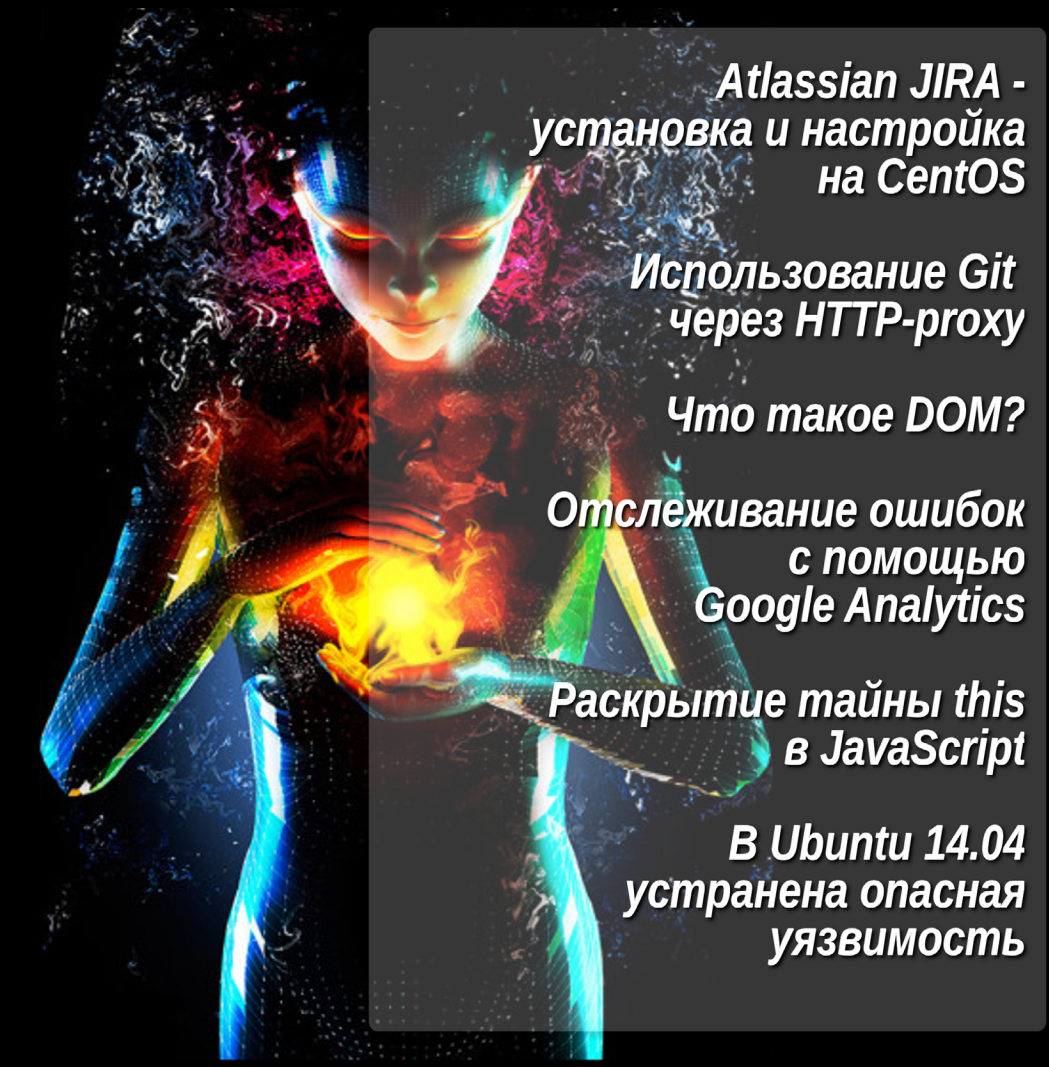
V14.05 (№9)
ISSN: 2223-6988

Попробуйте анонимную ОС, которой пользуется Сноуден

More Than USER

Приложение к журналу User And LINUX



Atlassian JIRA - установка и настройка на CentOS

Использование Git через HTTP-proxy

Что такое DOM?

Отслеживание ошибок с помощью Google Analytics

Раскрытие тайны this в JavaScript

В Ubuntu 14.04 устранена опасная уязвимость

Обновлен репозиторий UALinux для Ubuntu и Ubuntu-совместимых дистрибутивов. Добавлено и обновлено 35 приложений и 5 игр. На данный момент в репозитории игр - 375, приложений - 565.

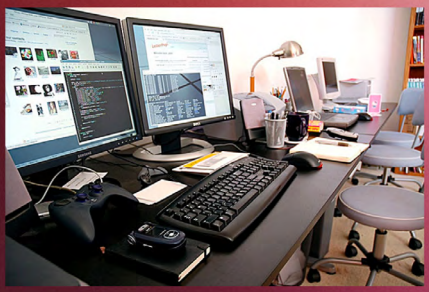
UA Linux
http://ualinux.com

ubuntu BusinessPack

Операционная система, которая идеально подходит для использования на персональных компьютерах и ноутбуках. Она ориентирована на простоту использования и удобство работы.

Включена необходимая подборка программного обеспечения, которая позволяет создать удобное рабочее окружение в корпоративной среде предприятия или на домашнем компьютере.

Ubuntu Business Pack это:



- **простая установка операционной системы не требующая особых знаний;**
- **уверенность в том, что на компьютере установлено только лицензионное программное обеспечение;**
- **это низкая цена по сравнению с аналогами;**
- **создание рабочего места без дополнительных финансовых затрат. Это существенно экономит бюджет организаций;**
- **идеальное решение для перехода на Linux с Windows, если вы все еще используете windows-приложения и игры;**
- **полная поддержка в системе русского, украинского и английского языков;**
- **отсутствие необходимости затрат на антивирусную защиту.**

Программное обеспечение имеет понятный графический интерфейс и полностью совместимо с популярными форматами документов, поэтому переход не вызывает никаких проблем с переносом данных и переквалификацией сотрудников.

поддержка широкого спектра современного оборудования;
дополнительные драйвера для видео-карт, wi-fi адаптеров и принтеров;
возможность использовать Windows-драйвера для WiFi-адаптеров USB;
управление веб-камерами.

безопасность и надежная защита от вирусов;
проверка файлов на вирусы в режиме реального времени (актуально в случае запуска windows-приложений);
защита от вирусных атак системы и электронной почты;
проверка на спам.

поддержка мультимедия (аудио - видео) различных форматов (avi, divX, mp4, mkv, amr, aac, Adobe Flash и многие другие)
просмотр защищенных, зашифрованных лицензионных, двухслойных DVD и Bluray дисков

полный набор офисных компонент (тексты, таблицы, презентации) совместимых с форматами MS Office
включена поддержка импорта файлов MS Visio
поддержка различных типов архивов (RAR, ACE, ARJ и других);

поддержка windows-приложений (гарантированный запуск более 130 приложений и более 600 игр)

OpenJDK

полноценная поддержка Java-приложений;
гарантированная работа онлайн банк-клиентов, таких как Приват24
гарантированная работа онлайн-бухгалтерии, таких как iFin.

от редакции

Дорогие читатели!

Мы спешим представить вам новый выпуск приложения «More Than USER».

Как вы могли заметить, мы сменили название и логотип журнала. Надеемся, что вам они придутся по вкусу.

Этот номер полон интересных и полезных статей. Вы узнаете как установить и настроить Atlassian JIRA на CentOS, научитесь использовать Git через HTTP-proxy, узнаете, что же такое DOM.

Также на страницах нашего журнала вы прочтете об опасной уязвимости в Ubuntu 14.04, которая была обнаружена и устранена, сможете попробовать анонимную ОС, которую использует всем известный Эдвард Сноуден. Для любителей JavaScript мы опубликовали статью, которая прольет свет на тайну функции this.

Приятного чтения, дорогие друзья! Оставайтесь с нами и оставляйте свои отзывы и предложения на нашем форуме http://ualinux.com/forum/magazine

Команда More Than USER

НАД ВЫПУСКОМ РАБОТАЛИ:

Звенигородская Анастасия
Попов Владимир
Шарай Игорь
Россошанский Андрей
Якимчук Сергей
Кирильчук Виктор
Безруков Марк

Содержание

SERVERS

Новый выделенный сервер: приемка и проверка 5
Atlassian JIRA: установка и настройка на CentOS 13

WORKSTATION

Использование Git через HTTP-proxy 20
MySQL: включаем логи 20

CONSOLE

BASH: скрипт бекапа с инкрементальным копированием
файлов и полным MySQL 22
Терминал Linux. Команды поиска файлов (продолжение) 25

PROGRAMMING

Что такое DOM? 28
Быстрая инициализация проекта 30
Отслеживание ошибок с помощью Google Analytics 37
Раскрытие тайны this в JavaScript 38
First Line Software 40

OTHERS

Файл подкачки: swap-файл и swap-раздел в Linux 42

SECURITY

Защита от ddos атак
В Ubuntu 14.04 устраненаопасная уязвимость 46

CYBERCRIME

Вышел первый релиз анонимной ОС, которой пользуется Сноуден 47
Через «дыру» в Adobe Flash Player шпионили за пользователями 49

Новый выделенный сервер: приемка и проверка

Начиная работу с новым сервером, нелишним будет проверить, соответствует ли он заявленной конфигурации. Многие начинающие пользователи испытывают затруднения в случаях, когда требуется просмотреть информацию о сервере с использованием команд, доступных только в консоли.

В этой статье мы расскажем о том, как можно получить спецификацию Linux-сервера в командной строке.

Общая информация о системе

Получить информацию о системе можно с помощью команды uname, которая пишет в стандартный вывод имя используемой операционной системы. Если указать одну из описанных ниже опций, в консоль будет выведена более специализированная информация о характеристиках системы:

```
# uname -a
Linux srv1 3.8.0-35-generic
#50-Ubuntu SMP Tue Dec 3
01:24:59 UTC 2013 x86_64 x86_64
x86_64 GNU/Linux
```

Имя операционной системы, дата компиляции ядра, версия и битность: Tue Dec 3 01:24:59 UTC 2013, 3.8.0-35-generic, i386 – 32 бита, x86_84 – 64 бита.

Ключи команды:

-a выводить всю возможную информацию;

-i показать информацию об идентификаторе ядра;

-m показать тип текущей аппаратной платформы;

-n показать имя системы;

-p показать тип процессорной архитектуры сервера;

-r показать информацию о релизе операционной системы;

-s показать имя используемой операционной системы;

-v показать информацию о версии используемой операционной систем.

Информацию об используемом дистрибутиве операционной системы можно также вывести при помощи команды cat /etc/issue:

```
# cat /etc/issue
Ubuntu 13.04 \n \l
```

Есть и другой способ просмотра информации о дистрибутиве:

```
# lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description: Ubuntu 13.04
Release: 13.04
Codename: raring
```

Информация об аппаратных компонентах. Утилита lshw

Утилита lshw выводит в консоль полный список аппаратных компонентов системы вместе с информацией об устройствах. lshw включена во многие современные дистрибутивы Linux по умолчанию; если же она отсутствует, ее можно установить стандартным менеджером пакетов:

```
# apt-get install lshw
```

Чтобы вывести в консоль информацию о «железе», нужно ввести следующую команду:

```
# lshw
```

Вывести эту информацию в сокращенном виде можно при помощи опции -short:

```
# lshw -short
```

С помощью lshw можно также просматривать и информацию об отдельных аппаратных компонентах системы. Для этого используется ключ -C, после которого указывается устройство, информацию о котором нужно вывести на консоль:

– процессор:

```
# lshw -C cpu
```

– память:

```
# lshw -C memory
```

– дисковая подсистема:

```
# lshw -C disk
```

Утилита lspci

С помощью утилиты lspci можно просмотреть информацию обо всех шинах PCI и подключенных к ним устройствах. Она входит в пакет pciutils, включенный в большинство современных дистрибутивов Linux; если же он по каким-либо причинам отсутствует, его можно установить при помощи стандартного менеджера пакетов.

По умолчанию lspci показывает краткий список устройств; более подробную информацию можно получить при помощи многочисленных опций.

Опция -t отображает информацию о шинах и подключенных устройствах в виде дерева. В выводе будут указаны только цифровые идентификаторы устройств:

```
# lspci -t
-[0000:00]-+-00.0
           +-01.0-[01]--+-00.0
           |            \-00.1
           +-03.0-[02]--+-00.0
           |            \-00.1
           +-07.0-[04]--
           +-09.0-[05]--
           +-14.0
           +-14.1
           +-1c.0-[03]----00.0
           +-1d.0
           +-1e.0-[06]----03.0
           +-1f.0
```

Просмотреть цифровые коды устройств можно с помощью опции -n:

```
# lspci -n
01:00.1 0200: 14e4:1639 (rev 20)
02:00.0 0200: 14e4:1639 (rev 20)
02:00.1 0200: 14e4:1639 (rev 20)
03:00.0 0104: 1000:0079 (rev 05)
06:03.0 0300: 102b:0532 (rev 0a)
```

В начале каждой строки в выводе указывается код устройства в формате «:.», а далее – код в формате «:».

Чтобы в список были включены не только коды, но и имена соответствующих им устройств, указывается опция -nn:

```
# lspci -nn
01:00.0 Ethernet controller
[0200]: Broadcom Corporation
NetXtreme II BCM5709 Gigabit
Ethernet [14e4:1639] (rev 20)
03:00.0 RAID bus controller
[0104]: LSI Logic / Symbios
Logic MegaRAID SAS 2108
[Liberator] [1000:0079] (rev 05)
06:03.0 VGA compatible
controller [0300]: Matrox
Electronics Systems Ltd. MGA
G200eW WPCM450 [102b:0532] (rev
0a)
```

Определить имя устройство по коду «:.» можно при помощи опции -s:

```
# lspci -s 03:00.0
03:00.0 RAID bus controller: LSI
Logic / Symbios Logic MegaRAID
SAS 2108 [Liberator] (rev 05)
```

Чтобы определить устройство по коду «:,» нужно воспользоваться опцией -d:

```
# lspci -d 1000:0079
03:00.0 RAID bus controller: LSI
Logic / Symbios Logic MegaRAID
SAS 2108 [Liberator] (rev 05)
```

После ключа -d можно указать только код поставщика или код устройства, например:

```
# lspci -d 8086:
```

```
# lspci -d :0532
```

В этом случае будет показан список всех устройств, соответствующих введенному коду.

Для просмотра информации о драйверах ядра, отвечающих за конкретные устройства, используется опция -k:

```
# lspci -k
00:1f.2 IDE interface: Intel
Corporation 82801IB (ICH9) 2
port SATA Controller [IDE mode]
(rev 02)
        Subsystem: Dell
PowerEdge R610 SATA IDE
Controller
        Kernel driver in use:
ata_piix
        Kernel modules: ata_
generic, pata_acpi, ata_piix
02:00.0 Ethernet controller:
Broadcom Corporation NetXtreme
II BCM5709 Gigabit Ethernet (rev
20)
        Subsystem: Dell
PowerEdge R610 BCM5709 Gigabit
Ethernet
        Kernel driver in use:
bnx2
        Kernel modules: bnx2
03:00.0 RAID bus controller: LSI
Logic / Symbios Logic MegaRAID
SAS 2108 [Liberator] (rev 05)
        Subsystem: Dell PERC
H700 Integrated
        Kernel driver in use:
megaraid_sas
        Kernel modules:
megaraid_sas
```

Псевдофайловая система /proc

Информация об аппаратных компонентах в Linux-системах хранится в так называемой псевдофайловой системе /proc. Она называется псевдофайловой, так как является виртуальной и вообще не занимает места на накопителе. Большинство хранимых в /proc псевдофайлов представлены в понятной для человека форме. Дерево /proc используют многие программы, выводящие информацию о системе.

Процессор

Информация о процессоре хранится в псевдофайле /proc/cpuinfo. Чтобы просмотреть его содержимое, введем следующую команду:

```
# cat /proc/cpuinfo
```

В выводе этой команды содержится много различной информации: о модели процессора, количестве ядер, поддерживаемых технологиях аппаратной виртуализации и т.п.

Самый объемный и трудный для понимания раздел вывода – это, конечно же, flags (флаги). Они содержатся и в выводе команды lshw.

Рассмотрим значения наиболее важных флагов:

ht (HyperThreading) – поддержка технологий одновременной многопоточности; присутствует в сериях процессоров Intel Xeon, Pentium 4, Atom, Core i3,

Core i5, Core i7;

lm (long mode) – указывается, если процессор выполнен по 64-битной

технологии;

vmx (для Intel), svd (для AMD) – поддержка процессором технологий аппаратной виртуализации; означает наличие инструкций для предоставления прямого доступа к ресурсам процессора из гостевых систем;

aes – поддержка расширения системы команд AES;

hypervisor – указывается, если ОС запущена под гипервизором;

smx – поддержка технологий TXT (TPM).

Память

Просмотреть информацию об общем объеме свободной и используемой памяти, включая swap, можно при помощи команды free. Вывод этой команды может выглядеть, например, так:

```
# free -m
             total       used       free     shared    buffers     cached
Memory:       3627       3216        410          0        107       1157
-/+ buffers/cached:       1950       1676
Swap:         3762         31       3731
```

Опция -m указывает, что объем свободной и используемой памяти нужно выводить в мегабайтах. Чтобы отображать объем в гигабайтах, нужно указать ключ -g; это удобно для серверов с большим (исчисляемым десятками, а то и сотнями гигабайт) объемом оперативной памяти.

Еще более подробная информация хранится в псевдофайле /proc/meminfo.

Вывод команды cat /proc/meminfo включает следующие основные параметры:

MemTotal – доступный объем оперативной памяти;

MemFree – показывает, какой объем памяти в данный момент не используется и доступен для выделения процессам;

Buffers – область памяти, в которой хранятся данные, ожидающие записи на диск;

Cached – объем, занятый под кэш чтения страниц с диска;

SwapCached – объем, который был перенесен в область подкачки, а затем перемещен обратно в оперативную память;

Active – объем памяти, занятый наиболее часто используемыми страницами;

Inactive – объем памяти, занятый страницами, которые в настоящий момент не используются;

Swap {total, free} – общий объем области подкачки;

Dirty – так называемые «грязные» страницы (т.е. находящиеся в оперативной памяти, но еще не сброшенные на диск);

Writeback – страницы, сбрасываемые на диск в настоящий момент;

AnonPages – анонимные страницы (данные, используемые программами, но не ассоциированные с каким-либо файлом);

Mapped – общий объем памяти, перенесенный в виртуальное адресное пространство процессов при помощи mmap;

Committed_AS – количество памяти, выделенное всем процессам (даже если они при этом не используют ее в полном объеме).

Дисковая подсистема

Для проверки разбивки и количества дисков используется команда:

```
# fdisk -l
```

Размер свободного и занятого дискового пространства во всех смонтированных файловых системах можно узнать с помощью команды df. С командой используются следующие опции:

-a вывести информацию обо всех файловых системах;

-h вывести информацию в человекочитаемом формате;

-T показать тип файловой системы;

-t вывести информацию только об указанных типах файловых систем.

Рассмотрим пример вывода команды df -h более подробно:

```
# df -h
Filesystem                Size
Used  Avail Use% Mounted on
/dev/mapper/vg0-vg0root    50G
15G   32G   32%  /
tmpfs                     5.9G
0     5.9G   0%  /dev/shm
/dev/sda1                 1008M
62M   895M   7%  /boot
/dev/mapper/vg0-var       2.7T
839G  1.7T  33%  /var
```

Информация о размере фактического свободного пространства отображается в разделе Available. Если сложить цифры, указанные в разделах Available и Used, то полученная сумма не будет равна цифре в разделе Size. Это связано с тем, что часть дискового пространства отводится под системные файлы и метаданные.

Просмотреть подробную информацию о состоянии жесткого диска можно при помощи утилиты smartctl, включенной в официальные репозитории большинства современных дистрибутивов Linux. Для просмотра полной информации нужно ввести команду:

```
# smartctl -a /dev/sda
```

Для отображения информации о физических томах используются команды pvdisplay, pvs и pvscan.

Команда pvscan проверяет все блочные устройства в системе на наличие физических томов:

```
# pvscan
  PV /dev/md0   VG vg0   lvm2
[462.96 GiB / 205.22 GiB free]
  Total: 1 [462.96 GiB] / in
use: 1 [462.96 GiB] / in no VG:
0 [0   ]
```

С помощью команды pvdisplay можно просмотреть подробный многострочный вывод для каждого физического тома:

```
# pvdisplay
  --- Physical volume ---
  PV Name               /dev/md0
  VG Name               vg0
  PV Size               462.96
GiB / not usable 1.87 MiB
  Allocatable           yes
  PE Size               4.00 MiB
  Total PE              118517
  Free PE               52536
  Allocated PE          65981
  PV UUID               Idm6eZ-
5vS0-IJCo-RDQq-WZNk-nJ22-eb7aDd
```

С помощью команды pvs можно настроить формат отображения данных (для каждого тома – одна строка).Это бывает полезно, например, при написании скриптов.

Для просмотра информации о логических томах жесткого диска используются утилиты lvs, lvscan и lvdisplay, входящие в пакет lvm2. lvm2 устанавливается при помощи стандартного менеджера пакетов:

```
# apt-get install lvm2
```

Команда lvscan выводит в консоль список всех имеющихся в системе логических томов (листинг фрагмента вывода):

```
# lvscan
  ACTIVE            '/dev/vg0/
root' [18.62 GiB] inherit
  ACTIVE            '/dev/vg0/
www' [200.00 GiB] inherit
```

С помощью команды lvdisplay можно вывести на консоль список атрибутов логических томов (имя, размер, разметка). Просмотреть информацию об атрибутах конкретного логического тома можно, воспользовавшись опцией -v и указав его имя (листинг фрагмента вывода):

```
# lvdisplay
  --- Logical volume ---
  LV Name                /dev/
vg0/root
  VG Name                vg0
  LV UUID                yPtVFt-
BON5-agWC-jXSr-cU4x-Tcu9-NRiwMF
```

```
  LV Write Access        read/
write
  LV Status
available
  # open                 1
  LV Size                18.62
GiB
  Current LE             4768
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           253:0

  --- Logical volume ---
  LV Name                /dev/
vg0/www
  VG Name                vg0
  LV UUID                reCzuE-
5dgN-A4eB-LubM-VtUA-Lclq-MUt5v6
  LV Write Access        read/
write
  LV Status
available
  # open                 1
  LV Size                200.00
GiB
  Current LE             51200
  Segments               2
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           253:2
```

Команда lvs аналогична рассмотренной выше команде pvs: позволяет настроить формат отображения данных и выводит по одному тому каждой строке.

```
# lvs
  LV    VG   Attr   LSize
Origin Snap%  Move Log Copy%
Convert
  root  vg0  -wi-ao  18.62g
  www   vg0  -wi-ao 200.00g
```

Информация о состоянии активных программных RAID-массивов хранится в псевдофайле /proc/mdstat. Просмотреть его содержимое можно с помощью команды:

```
# cat /proc/mdstat
Personalities : [raid1]
read_ahead 1024 sectors
md1 : active raid1 hda3[0]
hdc3[1]
      522048 blocks [2/2] [UU]
md0 : active raid1 hda2[0]
hdc2[1]
      4192896 blocks [2/2] [UU]
md2 : active raid1 hda1[0]
hdc1[1]
      128384 blocks [2/2] [UU]
```

В рассматриваемом нами примере в системе содержится три массива. Для каждого из них в псевдофайле /proc/mdstat имеется отдельный раздел, содержащий следующую информацию:

имя RAID-массива;
состояние RAID-массива;
уровень массива;
имена физических разделов, входящих в состав массива;
число настроенных устройств и число работающих устройств в массиве;
состояние каждого работающего устройства (U означает, что устройство

работает, а _ – что устройство отказало, sync – идет синхронизация).

Проверить, используется ли на сервере аппаратный RAID, можно с помощью уже знакомой нам команды lspci:

```
# lspci -nn | grep RAID
```

Если RAID используется, то на консоль будет выведен ответ типа:

```
02:00.0 RAID bus controller
[0104]: LSI Logic / Symbios
Logic MegaRAID SAS 2108
[Liberator] [1000:0079] (rev 04)
```

Информацию о состоянии аппаратного RAID при помощи штатных средств операционной системы получить невозможно. Для этого существуют специальные утилиты: MegaCLI для LSI-контроллеров и Adaptec Storage Manager (asm) для adaptec. В официальные репозитории Linux-систем они не включены. Скачать megacli можно отсюда hwraid.le-vert.net, а ASM – с сайта компании Adaptec www.adaptec.com.

Сетевые интерфейсы

Информация обо всех сетевых интерфейсах, подключенных к системе, содержится в псевдофайле /proc/net/dev. При вводе команды cat /proc/net/dev в консоль будет выведен список всех активных и неактивных сетевых интерфейсов.

Статус всех текущих интерфейсов можно просмотреть с помощью команды ip link show up.

Команда ip address выводит информацию обо всех сетевых интерфейсах:

```
# ip address
1: lo:  mtu 16436 qdisc noqueue
state UNKNOWN
      link/loopback
00:00:00:00:00:00 brd
00:00:00:00:00:00
      inet 127.0.0.1/8 scope
host lo
      inet6 ::1/128 scope host
  valid_lft forever preferred_
lft forever
2: eth0:  mtu 1500 qdisc pfifo_
fast state UP qlen 1000
      link/ether
00:30:48:f2:7a:a0 brd
ff:ff:ff:ff:ff:ff
      inet 5.178.83.252/29 brd
5.178.83.255 scope global eth0
      inet6
fe80::230:48ff:fef2:7aa0/64
scope link
  valid_lft forever preferred_
lft forever
3: eth1:  mtu 1500 qdisc noop
state DOWN qlen 1000
      link/ether
00:30:48:f2:7a:a1 brd
ff:ff:ff:ff:ff:ff
```

UP означает, что интерфейс работает; NO CARRIER означает отсутствие кабеля или трансивера в порту сетевой карты. Команда ip route (сокращенный вариант– ip r) выводит на консоль таблицы маршрутизации.

```
# ip r
default via 88.93.16.185 dev br0
50.178.87.0/24 via 192.16.122.1
dev br0
10.0.0.0/8 via 192.16.122.1 dev
```

```
br0
1.131.251.0/24 via 192.16.122.1
dev br0
192.16.122.0/24 dev br0  proto
kernel  scope link  src
192.16.122.2
88.93.16.184/29 dev br0  proto
kernel  scope link  src
88.93.16.186
```

Блог компании Селектел, habrahabr.ru

Atlassian JIRA: установка и настройка на CentOS

Установка будет производится на:

```
# cat /etc/redhat-release
CentOS release 6.4 (Final)
```

Хотя, т.к. JIRA работает на Java и Tomcat – то система роли особо не играет.

В качестве сервера базы данных будет использоваться MySQL:

```
# yum list installed | grep
mysql-server
mysql-server.i686
5.5.34-1.el6.remi          @remi
```

Установка будет проводится не под пользователем root, поэтому — создадим пользователя вручную:

```
# useradd jira
# passwd jira
# su -l jira
07:05:52 [jira@localhost ~] $
```

Создадим временную директорию для установки:

```
$ mkdir jira && cd jira
```

Для загрузки – переходим на страницу https://www.atlassian.com/software/jira/download и внизу кликаем на «All JIRA Download Options» Выбираем «JIRA 6.2.1 (Linux 32 Bit Installer) или JIRA 6.2.1 (Linux 64 Bit Installer)»

Качаем:

```
$ wget http://www.atlassian.com/
software/jira/downloads/binary/
atlassian-jira-6.2.1-x32.bin
$ ls -l
total 207848
-rw-r--r-- 1 jira jira 212835395
Mar 31 05:43 atlassian-jira-
6.2.1-x32.bin
```

Устанавливаем бит выполнения:

```
$ chmod u+x atlassian-jira-
6.2.1-x32.bin
```

И запускаем установку:

```
$ ./atlassian-jira-6.2.1-x32.bin
Unpacking JRE ...
Starting Installer ...
Mar 31, 2014 5:46:03
```

```
AM java.util.prefs.
FileSystemPreferences$1 run
INFO: Created user preferences
directory.
You do not have administrator
rights to this machine and as
such, some installation options
will not be available. Are you
sure you want to continue?
Yes [y, Enter], No [n]
y

This will install JIRA 6.2.1 on
your computer.
OK [o, Enter], Cancel [c]
o

Choose the appropriate
installation or upgrade option.
Please choose one of the
following:
Express Install (use default
settings) [1], Custom Install
(recommended for advanced users)
[2, Enter], Upgrade an existing
JIRA installation [3]
1

See where JIRA will be installed
and the settings that will be
used.
Installation Directory: /home/
jira/atlassian/jira
Home Directory: /home/jira/
atlassian/application-data/jira
HTTP Port: 8080
RMI Port: 8005
Install as service: No
Install [i, Enter], Exit [e]
iExtracting files ...
lib/rt.jar
...
Please wait a few moments while
JIRA starts up.
Launching JIRA ...
Installation of JIRA 6.2.1 is
complete
Your installation of JIRA 6.2.1
is now ready and can be accessed
via your
browser.
JIRA 6.2.1 can be accessed at
http://localhost:8080
Finishing installation …
```

Если выбрать «Custom Install» – можно задать свои параметры установки:

```
Express Install (use default
settings) [1], Custom Install
(recommended for advanced users)
[2, Enter], Upgrade an existing
JIRA installation [3]
2

Where should JIRA 6.2.2 be
installed?
[/home/jira/atlassian/jira]

Default location for JIRA data
[/home/jira/atlassian/
application-data/jira]

Configure which ports JIRA will
use.
JIRA requires two TCP ports that
are not being used by any other
applications on this machine.
```

```
The HTTP port is where you will
access JIRA
through your browser. The
Control port is used to Startup
and Shutdown JIRA.

Use default ports (HTTP: 8080,
Control: 8005) - Recommended [1,
Enter], Set custom value for
HTTP and Control ports [2]
2

HTTP Port Number
[8080]
8084
Control Port Number
[8005]
8007

Extracting files ...
  atlassian-jira/includes/
jquery/plugins/shorten/shorten-
min.js
```

Возвращаемся в домашнюю директорию:

```
$ cd ../
```

Установка произведена в каталог atlassian:

```
$ ls -l
total 8
drwxrwxr-x 4 jira jira 4096 Mar
31 05:54 atlassian
drwxrwxr-x 3 jira jira 4096 Mar
31 05:50 jira
```

Структура каталогов JIRA выглядит так:

```
$ tree -d -L 3 atlassian/
atlassian/
├── application-data
│   └── jira
└── jira
    ├── atlassian-jira
    │   ├── aui-examples
    │   ├── decorators
    │   ├── func
    │   ├── images
    │   ├── includes
    │   ├── META-INF
    │   ├── portlets
    │   ├── secure
    │   ├── static
    │   ├── static-assets
    │   ├── styles
    │   ├── template
    │   ├── ui
    │   ├── views
    │   └── WEB-INF
```

```
22
├── bin
23
│   └── apr
24
├── conf
25
├── external-source
26
├── __i4j_jre_backup
27
│   ├── bin
28
│   ├── lib
29
│   ├── man
30
│   └── plugin
31
├── jre
32
│   ├── bin
33
│   ├── lib
34
│   ├── man
35
│   └── plugin
36
├── lib
37
├── logs
38
├── temp
39
├── tomcat-docs
40
├── webapps
41
└── work
42
└── Catalina
```

Переходим в JIRA Installation Directory:

```
$ cd ~/atlassian/jira/
```

Управление запуском и остановкой выполняется через через скрипты start-jira.sh и stop-jira.sh в каталоге bin.

После установки JIRA уже должна быть запущена.

Проверяем:

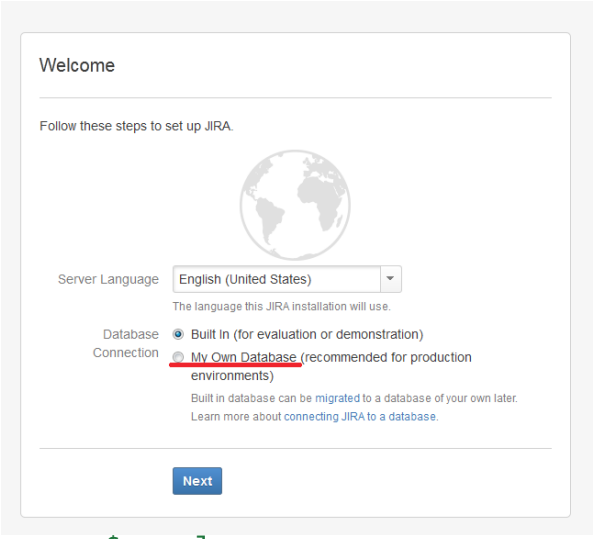
```
$ ps ux | grep jira | grep java
| grep -v grep
jira 4864 28.5 41.3 1283824
425916 pts/1 Sl 06:48 0:44 /
home/jira/atlassian/jira/
jre//bin/java -Djava.util.
logging.config.file=/home/jira/
atlassian/jira/conf/logging.
properties -Djava.util.
logging.manager=org.apache.
juli.ClassLoaderLogManager
-XX:MaxPermSize=384m
-Xms384m -Xmx768m -Djava.awt.
headless=true -Datlassian.
standalone=JIRA -Dorg.apache.
jasper.runtime.BodyContentImpl.
LIMIT_BUFFER=true -Dmail.
mime.decodeparameters=true
-Dorg.dom4j.factory=com.
atlassian.core.xml.
InterningDocumentFactory
-XX:+PrintGCDateStamps -XX:-
OmitStackTraceInFastThrow
-Djava.endorsed.dirs=/home/
jira/atlassian/jira/endorsed
-classpath /home/jira/atlassian/
```

```
jira/bin/bootstrap.jar:/
home/jira/atlassian/
jira/bin/tomcat-juli.jar
-Dcatalina.base=/home/jira/
atlassian/jira -Dcatalina.
home=/home/jira/atlassian/
jira -Djava.io.tmpdir=/
home/jira/atlassian/jira/
temp org.apache.catalina.
startup.Bootstrap start

$ netstat -np | grep 8080

(Not all processes could
be identified, non-owned
process info
will not be shown, you
would have to be root to
see it all.)
tcp 0 0
::ffff:10.***.***.104:8080
::ffff:10.***.***.15:58307 TIME_
WAIT -
tcp 0 0
::ffff:10.***.***.104:8080
::ffff:10.***.***.15:58306 TIME_
WAIT -
tcp 0 0
::ffff:10.***.***.104:8080
::ffff:10.***.***.15:58299 TIME_
WAIT -
tcp 0 0
::ffff:10.***.***.104:8080
::ffff:10.***.***.15:58305 TIME_
WAIT -
```

Заходим на страницу хоста, и видим предложение продолжить установку:



Переходим к серверу MySQL:

```
$ mysql -u root -p
Enter password:
```

Создаем базу:

```
mysql> CREATE DATABASE jiradb
CHARACTER SET utf8 COLLATE utf8_
bin;
Query OK, 1 row affected (0.07
sec)
```

Назначаем права доступа:

```
mysql> GRANT SELECT,INSERT
,UPDATE,DELETE,CREATE,DROP,
ALTER,INDEX on jiradb.* TO
'jiradb'@'localhost' IDENTIFIED
BY 'jiradb';
Query OK, 0 rows affected (0.01
sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.04
sec)
```

Пробуем зайти под созданным пользо-

вателем и проверяем доступ к базе:

```
$ mysql -u jiradb -pjiradb
```

```
Welcome to the MySQL monitor. Commands end with ; or \g.
...
1
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| jiradb             |
+--------------------+
2 rows in set (0.03 sec)

mysql> use jiradb;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
```

Для дальнейшей работы нам понадобится драйвер JDBC-коннектора для MySQL.

Качаем его со страницы http://dev.mysql.com/downloads/connector/j. Выбираем «Platform Independent» и тип tar.gz:

```
$ cd && mkdir tmp
$ cd tmp/
$ wget http://dev.mysql.com/get/Downloads/Connector-J/mysql-connector-java-5.1.29.tar.gz
--2014-03-31 07:00:50-- http://dev.mysql.com/get/Downloads/Connector-J/mysql-connector-java-5.1.29.tar.gz
...
Length: 3614703 (3.4M) [application/x-tar-gz]
Saving to: “mysql-connector-java-5.1.29.tar.gz”
...
2014-03-31 07:00:55 (823 KB/s) - “mysql-connector-java-5.1.29.tar.gz” saved [3614703/3614703]
```

Распаковываем архив:

```
$ tar xfp mysql-connector-java-5.1.29.tar.gz
```

Находим jar-файл с драйвером:

```
$ find mysql-connector-java-5.1.29 -name "mysql-connector-java-*-bin.jar"
mysql-connector-java-5.1.29/mysql-connector-java-5.1.29-bin.jar
```

И копируем его в директорию JIRA Installation Directory/lib:

```
$ cp mysql-connector-java-5.1.29/mysql-connector-java-5.1.29-bin.jar ~/atlassian/jira/lib/
```

Переходим в домашнюю директорию:

```
$ cd
```

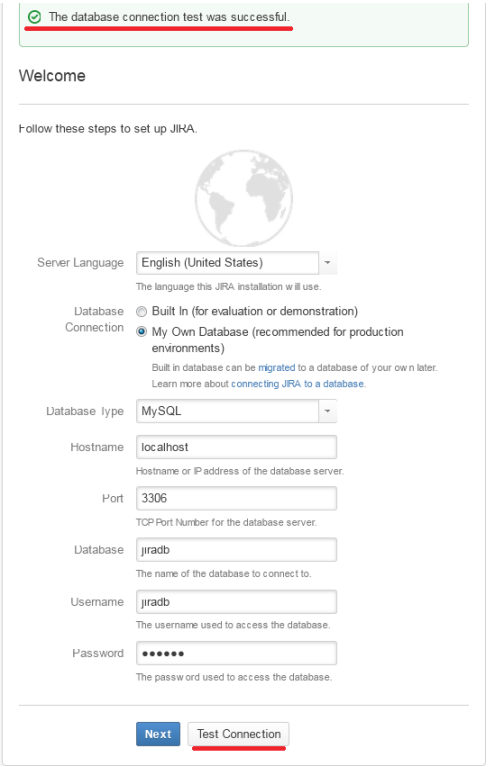
Перезапускаем JIRA:

```
$ ./atlassian/jira/bin/stop-jira.sh
executing as current user
...
```

```
$ ./atlassian/jira/bin/start-jira.sh
To run JIRA in the foreground, start the server with start-jira.sh -fg
executing as current user
...
```

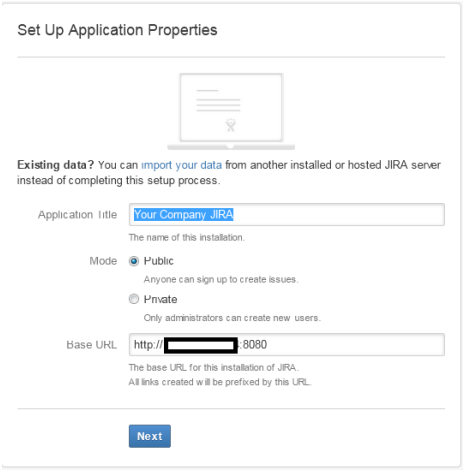
Обновляем страницу в браузере и продолжаем установку. После заполнения полей – жмем «Test connection» для проверки:

И «Next», если все в порядке. Ждем, пока JIRA заполнит таблицы в базе, и попадаем на следующую страницу:



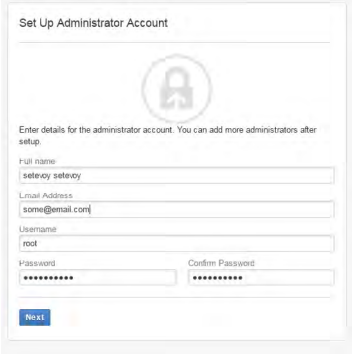
Нажимаем «Next» и попадаем на страницу лицензирования

Заполняем по своему усмотрению, и по-



падаем на следующую страницу, где создаем аккаунт администратора:

На следующей странице нам предложат настроить email-нотификацию, после чего – мы сможет, наконец-то, попасть в веб-интерфейс нашей новой JIRA:



Готово.

rtfm.co.ua

Использование Git через HTTP-proxy

При попытке использовать git на системе, находящейся за proxy-сервером – получаем сообщение об ошибке:

```
# git clone https://github.com/
graphite-project/carbon.git
Initialized empty Git repository
in /home/setevoy/carbon/.git/
error: Failed connect to
github.com:443; Operation now
in progress while accessing
https://github.com/graphite-
project/carbon.git/info/refs

fatal: HTTP request failed
```

Что бы настроить git на использование прокси – создаем файл настроек:

```
$ touch /home/setevoy/.gitconfig
$ cd /home/setevoy/
```

Следующей командой – добавляем в файл данные доступа к нашему прокси:

```
$ git config --global http.proxy
http://proxyuser:proxypass@
proxyaddress:8080
```

Проверим содержимое файла:

```
$ cat .gitconfig
[http]
proxy = http://
proxyuser:proxypass@
proxyaddress:8080
```

И еще раз попробуем скачать необходимое:

```
$ git clone https://github.com/
graphite-project/graphite-web.
git
Initialized empty Git repository
in /home/setevoy/nikita/
graphite-web/.git/
...
Receiving objects: 100%
(13833/13833), 17.
```

rtfm.co.ua

MySQL: включаем логи

Сам MySQL уже должен быть установлен, например – во время установки LAMP, как было описано в прошлом номере «Больше, чем USER».

Чтобы включить подробное логгирование запросов ко всем базам MySQL, надо добавить запись в файл my.cnf.

Сначала удостоверимся, что этот файл есть в нашей системе:

```
# locate my.cnf
# whereis my.cnf
# find my.cnd
```

Если никакая из приведенных команд результата не принесла – выполняем следующее.

При установке, MySQL создает несколько демонстрационных файлов и размещает их в каталоге /usr/local/share/mysql/. Среди них есть четыре варианта:

my-huge.cnf – для сервера с огромной нагрузкой;

my-large.cnf – для сервера с большой нагрузкой;

my-medium.cnf – средней нагрузкой;

my-small.cnf – маленькой нагрузкой.

Выберите нужный вариант, и выполните:

```
# cp /usr/local/share/mysql/my-small.cnf /var/db/mysql/my.cnf
```

где my-small.cnf – вариант файла конфигурации, который вы выбрали для своего сервера. Например my-small.cnf.

Путь, куда вы будете копировать может быть разным: /var/db/mysql/, /etc/ или /usr/local/etc/ – MySQL при запуске выполнит поиск файла во всех этих каталогах.

Далее, в файле, который вы скопировали (.те. /var/db/mysql/my.cnf, если вы не выбрали другой путь), найдите блок [mysqld] и в его конец добавьте строку:

```
log=/var/log/mysql.log
```

Опять-таки, расположение файла лога может любым, на ваше усмотрение.

Теперь – создайте сам файл и установите на него права для MySQL:

```
# touch /var/log/mysql.log
# chown mysql:mysql /var/log/mysql.log
```

Перезагрузите MySQL:

```
# /usr/local/etc/rc.d/mysql-server restart
```

Теперь все запросы к базам MySQL будут записываться в этот лог.

ВАЖНО: обязательно настроить ротацию лога, иначе он быстро разрастется до слишком больших размеров.

В файл конфигурации /etc/newsyslog.conf добавьте строку:

```
/var/log/mysql.log mysql:mysql 600 2 100 $W6D0 JB /var/db/mysql/akira.pid
```

где 100 – это размер файла лога, после которого он будет заархивирован, а akira.pid – название pid-файла с именем вашего сервера.

Перезапустите newsyslod для применения изменений:

```
# /etc/rc.d/newsyslog restart
Creating and/or trimming log files.
```

Теперь можно посмотреть сам лог:

```
# tail -f /var/log/mysql.log
```

rtfm.co.ua

BASH: скрипт бекапа с инкрементальным копированием файлов и полным MySQL

Скрипт предназначен для создания резервной копии JIRA, но может использоваться для любых целей.

Скрипт создает 1 раз в неделю (воскресенье) полную копию файлов (база данных дампится полностью каждый раз), и каждый день – копию файлов, которые были изменены за последние сутки + полную копию базы.

Также – может быть запущен вручную, для создания полной копии в отдельном каталоге.

Инкрементальные бекапы MySQL можно делать с помощью MySQL Enterprise Backup – но тут она не используется.

Как и другие скрипты – используется getopts () для удобства запуска.

```
$ ./jira_backup.sh -h
```

Usage: ./jira_backup.sh [options]
This script can create full or incremental backup of JIRA and it's database.
Available options are:

-h | help - show this help and exit;
-m | manual - run full backup in to directory "Files manual" and "MySQL manual" directories;
-a | auto - if started at Sunday will run full backup in to "Files weekly" and "MySQL weekly" directories, otherwise - will copy only files modified last 24 hour in to "Files daily" and "MySQL daily".

Directories list:
Files daily: /home/jira/backup/files/daily
Files weekly: /home/jira/backup/files/weekly
Files manual: /home/jira/backup/files/manual

MySQL daily: /home/jira/backup/mysql/daily
MySQL weekly: /home/jira/backup/mysql/weekly
MySQL manual: /home/jira/backup/mysql/manual

Daily backups stored 7 days. Weekly - 4 copies (i.e. - one month).
All activities written to log /var/log/jira_backup.log.

Запускается скрипт по cron-у, в 01:30:

```
$ crontab -l
```

30 1 * * * /home/jira/bin/jira_backup.sh -a

Содержимое скрипта:

```
#!/bin/bash
jira_home="/home/jira/atlassian"
bklog="/var/log/jira_backup.log"
dbkdir="/home/jira/backup/files/daily"
wbkdir="/home/jira/backup/files/weekly"
manbkdir="/home/jira/backup/files/
```

```
manual"

  mdbkdir="/home/jira/backup/mysql/daily"
  mwbkdir="/home/jira/backup/mysql/weekly"
  manmbkdir="/home/jira/backup/mysql/manual"
  curdate=`date +%Y-%M-%d-%H:%M:%S`
  ardate=`echo $curdate | sed -e 's/:/_/g' -e 's/-/_/g'`
  wday=`date +%a`
  usage () {
  cat << EOF
  Usage: $0 [ options ]
  This script can create full or incremental backup of JIRA and it's database.
  Available options are:
   -h | help - show this help and exit;
    -m | manual - run full backup in to directory "Files manual" and "MySQL manual" directories;
    -a | auto - if started at Sunday will run full backup in to "Files weekly" and "MySQL weekly" directories, otherwise - will copy only files modified last 24 hour in to "Files daily" and "MySQL daily".
  Directories list:
  Files daily: $dbkdir
  Files weekly: $wbkdir
  Files manual: $manbkdir
  MySQL daily: $mdbkdir
  MySQL weekly: $mwbkdir
  MySQL manual: $manmbkdir
  Daily backups stored 7 days. Weekly - 4 copies (i.e. - one month).
  EOF
  }
  clean_daily () {
  while [[ ! -z $@ ]]
  do
   echo -e "Cleaning directory $1 - deleting archives older than 7 days...\n"
    # в переданных аргументами директориях находим и удаляем архивы старше 7-ми дней
    find $1 -mtime +7 -exec rm -f {} \;
    shift
  done
  }
  clean_weekly () {
  while [[ ! -z $@ ]]
  do
   echo -e "Cleaning directory $1 - deleting old archives...\n"
    # в переданных аргументами директориях находим и удаляем все архивы, кроме последних 4-х
    cd $1
    [[ `pwd` == $1 ]] && rm -f `ls -1 | tail -n +5` || { echo "Wrong dir $1! Exit."; exit 1; }
    shift
  done
  }
  files_full_backup () {
   echo -e "Archiving files...\n"
     tar cjpf "$1/full_files_$ardate.tar.bz2" "$jira_home"
  }
  files_inc_backup () {
   echo -e "Archiving files...\n"
   find $jira_home -mtime -1 -exec tar cvjpf "$1/inc_files_$ardate.tar.bz2" {} \;
  }
  mysql_backup () {
   echo -e "Dumping database...\n"
   mysqldump -u dbuser -pdbpass dbname > $1/dbname_$ardate.sql
```

```
}
{
[[ -z $1 ]] && usage
auto=0
manual=0
while getopts "ham" opt; do
 case $opt in
      h)
       usage && exit 1
       ;;
      a)
       auto=1
       ;;
      m)
       manual=1
       ;;
      \?)
       usage && exit 1
       ;;
 esac
done
if [ $auto == 1 ]; then
    echo -e "Started automated backup process at $curdate.\n"
    if [ $wday != Sun ]
    then
     echo -e "As today is not Sunday - I'll start incremental backup.\n"
      files_inc_backup $dbkdir
      mysql_backup $mdbkdir
      clean_daily $dbkdir $mdbkdir
    else
     echo -e "As today is Sunday - I'll start full backup.\n"
      files_full_backup $wbkdir
      mysql_backup $mwbkdir
      clean_weekly $wbkdir $mwbkdir
    fi
  fi
  if [ $manual == 1 ]; then
   files_full_backup $manbkdir
   mysql_backup $manmbkdir
  fi
  echo -e "Backup finished at $curdate.\n"
  } 2>&1 | tee -a "$bklog"
```

rtfm.co.ua

Терминал Linux. 3 статья - команды поиска файлов (продолжение)

Цикл статей о терминале:
1. Терминал Linux. 1 статья - команды навигации в терминале.
2. Терминал Linux. 2 статья - команда поиска файлов и директорий в терминале
3. Терминал Linux. 3 статья - команды поиска файлов (продолжение)
4. Терминал Linux. 4 статья - создание, удаление, форматирование, монтирование разделов жесткого диска
5. Терминал Linux. 5 статья - создание aliases (псевдонимов) в Ubuntu

В этой статье мы поговорим о других командах поиска файлов в терминале, помимо find.

Помимо find, для поиска файлов существует еще команда locate.

Разница между ними в том, что locate использует собственную базу данных для хранения имен файлов, а find исследует директории в поисках заданного параметром командной строки имени файла.

Команда locate ищет файлы очень быстро, так как она производит поиск не по файловой системе, а по собственной базе данных.

Поэтому при использовании данной команды следует использовать команду, которая обновит базу данных индексов файлов в системе:

```
sudo updatedb
```

Если не выполнить ее, то в поиске могут быть выведены удаленные файлы или не будут выведены только что созданные.

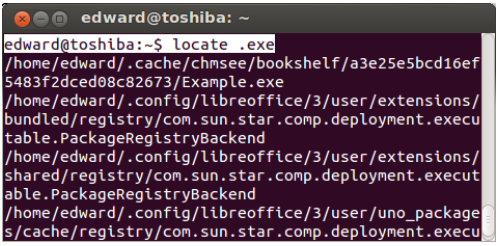
Начнем использование данной команды:

В обычном виде вводится команда locate и имя файла, который хотим найти:

```
locate file
```

Если мы хотим найти все файлы с расширением .exe, нужно использовать команду:

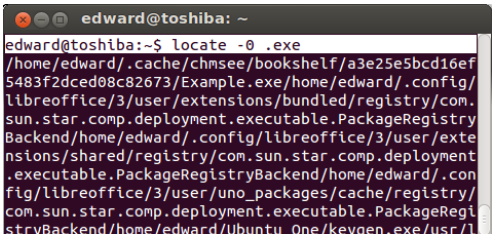
```
locate .exe
```



Если мы хотим вывести результаты поиска в одну строку, тогда используем опцию -0:

```
locate -0 .exe
```

Вот результат:



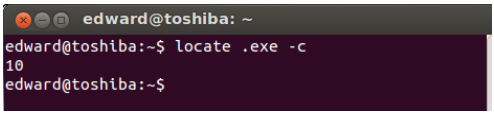
Регистр учитывается, поэтому если мы ищем файл с названием 'file' , то не найдутся файлы с именами в другом регистре: File, FILE и т.д.

Чтобы это исправить, нужно использовать опцию -i , то есть:

```
locate file -i
```

Если мы хотим узнать, сколько файлов имеют в своих именах текст "file" или файлов с расширением .exe и т.д., то нужно указать опцию -c:

```
locale .exe -c
```



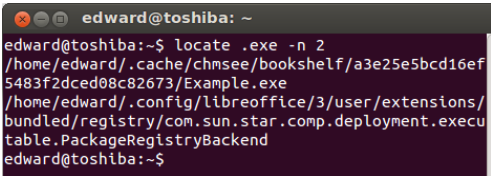
Вывод команды в примере означает, что в системе 10 файлов имеют расширение .exe

Если мы хотим ограничить число выводимых файлов в поиске по заданному параметру, мы ставим опцию -n и вводим число:

```
locate .exe -n 2
```

То есть данная команда с опцией -n 2 выведет нам только 2 первых найденных файла:

Эту команду лучше выполнять с опцией -e, тогда будут отображены файлы, которые существуют в системе.



Т.е. даже тогда, когда запись о файле находится в базе данных, все равно будет осуществлена проверка физического нахождения файла в системе Linux, перед выводом команды locate:

```
locate .exe -e
```

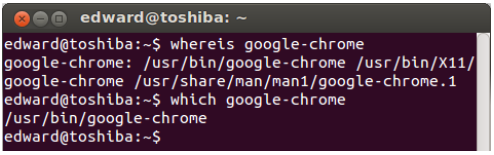
Полезные команды "whereis" и "which"

В нашей системе установлены различные программы и для того, чтобы узнать размещение бинарных файлов, исходных кодов и руководств, относящихся к установленной программе, неоходимо выполнить команду whereis. Для примера найдем информацию о пакете google-chrome:

```
whereis google-chrome
```

Команда же which отображает полный путь к установленной программе, также в качестве примера возьмем google-chrome:

```
which google-chrome
```

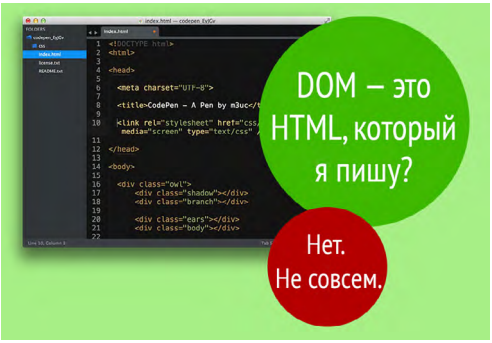


Магазин "TOTAL"

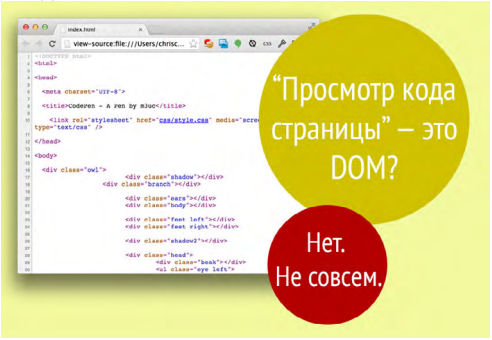
- персональные компьютеры;
- компьютерные комплектующие;
- ноутбуки, нетбуки, планшеты;
- принтеры, МФУ, расходники;
- сетевое оборудование;
- CD/DVD диски, флеш-накопители;
- и многое другое.

г. Кривой Рог, ул. Адмирала Головко, 40, Терновской р-н
тел. (067)-698-87-79, (097)-692-73-38

Что такое DOM?



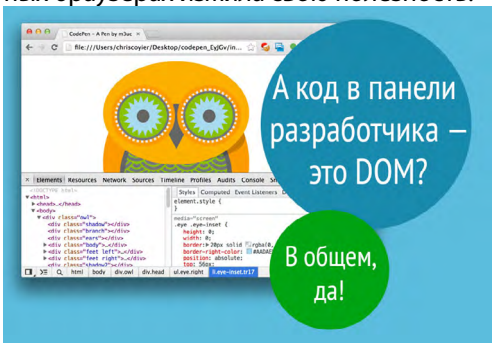
HTML-код, который вы пишете, анализируется и преобразуется браузером впоследствии в DOM.



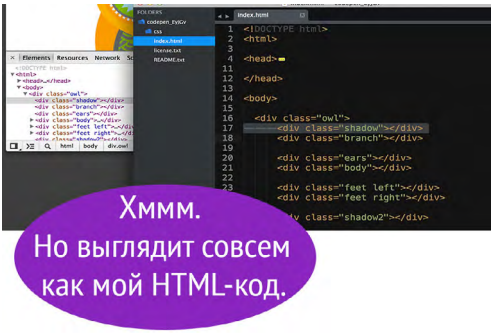
Функция «просмотра кода страницы» всего лишь показывает HTML-код, формирующий страницу. Это практически тот самый HTML-код, который вы написали в редакторе.

Этот код может выглядеть несколько иначе, если, например, вы работаете с файлами шаблона в бэкенде, и не очень-то часто заглядываете в HTML-код, получаемый на выходе. Или, если перед публикацией страницы на сайте ваш HTML-код проходит через некоторый «процесс сборки». Этот код может, например, сжиматься перед публикацией или изменяться каким-либо другим способом.

В целом, режим просмотра кода страницы – немного странная штука. На самом деле, этот код интересен только разработчикам, но в то же время, все современные браузеры уже имеют встроенные инструменты для разработчиков (DevTools и аналоги). Вероятно, эта функция в современных браузерах изжила свою полезность.



Если вы откроете панель инструментов разработчика и увидите там что-то вроде HTML, то вы обнаружите, что на самом деле это – визуальное представление DOM. Ура, мы наконец-то нашли DOM!



Да, это действительно так и есть. Помните, DOM формируется из написанного вами HTML-кода. В самом упрощенном варианте визуальное представление DOM будет выглядеть точно так же, как ваш простой HTML-код.

Но, чаще всего, все совсем не так.

Когда визуальное представление DOM отличается от HTML-кода?

Например, если в вашем HTML-коде есть ошибки, и браузер исправил их за вас. Допустим, у вас есть элемент <table>, и вы пропустили обязательный элемент <tbody>. Браузер просто добавит этот элемент <tbody> за вас. Он будет присутствовать в DOM, поэтому вы сможете найти его с помощью JavaScript, или стилизовать с помощью CSS, даже если его нет в написанном вами HTML-коде.

И тем не менее, наиболее вероятная причина в различиях между написанным кодом и фактическим DOM в том, что JavaScript может динамически изменять DOM.

Представьте, что в вашем HTML-коде есть пустой элемент, примерно вот такой:

```
<div id="container"></div>
```

А чуть ниже в коде есть немножко JavaScript'a:

```
<script>
        var container = document.
getElementById("container");
  container.innerHTML = "Абра-кадабра!";
</script>
```

Даже если вы не знаете JavaScript, вы можете понять смысл этого фрагмента кода. На экране вы увидите Абра-кадабра!, потому что пустой div был заполнен новым содержимым.

Если вы откроете панель инструментов разработчика, чтобы посмотреть на визуальное представление DOM, то увидите:

```
<div id="container">Абра-кадабра!</
div>
```

Этим и отличается DOM от оригинального HTML-кода или от того, что вы увидите при просмотре исходного кода страницы.

Ajax и шаблонизация

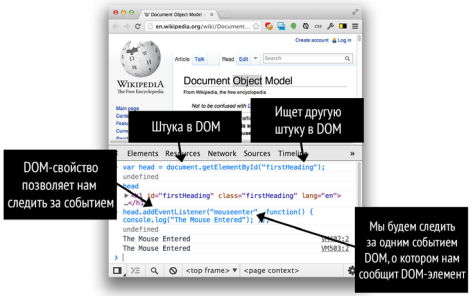
Даже не углубляясь в эту тему, можно быть уверенным, что вы понимаете, что DOM будет очень сильно отличаться от оригинального HTML-кода при использовании AJAX для получения контента извне и добавления его на страницу. Аналогичная ситуация происходит при загрузке данных и использовании их в клиентской шаблонизации.

Попробуйте открыть Gmail и посмотреть исходный код страницы. Вы обнаружите кучу скриптов и данных для первоначальной загрузки страницы. Совсем не похоже на интерфейс, который вы видите на экране.

JavaScript vs. DOM

По сути, JavaScript – это язык, понимаемый браузером, с помощью которого в нем делается множество операций. Все они происходят именно внутри DOM. Большая часть того, что вы бы назвали операциями JavaScript, на самом деле являются частью DOM API.

Например, мы можем написать код на JavaScript, который будет следить за срабатыванием события mouseenter на элементе. Этот элемент есть ни что иное, как узел DOM-дерева, и к этому узлу мы с помощью DOM-свойства подключаем обработчик. Срабатывание события происходит, потому что узел DOM-дерева распространяет его.



Ехал DOM через DOM

DOM – это то, что связывает весь код воедино, именно внутри DOM все и происходит. JavaScript – это просто синтаксис, язык. Он может быть использован и вне браузера, где DOM API нет и в помине. Например, обратите ваше внимание на платформу Node.js.

frontender.info

Быстрая инициализация проекта

Представьте себе фронтенд-разработчика типичной среднестатистической компании, занимающейся веб-разработкой. У него часто возникает ситуация, когда каждый следующий проект начинается с тех же самых действий, что и несколько предыдущих. Можно даже сказать, что с точки зрения разработчика 90-95% проектов в рамках одной такой компании начинаются приблизительно с одних и тех же действий. Поработав так некоторое время многие начинают задумываться над вопросом оптимизации рабочего процесса: как можно избежать клонирования одной и той же структуры файлов и каталогов вручную? Как избежать поиска и загрузки новых версий используемых библиотек и не писать новые конфигурационные файлы для любимых менеджеров задач, препроцессоров и других инструментов? Как можно автоматизировать этот процесс? Как сде-

лать инициализацию проектов более быстрой, гибкой и безболезненной? Давайте об этом поговорим.

Скаффолдинг

Наиболее часто употребляемый перевод для термина scaffolding – «инициализация проекта». На самом деле, обычно, смысл его немного глубже. Скаффолдинг – это процесс инициализации проекта, включающий развертывание файловой структуры проекта и генерацию некоторого (иногда опционального) количества кода, требуемого пользователю. А скаффолдер – это инструмент, реализующий этот самый процесс. Таким образом, в результате скаффолдинга вы получаете заданную структуру проекта, код, сгенерированный вашим инструментом (скаффолдером), и, возможно, некоторые опциональные инструменты, вроде автоматизации задач, пакетных менеджеров, тестировщиков и т.п. Но, случается, что скаффолдер нужен только для создания определенной файловой структуры проекта без включения дополнительных компонентов в него.

Инструментарий

Удобный инструмент – это самое важное в вопросе эффективного скаффолдинга. Я открыл для себя Yeoman, и постараюсь открыть его для вас. А также предложу к рассмотрению еще пару альтернативных скаффолдеров: Volo и grunt-init.

Yeoman

Yeoman – пожалуй, самый популярный в последнее время скаффолдер, поэтому о нем поговорим подробнее. На самом деле, сам Yeoman – это не просто скаффолдер, а целый набор инструментов, которые отлично дополняют друг друга в процессе скаффолдинга и гармонично сочетаются в процессе разработки. Yeoman «стоит на трех китах», задающих тон рабочему процессу современных разработчиков: скаффолдер Yo, менеджер пакетов Bower и менеджер задач Grunt. При установке Yo вам будут установлены также Bower и Grunt, если они не были установлены ранее.

Генераторы

Генератор Yeoman – это npm-пакет с инструкциями и шаблонами для Yo, которые описывают инициализацию проекта: какие директории создать, какие файлы и куда копировать, каким образом обрабатывать шаблоны и куда их разместить после обработки. В Yeoman шаблоны обрабатываются с использованием библиотеки Underscore.js.

На момент написания статьи в npm-репозитории уже доступно для установки более 300 генераторов. Устанавливаются они как обычные npm-пакеты с использованием команды

```
npm install <generator-name>
```

В конце статьи я приведу небольшой список полезных генераторов.

Использование Yeoman

Самый простой способ понять, как именно инициализировать проекты с Yeoman – это начать его использовать. Я рассмотрю процесс скаффолдинга с использованием генератора generator-

webapp и опишу использование каждого из трех компонентов Yeoman.

Yo

Установка Yo выполняется командой:

```
npm install -g yo
```

После установки необходимо обзавестись генератором приложения, которое вы хотите инициализировать. Просмотреть список доступных для установки генераторов можно по ссылке (http://yeoman.io/community-generators.html) или выполнив поиск пакета в npm по ключу «yeoman-generator»:

```
npm search yeoman-generator
```

Если среди существующих генераторов нет ничего подходящего, можно написать свой генератор. Но об этом позже.

Как уже было сказано, я опишу процесс скаффолдинга на примере генератора webapp. Для его глобальной установки выполните:

```
npm install -g generator-webapp
```

Теперь можно пробовать развернуть каркас будущего веб-приложения. Для запуска скаффолдинга в заранее созданной директории проекта выполните:

```
yo webapp
```

Генератор выполняет скаффолдинг по сценарию, который зависит от действий пользователя. Обычно на выбор предлагаются некоторые фундаментальные вещи, от которых будет существенно зависеть дальнейший процесс работы над проектом. В случае с webapp нам будет предложено выбрать, хотим ли мы в проекте использовать Bootstrap для Sass, RequireJS иModernizr.

По завершении скаффолдинга вы получаете следующую структуру:

```
app/
node_modules/
test/
.bowerrc
.editorconfig
.gitattributes
.gitignore
.jshintrc
bower.json
Gruntfile.js
package.json
```

В директории app находится dev-версия вашего проекта – там содержатся все файлы в исходном состоянии, не прошедшие конкатенацию, минификацию и прочие обработки. Дальнейшая разработка проекта обычно не выходит за пределы этой директории.

Некоторые генераторы могут иметь опции, которые указываются непосредственно в командной строке в момент инициализации. Обычно они описаны в документации, поэтому читайте внимательно. Webapp имеет три опции:

- skip-install – пропустить автоматический запуск установки пакетов Bower и npm после скаффолдинга.
- test-framework <framework> – по умолчанию установлен mocha. Может быть использован любой другой фреймворк для запуска тестов, вроде Jasmine.
- coffee – добавляет в проект поддержку CoffeeScript.

Grunt

Как вы уже поняли, конкатенация, минификация и другие преобразования файлов, не требующие человеческого вмешательства выполняются при помощи инструмента Grunt. Если вы до сих пор не знакомы с ним, предлагаю для ознакомления прочесть пост в блоге Артема Сапегина, вводный пост на сайте Smashing Magazine или краткое руководство ВСТАВИТЬ ССЫЛКИ)) на официальном сайте. Этого будет достаточно, чтобы начать его использовать. А пока вам необходимо знать только то, что запуск задач Grunt происходит по команде

```
grunt [<задача>]
```

Генератор в процессе скаффолдинга устанавливает пакеты, необходимые для выполнения задач Grunt. Большинство генераторов создают достаточно объемный Gruntfile.js, имея в распоряжении «багаж» полезных задач. Чтобы понять, какие задачи можно выполнять для проекта достаточно заглянуть в Gruntfile.js или выполнить

```
grunt -h
```

для директории с файлом Gruntfile.js. Webapp, например, имеет в качестве основных функциональных задач default, build, test, server.

Задача server выполняет очистку временной папки .tmp в корне проекта (создает ее, если директория .tmp отсутствует), обрабатывает имеющиеся Sass-файлы и кладет результат в.tmp/css, запускает слежку за изменением файлов в директории проекта, запускает сервер для обработки статичных файлов и открывает страницу проекта в браузере с возможностью запустить LiveReload, если у вас установлен плагин LiveReload для браузера. Эта задача обычно запущена во время работы над проектом.

Задача test собирает тестовую версию проекта, запускает тестировщик Mocha или Jasmine, в зависимости от того, что было выбрано при установке.

Задача default, которая также запускается по команде grunt, выполняет проверку JS-файлов из директории app/scripts на предмет ошибок с помощью JSHint и запускает задачу test.

Задача build выполняет компоновку релизной версии проекта с файлами, прошедшими конкатенацию, минификацию и переименование. Релизная версия проекта располагается в директории dist, которая будет создана в корне проекта при первом запуске этой задачи.

Стоит отметить, что если вы не используете в проекте RequireJS и хотите чтобы добавленные вами CSS-стили и JS-скрипты были доступны для автоматической конкатенации и минификации (если вы планируете использовать Grunt) их нужно записывать в index.html указывая тип файла и путь к нему в релизной версии проекта.

Важно понимать, что такое поведение – лишь частный случай генератора, т.к. в Webapp используется задача useminPrepare (плагин grunt-usemin), который берет на обработку файлы, описанные в HTML-файлах особым образом.

Например, добавляя в проект app/css/main.css, который в релизной версии бу-

дет расположен по пути dist/css/main.css, мы должны добавить в index.html следующие строки:

```
<!-- build:css(.tmp) css/main.css
-->
<link rel="stylesheet" href="css/
main.css">
<!-- endbuild -->
```

Подобная ситуация и с JavaScript:

```
<!-- build:js js/vendor.js -->
<script src="bower_components/
jquery/jquery.js"></script>
<!-- endbuild -->
```

Также можно осуществлять конкатенацию нескольких скриптов из dev-версии в один скрипт в релизной версии при помощи той же схемы:

```
<!-- build:js js/vendor.js -->
<script src="bower_components/
jquery/jquery.js"></script>
<script src="bower_components/
foundation/js/foundation/
foundation.js"></script>
<!-- endbuild -->
```

Так нужно делать только в случае со всеми CSS-файлами и JS-библиотеками, добавленными в проект вручную. Инструмент Bower, который будет рассмотрен следующим, может автоматизировать процесс добавления библиотек в проект. Если ваш Gruntfile.js содержит задачу bower-install, то можно воспользоваться следующим алгоритмом:

1. убедиться, что в HTML-файле имеется следующие комментарии:

<!-- bower:js -->
<!-- endbower -->

2. установить требуемую библиотеку, выполнив:

```
bower install jquery --save
```

3. вызвать задачу bower-install:

```
grunt bower-install
```

Bower установит необходимую библиотеку, попытается записать файл библиотеки между комментариями из пункта 1 и сообщит вам, удалось ли ему это.

Bower

Третьим немаловажным компонентом Yeoman является Bower – менеджер пакетов для веб-приложений. Этот инструмент упрощает поиск, установку и обновление библиотек для вашего проекта. В процессе скаффолдинга Bower позволяет генератору загружать необходимые версии библиотек, зависимости которых определены в файле bower.json. Часто bower.json представляет собой шаблон, обрабатываемый генератором во время инициализации проекта. В таком случае он способствует опциональной загрузке библиотек в процессе скаффолдинга. Завершив скаффолдинг проекта, вы сможете без лишних движений доставить требуемые библиотеки при помощи Bower.

Все, что для этого нужно: команды bower search [<ключевое_слово>] иbower install <имя_пакета>. Установка пакетов идет в директорию, путь к которой прописан в конфигурационном файле .bowerrc. С более тонкой настройкой Bower вы сможете ознакомиться тут.

Bower работает с протоколами Git и HTTP(S) и способен устанавливать пакеты как из репо-

зитория, так и из простого zip-архива.

Некоторые могут задуматься над вопросом: «Зачем нужен Bower, если есть npm?». Наиболее подходящее объяснение описано на официальном сайте и состоит в том, что основное его предназначение – довольно простое управление отдельными пакетами с точки зрения фронтенд-разработчика за счет плоского дерева зависимостей и удобный API, который может быть использован для реализации более опциональных рабочих процессов. К тому же, Bower имеет отдельную от npm-плагинов директорию в проекте, что позволяет разработчику версионировать проект вместе с установленными пакетами и игнорировать локально установленные плагины для Node.js, которые зачастую занимают внушительное дисковое пространство.

Свой собственный генератор

Как уже было сказано ранее, уже существует более 300 генераторов Yeoman, доступных из npm-репозитория. Но может случиться так, что ни один из них не будет удовлетворять вашим потребностям. В таком случае вы всегда можете написать свой или внести изменения в существующий, если его лицензия позволяет это. Подробнее о том, как сделать свой генератор читайте здесь.

Полезные генераторы Yeoman

• Web App – генератор для инициализации веб-приложения на основе HTML5Boilerplate и опциональными компонентами. Рассмотрен в статье.

• AngularJS – генератор для инициализации веб-приложения на основе AngularJS. Также опционально устанавливает Twitter Bootstrap и дополнительные модули AngularJS, такие как angular-resource.

• Backbone – генератор для инициализации веб-приложения на основе Backbone. Также вы получаете доступ к нескольким суб-генераторам, которые могуты быть использованы для простого создания моделей (models), представлений (views), коллекций (collections) и т.д.

• Ember – генератор для инициализации веб-приложения на основе Ember.js.

• Mobile App – генератор для инициализации веб-приложений базирующихся на подходе mobile-first. Предоставляет возможность выбрать фреймворк из следующих: Twitter Bootstrap 3, TopCoat, Zurb Foundation и Pure.

• Jasmine – генератор, который инициализирует проект, использующий Jasmine в качестве фреймворка для тестирования проекта.

• H5BP – генератор для инициализации проекта на основе HTML5Boilerplate.

Bootstrap LESS – генератор для инициализации проекта с использованием LESS-версии фреймворка Twitter Bootstrap.

Volo

Volo будет рассмотрен менее подробно, так как он обладает меньшими возможностями, в отличии от Yeoman. Volo, как нам сообщают на сайте разработчика:

Volo – это инструмент, который дает возможность быстро создавать проекты, добавлять библиотеки и автоматизировать простые задачи используя Node.js и JavaScript.

Посмотрим, что он может:

Создание проекта

```
volo create [<имя_проекта>]
[<репозиторий> | <ссылка_на_шаблон>]
```

Примеры:

```
volo create foo h5bp/html5-boilerplate/4.0.0
volo create foo http://mozilla.github.com/mortar/builds/app-stub.zip
```

Добавление библиотек в проект:

```
volo add jquery
volo add backbone
volo add requirejs/~2
volo add amdjs/backbone
```

Для описания автоматизации процесса сборки проекта Volo использует свой конфигурационный файл – volofile. При этом, естественно, необходимо предварительно установить node-плагины, используемые для автоматизации задач таких, как jshint или uglify-js. Подробнее об автоматизации с Volo можно узнать здесь.

Скаффолдинг с Grunt-init

Grunt-init – инструмент от разработчиков Grunt, позволяющий разворачивать проекты, используя шаблоны. То, что в Yeoman называлось генераторами, у grunt-init называется шаблонами. Для установки плагина вам необходимо выполнить:

```
npm install -g grunt-init
```

Использование grunt-init концептуально ничем не отличается от других скаффолдеров. Вам аналогично необходимо загрузить шаблон и запустить скаффолдер. Однако, grunt-init имеет специфику в установке шаблонов. Процесс установки шаблона заключается в клонировании git-репозитория с исходным кодом шаблона в директорию ~/.grunt-init/. Выполнив клонирование шаблона, можно запускать скаффолдер командой:

```
grunt-init <TEMPLATE_NAME>
```

Небольшой список доступных шаблонов можно найти на официальном сайте, а также врепозитории Артема Сапегина.

Кроме того, вы можете написать свой шаблон. Подробные инструкции можно найти на том же официальном сайте и в блоге Артема Сапегина.

skype matmuchrapna доброму Владимиру Старкову.

Автор:
Денис Демченко, frontender.info

Отслеживание ошибок с помощью Google Analytics

Google Analytics всегда был больше, чем просто обыденным счетчиком посещений и инструментом статистики. Вы можете замерить эффективность рекламной кампании, понять, насколько глубоко пользователи продвигаются с точки входа на сайт до покупки, узнать, какими браузерами пользуются и на каких языках говорят ваши пользователи.

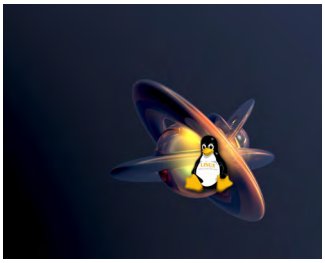
Но все это приятные мелочи созданы для маркетологов, а не для нас — разработчиков. А как гики могут использовать Google Analytics? Например, для ослеживания ошибок с помощью пользовательских событий.

```
// Отслеживание основных
JavaScript-ошибок
window.addEventListener('error',
function(e) {
    _gaq.push([
        '_trackEvent',
        'JavaScript Error',
        e.message,
        e.filename + ':  ' +
e.lineno,
        true
    ]);
});

// Отслеживание AJAX-ошибок
$(document).ajaxError(function(e,
request, settings) {
    _gaq.push([
        '_trackEvent',
        'Ajax error',
        settings.url,
        e.result,
        true
    ]);
});
```

Теперь, когда вы познакомились с Google Analytics поглубже, вы сможете наблюдать техническую статистику вашего сайта наряду с обычной. Конечно, вам стоит убедить маркетологов, что это фичи, а не баги, но это уже другая история. Внедрите отслеживание ошибок с помощью Google Analytics прямо сейчас, а меня всегда сможете поблагодарить позже.

frontender.info



Раскрытие тайны this в JavaScript

В этой статье мы попытаемся пролить свет на использование this в JavaScript и привнести немного ясности и понимание о том, как this работает. Это знание не относится к черной магии и не является запретным, наоборот – понимание механизма работы this чрезвычайно полезно для всех JavaScript-разработчиков.

Пока вы до конца не осознаете this, вы скорее всего будете чувствовать себя неуверенно.

Как это работает

Если метод был вызван из объекта, тогда this в контексте метода является ссылкой на родительский объект.

```
var parent = {
    method: function () {
        console.log(this);
    }
};
```

parent.method(); // ссылается на родительский объект

Заметьте, что этот паттерн очень уязвим, так как при вызове метода по ссылке this не будет больше ссылаться на родительский объект parent. Вместо этого, this теперь будет ссылаться на глобальный объект Window. Это обстоятельство приводит в замешательство большинство разработчиков.

```
var parentless = parent.method;
```

parentless(); // ссылается на Window

В последней строчке примера вы должны обратить внимание на то, как именно вызывается функция - возможны два варианта. Либо функция вызывается как свойство объекта, либо она сама по себе. И если она вызывается как свойство, то this будет ссылаться на данное свойство, в противном случае – на глобальный объект. В данном примере это - Window, но в строгом режиме this будет возвращать undefined.

В случае с конструктором this ссылается на созданный экземпляр при условии использования ключевого слова new.

```
function ThisClownCar () {
  console.log(this);
}
```

new ThisClownCar(); // ссылается на ThisClownCar {}

Стоит заметить, что поведение this в конструкторе весьма неочевидное: если вы случайно забудете дописать new, то this опять будет ссылаться на глобальный объект, как мы уже наблюдали в примере с parentless.

```
ThisClownCar();
// ссылается на Window
```

Ручное управление над this

Методы .call, .apply, и .bind используются для управляемого вызова функций, помогая определять оба передаваемых

значения – this, он же контекст функции, и arguments.

Метод Function.prototype.call может принимать любое количество аргументов. Первый из них будет использоваться как this, а оставшиеся будут переданы вызываемой функции как аргументы.

```
Array.prototype.slice.call([1, 2,
3], 1, 2)
  // ссылается на [2]
```

Метод Function.prototype.apply очень похож на предыдущий, только аргументы в нем передаются одним массивом, вместо неопределенного количеств аргументов, как в методе call.

```
String.prototype.split.
apply('13.12.02', ['.'])
  // ссылается на ['13', '12', '02']
```

Метод Function.prototype.bind возвращает специальную функцию, которая, в свою очередь, может быть использована для вызова еще одной функции. От ее имени и будет вызван bind. Функция всегда будет использовать переданный ей this, и в тоже время, ей можно передать несколько аргументов, в качестве всегда используемых в возвращаемой функции. Это может быть очень удобно для карринрования оригинальной функции.

```
var arr = [1, 2];
var add = Array.prototype.push.
bind(arr, 3);

  // эквивалентно arr.push(3)
add();
  // эквивалентно arr.push(3, 4)
add(4);

console.log(arr);
  // ссылается на [1, 2, 3, 3, 4]
```

Область видимости и this

В следующем случае this будет неизменным в разных областях видимости. Это исключение в правиле и оно часто приводит к ошибкам среди начинающих разработчиков.

```
function scoping () {
  console.log(this);

  return function () {
    console.log(this);
  };
}

scoping()();
  // ссылается на Window
```

Распространенный способ обойти создавшуюся проблему – это создать локальную переменную, содержащую ссылку на this в текущей функции, а значит, и в ее области видимости. Она останется доступной во вложенной функции. Вложенная функция, в свою очередь, будет иметь свою собственную переменную this, что означает невозможность использования this родительской функции напрямую.

```
function retaining () {
  var self = this;

  return function () {
    console.log(self);
  };
```

```
}

retaining()();
  // ссылается на Window
```

Если вы все-таки по неизвестным причинам захотите использовать родительский this в качестве контекста для вложенной функции, то можно использовать метод функции .bind. Эта функция может быть использована в том числе для того, чтобы пробросить родительский контекст во вложенную функцию.

```
function bound () {
  return function () {
    console.log(this);
  }.bind(this);
}

bound()();
  // ссылается Window
```

frontender.info

First Line Software

First Line Software пополнила свой продуктовый портфель решениями «1С-Битрикс»

Компания First Line Software, глобальный производитель заказных информационных систем для оптимизации бизнеса, пополнила продуктовый портфель решениями компании «1С-Битрикс». В качестве авторизованного партнера First Line будет разрабатывать для своих клиентов на базе продуктов вендора специализированные программные решения: внутрикорпоративные информационные ресурсы и e-commerce-системы.

«Значительная часть клиентов First Line Software сосредоточена в Европе и США, но пул российских заказчиков постоянно увеличивается. Наша компания работает с широкой линейкой производителей софта для создания веб-решений и интранет-порталов от зарубежных производителей. Со стороны российских клиентов мы увидели большой спрос на решения "1С-Битрикс", потому было принято решение пополнить портфель First Line Software продуктами игрока российского рынка CMS-систем. Компания "1С-Битрикс" предлагает программное решение с внушительным набором функций, позволяющее выстраивать интернет- и интранет-системы любого уровня сложности», — отметил генеральный директор First Line Software Александр Поздняков.

По информации First Line Software, в настоящее время проекты с использованием продуктов «1С-Битрикс» выполняются для двух российских компаний.

cnews.ru

ED.ua

Школьный электронный дневник

НАПИСАТЬ НАМ: info@ed.ua
Крым, г.Феодосия, ул. Крымская, 21-А Тел.: +38 (06562) 7-44-79

- ► Электронная база данных
- ► Персональный сайт школы
- ► Новости, события, праздники
- ► Домашнее задание, оценки
- ► Посещаемость и замечания
- ► Мобильная версия сайта
- ► Электронная очередь детских садов
- ► Отчеты, статистика, бланки
- ► Рейтинги школы
- ► Акции, скидки, праздники

с ED.ua сбудется моя мечта!

ПОТОМУ ЧТО НА САЙТЕ ED.UA ЕСТЬ ПОЛНОЕ ДОМАШНЕЕ ЗАДАНИЕ!

Функции постоянно добавляются и модернизируются! Новым пользователям предоставляется бесплатный тестовый период!

Файл подкачки: swap-файл и swap-раздел в Linux

Все мы знаем, что swap-файлы в Linux делаются просто и легко - настолько просто, что иногда забываем, как это делается. Прежде, чем что-то создавать, хорошо бы узнать, сколько swap-пространства у нас уже имеется в системе - для этого следует дать команду в консоли от рута:

```
# sudo swapon -s
```

Результат будет в виде:

```
Filename Type Size Used Priority
/dev/hda1 partition 289128 0 -1
```

Описание вывода команды:

• **Filename** описывает имеющиеся у вас своп-пространства и где они находятся.

• **Type** указывает тип пространства: partition (раздел) или file (файл).

• **Size** сообщает общий размер Swap-пространств.

•**Used** говорит о том, сколько сейчас свопа задействовано.

•**Priority** указан приоритет, т.е. какие пространства системе использовать вначале.

Тот же самый результат мы получим по команде cat /proc/swaps

Создание swap-файла в Linux

1. Открываем консоль\терминал и получаем полномочия root или используем sudo:

```
$ su
```

2. Думаем*, какой размер swap-файла нам нужен в мегабайтах. Подумав, даем команду:

```
sudo dd if=/dev/zero of=/swapfile
bs=1M count=500
```

или

```
# dd if=/dev/zero of=/swapfile
bs=1M count=500
```

В команде dd для задания размеров можно использовать суффиксы K, M, G для килобайт, мегабайт и гигабайт соответственно. В данном примере это 500 Мегабайт файла подкачки.

* Многие задаются фундаментальными вопросами бытия вроде "каков рекомендуемый размер swap в linux"? Можно не думать, а просто создать SWAP-файл по размеру оперативной памяти, периодически посматривая на ее, подкачки, использование с помощью команды top. При необходимости добавить\уменьшить своппинг системе. Можно использовать несколько файлов подкачки.

3. Поясняем системе, что созданный пустой файл это все-таки файл подкачки для Linux:

```
sudo mkswap /swapfile
```

или

```
# mkswap /swapfile
```

4. Подключаем созданный swap-файл:

```
sudo swapon /swapfile
```

или

```
# swapon /swapfile
```

При этом в выводе команды top или команды free должно появиться упоминание, что свопинга в системе поприбавилось. Чтобы отключить файл подкачки, пишем

```
sudo swapoff /swapfile
```

или

```
# swapoff /swapfile
```

Чтобы не подключать swap-файл или swap-раздел каждый раз, полезно занести запись в /etc/fstab следующего содержания:

```
/swapfile none swap sw 0 0
```

На всякий случай отметим, что каждый раз создавать swap-файл не нужно: просто подключаете и отключаете его с помощью swapon/swapoff. Работа со swap-разделами в Linux происходит аналогичным образом.

Приоритет SWAP-файлов

Создавать и использовать swap-файлов в Linux можно любое количество. При этом можно указать приоритет подключаемого swap-файла или раздела (хотя ядро умеет самостоятельно распределять по разделам/файлам подкачки).

Например, высший приоритет для файла подкачки задается так:

```
swapon -p 1 /opt/swapfile
```

Приоритет является целым числом от 0 до 32767.

Очистка swap-пространства после ресурсоемких приложений

Командой swapoff -a, запущенной от имени root, можно отключить использование всех разделов и файлов подкачки. После ввода команды содержимое свопа за несколько минут загружается обратно в оперативную память, а сам раздел подкачки отключается.

После загрузки содержимого свопа в оперативную память включаем своп обратно командой swapon -a.

2. Системные настройки использования своппинга – Linux

За интенсивность обращения системы к swap-файлам и swap-разделам отвечает параметр swappiness, равный по умолчанию 60. Значение параметра может быть в пределах от 0 - наименьшее использование подкачки, до 100 - подкачка используется часто.

Насчет оптимального значения параметра swappiness есть много разных мнений. Так, например, один из ведущих разработчиков ядра Эндрю Мортон считает, что для десктопа лучше ставить большое значение, чтобы всякое bloatware скинуть в своп и использовать оперативную память для чего-то нужного.

Чрезмерное значение здесь приведет к интенсивному использованию swap-файла, что нежелательно. Слишком маленькое значение может привести к тому, что при заполнении памяти будет принудительно запущен OOMkiller (процесс, запускающийся при исчерпании памяти и убивающий наиболее ресурсоемкие задачи).

Временно (до перезагрузки системы) изменить этот параметр можно с

помощью команды:

```
echo 50 > /proc/sys/vm/
swappiness
```

Чтобы изменить значение по умолчанию, необходимо изменить параметр vm.swappiness:

```
vm.swappiness=50
```

в файле /etc/sysctl.conf

Следует, впрочем, отметить, что со vm.swappiness сильно перегибать палку не стоит. При больших значениях система потеряет в отзывчивости (будет вытеснять память, с которой работают приложения, в своп, хотя оперативной памяти еще много). При малых значениях система работает отзывчивей, но когда оперативная память заканчивается, система начинает активно свопиться и притормаживать.

Также можно попробовать увеличить\уменьшить объем потребляемой системой памяти за счет изменения размеров дискового кеша. Уровень выделяемой под кеш памяти хранится в

/proc/sys/vm/vfs_cache_pressure

Значение по умолчанию: 100. Чтобы использовать меньше памяти под дисковые кеши (что вообще говоря не есть хорошая идея), ставим значение 50. Если, наоборот, хочется больше отзывчивости системы, увеличиваем размер кеша не скупясь:

```
echo 1000 > /proc/sys/vm/vfs_
cache_pressure
```

и далее продолжаем играться с параметрами вплоть до полного удовлетворения. Для того, чтобы настройки стали постоянными, заносим параметр

vm.vfs_cache_pressure = 1000

в файл /etc/sysctl.conf и со следующей загрузки дисковые кеши будут с радостью пользоваться вашей оперативной памятью.

mydebianblog.blogspot.com

Защита от ddos атак

Denial of Service – буквально «отказ в обслуживании». Сам термин – это сокращение от английского Distributed Denial Of Service Attack, то есть дословно «распределенная атака типа „отказ в обслуживании“, выполнение одновременно с большого числа компьютеров или кратко называется DDOS атакой. Если говорить о технической стороне вопроса, то ее главная цель – парализовать работу веб-узла, путем выведения вычислительной системы из строя.

Еще несколько лет назад данный термин был практически неизвестен широкому кругу пользователей, а использовался в основном специалистами. Сегодня мы постоянно слышим в новостях о том, что тот или иной сайт подвергся действиям, направленным на отказ в обслуживании. При этом цели, которые преследуются DDOS атаками, могут быть самыми разными: обрушить конкурента на рынке, выставить свои требования владельцам, привлечь внимание и пр, взломать сервер для получения некой информации (при внештатной ситуации ПО может выдать какую-либо критическую информацию, например, версию, часть программного кода) и пр.

Список команд, полезных для определения ддоса (DDos – отказ в обслуживании) или флуда, а так же для отражения нераспределенных атак.

```
netstat -ntu | awk ' $5 ~ /^[0-9]/ {print $5}' | cut -d: -f1 | sort | uniq -c | sort -n
netstat -anp |grep 'tcp\|udp' | awk '{print $5}' | cut -d: -f1 | sort | uniq -c | sort -n
netstat -ntu | awk ' $5 ~ /^(::ffff:|[0-9|])/ { gsub („::ffff:“,»«,$5); print $5}' | cut -d: -f1 | sort | uniq -c | sort -nr
netstat -ant | awk '/^tcp/ { split ($5,IP,»:"); CNT[IP[1]]++ }; END { OFS="\t"; for (ip in CNT) { print CNT[ip],ip };}'
  # Без сортировки
netstat -ant | awk '/^tcp/ { split ($5,IP,":"); CNT[IP[1]]++ }; END { OFS="\t"; for (ip in CNT) { print CNT[ip],ip };}'

  # С разбивкой по портам
netstat -nt | awk -F":" '{print $2}' | sort | uniq -c | sort -n
```

Число коннектов

Число коннектов на 80 порт:

```
netstat -na | grep :80 | wc -l
# То же, в статусе SYN
netstat -na | grep :80 | grep
syn
```

iptables – для фильтрации нежелательного трафика

Вот так добавляем в фильтр IP

```
iptables -A INPUT -p tcp --dport
80 -s 95.153.67.144 -j DROP
# если надо блочить IP полностью
iptables -A INPUT -s
95.153.67.144 -j DROP
```

Вот так смотрим список уже добавленных

```
iptables -L -n
iptables -L -n --line-numbers
```

А вот так убираем из фильтра IP

```
iptables -D INPUT -s 1.2.3.4 -j
DROP
```

или удалить по номеру правила

```
iptables -L -n --line-numbers
iptables -D INPUT 4
```

А вот так очищаем вообще весь список

```
iptables -F
```

world-blog.ru

В Ubuntu 14.04 устранена опасная уязвимость

В Ubuntu 14.04 Trusty обнаружена опасная уязвимость, позволяющая получить доступ к сеансу пользователя без ввода пароля. Данная проблема наблюдается только при использовании оболочки Unity.

Для доступа к сеансу пользователя достаточно:

• При заблокированном экране несколько раз кликнуть по индикаторам (например, заряд аккумулятора);

• Нажать сочетание клавиш ALT + F2;

• Выполнить произвольную команду от имени пользователя (например, перезапустить Compiz).

На данный момент уязвимость устранена в пакете unity-7.2.0+14.04.20140423-0ubuntu1.1. Пользователям рекомендуется пройти процедуру обновления.

Демонстрация проблемы:
https://www.youtube.com/watch?v=d4UUB0sI5Fc

Подробности:
https://bugs.launchpad.net/ubuntu/+source/unity/+bug/1313885

linux.org.ru

Вышел первый релиз анонимной ОС, которой пользуется Сноуден

Tails
theamnesicincognitolivesystem

Операционная система Tails 1.0, позволяющая полностью скрывать свое присутствие в Сети, стала доступна для пользователей. Система полностью размещается в ОЗУ и не копирует файлы на жесткий диск, а для всех типов коммуникаций использует анонимную сеть Tor, которая не позволяет отследить местоположение пользователя. Для запуска Tails можно использовать любой ПК, не оставляя следов.

Проект The Amnesic Incognito Live System или просто Tails объявил о выходе первого релиза анонимной операционной системы с одноименным названием – Tails 1.0.

На разработку Tails ушло более 5 лет. Первая публичная версия ОС была представлена в июне 2009 г. и называлась Amnesia. С тех пор было выпущено 36 стабильных версий, включая Tails 1.0.

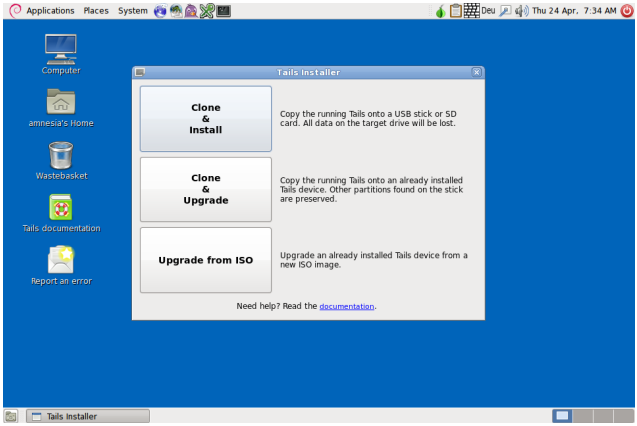
Tails – это полноценная ОС, базирующаяся на Debian GNU/Linux. У системы открытый исходный код, созданный сообществом анонимных разработчиков.

ОС предлагается со всем необходимым встроенным ПО, включая веб-браузер, клиент для обмена мгновенными сообщениями, пакет офисных приложений, редакторы изображений и звука и др.

Кроме того, в состав Tails входит набор средств шифрования трафика для всех видов коммуникаций: LUKS – для шифрования файлов, HTTPS Everywhere – веб-трафика, OpenPGP – писем, а OTR – мгновенных сообщений. Также в комплект входит шредер файлов Nautilus Wipe.

Основная задача Tails – скрывать присутствие пользователя сети. Помимо этого система позволяет преодолевать цензуру в интернете, то есть помогает открывать заблокированные по этой причине сайты.

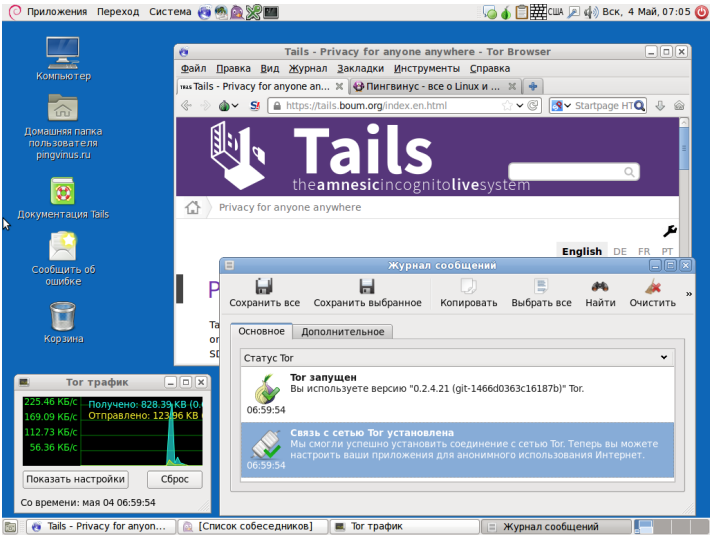
Рабочий стол Tails 1.0



Для выполнения этих задач Tails использует анонимную сеть Tor. Tor – это сеть пользовательских компьютеров, которые пропускают через себя трафик, стирая его след. В результате, становится невозможно вычислить, кто именно в сети просматривает конкретный сайт.

Изначально сеть Tor была военным проектом США, но впоследствии разработчики открыли ее код и сделали доступной всем желающим. Тогда Tor стали использовать люди, которые по разным причинам нуждаются в интернет-анонимности. Социальные работники пользуются Tor при общении с учётом тонкой специфики своей работы: в чатах и веб-форумах для жертв насилия, конфликтов, беженцев. Корпорации используют Tor как безопасный способ проведения анализа на конкурентном рынке, а также в качестве дополнения к виртуальным частным сетям. Журналисты с помощью Tor обеспечивают безопасное общение со своими информаторами. Например, Эдвард Сноуден с помощью Tor передал информацию о PRISM газетам The Washington Post и The Guardian, еженедельник The New Yorker запустил специализированный сервис Strongbox для приёма компромата, а итальянские интернет-активисты создали сайт MafiaLeaks по сбору информации о деятельности мафии. Правоохранительные органы используют Tor для скрытого посещения веб-сайтов, чтобы не оставлять при этом IP-адреса своих учреждений в логах соответствующих веб-серверов, а также для обеспечения безопасности сотрудников при проведении спецопераций.

Если какое-либо приложение в Tails попытается подключиться к интернету напрямую, минуя сеть Tor, соединение будет заблокировано.



Запустить Tails можно с DVD, USB-флешки или SD-карты независимо от того, какая ОС установлена на компьютере. При этом система не копирует файлы на жесткий диск, а полностью размещается в ОЗУ. Когда пользователь заканчивает работу в Tails или выключает компьютер, все данные ОС из ОЗУ исчезают. Таким образом, пользователь не оставляет следов,

а ПК затем можно продолжить использовать с той ОС, которая на нем стояла.

Этой операционной системой пользуется Эдвард Сноуден (Edward Snowden) для связи с журналистами. Сноуден – бывший сотрудник АНБ и ЦРУ, который раскрыл множество сведений о работе американской разведки и нашел убежище в России.

Согласно официальному сайту проекта, работа над Tails будет продолжена. Ближайшее обновление – Tails 1.1 – планируется выпустить в июне.

Также в планах разработчиков уже прописаны Tails 2.0 и Tails 3.0. Во второй версии они намерены сконцентрироваться на функциях обслуживания ОС, скорости выпуска обновлений; в третьей – на повышении уровня безопасности, включая внедрение технологии изолирования приложений. Сроки выпуска этих версий не сообщаются.

Сомнения вокруг этой анонимной операционной системы вызывает только, как это ни странно, анонимность ее создателей. До сих пор не известно, а этим вопросом задаются сейчас все журналисты мировых IT-изданий, "кто на самом деле стоит за Tails?" Судя по документации на сайте проекта самые большие пожертвования для существования Tails делает проект Tor.

А он был создан в исследовательской лаборатории Военно-морских сил США. И нет никакой уверенности, что и Tor, и Tails – это не какой-то очередной эксперимент американского правительства, предназначенный слежки за теневыми играми корпораций и правительств разных стран, а также для перехвата ценной анонимной информации.

cnews.ru

Через «дыру» в Adobe Flash Player шпионили за пользователями

Adobe устранила критическую уязвимость во Flash Player, затрагивающую пользователей платформ Windows, Mac и Linux. Эксперты полагают, что уязвимость была использована сирийскими диссидентами, выступающими против правительства, в шпионских целях.

Adobe выпустила обновление для Adobe Flash Player, устраняющее уязвимость CVE-2014-0515, с помощью которой злоумышленники могли получить контроль над компьютером жертвы.

Уязвимости содержатся во Flash Player 13.0.0.182 и более ранних версий для

Windows, Flash Player 13.0.0.201 и более ранних версий для OS X и Flash Player 11.2.202.350 и более ранних версий для Linux.

Обновленные версии с устраненными уязвимостями: Flash Player 13.0.0.206, Flash Player 13.0.0.206 и Flash Player 11.2.202.356 соответственно.

Пользователям рекомендуется установить новые версии как можно скорее. Пользователям веб-браузеров Google Chrome, Internet Explorer 10 и Internet Explorer 11 вручную обновляться не нужно – браузер сам автоматически загрузит новую версию.

Уязвимость была обнаружена в середине апреля 2014 г. командой «Лаборатории Касперского». Аналитики отправили в Adobe два эксплойта, использующие эту уязвимость, после чего та подтвердила наличие бреши и присвоила ей номер CVE-2014-0515. Брешь содержится в компоненте Flash Player под названием Pixel Bender, который отвечает за обработку видео и изображений.

Оба найденных эксплойта распространялись с сайта, расположенного по адресу jpic.gov.sy. Этот сайт был запущен в 2011 г. Министерством юстиции Сирии и представляет собой онлайн-форму для отправки жалоб граждан на нарушение правопорядка. Мы полагаем, что целью атаки были сирийские диссиденты, выступающие против правительства. Сайт уже был взломан в сентябре 2013 г., о чем сообщил предположительно сам хакер в своем твиттере, сообщил эксперт «Лаборатории Касперского» Вячеслав Закоржевский.

Один из эксплойтов пытался подключиться к решению видеоконференцсвязи Cisco MeetingPlace Express на компьютере. По мнению аналитиков, вероятно, злоумышленники намеревались шпионить за своими жертвами. Однако точно установить их намерения оказалось невозможно, так как сразу после детектирования первых атак файлы, которые должны были устанавливаться в целевой системе, были удалены с серверов.

На 24 апреля 2014 г. продукты ЛК на найденные эксплойты среагировали более 30 раз, при этом срабатывания были зафиксированы у 7 уникальных пользователей – и все они из Сирии, добавил Закоржевский.

«Вероятно, атака была тщательно спланированной, и за ней стоят профессионалы достаточно высокого уровня. Об этом свидетельствует использование профессионально написанных 0-day эксплойтов, которыми был заражен единственный ресурс», – заключил эксперт.

cnews.ru

Хотите стать частью команды UserAndLINUX?

Если вы давно используете Linux, либо только начали интересоваться продуктами OpenSoure, либо же просто интересуетесь компьютерными и техническими новинками, то мы с радостью примем вас в нашу дружную команду!

Каждый из нас именно так и попал в команду журнала UserAndLINUX – мы просто любим Linux и считаем, что обязаны нести эту любовь в массы. И мы знаем, как отплатить нашей любимой операционной системе – создать сообщество людей с общими интересами, поддерживать друг друга и помогать новичкам в этом интересном деле.

И не имеет значения, опытный ли вы программист, или одаренный школьник, или дизайнер, инженер, секретарь... Ваши идеи в совокупности с нашими могут помочь другим людям узнать, что такое Linux и с чем его едят!

Чем же вы можете нам помочь?

У вас есть творческие способности, креативное чувство стиля? Обладаете вкусом? Создавайте красивые темы и фоны (людям нравятся красивые картинки!) с командой наших дизайнеров!

Владеете иностранным языком? Переводите статьи с англоязычных ресурсов, чтобы наши читателя всегда имели удовольствие читать свежие интервью и новости со всего мира!

Любые навыки, которыми вы владеете, могут помочь команде UserAndLINUX – присоединяйтесь!
Предлагайте свои идеи. Учитесь вместе с нами. Развивайте новые умения, способности.

Спрашивайте. Задавайте вопросы, не стесняйтесь! Нам всегда нужны люди. Не думайте, что вы не сможете помочь, потому что вы не умеете программировать, администрировать или только начинаете работать в Linux как в системе, которая установлена у вас на компьютере.

Существует миллион способов внести свой вклад.
Присоединяйтесь к работе над журналом UserAndLINUX и приложением «Больше чем USER»!

Оставляйте свои вопросы, координаты и описание того, чем бы вы хотели помочь:
magazine@ualinux.com
на форуме ***http://ualinux.com/forum/userandlinux***
в нашей группе Вконтакте - ***https://vk.com/userandlinux***
или группе на Facebook - ***https://www.facebook.com/groups/userandlinux/***

С уважением,
коллектив журнала UserAndLINUX

User And LINUX

More Than USER
Приложение к журналу
User And LINUX

Адрес журнала в Интернете:
http://ualinux.com/journal

Обсуждение журнала
на форуме:
http://ualinux.com/forum

По вопросам
приобретения журнала:
http://ualinux.com/pay

Адрес редакции:
Украина, 03040,
г.Киев, а/я 56

Email:
magazine@ualinux.com

Тип издания:
электронный

Регулярность:
ежемесячный

Дата выпуска:
19.05.2014

Тираж: *более 25 000 копий

* указано сумарное количество загрузок прошлого выпуска журнала с первичных источников, а также загрузок с других известных ftp, http и torrent серверов

Свидетельство о гос. регистрации КВ №18270-7070Р октябрь 2011 г.
ISSN: 2223-6988


More Than USER
Приложение к журналу
User And LINUX