user And LINUX

№ 2 ФЕВРАЛЬ 2011

Больше чем user

Модем для FreshTel под Linux

Видеонаблюдение в Linux

Четыре инструмента архивации для администратора сервера Linux

Принтеры Canon из серии LBP в сети. Заставляем работать на Ubuntu

Программирование на Bash - часть II

Настройка GRUB2

От редакции

Мы рады приветствовать Вас, наши дорогие читатели, на страницах нового выпуска приложения к журналу UserAndLINUX - Больше чем User!

Благодаря Вам и Вашим советам, мы находимся в постоянном поиске того, что было бы интересно знать начинающим системным администраторам, программистам и тем, кто стремится узнать что-то большее из мира СПО.

Для некоторых материал публикуемый на страницах приложения послужит напоминанием, другие найдут наоборот, что-то новое и интересное для себя. Результатом нашей общей работы стало появление новой рубрики - Servers, в которой мы на протяжении всех выпусков будем рассматривать различные варианты настройки серверов. И пусть эта рубрика послужит отправной точкой в более детальном изучении настройки серверов и серверных приложений.

Но кроме этого в приложении есть еще множество других не менее интересных рубрик. Мы желаем удачи Вам и надеемся что наша информация будет Вам интересна.

Ирина Сикач
Главный редактор журнала
«Больше чем User»

НАША КОМАНДА:

Главный редактор:
Ирина Сикач

Редакторы:
Алексей Невенчаный
Владимир Попов
Надежда Козаченко
Дмитрий Бутолин

Дизайн и верстка:
Руслан Гордиевич
Игорь Шарай
Нина Копытина
Александр Никитин

Подбор материала:
Владимир Попов
Ирина Сикач
Алексей Невенчаный

Адрес журнала в интернете:
http://ualinux.com/index.php/journal

Обсуждение журнала в форуме:
http://ualinux.com/index.php/forum

По вопросам
преобретения журнала:
http://ualinux.com/index.php/get-journal

Адрес редакции:
Украина, 03040, г.Киев,а/я 56
Email: journal@ualinux.com

Тип издания:
электронный/печатный
Регулярность: ежемесячный
Дата выпуска: 14.02.2011
Тираж: *более10 000 копий.

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

UA Linux
open source ■ open future

Содержание

Console

Hi-tech

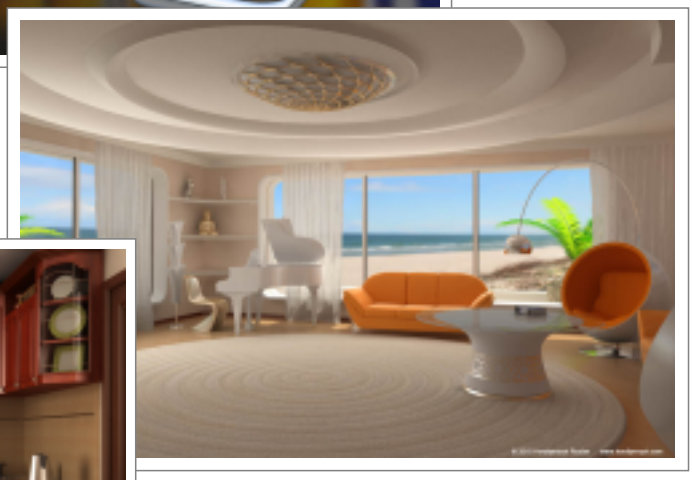
Programming

Servers

Other

Ruslan Hordiyevych

дизайн-студия Руслана Гордиевича



дизайн

3D-моделирование

визуализация

www.hordiyevych.com

10 полезных возможностей nmap

Nmap (Network Mapper) — свободная утилита с открытым исходным кодом, предназначенная для сканирования сетей и аудита сетевой безопасности. Nmap использует множество различных методов сканирования (UDP, TCP, TCP SYN, FTP, ICMP и т.д.), а также поддерживает большой набор дополнительных возможностей. Ниже описаны некоторые полезные возможности этой замечательной утилиты. Для выполнения большинства операций nmap требуются полномочия пользователя root. При запуске nmap от имени обычного пользователя значительная часть функций будет не доступна.

1. Получение информации об удаленном хосте и определение операционной системы

Nmap используется в следующем виде:

```
$sudo nmap -sS -P0 -sV -O
<target>
```

Где:

target — IP, хост или подсеть;

-sS — TCP SYN сканирование (полуоткрытое);

-P0 — отключение ICMP сканирования;

-sV — определение закрытых и фильтруемых портов;

-O — определение версии операционной системы.

Еще опции:

-A — включает определение «отпечатка» и версии операционной системы;

-v -vv — уровень вывода диагностических сообщений.

При использовании альтернативных опций, команда выглядит следующим образом:

```
$sudo nmap -sS -P0 -A -v <
target >
```

2. Определение списка серверов с открытым портом

Nmap используется в следующем виде:

```
$sudo nmap -sT -p 22 -oG -
192.168.1.* | grep open
```

Номер порта указывается после опции «-p». В данном примере выполняется поиск машин, для которых возможен вход по ssh (если, конечно, не изменен порт по умолчанию для ssh).

3. Поиск активных IP адресов в сети

Nmap используется в следующем виде:

```
$sudo nmap -sP 192.168.0.*
```

Чтобы опросить конкретную подсеть, можно использовать следующие параметры:

```
$sudo nmap -sP
192.168.0.0/24
```

4. Опросить (пропинговать) диапазон адресов

Nmap используется в следующем виде:

```
$sudo nmap -sP
192.168.1.100-254
```

Nmap понимает много нотаций IP адресов.

5. Поиск неиспользуемых IP адресов в подсети

Nmap используется в следующем виде:

```
$sudo nmap -T4 -sP
192.168.2.0/24 && egrep
“00:00:00:00:00:00 /proc/
net/arp
```

6. Поиск вируса Conficker в подсети

Nmap используется в следующем виде:

```
$sudo nmap -PN -T4
-p139,445 -n -v \
-script=smb-check-vulns \
-script-args \
safe=1 192.168.0.1-254
```

Чтобы скорректировать список IP адресов, нужно заменить «192.168.0.1-256» на свой вариант.

7. Поиск в сети мошеннических точек доступа (AP)

Nmap используется в следующем виде:

```
$sudo nmap -A -p1-
85,113,443,8080-8100 \
-T4 -min-hostgroup 50 \
-max-rtt-timeout 2000 \
-initial-rtt-timeout 300 \
-max-retries 3 \
-host-timeout 20m \
-max-scan-delay 1000 \
-oA wapscan 10.0.0.0/8
```

8. Декорирование истинного IP адреса при сканировании сети

Nmap используется в следующем виде:

```
$sudo nmap -sS 192.168.0.10
-D 192.168.0.2
```

В данном примере выполняется поиск открытых портов на машине 192.168.0.10, при этом, в качестве адреса, откуда ведется сканирование, указан адрес 192.168.0.2. Таким образом, в логах машины 192.168.0.10 будет отображен не действительный IP адрес, с которого ведется сканирование, а указанный — 192.168.0.2.

9. Список обратных DNS записей для подсетиNmap используется в следующем виде:

```
$sudo nmap -R -sL
209.85.229.99/27 | \
awk '{if($3==»not»)print»(«
$2 «) no PTR»;else print $3
« is «$2}' | \
grep ‘(‘
```

В этом примере nmap выполняет поиск обратных DNS записей для подсети. Результатом поиска будет список IP адресов с соответствующими PTR записями для подсети. Чтобы выполнить запрос через конкретный DNS сервер, необходимо добавить «-dns-servers x.x.x.x» после опции «-sL».

10. Подсчет Linux/Windows машин в сети

Nmap используется в следующем виде:

```
$sudo nmap -F -O
192.168.0.1-255 | \
grep «Running: « > /tmp/os;
\
echo «$(cat /tmp/os |
grep Linux | wc -l) Linux
device(s)»; \
echo «$(cat /tmp/os | grep
Windows | wc -l) Window(s)
devices»
```

www.unixhome.org.ua

Четыре инструмента архивации для администратора сервера Linux

Существует множество приложений резервного копирования самых разных видов – от простых и бесплатных, до сложных и дорогих. Но все равно остаются существенные проблемы со скоростью работы, простотой и удобством использования созданных резервных копий. Старыми испытанными методами архивного резервирования Linux/Unix продолжают пользоваться и современные администраторы серверов Linux; данные методы быстры, просты, удобны в использовании и легко поддаются управлению с помощью скриптов.

TAR

Иногда кажется, что tar существовал всегда. Он объединяет наборы файлов в один большой архив (что, кстати, упрощает задачу обмена файлами между разными людьми), но он не сжимает эти файлы, следовательно – не экономит дисковое пространство. Однако, несмотря на это, tar поддерживает параметры командной строки, которые позволяют сжимать архивы tar при помощи gzip или bzip2 (это будет продемонстрировано ниже).

Когда tar создает архив, он объединяет вместе файлы, каждый со своим заголовком, содержащим метаданные: имя файла, владелец, права доступа к файлу и любая другая информацию, описывающая файл. Метаданные для мобильности помещаются в архив в формате ASCII. Таким образом, потом архив предоставит информацию о файле вместе с самими данными, содержащимися в этом файле.

Одна из случайных проблем, которые могут возникнуть во время извлечения файлов из архива, – извлечение файлов «как есть», то есть прямо в текущую директорию (вместо того, чтобы создать свою собственную директорию). В лучшем случае, это может просто «загрязнить» директорию, в худшем же это может привести к перезаписыванию уже существующих файлов. Избежать такой неприятности можно, используя команду tar -tf file.tar, чтобы получить листинг файлов перед их извлечением; затем можно переместить архив в новую пустую директорию, если это необходимо.

Команды tar:

```
tar -cf archive.tar mydirectory/
```

Создает тарболл archive.tar, содержащий директорию mydirectory/ (извлечение произойдет с созданием директории mydirectory).

```
tar -xf archive.tar
```

Извлекает содержимое тарболла archive.tar в текущую директорию.

```
tar -zxf archive.tar.gz
```

Извлекает содержимое тарболла, сжатого gzip. Используйте -j вместо -z для архива bzip2.

```
tar -vxf archive.tar
```

Извлекает содержимое тарболла с подробным выводом (отображает список всех извлекаемых файлов).

ZIP

zip может одновременно и архивировать, и сжимать файлы, – то есть вы можете сжать различные файлы, поместив их при этом в архив. Метод известен еще с 1998 года и реализован на многих платформах, поэтому это один из наиболее мобильных вариантов (особенно если вам необходим доступ к вашему архиву из системы Windows).

Архив zip включает в себя главную директорию с именами файлов и метаданными файлов, находящуюся в конце файла. Это очень ускоряет процесс вывода списка файлов, находящихся в архиве, поскольку нет необходимости читать весь архив, достаточно прочитать только одну директорию.

Компрессия в большинстве случаев от zip-архивов не требуется, но по умолчанию она используется. Обычно используется метод DEFLATE, который устраняет повторяющиеся строки и записывает короткие варианты общих символов (и длинные варианты коротких вариантов). Каждый файл сжимается отдельно, что предпочтительнее, чем сжатие всего архива целиком. Это обеспечивает более быстрый доступ к архиву, но, поскольку метаданные не сжимаются, архив с большим количеством маленьких файлов не будет настолько уменьшен, как архив с малым числом больших файлов.

Команды zip:

```
zip -r myarchive dir
```

Создает архив myarchive.zip с содержимым директории dir.

```
unzip myarchive.zip
```

Извлекает содержимое архива myarchive.zip в текущую директорию, создавая вложенные директории (используйте -j чтоб не создавать вложенных директорий).

GZIP

gzip может сжимать файлы, но не архивировать. Однако он прекрасно работает, если его использовать вместе с архивирующим инструментом – как бывает на практике, gzip и tar отлично работают вместе.

gzip, как и zip, использует вариант алгоритма DEFLATE. Однако gzip сожмет tar-архив лучше, чем zip, потому что он сжимает архив в целом лучше, нежели каждый файл по-отдельности, он, таким образом, может найти (и устранить!) повторения как между файлами, так и внутри них.

Команды gzip:

```
gzip file.tar
```

Сжимает файл file.tar и создает file.tar.gz.

```
tar czf file.tar.gz mydir/
```

Архивирует и сжимает mydir/ в файл file.tar.gz.

```
gunzip file.tar.gz или
gzip -d file.tar.gz
```

Распаковывает file.tar.gz

```
tar zxf file.tar.gz
```

Распаковывает архив file.tar.gz и извлекает из него файлы.

Кстати, для распаковки напрямую

в стандартный вывод можно использовать gzcat (используется так же, как gunzip).

BZIP2

bzip2, как и gzip, сжимает, но не архивирует. И, опять-таки, он отлично работает с tar, если вы хотите получить и архивацию, и сжатие. Bzip2 сжимает лучше, чем gzip, но процесс сжатия у bzip2 занимает больше времени, чем у gzip.

Bzip2 использует последовательность преобразований сжимаемых данных, наиболее важное из которых – преобразование Барроуза-Уилера (англ. Burrows-Wheeler transform – прим. Перев.) Данное преобразование сортирует блоки файла так, чтобы максимально увеличить число повторяющихся символов. Цепочки повторяющихся символов заменяются символом и кодом, указывающим на длину цепочки. Также применяется кодирование Хаффмана (которое применяется и в алгоритме DEFLATE). Словом, все это занимает очень много времени.

Команды bzip2:

```
bzip2 file.tar
```

Сжимает файл file.tar и создает file.tar.bz2.

```
bunzip2 file.tbz
```

Распаковывает file.tbz и создает file.tar

```
tar jxf file.tbz
```

Распаковывает архив file.tbz и извлекает из него файлы.

Кстати, для распаковки напрямую в стандартный вывод можно использовать bzcat.

Если вы заинтересовались алгоритмами сжатия, можете также обратить внимание на 7-zip (консольная версия под Linux – p7zip), который использует алгоритм LZM. В большинстве случаев он сжимает лучше, чем bzip2, и быстрее распаковывает (зато дольше сжимает). Однако он намного менее популярен.

Оригинал: «4 Archiving Tools for Linux Server Admins
Автор: Juliet Kemp
Дата публикации: 11 октября 2010 г.
Перевод: Д.Оводов
Дата перевода: январь 2011 г.

www.rus-linux.net

Эффективно работаем в консоли Линукса

Я очень много времени провожу под консолью. Да, именно под настоящей консолью, а не эмулятором терминала. И я не содержу сервер, как могут подумать. Просто это на самом деле удобно, как для меня. А для того, чтобы было еще удобнее, я ее настроил под себя — так и удобнее, и производительность труда немного выше. Для тех, кто тоже хочет посвятить себя ее изучению, дам несколько рекомендаций, что нужно сделать.

СОВЕТ №1 ВКЛЮЧИТЬ И НАСТРОИТЬ FRAMEBUFFER

В стандартном виде консоль линукса просто текстовая. Ее разрешение составляет всего 80x25 символов (это текстовый режим), а мой монитор умеет все 1280x1024 пикселей (графический режим), и это уже не рациональное использование ресурсов. Конечно, работать в ней можно, но не очень удобно — имена файлов могут не влезать, большие, прям таки огромные буквы и, как говорят, ШГ. А то, во что превращается Midnight Commander, им только детей пугать. Но время идет и прогресс коснулся и консоли Linux! Что же этот прогресс может нам дать? А дать он может многое, и все благодаря использования технологии кадрового буфера, или Framebuffer.

Кадровый буфер (framebuffer, буфер кадра, видеобуфер, фреймбуфер) — виртуальное электронное устройство или область памяти для кратковременного хранения одного или нескольких кадров в цифровом виде перед его отправкой на устройство видеовывода. Буфер может быть использован для выполнения над кадром различных предварительных операций, организации стоп-кадра, устранения мерцания изображения и др. Обычно кадр хранится в виде последовательности цветовых значений каждого пикселя изображения. Цветовые значения чаще всего хранятся в следующих форматах: одноразрядный (монохромный; 1 бит), 4/8-битный (палитровый), 16-битный (High Color) и 24-битный (True Color); также может присутствовать альфа-канал. Размер памяти, необходимый для хранения одного кадра, зависит от разрешения и глубины цвета. При этом управление происходит уже не через устройства /dev/tty*, а через /dev/fb*.

Как его включить? Если у Вас современный дистрибутив с ядром версии старше 2.4.5 (рекомендуется ядро ветки 2.6), даю 99,9%, что фреймбуфер у вас уже включен по-умолчанию. Нужно его только настроить. Если Вы на 100% уверены, что эта технология все-таки у вас не работает, тогда только пересобирать ядро. Но как это делать, описывать не буду. Может как-то потом.

Настройка фреймбуфера. Для того, что бы от фреймбуфера было больше толка, давайте настроим его. Во-первых, выставим для него оптимальное разрешение — обычно настраивают максимальное, которое поддерживает LCD монитор. Мой имеет диагональ 17», значит наше рабочее разрешение будет 1280x1024. Глубину цвета возьмем тоже побольше — 24bit. Смотрим в таблицу и по этим параметрам узнаем код (больше можно узнать на английской вики):

Предупреждение: режимы для разрешений более 1280x1024 еще не достаточно оттестированы, поэтому никаких гарантий не даю!

Итак, для меня подходит значение 0x31b. Его можно таким и оставить, а можно перевести в десятичную, более привычную систему счисления: запускаем редактор vim и в нем вводим команду: echo 0x31b. Я получил

результат 795. Одно из этих значений нужно передать в ядро при загрузке через параметр «vga». Теперь настройте свой загрузчик. К примеру, вот вырезка из моего /boot/grub/menu.lst:

```
   title linux
   root (hd0,0)
   kernel /boot/vmlinuz
splash=verbose vga=795
   initrd /boot/initrd.img
```

Для отключения фреймбуфера просто убираем параметр vga.

Что нам это дало? Во-первых глазам приятнее. Если еще поставить красивый шрифт, как я написал тут, то прямо загляденье. Во-вторых, удобнее работать. Представьте только, как на серверном 22» мониторе будет удобно читать логи! В-третьих, сейчас уже очень много программ и игр работают под этой чудо-технологией. К примеру, qemu, mplayer (один раз у меня сгорела видеокарта и X просто не мог загрузится. Тогда я через mplayer под консолью фильмы с аниме гонял), frozen-bubble, mmc и так далее. Можно даже попробовать отказаться от X.org :)

```
-noquiet -nofs
-nomouseinput -vc
coreserve, -sub-fuzziness 1
-identify -slave -vo xv -ao
pulse -stop-xscreensaver
-ass -embeddedfonts -ass-
line-spacing 0 -ass-font-
scale 1 -ass-force-style Pl
ayResX=512,PlayResY=320,N
ame=Default,Fontname=Aria
l,Fontsize=20,PrimaryColo
ur=\&H00ffffff,BackColour
=\&H00000000,OutlineColou
r=\&H00000000,Bold=0,Ital
ic=0,Alignment=2,BorderSt
yle=1,Outline=1,Shadow=2,
MarginL=20,MarginR=20,Ma
rginV=8 -fontconfig -font
Arial -subfont-autoscale
0 -subfont-osd-scale 20
-subfont-text-scale 20
-cache 2000 -osdlevel
1 -vf-add screenshot
-slices -channels 2
-af scaletempo,equaliz
er=4:3:2:1:0:0:1:2:3:4
-softvol -softvol-max 110
-subcp enca:ru:cp1251»
```

Вы можете придумать что-то свое.

	640×480	800×600	1024×768	1280×1024	1600×1200
256 / 8bit	0x301 / 769	0x303 / 771	0x305 / 773	0x307 / 775	796
32k / 15bit	0x310 / 784	0x313 / 787	0x316 / 790	0x319 / 793	797
64k / 16bit	0x311 / 785	0x314 / 788	0x317 / 791	0x31A / 794	798
16M / 24bit	0x311 / 786	0x314 / 789	0x317 / 792	0x31b / 795	799

СОВЕТ№2 НАСТРОИТЬ ~/.BASHRC

Но это опционально. Я, например, добавил в запрос счетчик количества фоновых приложений — такое себе подобие панели задач. Для этого я дописал в файл ~/.bashrc такую строку:

```
   export PS1=»[\u'\j'@\H
\W]$ «
```

Или вот, например, чтобы не набирать всю команду перед каждым просмотром видео, создадим псевдоним для mplayer. Теперь можно просто набрать в консоли mplay с именем файла и смотреть с любимыми настройками:

```
   alias mplay=»mplayer
```

СОВЕТ№3. НАУЧИТСЯ ЭФФЕКТИВНО УПРАВЛЯТЬ ЗАДАЧАМИ

Linux, в отличии от MS-DOS, всегда был многозадачной ОС. Конечно, пользы от этого было бы мало, если бы не придумали нормального механизма управления процессами. И это действительно сделали. Далее я опишу простейшие манипуляции с процессами, которые нужно знать для начала.

Запуск приложений. Большинство программ для Линукс умеют работать и под консолью, даже если они имеют графический интерфейс. Для запуска просто вводим команду и по возможности параметры. Для примера запустим консольный проигрыватель музыки с плейлистом:

```
   [keed'0'@mandrivka ~]$
mpg123 -@ ~/enigma.m3u
   High Performance MPEG
1.0/2.0/2.5 Audio Player
for Layers 1, 2 and 3
   version 1.12.1; written
and copyright by Michael
Hipp and others

   free software (LGPL/GPL)
without any warranty but
with best wishes

   Directory: /data/Music/
Enigma/1991 - MCMXC a D
Limited Edition/
   Playing MPEG stream 1
of 107: 01 - The Voice Of
Enigma.mp3 ...
   Title: The Voice Of
Enigma Artist: Enigma
   Album: MCMXC a D Limited
Edition
   Genre: New Age
   MPEG 1.0 layer III, 192
kbit/s, 44100 Hz stereo
```

С музыкой жить веселее. Вот только теперь наша консоль получилась занятой и ничего нового ввести не получается. Что делать?

Можно плеер «свернуть». Для этого просто жмем клавиши Ctrl+Z. Это доступно для любых приложений, не только для плеера. Выведется примерно такое сообщение и музыка прекратит играть:

```
   ^Z
   [1]+ Stopped mpg123 -@ ~/
enigma.m3u
   [keed'1'@mandrivka ~]$
```

Нет, мы не закрыли приложение, мы его временно приостановили — он все еще запущен и занимает ресурсы компьютера. Для продолжения в фоновом режиме вводим команду bg (от англ. background — фон):

```
   [keed'1'@mandrivka ~]$ bg
   [1]+ mpg123 -@ ~/enigma.
m3u &
   [keed'1'@mandrivka ~]$
```

Музыка снова заиграла — процесс продолжился в фоне. Посмотрите на вторую строку — там написана наша команда, а в конце стоит знак «&». Для чего он нужен? Так вот, если его дописать в конец команды и выполнить ее, то программа сразу запустится в фоне. В нашем случае мы запустили

плеер в обычном режиме, потом остановили и перезапустили его в фоне. Для примера давайте еще что-то еще запустим, скажем, одну бесконечную и бестолковую команду:

```
   [keed'1'@mandrivka ~]$
cat /dev/zero > /dev/null &
   [2] 15335
   [keed'2'@mandrivka ~]$
```

Процессу присваивается PID 15335, условный идентификатор 2 и приложение уходит выполнятся на задний план. Запустим еще что-то и остановим его:

```
   [keed'2'@mandrivka ~]$
ping 127.0.0.1
   PING 127.0.0.1
(127.0.0.1) 56(84) bytes of
data.
   64 bytes from 127.0.0.1:
icmp_seq=1 ttl=64
time=0.068 ms
   64 bytes from 127.0.0.1:
icmp_seq=2 ttl=64
time=0.060 ms
   ^Z
   [3]+ Stopped ping
127.0.0.1
   [keed'3'@mandrivka ~]$
```

Теперь у нас в фоне 3 процесса. Как с ними работать? Очень даже просто. Давайте глянем, что у нас запущено и в каком состоянии оно находится. Тут нам пригодится команда jobs:

```
   [keed'3'@mandrivka ~]$
jobs
   [1] Running mpg123 -@ ~/
enigma.m3u &
   [2]- Running cat /dev/
zero > /dev/null &
   [3]+ Stopped ping
127.0.0.1
   [keed'3'@mandrivka ~]$
```

Два работающих процесса, один приостановленный. Теперь каждый из них можно «развернуть», то есть перевести на передний план. Для этого есть программа fg (от англ. foreground — передний план). Используем ее так: fg, далее ставим знак «%» и без пробела идентификатор процесса — число, что стоит в первом столбике вывода jobs в квадратных скобках. Проверим, пингуется еще сеть или уже нет и опять остановим процесс:

```
   [keed'3'@mandrivka ~]$ fg
%3
   ping 127.0.0.1
   64 bytes from 127.0.0.1:
icmp_seq=21 ttl=64
time=0.074 ms
   64 bytes from 127.0.0.1:
icmp_seq=22 ttl=64
time=0.058 ms
   ^Z
   [3]+ Stopped ping
127.0.0.1
   [keed'3'@mandrivka ~]$
```

Завершение процессов. Завершить процесс можно «мягко» и «жестко». Что бы это сделать мягко, переключимся на его и нажмем «Ctrl+C» (если для этого в программе не придумано специальных горячих клавиш, как в mc, например — F10):

```
   [keed'3'@mandrivka ~]$ fg
%2
   cat /dev/zero > /dev/null
   ^C
   [keed'2'@mandrivka ~]$
```

Но бывает так, что программа начинает вести себя странно и/или потреблять довольно много ресурсов. Мягкий способ при этом не всегда помогает. Придется процесс убить — команда kill поможет нам. Используется она также, как и fg — убивает по PID`у или числовому идентификатору:

```
   [keed'3'@mandrivka ~]$
jobs
   [1] Running mpg123 -@ ~/
enigma.m3u &
   [2]- Running cat /dev/
zero > /dev/null &
   [3]+ Stopped ping
127.0.0.1
   [keed'3'@mandrivka ~]$
```

Тут понятно, что мы убили ping. Однако, завершая процессы таким способом, Вы скорее всего потеряете все несохраненные данные. Поэтому только в крайних случаях! Убить процесс можно и не зная его числовых значений, а зная только имя:

```
   [keed'3'@mandrivka ~]$
killall ping
   [keed'2'@mandrivka ~]
```

Используйте top. top — замечательная стандартная утилита для управления процессами. Описывать ее огромный функционал я не буду — это за меня уже сотни раз было сделано. Читайте man top в консоли, это полезно :) И на всякий случай приведу маленькую шпаргалку по тем командам, которые мы сегодня выучили:

команда & — запуск команды в фоновом режиме;

bg — перевод команды в фоновую работу;

fg %n— перевод процесса [n] в развернутый режим;

jobs — вывод списка запущенных процессов;

kill %n — убить процесс [n];

killall команда — завершить команду по ее названию;

nohup команда — запуск команды без привязки к консоли;

Ctrl+Z — приостановить процесс;

Ctrl+C — мягкое завершение процесса;

top — консольный диспетчер задач

Для еще более гибкой работы в консоли попробуйте изучить screen.

СОВЕТ №4

Освойте скрипты. Это просто находка для автоматизации процессов. И это совсем не такие скрипты, как bat и cmd из мира Microsoft. Это на самом деле очень серьезный и удобный инструментарий для системного администрирования — тот не админ, который не знает bash. Настоятельно рекомендую руководство Advanced Bash-Scripting Guide. Скачать его бесплатно можно отсюда — http://nklug.org.ua/lg/rus/articles/index-abs-guide.html. Руководство очень хорошо написано, будет полезно как новичкам, так и опытным программистам. Я сам по нему учился и знаю, что говорю. Даже если вы далеки от программирования, научитесь хотя бы читать сценарии. Начните с легкого — с Bash, а далее, если втянитесь, самим захочется освоить что-то посерьезнее вроде Perl или Python.

СОВЕТ №5

Запустите сервис консольной мыши. Для этого установите

и запустите сервис gpm:

```
   [keed'0'@mandrivka ~]$
sudo urpmi gpm
   [keed'0'@mandrivka ~]$
sudo /etc/init.d/gpm start
```

Теперь прямо в консоли можно выделять, копировать и вставлять текст, а также более эффективно работать с vim или mc, например.

Андрей 'keedhost' Кондратьев
www.keedhost.blogspot.com

Модем для FreshTel под Linux

Идея заставить работать модем для Freshtel под Linux появилась еще до появления в руках самого модема, собственно потому, что доступ к интернету необходим был в том месте где Freshtel с раздачей его по Wi-Fi был единственным вариантом, а цена роутера от Freshtel была неоправданно большой.

В счастливые времена раздачи модемов по 200 грн. я его и заполучил! Изучил все внутренности и начал поиски способа запустить его под Линуксом. Первой мыслью было, а может ndiswrapper?, но для начала он должен был работать собственно в режиме модема, с чем помог Josua Dietze (автор программы usb_modeswitch)

С ndiswrapper'ом ничего не получилось, была найдена информация о том, что чипы GDM7213 будут использоваться в коммуникаторах с Google Android, но это была всего лишь новость.

Во время поисков я часто сталкивался с драйвером madwimax Александра Гордеева, решил проверить как работает связка модема с ПК, вдохновился статьей про Реверс инжиниринг с сайта Гордеева. В этом мне помогла программа UsbSniff, я пользовался ей и раньше, снимая логи для Josh'a, но тогда я не вникал что там и как. Разбираться в логах оказалось очень интересно и увлекательно, меня это просто затянуло, это было интересней чем разгадывать кроссворды, ты ищешь, что тот или иной знак значит, а когда начинаешь понимать....

Возможно я когда-то и выложу часть логов если кому-то будет интересно.

Когда все окончательно прояснилось, нужно было переходить к написанию самого драйвера, с одной стороны это было безумно интересно, но возникла другая проблема - я программист-самоучка и до этого ничего не писал на «С» тем более под Линуксом.

За основу был взят драйвер madwimax-0.1.1, спасибо Александру за то, что там все очень понятно написано, и началось...

Ясное дело модемы совсем разные, была полностью переписана часть программы по роботе с протоколом, сам метод аутентификации тоже отличался, в Самсунговском модеме он проходит на уровне железа, а у модема для Freshtel - софтверно. С ней пришлось больше всего повозится, благо у создателя wpa_supplicant очень внятная документация для разработчиков.

От первоначального драйвера в оригинальном виде остались работа с TUN/TAP интерфейсом и вывод логов. Кстати есть предположение что драйвер будет работать не только с чипами GDM7213, но и GDM7205.

Драйвер мог появится и раньше, примерно месяца на 4, в тот момент когда я вроде бы все дописал, он не проходил аутентификацию на сервере Freshtel. Я с этим долго бился, но никак не мог побороть, начиналась учеба и я из-за недостатка времени, закинул это дело. Как то вечером на очередных каникулах я все таки понял в чем проблема, на самом деле даже смешно говорить вот приведу строчку кода который был.

Было:

```
eap_ctx.eap_config.
phase2 = (char *) os_
strdup(«eapauth=
MSCHAPV2»);
```

а нужно:

```
eap_ctx.eap_config.
phase2 = (char *) os_
strdup(«auth=
MSCHAPV2»);
```

и все прошло гладко как по маслу.

Драйвер вы можете скачать по адресу http://code.google.com

Его обсуждение на форуме UALinux www.ualinux.com.

Почему gctWiMAX? наверное мне было лень придумывать что то оригинальней, а буквы gct были взяты из названя фирмы производителя чипов для наших модемов GCT Semiconductor, Inc.

Буду рад любой помощи в тестировании и доведении драйвера до ума.

Ярослав Левандовський

Настраиваем сканер Mustek BearPaw 2400 CS Plus

Давно прошли те времена, когда с поддержкой периферийных устройств в linux были проблемы. ;-) Сейчас я вам на примере сканера Mustek покажу, как легко и просто подключаются и настраиваются разные непонятные китайские железки.

Итак. Есть USB-сканер Mustek, на корпусе написано «BearPaw 2400 CS Plus». Для начала нам нужно понять, видит ли вообще ядро linux на usb-портах хоть что-нибудь, похожее на сканер:

```
[fog@computer ~]$ lsusb | grep
Mustek
Bus 003 Device 002: ID 055f:0219
Mustek Systems, Inc. BearPaw 2400
TA Plus
```

Так, отлично, сканер мы видим. Теперь нужно нечто, чем можно сканировать.

Погуглив «линукс сканнер», находим многочисленные ссылки на проект «SANE» (Scanner Access Now Easy), где на странице «Supported devices» обнаруживаем упоминание нашей модели сканера. Чудненько. Теперь нам нужна GUI'овина для этой штуки. Опять же, на сайте проекта, но теперь уже на странице «Frontends» видим упоминание XSane. Видимо оно-то нам и нужно. Открываем PackageKit и устанавливаем соответствующие пакеты. Запускаем.

```
[fog@computer ~]$ xsane
[gt68xx] Couldn't open firmware file
(`/usr/share/sane/gt68xx/A2Dfw.
usb'): Нет такого файла или ката-
лога
```

Опаньки. Какая-то непонятная фигня. Не может найти какую-то firmware. Что делать?

Возвращаемся на станицу «Supported Devices». Ага. В табличке есть ссылка на какой-то «Backend». Причем, название файла подозрительно похоже на то, что не может найти XSane. Переходим по ссылкам (там кстати даже есть описание на русском), скачиваем файл и кладем его в нужную папочку.

```
[root@computer fog]# mkdir /usr/
share/sane/gt68xx
[root@computer fog]# cp A2Dfw.usb
/usr/share/sane/gt68xx/A2Dfw.usb
```

Запускаем еще разок xsane и вот результат:

Как видите, сканер в линукс подключается элементарно. :-)

http://open-life.org/

Как сделать офисный сканер сетевым

В одной маленькой организации (~10 компьютеров) было необходимо организовать резервное копирование информации. Для этого было решено установить компьютер с большим жестким диском и Ubuntu внутри, а на компьютеры пользователей – Cobian backup. Пытливый читатель спросит: «А зачем ты это пишешь, автор? И при чем тут какой-то сканер?» Дело в том, что в этой организации была только одна возможность отсканировать документ – попросить это сделать счастливого обладателя МФУ Samsung SCX-4200 (название, конечно же, непринципиально, но все произошло именно из-за этого устройства).

Во время установки Ubuntu на «сервер для резервного копирования» и возникла идея: «А если подключить это МФУ сюда, оставить клавиатуру подключенной, и по нажатию определенных клавиш заставить сканер сканировать, сохраняя результат в общую папку? Ведь тогда человек не будет отвлекаться от работы для сканирования чужих документов!»

Конечно же, в первую очередь искались готовые скрипты. Наиболее интересным показался этот –

```
www.opennet.ru/base/sys/
net_scanner.txt.html
```

Однако при ближайшем рассмотрении он оказался не таким уж хорошим, т.к. у меня совсем нет желания перекомпилировать программу для изменения настроек сканера, учить пользователей конвертировать картинки и создавать .pdf-файлы.

Попробую написать свой, хороший…

ИТАК, ЗАДАЧА ПОСТАВЛЕНА

1. Подключить к Ubuntu МФУ, сделать принтер общим, настроить сканер.
2. Написать скрипт, который будет ожидать нажатия клавиши на клавиатуре

Esc – отмена незавершенного сканирования
1 – режим сканирования в цвете
2 – режим сканирования в оттенках серого
0 – сохранить отсканированный файл (jpg, если была отсканирована одна страница или pdf, если несколько)
Enter – отсканировать страницу.

РЕШЕНИЕ

Пункт 1 отпал сам собой. Подключение принтера произошло неожиданно просто и не вызвало ни одного вопроса. Достаточно было просто подключить кабель USB к компьютеру, и через несколько секунд Ubuntu сообщила, что принтер должен печатать. Отправил пробную страницу – действительно печатает! Теперь самое интересное.

Посмотрим, какие сканеры у нас есть в системе:

```
scanimage -L
```

Если система не может найти эту команду – значит, нужно установить пакет sane-utils:

```
sudo apt-get install sane-
utils
```

Программа нашла устройство с именем `xerox_mfp:libusb:001:002`
Если сканер единственный в системе, то параметр «имя устройства» можно опустить; scanimage будет сканировать единственно возможным сканером.

Попробуем отсканировать страницу:

```
scanimage -d “xerox_
mfp:libusb:001:002”
--resolution 150 --mode
Color --format=tiff > test.
tif
```

Разрешение 150 dpi выбрано из-за желания уменьшить время сканирования и размер выходного файла, но при этом оставить пользователю возможность напечатать документ с читабельным текстом. Если когда-нибудь потребуется распознавание текста или сканирование фотографий – тогда появятся дополнительные опции.
Итак, у нас есть tiff. Либо один, либо несколько (отсканировали многостраничный документ). Понятно, что пользователи в 90% случаев сканируют документы для отправки по электронной почте, а tiff – формат не компактный. Значит, нужно сконвертировать результат в .jpg или .pdf.

Ставим пакет для редактирования/конвертирования растровых изображений:

```
sudo apt-get install
imagemagick
```

Сжимаем единственное изображение:

```
convert -quality 60% test.
tif test.jpg
```

Или все сразу:

```
convert -compress jpeg
```

```
-quality 60% *.tif all.pdf
```

Остался только один момент. Я не собираюсь оставлять подключенным к этому компьютеру монитор – значит, пользователь остается без обратной связи. Ну что ж, сделаем компьютер говорящим.

Составим список звуков, которые должны воспроизводиться в ответ на нажатие той или иной клавиши на клавиатуре, берем в руки микрофон и записываем наши мини-фразы. Они максимально короткие по времени, ведь цель всего действа – не прослушивание mp3, а сканирование.

1. Жду команду (waitcommand.mp3)
2. Цветное сканирование (color.mp3)
3. Черно-белое сканирование (bw.mp3)
4. Сканирую страницу (scanpage.mp3)
5. Сохраняю результат (saveresult.mp3)

Ах да, ведь у нас «чистая» система, которая не умеет воспроизводить mp3 из консоли… Исправляем ситуацию:

```
sudo apt-get install mpg321
```

Наслаждаемся:

```
mpg123 -q waitcommand.mp3
```

ТЕПЕРЬ, КАЖЕТСЯ, ВСЕ ГОТОВО

Пишем скрипт /mnt/2tb/scan.sh

```
#!/bin/bash

# ========================
========================
```

```
# Настройки
scannerdevice=»xerox_
mfp:libusb:001:002»
#scanimage -L
workdir=»/tmp/scanworkdir»
destdir=»/mnt/2tb/
Share/1Scanner»
dpi=»150»
jpegquality=»60%»
# ========================
========================
# Собственно скрипт
sleep 10s
mkdir -p $workdir
rm $workdir/*.* 2>/dev/null
numpages=0
color=»Gray»
mpg123 -q waitcommand.mp3
while true
do

#Нажмите клавишу
read -sn 1 Keypress
case «$Keypress» in
$'\e')

#Жду команду
rm $workdir/*.* 2>/dev/null
numpages=0
color=»Gray»
mpg123 -q waitcommand.mp3
;;
$'1')

#Цвет
color=»Color»
mpg123 -q color.mp3
;;
$'2')

#ч/б
color=»Gray»
mpg123 -q bw.mp3
;;
$'0')

#Сканирование завершено
mpg123 -q saveresult.mp3
filename=`date +%Y%m%d-
%H%M%S`
if [ $numpages = 1 ]; then
convert -quality
$jpegquality $workdir/1.tif
$destdir/$filename.jpg
fi
if [ $numpages \> 1 ]; then
convert -compress jpeg
-quality $jpegquality
$workdir/*.tif
$destdir/$filename.pdf
fi
rm $workdir/*.* 2>/dev/null
numpages=0
color=»Gray»
mpg123 -q waitcommand.mp3
;;
$'')

#Новая страница
mpg123 -q scanpage.mp3
let «numpages=numpages+1»
scanimage -d $scannerdevice
--resolution $dpi --mode
$color --format=tiff
>$workdir/$numpages.tif
mpg123 -q waitcommand.mp3
;;
esac
done
```

Говорим

```
chmod +x /mnt/2tb/scan.sh
```

ЗАПУСКАЕМ

Да, все работает как ожидалось. Теперь идем в меню Ubuntu Система -> Параметры -> Запускаемые приложения, Добавить, Обзор, выбираем файл со скриптом. Перезагружаем компьютер, и… сканер начинает непрерывно что-то сканировать, звуки не воспроизводятся.

Ладно, нажимаю Ctrl+C, читаю еще раз то что написал… Со звуками все тривиально – их нет в той папке, откуда вызывается скрипт. Непрерывное сканирование происходит, видимо, из-за $') внутри case'а.

Я не стал с этим детально разбираться, а просто изменил команду автозапуска на

```
/usr/bin/gnome-terminal -e
/mnt/2tb/scan.sh --working-
directory /mnt/2tb
```

Еще раз перезагружаю компьютер – все работает, пользователи довольны.

http://habrahabr.ru/blogs/

Как заставить работать на Ubuntu принтеры Canon из серии LBP в сети

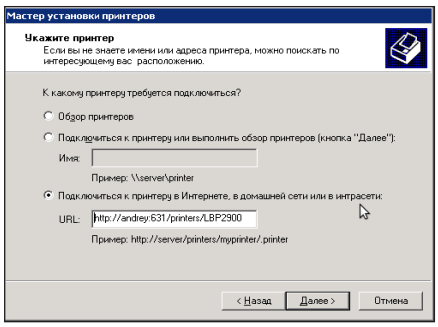
В предыдущей статье я описал как можно принтеры Canon LBP прикрутить и заставить работать на Ubuntu. Теперь усложним задачу: у нас еще появился компьютер с Windows, и этому компьютеру с виндой тоже надо печатать, и печатать на нашем принтере, прикрученного к Ubuntu-машинке. Как это сделать? Все тоже очень просто: Надо зайти в настройки CUPS и поставить несколько галочек – так мы расшарим принтер в сетке. Для этого набираем в броузере `http://localhost:631/admin`

и ставим галочки как показано на картинке:
у вас запросит пароль, введете его.

Теперь на компьютере с Windows нажмите Пуск – Принтеры и факсы – Установка принтера – Далее – сетевой принтер – поставите галочку на Подключится к принтеру и выбрать обзор принтеров. В окошечке наберете `http://192.168.1.1:631/printers/LBP2900`

где 192.168.1.1 – IP вашего компьютера, можно написать и имя компьютера, тогда надпись будет выглядеть так: `http://andrey:631/printers/LBP2900`,

но тут надо учесть, что ваши компьютеры должны быть в одной локальной сетке. В Windows, я надеюсь, вы знаете как это делать (правой кнопкой по Мой компьютер – свойства – вкладочка Имя компьютера – кнопочка Изменить – и откроется окошечко как изменить имя вашего компа и имя рабочей группы). А в Ubuntu в терминале набираем `sudo gedit`



откроется Gedit под рутом, в нем выбираете Файл – Открыть – выбираете Файловая система – etc/samba/smb.conf – открывается файлик, в этом файлике находите такие строчки:

```
#======================
Global Settings
======================
[global]
## Browsing/Identification
###
# Change this to the
workgroup/NT-domain name
your Samba server will
part of
workgroup = WORKGROUP
```

в последней строчке, после знака «=» стоит имя вашей сетки, ее можете сразу поменять и сохранить. И последняя надпись LBP2900 – это сетевое имя моего принтера, у вас оно может быть другим – это зависит от модели вашего принтера,)).

Все, после таких несложных манипуляций нажимаем кнопочку «далее», выбираем драйвер принтера из списка или ставим его с диска и все – ваш принтер должен заработать. Если после таких манипуляций не заработал, то попробуйте перезагрузить обе машинки, так как принтеры canon очень вредные и коварные)).

Кстати, эта инструкция подходит для любых принтеров, а не только для Canon, если с Windows надо печатать на принтер, подключенный к Ubuntu.

Дмитрий Бутолин
http://ubuntu.dp.ua/vau/

Подключаем 3G модем в Linux Ubuntu

Основной нашей задачей будет настройка и дальнейшее использование 3G модема в ОС Ubuntu Linux . Итак подключаем модем и …. Ну собственно и ничего. Исследуем более детально особенности работы 3G модема. Такие модемы используют технологию ZeroCD, т.е. модем определяется как CD-ROM, включает автозапуск, который проверяет наличие драйверов в системе и либо запускает специальную программу связи, либо устанавливает ее вместе с драйверами на устройство. Обычно под Windows – это QMobile или VZAccess. Основная задача – это перевести модем непосредственно в режим модема. Открываем Synaptic и ищем usb-modeswitch, или же выполняем в командной строке: `$sudo apt-get instal usb-modeswitch`

После перезагрузки устройство уже определяется как модем на порту /dev/ttyACM0 или что-то подобное (в зависимости от системы может различаться). Далее уже дело техники – или как кому по вкусу. Можно установить консольную утилитку wvdial или GUI-вый Gnome-ppp (так я и поступил) или использовать стандартный Network-Manager.

Создаем необходимое подключение.
Простейший конфиг для wvdial:

```
[Dialer Defaults]
Init1 = ATZ
Init2 = ATQ0 V1 E1 S0=0 &C1
&D2 +FCLASS=0
Modem Type = USB Modem
Baud = 460800
New PPPD = yes
Modem = /dev/ttyACM0
ISDN = 0
Phone = #777
Password = IT
Username = IT
```

http://spider.bsyteam.net

Программирование на Bash

Краткое введение. Часть II

В своей второй статье Гарольд продолжает первоклассное введение в программирование на bash. На этот раз он объясняет, как выполнять арифметические операции в скриптах bash и как определить функции в своих программах. Завершается статья введением в такие продвинутые вещи как чтение пользовательского ввода, обработка скриптом аргументов, перехватывание сигналов и обработка кодов завершения программ.

Безусловно, результаты прочтения превзойдут все ожидания! После этой статьи вас уже нельзя будет назвать новичком. Ведь вы на пути к тому, чтоб называться мастером программирования на bash!

АРИФМЕТИКА И BASH

bash позволяет выполнять арифметические операции. Как вы уже видели в предыдущей статье, арифметика выполняется с помощью команды expr. Однако, подобно команде true, этот вариант считается медленным. Причина кроется в том, что для использования true и expr оболочка должна предварительно запустить их. Лучше всего использовать встроенную в bash функцию, которая работает быстрее. Аналогично тому, что альтернативой true является команда «:», альтернатива expr – заключение арифметического выражения в конструкцию вида $((...)). Будьте внимательны, она отличается от $(...). Отличие тут в количестве скобок. Так давайте же испробуем это:

```
#!/bin/bash
x=8 # присваиваем x значение 8
y=4 # присваиваем y значение 4
# результат сложения x и y сохраняем в z:
z=$(($x + $y))
echo «Сумма $x и $y равна $z»
```

Как обычно, выбор используемого метода вычислений за вами. Если использование expr для вас более комфортно и привычнее, чем $((…)), используйте его.

bash умеет выполнять сложение, вычитание, умножение, целочисленное деление и получение остатка от деления. Каждое арифметическое действие имеет соответствующий ему оператор:

Действие	Оператор
Сложение	+
Вычитание	-
Умножение	*
Целочисленное деление	/
Остаток от деления	%

Большинство из вас должно быть знакомо с первыми четырьмя операциями. Если вы не знаете, что такое деление по модулю, то это просто число равное остатку от деления одного целого числа на другое. Ниже приведен пример выполнения арифметических операций в bash:

```
#!/bin/bash
x=5 # устанавливаем x равным 5
y=3 # устанавливаем y равным 3

# сохраняем сумму x и y в переменную add
add=$(($x + $y))

# сохраняем разность x и y в переменную sub
sub=$(($x - $y))

# умножаем x на y и сохраняем результат в переменную mul
mul=$(($x * $y))

# в переменную div сохраняем результат деления x на y
div=$(($x / $y))

# получаем остаток от деления x на y и сохраняем его в переменную mod
mod=$(($x % $y))

# печатаем ответы
echo «Сумма равна: $add»
echo «Разность равна $sub»
echo «Произведение равно $mul»
echo «Результат деления $div»
echo «Остаток от деления $mod»
```

Код, приведенный выше, можно было бы написать с использованием expr. Например, вместо `add=$(($x + $y))` мы могли бы использовать `add=$(expr $x + $y)` или `add=`expr $x + $y``.

ЧТЕНИЕ ВВОДА ПОЛЬЗОВАТЕЛЯ

А теперь – самое интересное. Мы напишем свой скрипт так, что он будет взаимодействовать с пользователем, а пользователь с ним. Команда для получения данных от пользователя – read. Это встроенная в bash команда, сохраняющая ввод пользователя в указанной переменной:

```
#!/bin/bash
# спросить у пользователя его имя и поздороваться с ним
echo -n “Введите свое имя: ”
read user_name
echo “Привет $user_name!”
```

Переменная здесь – это user_name. Конечно, мы могли бы назвать ее как угодно. read прервет выполнение скрипта и будет ждать, пока пользователь введет что-нибудь и нажмет клавишу ENTER. Если клавиша ENTER была нажата без ввода чего-либо, read запустит следующую строку кода. Попробуйте это сделать.

Ниже приведен тот же пример, только на этот раз мы проверяем, вводит ли что-то пользователь:

```
#!/bin/bash
# спрашиваем имя пользователя и выводим приветствие

echo -n «Введите имя: »
```

```
read user_name
# проверка ввода пользователя
if [ -z «$user_name» ]; then
echo «Вы не сказали мне свое имя!»
exit
fi
echo «Привет $user_name!»
```

В приведенном примере, если пользователь нажал ENTER и не ввел ничего при этом, наша программа напишет об этом и завершит свою работу. В противном случае она напечатает приветствие. Получение пользовательского ввода полезно для интерактивных программ, которые требуют от пользователя ввести какие-то данные.

ФУНКЦИИ

Использование функций делает проще поддержку своих скриптов. Проще говоря, это хороший способ разделить программу на более мелкие куски. Функция выполняет определенное действие и может возвращать то значение, какое вы пожелаете. Прежде чем продолжать, я приведу пример скрипта, написанного с использованием функции:

```
#!/bin/bash
# функция hello() печатает сообщение
hello()
{
echo «Вы находитесь в функции hello()»
}
echo «Вызываем функцию hello()…»
hello
```

Попробуйте запустить код из примера выше. Функция hello() в нем имеет только одно предназначение – просто напечатать сообщение. Но, конечно же, они могут решать и более сложные задачи. Выше мы вызвали функцию hello(), используя строку:

```
hello
```

Когда запускается эта строка, bash ищет скрипт для строки hello(). Он находит его в начале файла и выполняет его содержимое. Функции всегда вызываются по своему имени, что мы и видели выше. При написании функции вы можете обьявить ее, просто указав имя_функции(), как это сделано выше, или если вы хотите сделать ее объявление более явным, можете объявить ее так: function имя_функции(). Ниже представлен альтернативный способ написания функции `hello()` :

```
function hello()
{
echo «Вы находитесь в функции hello()»
}
```

Функции имеют в имени пустые открывающую и закрывающую скобки: «()», за ними следует пара фигурных скобок: «{...}», содержащих тело функции. Другими словами, весь код функции заключен в эти фигурные скобки. Функции всегда должны быть предварительно объявлены до своего вызова. Давайте попробуем в приведенном выше примере вызвать функцию до ее объявления:

```
#!/bin/bash
echo «Вызов функции hello() …»
hello
# функция hello() просто выводит сообщение
hello()
{
echo «Вы находитесь в функции привет ()»
}
```

Вот что мы получим, когда попытаемся запустить этот скрипт:

```
$ ./hello.sh
Вызов функции привет () …
./hello.sh: hello: command not found
```

Как видите, мы получили сообщение об ошибке. Поэтому стоит всегда размещать ваши функции в начале кода или, по крайней мере, непосредственно перед вызовом функции. Еще один пример использования функции:

```
#!/bin/bash
# admin.sh – инструмент для администратора
# функция new_user () создает новую учетную запись пользователя
new_user()
{
echo «Подготовка к созданию новых пользователей …»
sleep 2
# запускаем программу adduser
adduser
}
echo «1. Добавить пользователя»
echo «2. Выход»
echo «Укажите, что вы хотите сделать:»
read choice
case $choice in
1) new_user # вызов функции new_user()
;;
*) exit
;;
esac
```

Для того чтобы приведенный скрипт работал правильно, вам необходимо запустить его из-под пользователя root, т. к. иначе программа adduser не сможет создать новых пользователей. Надеюсь, этот пример (хоть он и краток) показывает положительный эффект от использования функций.

ПЕРЕХВАТ СИГНАЛОВ

Вы можете использовать встроенную в bash программу trap для перехвата сигналов в своих программах. Это хороший способ изящно завершать работу программы. Например, если пользователь, когда ваша программа работает, нажмет CTRL-C – программе будет отправлен сигнал interrupt (SIGINT, signal (7)), который завершит ее. Trap позволит вам перехватить этот сигнал, что даст возможность либо продолжить выполнение программы, либо сообщить пользователю, что программа завершает работу. Синтаксис этой команды такой:

```
trap action signal
```

Здесь:

`action` – то, что вы хотите делать, когда сигнал получен;

`signal` – сигнал, на который стоит реагировать.

Список сигналов можно посмотреть с помощью команды trap -l. При указании сигналов в своих скриптах можно опустить первые три буквы названия сигнала, т. е. SIG. Например, сигнал прерывания это – SIGINT. В ва-

шем скрипте, в качестве его имени, можно указать просто INT. Вы также можете использовать номер сигнала, указанный рядом с его именем. Например, числовое значение сигнала SIGINT – 2. Попробуйте написать и запустить приведенный ниже пример:

```
#!/bin/bash
# использование команды trap
# перехватываем нажатие CTRL-C и запускаем функцию sorry()
trap sorry INT

# function sorry() prints a message
sorry()
{
echo «Извини меня, Дэйв. Я не могу этого сделать»
sleep 3
}
# обратный отсчет от 10 до 1:
echo «Подготовка к уничтожению системы»
for i in 10 9 8 7 6 5 4 3 2 1; do
echo «Осталось $i секунд до уничтожения...»
sleep 1
done
echo «Запуск программы уничтожения!»
```

Наберите и запустите приведенный пример. Когда программа будет работать и вести обратный отсчет, нажмите CTRL-C. Это действие отправит программе сигнал прерывания – SIGINT. Тем не менее сигнал будет перехвачен командой trap, которая, в свою очередь, выполнит функцию sorry(). Вы можете заставить trap игнорировать сигнал, указав символ кавычек вместо указания действия. Также вы можете отключить ловушку с помощью тире: «-». Например:

```
# запускать функцию sorry(), если получен сигнал SIGINT
trap sorry INT

# отключение ловушки
trap – INT
# ничего не делать при получении сигнала SIGINT
trap ” INT
```

Если вы отключаете ловушку, программа работает как обычно – при получении сигнала прерывается ее исполнение и она завершает работу. Когда вы говорите trap ничего не делать при получении сигнала – она делает именно это. Ничего. Программа будет продолжать работать, игнорируя сигнал.

ЛОГИЧЕСКИЕ И И ИЛИ

Вы уже видели, что такое управляющие структуры и как их использовать. Для решения тех же задач есть еще два способа. Это логическое И – «&&» и логическое «ИЛИ» – « || ». Логическое И используется следующим образом:

```
выражение_1 && выражение_2
```

Сначала выполняется выражение, стоящее слева, если оно истинно, выполняется выражение, стоящее справа. Если выражение_1 возвращает ЛОЖЬ, то выражение_2 не будет выполнено. Если оба выражения возвращают ИСТИНУ, выполняется следующий набор команд. Если какое-то из выражений не истинно, приведенное выражение считает ложным в целом. Другими словами, все работает так:

```
если выражение_1 истинно и выражение_2 истинно, тогда выполнять...
```

Примечание переводчика: при работе с булевыми переменными ИСТИНА и ЛОЖЬ (True и False), bash ведет себя отлично от других языков программирования. В других языках 0 соответствует False (Ложь), а 1 – True (Истина). В bash все наоборот. Это связано с кодами завершения программ (см. ниже). Об этом следует всегда помнить при написании своих скриптов!

Пример использования:

```
#!/bin/bash
x=5
y=10
if [ «$x» -eq 5 ] && [ «$y» -eq 10 ]; then
echo «Оба условия верны»
else
echo «Условия не верны»
fi
```

Здесь мы обнаруживаем, что переменные x и y содержат именно те значения, которые мы проверям, поэтому проверяемые условия верны. Если вы измените значение с x = 5 на x = 12, а затем снова запустите программу, она выдаст фразу «Условия не верны».

Логическое ИЛИ используется аналогичным образом. Разница лишь в том, что оно начинает проверять ложность первого условия, если оно ложно, начинает выполняться следующий оператор:

```
выражение_1 || выражение_2
```

Данное выражение в псевдокоде выглядит так:

```
если выражение_1 истинно ИЛИ выражение_2 истинно, выполняем ...
```

Таким образом, любой последующий код будет выполняться, если хотя бы одно из выражений истинно:

```
#!/bin/bash
x=3
y=2
if [ «$x» -eq 5 ] || [ «$y» -eq 2 ]; then
echo «Одно из условий истинно»
else
echo «Ни одно из условий не является истинным»
fi
```

Здесь вы видите, что только одно из выражений истинно. Попробуйте изменить значение y и повторно запустите программу. Вы увидите сообщение, что ни одно из выражений не является истинным.

Аналогичная реализация условия с помощью оператора if будет большего размера, чем вариант с использованием логического И и ИЛИ, поскольку потребует дополнительного вложенного if. Ниже приведен код, реализующий тот же функционал, но с использованием оператора if:

```
#!/bin/bash
x=5
y=10
if [ «$x» -eq 5 ]; then
if [ «$y» -eq 10 ]; then
echo «Оба условия верны»
else
echo «Оба условия неверны»
fi
fi
```

Приведенный код менее нагляден для чтения и требует больших усилий

для написания. Но у вас остается шанс избавить себя от этих трудностей путем использования логических операторов И и ИЛИ.

ИСПОЛЬЗОВАНИЕ АРГУМЕНТОВ

Возможно, вы уже заметили, что большинство программ в Linux не интерактивны. Вы должны указать им какие-то параметры, в противном случае вы получите сообщение со списком возможных аргументов. Возьмем, к примеру, команду more. Если вы не укажете имя файла, она выдаст краткую справку по использованию программы. Также можно сделать так, чтобы ваши скрипты также могли принимать аргументы. Для этого вам нужно знать, что такое переменная вида $#. В ней содержится общее количество аргументов, переданных программе. Например, если вы запустите программу следующим образом:

```
$ что-то параметр
```

то значение переменной $# будет равно единице, потому что программе передан только один аргумент. Для двух аргументов ее значение будет равно двум и так далее. Также стоит знать о том, что каждый параметр командной строки (включая даже имя скрипта!!!) сохраняется в соответствующей переменной.

Так, имя нашей программы «что-то» будет сохранено в переменной $0. Аргумент программы – параметр сохранится в переменной $1. Вы можете использовать до 9 переменных, начиная с $0 (обозначающего имя скрипта), а затем $1-$9, обозначающие аргументы программы. Давайте посмотрим, как это работает:

```
#!/bin/bash
# скрипт, печатающий свои аргументы
# проверяем, переданы ли скрипту аргументы:

if [ «$#» -ne 1 ]; then
echo «корректный запуск программы: $0 <параметр>»
fi
echo «Переданный параметр - $1»
```

Приведенный скрипт ожидает один и только один аргумент для своего запуска. Если вы не укажете ему аргументов – будет выводиться справочная информация. В противном случае, если при запуске указан какой-то аргумент – он будет передан в наш скрипт, который выведет его на экран. Напоминаю, что $0 – это имя скрипта. Именно поэтому эта переменная используется в справочном сообщении. Последняя строка выводит переданный программе параметр – $1.

РАБОТА С ВРЕМЕННЫМИ ФАЙЛАМИ

Довольно часто вам будет необходимо создавать временные файлы. Обычно это файл, в котором хранятся какие-то используемые скриптом данные, либо что-то еще. Как только работа скрипта будет завершена, этот файл нужно удалить. При создании такого файла вы должны задать его имя. Проблема тут кроется в том, что файл, создаваемый вами, не должен случайно переписать уже существующий в той же директории, если их имена совпадут.

Для того, чтобы создать временный файл с гарантированно уникальным именем, вам нужно использовать символ «$$», либо как префикс, либо как суффикс к имени создаваемого файла. Предположим, вы хотите создать временный файл с именем hello. Возможно, что у пользователя, который работает с нашим скриптом, уже есть файл с таким именем. Создавая файл с именем hello.$$ или $$hello, вы создадите файл с уникальным именем. Например:

```
$ touch hello
$ ls
hello

$ touch hello.$$
$ ls
hello hello.689
```

Примерно так и будет выглядеть имя вашего временного файла.

Примечание переводчика: В переменной $$ обычно хранится следующий свободный PID. Именно поэтому использование такой переменной гарантирует уникальные имена для вновь создаваемых файлов.

КОДЫ ЗАВЕРШЕНИЯ ПРОГРАММ

Большинство программ возвращают в операционную систему какое-то число, показывающее, насколько удачно программа завершила свою работу. Например, man-страница grep говорит, что grep вернет 0, если заданный шаблон найден, и 1, если совпадений не найдено. Почему нас так волнуют эти коды завершения? По разным причинам. Допустим, мы хотим проверить – есть ли пользователь с данным именем в системе? Один из способов сделать – использовать команду вида: grep имя_пользователя /etc/passwd. Допустим, имя пользователя — vasya:

```
$ grep vasya /etc/passwd
$
```

Ничего не вывелось. Это означает, что grep не обнаружила заданного пользователя. Но для нас было бы значительно лучше получить сообщение об этом. Это как раз тот случай, когда нужно использовать код завершения программы. Он сохраняется в переменной с именем $?. Посмотрим на следующий фрагмент кода:

```
#!/bin/bash
# ищем пользователя vasya в /etc/passwd,
# весь вывод перенаправляем в /dev/null
grep vasya /etc/passwd > /dev/null 2>&1

# смотрим код завершения и действуем по обстоятельствам:
if [ «$? -eq 0 ]; then
echo «Пользователь vasya найден»
exit
else
echo «Пользователь vasya не найден»
fi
```

Теперь, когда вы запустите скрипт, он будет перехватывать и анализировать код завершения grep. Если он равен 0, значит пользователь найден и мы выводим соответствующее сообщение об ошибке. В противном случае скрипт напечатает, что пользователя найти не получилось. Это очень простой способ использования получаемого кода завершения программы. По мере практики вы сами будете

понимать, для решения какой задачи вам нужно использовать эти коды завершения.

Возможно вас озадачивает конструкция вида 2>&1, но все довольно просто. В Linux этими числами обозначаются дескрипторы файлов. 0 – стандартный ввод (по умолчанию, клавиатура), 1 стандартный вывод (по умолчанию, монитор) и 2 – вывод стандартных ошибок (по умолчанию, монитор).

Весь вывод команды идет в файл с дескриптором 1, любые ошибки отправляются в файл с дескриптором 2. Если вы не хотите, чтобы сообщения об ошибках появлялись на экране, просто перенаправьте его в /dev/null. Но это не прекратит вывод на экран обычной информации. Например, если у вас нет разрешения на чтение домашнего каталога другого пользователя, вы не сможете просмотреть список его содержимого:

```
$ ls /root
ls: /root: Permission
denied

$ ls /root 2> /dev/null
$
```

Как видите, во второй раз информация об ошибке не была напечатана. Все то же самое относится к другим программам и дескриптору 1. Если вы не хотите видеть нормальный выход из программы, то есть хотите, чтобы она работала молча, вы можете перенаправить в /dev/null и его. Теперь, если вы не хотите видеть вообще никакого вывода программы – добавьте в нее следующее:

```
$ ls /root > /dev/null
2>&1
```

Это означает, что программа будет отправлять свой вывод и ошибки, которые возникают в /dev/null, т. е. будет работать молча, что нам и нужно.

Примечание переводчика: на самом деле все работает так:

Конструкция вида 2>&1 перенаправляет вывод ошибок (дескриптор 2) на стандартный вывод (дескриптор 1). Знак «загогулины» – & – тут нужен для того, чтобы пояснить bash, что вы имеете в виду не файл с именем 1, а именно файл с дескриптором 1, т. е. стандартный вывод. Если вы укажете что-то вроде:

```
$ команда 2>1
```

то стандартный вывод ошибок пойдет в файл с именем 1. Конструкцией 2>&1 мы «сцепляем» вывод команды и вывод ошибок вместе. А первым перенаправлением (первым символом > в комментируемой команде) мы отправляем весь вывод команды в /dev/null. Чтобы дополнительно понять, как все работает, можете поэкспериментировать, убрав 2>&1 из команды и перезапустив ее.

А что если вы хотите, чтобы ваш скрипт тоже возвращал какой-нибудь код завершения при выходе? Команда exit может принимать один аргумент – тот самый код завершения. Обычно число 0 используется для обозначения успешного завершения работы. Число, отличное от нуля означает, что произошла какая-то ошибка. Какое число возвращать – решает сам программист. Посмотрим приведенный пример:

```
#!/bin/bash
if [ -f «/etc/passwd» ];
then
  echo «Файл passwd суще-
ствует»
  exit 0
  else
  echo «Нет такого файла»
  exit 1
  fi
```

Задавая значение кода завершения, вы делаете возможным для других скриптов, использующих ваш скрипт, анализировать результаты его работы.

ПЕРЕНОСИМОСТЬ ВАШИХ СКРИПТОВ НА BASH

При написании ваших собственных скриптов важно делать это так, чтобы они оставались переносимыми. Термин «переносимость» означает, что если ваш скрипт работает под Linux, то он должен работать в другой Unix-системе с малыми изменениями или вообще без них. Чтобы добиться этого, вы должны быть осторожны при вызове внешних программ. Разработчик должен при этом ответить на вопрос: «Будет ли эта программа доступна на другом варианте Unix?» (и что более важно – будет ли она там работать также, как на Linux – прим. перев.). Допустим, вы используете программу foo, которая на Linux работает аналогично echo, поэтому вместо echo вы используете ее. Но если ваш скрипт будет работать на других системах, где нет программы foo, он начнет выдавать сообщения об ошибках. Кроме того, имейте в виду, что разные версии bash могут иметь разные методы для одних и тех же операций. Например, конструкция VAR = $(ps) делает то же самое, что и VAR = `ps`, но на самом деле старые версии оболочек, например Bourne shell (sh), признают только последний синтаксис. Если вы собираетесь распространять свои скрипты, обязательно включайте текстовый файл README, который будет предупреждать пользователя о любых сюрпризах, в том числе и о том, что скрипт проверялся на такой-то версии bash. Желательно также указать, какие программы и библиотеки (и каких версий) будут нужны скрипту.

Примечание переводчика: Для проверки наличия в скрипте команд и функций специфичных для bash в ALT Linux есть пакет checkbashisms, который взят из пакета devscripts Debian.

ЗАКЛЮЧЕНИЕ

Пришла пора завершить это краткое введение в написание скриптов на bash. Однако ваше обучение этому умению еще не завершено. В тоже время, написанного вполне достаточно, чтобы вы могли модифицировать имеющиеся скрипты и писать собственные. Если вы действительно хотите стать мастером написания скриптов на bash, я рекомендую приобрести книгу «Learning the bash shell» (Изучение оболочки bash), 2-е издание издательства O'Reilly & Associates, Inc.

Скрипты на bash идеально подходят для повседневной работы по администрированию. Но если вы планируете что-то более серьезное, следует использовать гораздо более мощный язык, такой как C или Perl. Удачи!

Антон Чернышов
Linux-преподаватель R-Style
www.tux-the-penguin.blogspot.

ITOutsourcing

Доверьте свои сервера профессионалам

http://itoutsourcing.org.ua

+38 (050) 022-33-61

Видеонаблюдение в Linux

Ни для кого не секрет, что в наше время все следят друг за другом. Нет, не стоит пугаться, это не правительство и не инопланетяне (хотя, в общем не стоит исключать и таких вариантов). Все гораздо проще. Практически в каждой коммерческой фирме или госконторе, а зачастую, в школах и университетах, начальство устанавливает охранные системы. Под охранными системами обычно считают Охранно-Пожарные Системы(ОПС), Системы Контроля Доступа(СКД) и Системы видеонаблюдения. Вот про последние мы сегодня с вами и поговорим.

ВВЕДЕНИЕ

Все существующие системы видеонаблюдения можно условно разделить на два лагеря: построенные на базе видеорегистраторов и построенные на базе компьютера. Если с видеорегистратором все понятно (подключили к нему монитор пошире, камер побольше, посадили охранника посмышленее и система работает), то с системой на базе компьютера (видеосервера) все немного сложнее. Во-первых, нужно определиться с операционной системой, под которой всё это будет работать, во-вторых, нужно выбрать тот или иной программный продукт, который и будет записывать интересные моменты из жизни офиса ну и, в-третьих, нужно купить компьютер и платы видеозахвата (в случае использования аналоговых камер).

Казалось бы, что вариант с видеорегистратором куда проще и доступнее, и, возможно, для небольших офисов/магазинов – самый лучший вариант, но что делать, если нужно организовать систему по принципу – сидит охранник и мониторит 30 камер в 30-ти помещениях большого торгового центра, сидит начальник охраны и смотрит, что же делают охранники в его отсутствие, директор наблюдает за тем, как работают его подчиненные (включая начальника охраны и секретаршу Машу которая любит сидеть в «Одноклассниках»), а тем временем, Генеральный Директор сидит дома и время от времени смотрит как работают люди в 4-ех его магазинах и 5-ти офисах, ну и наконец за всеми этими людьми следит Самый Главный Человек… – Системный Администратор. Тут уже одним простым видеорегистратором не обойтись, нужно ставить (и не один) видеосервер, способный отдавать картинку с камер по сети с разграничением прав и пользователей.

ВЫБИРАЕМ

Во-первых, нужно выбрать ОС. Как видно из названия статьи, я рекомендую использовать Linux, потому что:

1. Это выгодно. Не нужно платить за лицензию на каждый сервер и удаленный компьютер наблюдения.

2. Это надежно. Правильно настроенный сервер на Linux'е будет служить вам верой и правдой не один год.

Во-вторых, нужно выбрать программный продукт для регистрации видео.

Здесь кроется еще один плюс выбранной операционной системы. Дело в том, что многие из существующих систем видеонаблюдения под линукс распространяются бесплатно (будь то ZoneMinder, Motion или LinuxDVR (до 4-ех камер в системе)), в то время как аналогичные системы под «другие ОС» стоят порядка 50-ти тысяч рублей (в зависимости от предполагаемого количества камер и количества компьютеров и серверов работающих в системе).

ИСТОРИЯ УСПЕХА

Для себя я выбрал Motion.

Motion представляет из себя простую консольную программку со встроенным веб-сервером, выводящим по указанному порту в формате MJPEG видеопоток с устройства video4linux(/dev/video*). Кроме того, motion имеет встроенную систему регистрации движений в поле зрения камеры. Т.е. на диск пишется не все видео в режиме 24/7, а только тогда, когда камера «видит», что что-то происходит, что существенно экономит место на диске и увеличивает максимальное время записи полезной информации на диск (впрочем, если нужно, то можно включить и постоянную запись).

НАСТРОЙКА

Настройка motion происходит через конфигурационный файл, который по умолчанию расположен здесь: /etc/motion/motion.conf

Про все пункты в этом файле вы можете прочитать в команде man motion или же найти в гугле (благо в интернете информации по этому пакету достаточно), но пару пунктов из этого файлика я бы хотел выделить.

```
   control_port 8080
   control_localhost off
   control_html_output on
   control_authentication
user:password
```

В этих четырех строках описан способ доступа к Web GUI настройки motion. В первой строке указан порт по которому нужно будет постучаться чтобы попасть в GUI, во второй строке разрешается/запрещается доступ к админке с адресов, отличных от localhost'а(127.0.0.1), ну и в четвертой – параметры авторизации (логин:пароль) для доступа к админке.

В админке есть все пункты настройки motion (даже те, которые не прописаны в конфигурационном файле – стоят значения по умолчанию), с их кратким описанием. Кроме того, конфигурационные файлы очень удобно разбивать на «модули». Т.е. основные настройки для всей системы хранить в файле /etc/motion/motion.conf, а настройки для каждой камеры хранить в отдельном файле, который потом можно будет подключить к основному файлу командой thread.

Вот так у меня выглядит конфигурационный файл для системы с четыремя камерами:

/etc/motion/motion.conf:

```
   framerate 15
   ffmpeg_cap_new on
   ffmpeg_cap_motion off
   ffmpeg_timelapse 0
   ffmpeg_timelapse_mode
daily
   ffmpeg_bps 400000
   ffmpeg_variable_bitrate 0
   ffmpeg_video_codec ffv1
#Вот тут мы выбираем кодек,
которым мы будем сохранять
видео
   ffmpeg_deinterlace off
   max_mpeg_time 600
   text_right %Y-%m-%d\n%T
   text_changes off
   text_event %Y%m%d%H%M%S
   text_double off
   snapshot_filename
%v-%Y%m%d%H%M%S-snapshot
   jpeg_filename
```

```
%v-%Y%m%d%H%M%S-%q
   movie_filename
%Y.%m.%d/%H:%M:%S
   timelapse_filename %Y%m%d-
timelapse
   output_normal off
   webcam_quality 50
   webcam_motion off
   webcam_maxrate 50
   webcam_localhost off
   webcam_limit 0
   control_port 8080
   control_localhost off
   control_html_output on
   control_authentication
user:password
   quiet on
   thread /etc/motion/
thread1.conf
   thread /etc/motion/
thread2.conf
   thread /etc/motion/
thread3.conf
   thread /etc/motion/
thread4.conf
```

Выделил 4 файла настроек для каждой камеры. Вот один из них:

/etc/motion/thread1.conf:

```
   #Выбираем видеосутройство
(карта видеозахвата)
   videodevice /dev/video0
   #Выбираем номер канала
   input 1
   #Выбираем normid( PAL-DK,
SECAM, etc. в зависимости
от вашей камеры )
   norm 3
   #Вот тут самое интересное
– порт, на который будет
транслироваться видео поток
в формате mjpeg
   webcam_port 8081
   #Текст, который будет
выводиться в левом нижнем
углу картинки (Сюда можно
название камеры написать)
   text_left = «Camera
Color»
   #Директория, куда будет
сохраняться записанный ви-
деоматериал
   target_dir /opt/lampp/
htdocs/bigbrother/Camera 1/
```

ИНТЕРФЕЙС

Систему мы настроили и запустили. Камеры регистрируют любое движение и записывают все происходящее в специально подготовленную директорию. Остался один маленький вопрос. Куда смотреть, чтобы в режиме реального времени следить за происходящим то?

Помните строчку в конфигах webcam_port 8081? Вот если в адресной строке браузера зайти по адресу http://localhost:8081, то мы и увидим картинку с камеры в режиме реалтайм.

Как видите ни о каком интерфейсе разговора нет. Можно конечно оставить все как есть. И для каждой камеры открывать отдельное окно браузера со своим портом. Но ведь гораздо приятней (да и удобней) видеть все в одном окне, как тут:

Тут то нам на помощь и придут базовые знания в html (а на HTML5, CSS3, JQuery, etc. можно нарисовать интерфейс, который не будет уступать небраузерным клиентам). Делаем табличку, втыкаем туда 4 тега <img>, в качестве параметра src указываем http://localhost:8081, и любой бразуер будет показывать видеопоток в формате MJPEG.

Да, такая система требует небольших познаний в html, но в итоге получается очень гибкой и легко настраиваемой. Вот код моего рабочего примера на 4 камеры(см. скрин выше):

```
   <html>
   <head>
   <title>BigBrother</title>
   <META content=»text/html;
charset=utf-8 http-equiv=Content-
Type>
   <style></style>
   </head>
   <body bgcolor=gray>
   <center>
   <a href=»bigbrother»>Перейти к
архиву</a> | <a href=#>Справка</a> |
<a href=#>О программе</a>
   <br><br>
   <table border=1 bordercolor=green
width=80%>
   <tr>
   <td width=50%>
   <img
src=»http://192.168.10.56:8081»
width=100%>
   </td>
   <td width=50%>
   <img
src=»http://192.168.10.56:8082»
width=100%>
   </td>
   </tr>
   <tr>
   <td width=50%>
   <img
src=»http://192.168.10.56:8083»
width=100%>
   </td>
   <td width=50%>
   <img
src=»http://192.168.10.56:8084»
width=100%>
```

```
   </td>
   </tr>
   </table>
   </center>
   </body>
   </html>
```

Кроме того, если все камеры подключены к серверу, а наблюдение ведется с удаленного компьютера, не совсем удобно добираться к записанному видеоматериалу. Нужно либо идти к серверу с флешкой (что совсем неправильно), либо поднимать ftp-сервер с авторизацией, либо написать еще один небольшой скрипт на php, который будет следить за появление новых роликов и отдавать их клиентам. Тут уже решать Вам самим. Для себя я выбрал последний вариант, потому-что его можно безболезненно встроить в интерфейс, он прост в понимании для охранников и позволяет им не отрываясь от своего любимого дела (наблюдением за нарушителями на охраняемой ими территории), не закрывая интерфейса программы, скачивать с сервера файл за тот или иной период записи.

ВЫВОДЫ

Вообще, система motion очень удобная в использовании и простая в настройке. Но пользоваться ей без дополнительных средств слегка неудобно. Поэтому я рекомендую использовать связку motion+xampp+html, такой способ позволит сделать доступ к системе с любой машины в сети на которой установлен обычный браузер. В принципе эту систему можно расширять сколько угодно долго. Добавить модули поворотных устройств (работающих по протоколам PELCO, PANASONIC и прочих – благо протоколы простые в понимании и подробно описаны в интернете), интегрировать с ОПС и СКД (все опять же упирается в протоколы передачи данных) и т.д. оно и понятно – к Web интерфейсу можно дописать чего угодно.

Михаил Павлов
www.todeus.ru/

Почта БЕЗ спама на UBUNTU 10.10

Установим
Веб сервер: Nginx v0.8.53/Uwsgi v0.9.6.5
БД: MySQL v5.1.49
Почтовый сервер: Postfix v2.7.1
DNS сервер: Dnsmasq 2.55
Фильтр: MailScanner v4.81.4
Фронт энд: Baruwa v1

Настраиваем сеть:
/etc/network/interfaces

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).
# The loopback network interface
auto lo
iface lo inet loopback
# The primary network interface
auto eth0
iface eth0 inet static
address 192.168.0.100
netmask 255.255.255.0
network 192.168.0.0
broadcast 192.168.0.255
gateway 192.168.0.1
```

Перезагружаем сеть:
/etc/init.d/networking restart

Настраиваем хосты:
/etc/hosts

```
127.0.0.1 localhost.localdomain localhost
192.168.0.100 server1.example.com server1
# The following lines are desirable for IPv6 capable hosts
::1 localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
ff02::3 ip6-allhosts
```

Выполним:
```
echo server1.example.com > /etc/hostname
reboot now
```

После выполним:
```
hostname
hostname -f
```

Выберем шэлл по умолчанию:
```
dpkg-reconfigure dash
```

На вопрос отвечаем НЕТ:
```
Install dash as /bin/sh? <-- No
```

Установим требующиеся пакеты:
```
apt-get install binutils cpp fetchmail flex gcc libarchive-zip-perl libc6-dev libcompress-zlib-perl libdb4.6-dev libpcre3 libpopt-dev lynx m4 make ncftp nmap openssl perl perl-modules unzip zip zlib1g-dev autoconf automake1.9 libtool bison autotools-dev g++ build-essential telnet wget gawk
```

Установим Dnsmasq:
```
aptitude install dnsmasq
```

Редактируем конфиг:
/etc/dnsmasq.conf

```
listen-address=127.0.0.1
```

Редактируем resolv.conf
/etc/resolv.conf

```
nameserver 127.0.0.1
```

Установим Mysql:
```
aptitude install mysql-client mysql-server libdbd-mysql-perl
```

При установке вам будет задано два вопроса, нужно указать пароль для администратора:

New password for the MySQL «root» user: <-- Пароль
Repeat password for the MySQL «root» user: <-- Повтор пароля

Установим Postfix:
```
aptitude install postfix postfix-mysql postfix-doc procmail
```

В процессе установки будет задано два вопроса:
```
General type of mail configuration: -> Internet Site
System mail name: -> server1.example.com
```

Остановим Postfix:
```
postfix stop
```

Редактируем файл master.cf:
/etc/postfix/master.cf

```
pickup fifo n - - 60 1 pickup
-o content_filter=
-o receive_override_options=no_header_body_checks
```

Редактируем файл postfix.sh:
/usr/src/postfix.sh

```
#!/bin/sh
postconf -e «alias_maps = hash:/etc/aliases»
newaliases
postconf -e «myorigin = domain.tld»
postconf -e «myhostname = server1.domain.tld»
postconf -e «mynetworks = 127.0.0.0/8, 192.168.0.0/24
password = ПАРОЛЬ
dbname = baruwa
query = select concat('smtp:[', mail_hosts.address, ']', ':', port) 'transport' from mail_hosts, user_addresses where user_addresses.address = '%s' AND user_addresses.id = mail_hosts.useraddress_id;
hosts = 127.0.0.1
EOF
```

Установим права:
```
chmod +x /usr/src/postfix.sh
```

И запустим скрипт:
```
./usr/src/postfix.sh
```

Редактируем:
/etc/postfix/main.cf

```
verify_recipient = reject_unknown_recipient_domain, reject_unverified_recipient
look_ahead = check_recipient_access hash:/etc/postfix/access
unverified_recipient_reject_code = 550
address_verify_map = btree:/var/lib/postfix/verify
```

Добавьте ваши smtpd_restriction_classes:
```
verify_recipient, look_ahead
```

Добавьте это к smptd_recipient_restrictions:

```
smtpd_recipient_restrictions = permit_mynetworks, permit_sasl_authenticated, reject_unauth_destination, look_ahead, whitelist_policy, grey_policy, rbl_policy, spf_policy, permit
```

Выполним:

```
touch /etc/postfix/access
```

Добавьте ваши домены:

```
domainA.com verify_recipient
domainB.com verify_recipient
```

Выполним:

```
postmap /etc/postfix/access
```

Проверим main.cf:

```
less /etc/postfix/main.cf
```

Запускаем Postfix:

```
postfix start
```

Проверим, отвечает ли Postfix:

```
telnet 127.0.0.1 25
```

Должны увидеть:

```
220 [yourFQDNhere] ESMTP Postfix (Ubuntu)
```

Установим MailScanner, Apparmor, Clamav, DCC, Pyzor, Razor and Spamassassin:

```
cd /usr/src
wget http://http.us.debian.org/debian/pool/main/libt/libtool/libltdl3_1.5.26-4+lenny1_i386.deb
dpkg -i libltdl*
aptitude install razor pyzor clamav-daemon libclamav6 apparmor
```

Настроим Apparmor для Clamav:

```
usermod -a -G www-data clamav
```

Редактируем профиль для Clamav:

```
/etc/apparmor.d/usr.sbin.clamd
```

Добавим папку «Входящие» к списку папок:

```
/usr/sbin/clamd {
#clamav
/var/spool/MailScanner/** rw,
/var/spool/MailScanner/incoming/** rw,
}
```

Перезагрузим apparmor:

```
/etc/init.d/apparmor reload
```

Настройка DCC:

```
cd /tmp
wget http://ppa.launchpad.net/jonasped/ppa/ubuntu/pool/main/d/dcc/dcc-common_1.3.130-0ubuntu1~ppa1~karmic1_i386.deb && dpkg -i dcc-common_1.3.130-0ubuntu1~ppa1~karmic1_i386.deb
wget http://ppa.launchpad.net/jonasped/ppa/ubuntu/pool/main/d/dcc/dcc-client_1.3.130-0ubuntu1~ppa1~karmic1_i386.deb && dpkg -i dcc-client_1.3.130-0ubuntu1~ppa1~karmic1_i386.deb
```

Проверим установку:

```
cdcc info
```

Настроим Pyzor:

```
#!/usr/bin/python -Wignore::DeprecationWarning
```

Выполним:

```
mkdir /var/lib/MailScanner
pyzor –homedir=/var/lib/MailScanner discover
pyzor ping
```

Настроим Razor:

```
cd && rm /etc/razor/razor-agent.conf
mkdir /var/lib/MailScanner/.razor
razor-admin -home=/var/lib/MailScanner/.razor -create
razor-admin -home=/var/lib/MailScanner/.razor -discover
razor-admin -home=/var/lib/MailScanner/.razor -register
```

Редактируем:

```
/var/lib/MailScanner/.razor/razor-agent.conf
debuglevel = 0
razorhome = /var/lib/MailScanner/.razor/
```

Установим зависимости:

```
aptitude install libconvert-tnef-perl libdbd-sqlite3-perl libfilesys-df-perl libmailtools-perl libmime-tools-perl libmime-perl libnet-cidr-perl libsys-syslog-perl libio-stringy-perl libfile-temp-perl libole-storage-lite-perl libarchive-zip-perl libsys-hostname-long-perl libnet-cidr-lite-perl libhtml-parser-perl libdb-file-lock-perl libnet-dns-perl libncurses5-dev libdigest-hmac-perl libdigest-sha1-perl libnet-ip-perl liburi-perl libfile-spec-perl spamassassin libnet-ident-perl libmail-spf-query-perl libmail-dkim-perl dnsutils libio-socket-ssl-perl
```

Настроим SpamAssassin:

Отключаем файл конфигурации по умолчанию:

```
mv /etc/spamassassin/local.cf /etc/spamassassin/local.cf.disabled
```

Восстановим резервную копию файла конфигурации SpamAssassin из MailScanner:

```
cp /opt/MailScanner/etc/spam.assassin.prefs.conf /opt/MailScanner/etc/spam.assassin.prefs.conf.back
```

Настроим SpamAssassin SQL Bayes:

```
SpamAssassin Bayes Database Name: sa_bayes
SpamAssassin Bayes Database UserName: sa_user
SpamAssassin Bayes Database Password: sa_password
```

Создадим базу данных:

```
mysql -u root -p
mysql> create database sa_bayes;
mysql> GRANT ALL ON sa_bayes.* TO sa_user@localhost IDENTIFIED BY ‘sa_password’;
mysql> flush privileges;
```

Импортируем базу:

```
mysql -u sa_user -p sa_bayes < /usr/share/doc/spamassassin/sql/bayes_mysql.sql
```

Включаем DCC, редактируем v310.pre:

```
/etc/spamassassin/v310.pre
loadplugin Mail::SpamAssassin::Plugin::DCC
```

Создадим папку:

```
mkdir /var/www/.spamassassin
```

Редактируем файл spam.assassin.prefs.conf:

```
/opt/MailScanner/etc/spam.assassin.prefs.conf

#pyzor
use_pyzor 1
pyzor_options –homedir /var/lib/MailScanner/
pyzor_add_header 1
#razor
use_razor2 1
razor_config /var/lib/MailScanner/.razor/razor-
```

```
agent.conf
```

Исправим путь к DCC:

```
cc_path /usr/bin/dccproc
```

Обновим в начале:

```
bayes_ignore_header
X-YOURDOMAIN-COM-MailScanner
bayes_ignore_header
X-YOURDOMAIN-COM-MailScanner-
SpamCheck
bayes_ignore_header
X-YOURDOMAIN-COM-MailScanner-
SpamScore
bayes_ignore_header
X-YOURDOMAIN-COM-MailScanner-
Information
#use_auto_whitelist 0
```

Добавьте строки подключения к бд:

```
bayes_store_module Mail::Spa
mAssassin::BayesStore::SQL
bayes_sql_dsn DBI:mysql:sa_
bayes:localhost
bayes_sql_username sa_user
bayes_sql_password sa_
password
bayes_sql_override_username
root
```

Редактируем v310.pre:

```
#loadplugin Mail::SpamAssass
in::Plugin::DomainKeys
```

Добавим задание в крон:

```
30 01 * * * /usr/bin/sa-
learn –force-expire –sync -p
/opt/MailScanner/etc/spam.
assassin.prefs.conf
```

Установим пакеты:

```
perl -MCPAN -e shell
install IP::Country::Fast
install Encode::Detect
install Crypt::OpenSSL::RSA
```

Установим права:

```
chown -R postfix:www-data /
var/spool/postfix/hold
chmod -R ug+rwx /var/spool/
postfix/hold
```

Проверим установку:

```
spamassassin -x -D -p /opt/
MailScanner/etc/spam.assassin.
prefs.conf –lint
```

Проверьте эти строки:

```
debug: bayes: Database
connection established
debug: bayes: found bayes db
version 3
debug: bayes: Using userid:
2
```

Скачиваем и устанавливаем MailScanner:

```
cd /usr/src && wget
http://www.mailscanner.info/
files/4/tar/MailScanner-
install-4.81.4-1.tar.gz
tar xvfz MailScanner-
install-4.81.4-1.tar.gz && cd
MailScanner-install-4.81.4
./install.sh
```

Добавим задания в крон:

```
37 5 * * * /opt/MailScanner/
bin/update_phishing_sites
07 * * * * /opt/MailScanner/
bin/update_bad_phishing_sites
58 23 * * * /opt/
MailScanner/bin/clean.
quarantine
42 * * * * /opt/MailScanner/
bin/update_virus_scanners
3,23,43 * * * * /opt/
MailScanner/bin/check_
mailscanner
```

Настроим MailScanner:

```
mkdir /var/spool/
MailScanner/spamassassin
```

Сделаем бэкап файла конфигурации MailScanner.conf:

```
cp /opt/MailScanner/
etc/MailScanner.conf /opt/
MailScanner/etc/MailScanner.
conf.dist
```

Редактируем:

```
/opt/MailScanner/etc/
MailScanner.conf
```

Измените следующие параметры в MailScanner.conf со следующим сценарием:

```
/usr/src/mailscanner.sh
```

Установим права:

```
chmod +x mailscanner.sh
```

И запускаем:

```
./usr/src/mailscanner.sh

sed -i «/^%org-name% =/
c\%org-name% =orgname» /opt/
MailScanner/etc/MailScanner.
conf
sed -i «/^%org-long-
name% =/ c\%org-long-name% =
longorgname» /opt/MailScanner/
etc/MailScanner.conf
sed -i «/^%web-site% =/
c\%web-site% = www.domain.
tld» /opt/MailScanner/etc/
MailScanner.conf
sed -i «/^Run As User =/ c\
Run As User = postfix» /opt/
MailScanner/etc/MailScanner.
conf
.....
sed -i «/^Non Spam Actions
=/ c\Non Spam Actions
= deliver store» /opt/
MailScanner/etc/MailScanner.
conf
sed -i «/^SpamAssassin User
```

Редактируем скрипт запуска:

```
/etc/init.d/mailscanner
```

Установим права:

```
chmod +x /etc/init.d/
mailscanner

#! /bin/sh
### BEGIN INIT INFO
# Provides: MailScanner
daemon
# Required-Start: $local_fs
$remote_fs
# Required-Stop: $local_fs
$remote_fs
# Default-Start: 2 3 4 5
# Default-Stop: 0 1 6
# Short-Description:
Controls mailscanner instances
# Description: MailScanner
is a queue-based spam/virus
filter
### END INIT INFO
# Author: Simon Walter
# PATH should only include
/usr/* if it runs after the
mountnfs.sh script
PATH=/usr/sbin:/usr/bin:/
bin:/sbin:/opt/MailScanner/bin
DESC=»mail spam/virus
scanner»
NAME=MailScanner
PNAME=mailscanner
DAEMON=/opt/MailScanner/
bin/$NAME
STARTAS=MailScanner
SCRIPTNAME=/etc/
init.d/$PNAME
CONFFILE=/opt/MailScanner/
etc/MailScanner.conf
# Exit if the package is not
installed
[ -x «$DAEMON» ] || exit 0
run_nice=0
stopped_lockfile=/var/lock/
subsys/MailScanner.off
# Read configuration variable
file if it is present
[ -r /etc/default/$PNAME ]
&& . /etc/default/$PNAME
# Load the VERBOSE setting
and other rcS variables
. /lib/init/vars.sh
# Define LSB log_* functions.
# Depend on lsb-base (>=
3.0-6) to ensure that this file
is present.
```

```
   . /lib/lsb/init-functions
   # sanity check for
permissions
   fail()
   {
   echo >&2 «$0: $1
   exit
   1
   }
   check_dir()
   {
   if [ ! -d $1 ]; then
   mkdir -p «$1 || \
   fail «directory $1: does not
exist and cannot be created»
   fi
   actual=»$(stat -c %U $1)»
   if [ «$actual» != «$2» ];
then
   chown -R «$2 «$1 || \
   fail «directory $1: wrong
owner (expected $2 but is
$actual)»
   fi
   actual=»$(stat -c %G $1)»
   if [ «$actual» != «$3» ];
then
   chgrp -R «$3 «$1 || \
   fail «directory $1: wrong
group (expected $3 but is
$actual)»
   fi
   }
   user=$(echo $(awk -F= '/^Run
As User/ {print $2; exit}'
$CONFFILE))
   group=$(echo $(awk -F=
'/^Run As Group/ {print $2;
exit}' $CONFFILE))
   check_dir /var/spool/
MailScanner ${user:-postfix}
${group:-www-data}
   check_dir /var/lib/
MailScanner ${user:-postfix}
${group:-www-data}
   check_dir /var/run/
MailScanner ${user:-postfix}
${group:-www-data}
   check_dir /var/lock/subsys
${user:-root} ${group:-root}
#Required to Create Folder
   check_dir /var/lock/subsys/
MailScanner ${user:-postfix}
${group:-www-data}
   #
   # Function that starts the
daemon/service
   #
   do_start()
   {
   # Return
   # 0 if daemon has been
started
   # 1 if daemon was already
running
   # 2 if daemon could not be
started
   start-stop-daemon -start -
quiet -startas $STARTAS -name
$NAME -test > /dev/null \
   || return 1
   start-stop-daemon -start -
quiet -nicelevel $run_nice -
chuid postfix:www-data -exec
$DAEMON -name $NAME - $DAEMON_
ARGS \
   || return 2
   # Add code here, if
necessary, that waits for the
process to be ready
   # to handle requests from
services started subsequently
which depend
   # on this one. As a last
resort, sleep for some time.
   # Set lockfile to inform
cronjobs about the running
daemon
   RETVAL=»$?»
   if [ $RETVAL -eq 0 ]; then
   touch /var/lock/subsys/
mailscanner
   rm -f $stopped_lockfile
   fi
   if [ $RETVAL -eq 0 ]; then
   echo «MailScanner Started»
   fi
   }
   #
   # Function that stops the
daemon/service
   #
   do_stop()
   {
   # Return
   # 0 if daemon has been
stopped
   # 1 if daemon was already
stopped
   # 2 if daemon could not be
stopped
   # other if a failure
occurred
   start-stop-daemon -stop -
retry=TERM/30 -name $NAME
   RETVAL=»$?»
   [ «$RETVAL» = 2 ] && return
2
   # Remove lockfile for
cronjobs
   if [ $RETVAL -eq 0 ]; then
   rm -f /var/lock/subsys/
mailscanner
   touch $stopped_lockfile
   fi
   if [ $RETVAL -eq 0 ]; then
   echo «MailScanner Stopped»
   fi
   }
   #
   # Function that sends a
SIGHUP to the daemon/service
   #
   do_reload() {
   start-stop-daemon -stop -
signal 1 -quiet -name $NAME
   return 0
   }
   case «$1 in
   start)
   [ «$VERBOSE» != no ] && log_
daemon_msg «Starting $DESC»
«$NAME»
   do_start
   case «$?» in
   0|1) [ «$VERBOSE» != no ] &&
log_end_msg 0 ;;
   2) [ «$VERBOSE» != no ] &&
log_end_msg 1 ;;
   esac
   ;;
   stop)
   [ «$VERBOSE» != no ] && log_
daemon_msg «Stopping $DESC»
«$NAME»
   do_stop
   case «$?» in
   0|1) [ «$VERBOSE» != no ] &&
log_end_msg 0 ;;
   2) [ «$VERBOSE» != no ] &&
log_end_msg 1 ;;
   esac
   ;;
   restart|force-reload)
   #
   # If the «reload» option is
implemented then remove the
   # 'force-reload' alias
   #
   log_daemon_msg «Restarting
$DESC» «$NAME»
   do_stop
   case «$?» in
   0|1)
   do_start
   case «$?» in
   0) log_end_msg 0 ;;
   1) log_end_msg 1 ;; # Old
process is still running
   *) log_end_msg 1 ;; # Failed
to start
   esac
   ;;
   *)
   # Failed to stop
   log_end_msg 1
   ;;
   esac
   ;;
   *)
   echo «Usage: $SCRIPTNAME
{start|stop|restart|force-
reload}» >&2
   exit 3
   ;;
   esac
   exit 0
```

Создадим симлинки:

```
   chmod 755 /etc/init.d/
mailscanner
   ln -s ../init.d/mailscanner
/etc/rc0.d/K20mailscanner
   ln -s ../init.d/mailscanner
/etc/rc1.d/K20mailscanner
   ln -s ../init.d/mailscanner
/etc/rc2.d/S20mailscanner
   ln -s ../init.d/mailscanner
/etc/rc3.d/S20mailscanner
   ln -s ../init.d/mailscanner
/etc/rc4.d/S20mailscanner
   ln -s ../init.d/mailscanner
```

```
/etc/rc5.d/S20mailscanner
   ln -s ../init.d/mailscanner
/etc/rc6.d/K20mailscanner
   ln -s /opt/MailScanner/bin/
Quick.Peek /usr/sbin/Quick.
Peek
```

Запускаем:

```
   /etc/init.d/mailscanner
start
   /etc/init.d/postfix start
```

Проверим логи:

```
   tail -f /var/log/mail.log
```

Установим Baruwa:

```
   cd /usr/src
   wget baruwa_1.0.0-3_all.deb
Updated 120310
   wget baruwa-doc_1.0.0-3_all.
deb
   gdebi baruwa*.deb
```

Редактируем:

```
   /usr/share/pyshared/baruwa/
settings.py

   QUARANTINE_REPORT_HOSTURL =
'http://baruwa-alpha.local'
```

Редактируем:

```
   /opt/MailScanner/etc/
MailScanner.conf

   Always Looked Up Last =
&BaruwaSQL
   Is Definitely Not Spam =
&BaruwaWhitelist
   Is Definitely Spam =
&BaruwaBlacklist
   Required SpamAssassin Score
= &BaruwaLowScore
   High SpamAssassin Score =
&BaruwaHighScore
```

Установим Nginx с Uwsgi:

```
   wget https://launchpad.
net/~chris-lea/+archive/nginx-
devel/+files/nginx_0.8.53-
1chl1%7Emaverick1_i386.deb
   wget https://launchpad.
net/~chris-lea/+archive/
uwsgi/+build/2038401/+files/
uwsgi-common_0.9.6.5-
1chl1~maverick1_i386.deb
   wget https://launchpad.
net/~chris-lea/+archive/
uwsgi/+build/2038401/+files/
uwsgi-extra_0.9.6.5-
1chl1~maverick1_i386.deb
   wget https://launchpad.
net/~chris-lea/+archive/
uwsgi/+build/2038401/+files/
uwsgi-python2.6_0.9.6.5-
1chl1~maverick1_i386.deb
   wget https://launchpad.
net/~chris-lea/+archive/
uwsgi/+build/2038401/+files/
uwsgi_0.9.6.5-
1chl1%7Emaverick1_all.deb
   apt-get install libsctp1
   dpkg -i nginx*
   dpkg -i uwsgi*
```

Редактируем:

```
   /etc/uwsgi/uwsgi-python2.6/
baruwa.ini

   [uwsgi]
   socket = /var/run/uwsgi/
uwsgi-python2.6/baruwa/baruwa.
sock
   master = true
   processes = 2
   env = DJANGO_SETTINGS_
MODULE=baruwa.settings
   module = django.core.
handlers.wsgi:WSGIHandler()
```

Редактируем:

```
   /etc/nginx/sites-available/
baruwa.conf

   server {
   listen 80;
   server_name example.com;
   root /usr/share/pyshared/
baruwa;
   #main access log
   access_log /var/log/nginx/
access.log;
   #main error log
   error_log /var/log/nginx/
error.log;
   location /static {
   root /usr/share/pyshared/
baruwa/static/;
   }
   # static resources
   location ~* ^.+\.(html|jpg|
jpeg|gif|png|ico|css|zip|tgz|
gz|rar|bz2|doc|xls|exe|pdf|pp
t|txt|tar|mid|midi|wav|bmp|rt
f|js)$
   {
   expires 30d;
   break;
   }
   location / {
   uwsgi_pass unix:///var/run/
uwsgi/uwsgi-python2.6/baruwa/
baruwa.sock;
   include uwsgi_params;
   }
   }
```

Удаляем виртуальные хосты, созданные по умолчанию, и копируем uwsgi_params:

```
   rm -r /etc/nginx/sites-
enabled/default
   cp /usr/share/doc/uwsgi-
extra/nginx/uwsgi_params /etc/
nginx/uwsgi_params
   ln -s /etc/nginx/sites-
available/baruwa.conf /etc/
nginx/sites-enabled/baruwa.
conf
```

Перезапускаем:

```
   /etc/init.d/uwsgi-python2.6
restart && /etc/init.d/nginx
restart
```

Создадим симлинки:

```
   ln -s /usr/share/pyshared/
baruwa/manage.py /usr/bin/
manage.py
   chmod +x /usr/bin/manage.py
```

Добавим задания в крон:

```
   @daily manage.py
cleanquarantine #Clean
quarantine
   * * * * 5 manage.py
sendquarantinereports #Send
quarantine reports
   @monthly manage.py dbclean
#Clean maillog
   @weekly manage.
py updatesarules #Update
spamassassin rules
   @daily manage.py
sendpdfreports #Send PDF
Reports
```

Запустим MailScanner:

```
   /etc/init.d/mailscanner
start
```

Установим и настроим SPF:

```
   cd /usr/src
   wget http://www.openspf.
org/blobs/postfix-policyd-spf-
perl-2.007.tar.gz
   tar xvfz postfix-policyd-spf-
perl-2.007.tar.gz
   cd postfix-policyd-spf-
perl-2.007
   cp postfix-policyd-spf-perl
/usr/lib/postfix/policyd-spf-
perl
```

Редактируем:

```
   /etc/postfix/master.cf
```

Добавим в конец файла:

```
   policy unix - n n - - spawn
   user=nobody argv=/usr/bin/
perl /usr/lib/postfix/policyd-
spf-perl
```

Перезагрузим postfix:

```
   /etc/init.d/postfix restart
```

Григор Андрей
www.григор.рф

Установка и настройка Принт-Сервера на основе Linux Ubuntu Server 10.04.1 LTS

Устанавливаем ОС Linux Ubuntu Server 10.04.1 LTS (LongTimeSupport) по причине долгой поддержки со стороны Canonical (создателя дистрибутива) – 5лет.

ЗАМЕТКИ:

- Установка подразумевает постоянный доступ к сети Интернет.
- Программу «aptitude» можно заменить программой «apt-get».
- Лучше ставить на чистый диск.
- Если использовать дистрибутив «10.04 LTS», вылетает ошибка и просит вставить диск, при использовании поправленного дистрибутива «10.04.1 LTS» ошибка ушла, по всей видимости это происходит из-за обновленных файлов в репозиториях.
- При установке системы на любом этапе можно нажать «Назад» и выбрать любой раздел для настройки, также можно всегда вызвать справку клавишей F1
- Если настраивать вручную через HPLIP, то находим файл « /etc/hp/hpip.conf » и исправляем значение параметра: «gui-build = no» для того, чтобы не требовал GUI.

ПОЛЕЗНЫЕ ССЫЛКИ ПО ТЕМЕ:

HPLIP — www.hplipopensource.com
CUPS – www.cups.org
Cups Документация - www.cups.org
Детально о cupsd.conf - www.cups.org
Русский ман - www.wiki.archlinux.org
Загрузка PPD файлов - www.openprinting.org
Специальный драйвер - www.foo2zjs.rkkda.com
Samba на русском — www.smb-conf.ru

УСТАНОВКА ОС

1. Загружаемся с LiveCD и начинаем установку сервера.
2. Выбираем в автозапуске «Русский язык».
3. Установить Ubuntu server.
4. Регион — «Украина».
5. Не определять раскладку автоматически.
6. Раскладка клавиатуры – «США».
7. В большинстве случаев IP адрес получается автоматом по средством DHCP.
8. Имя сервера — «SERVER-PRINT».
9. Time zone correct – выбираем «Нет».
10. Выбираем из списка «KIEV».
11. Разметка диска, файловая система «ext4», метод разбиения разделов (разметка диска приведена как пример):

- swp = 2xRAM (вдвое больше чем ОЗУ и не меньше 1Gb);
- / = Остаток;
- /home = 4Gb минимум;
- /usr = 10Gb минимум.

12. Начнётся установка базовой системы, необходимо соединение с интернетом (при использовании 10.04, попросит вставить диск с дистрибутивом из-за отсутствия файлов).
13. Заводим пользователя «administrator» и ставим стандартный пароль.
14. Не шифровать домашний каталог.
15. Выбираем «Без прокси», поля оставляем пустыми, если доступ в Интернет происходит напрямую.
16. Теперь нужен доступ в интернет для скачивания недостающих пакетов.
17. Ставим опцию «Без автоматического обновления» (для обновления вручную нужно набрать сначала «aptitude update», затем «aptitude upgrade»).
18. Настраиваем службы на сервере — OpenSSH, Print server, Samba.
19. Снова нужен интернет, так как будут скачиваться выбранные пакеты и локализация выбранного языка (рус).
20. Соглашаемся на установку GRUB загрузчика в «MBR».
21. Установка закончилась, перезагружаемся.

НАСТРОЙКА СИСТЕМЫ

1. Входим в систему под пользователем «administrator».
2. Повышаем привилегии до уровня ROOT — « sudo -s ».
3. Создаем пароль для пользователя « root », тем самым мы его включим – «passwd root».
4. Система увидела большое количество обновлений, – их установку необходимо выполнить вручную, выполняем для этого команды «aptitude update», затем «aptitude upgrade -y». Это может занять до ~30мин при 100кб/с, плюс ещё минут 15 на установку.
5. После такого серьёзного апдейта надо перезапустить систему – «reboot».
6. Заходим под «administrator» и вводим команду «su root». После этого мы увидим add user – это значит, что профиль «root» создан и можно под ним войти в систему. Вводим команду «exit» и теперь можно ввести имя пользователя «root» и его пароль.
7. Устанавливаем ПО «первой необходимости» – команда «aptitude install mc trafshow traceroute -y» – в качестве параметра ставим в очередь названия программ через пробел.
8. Теперь надо попробовать зайти на сервер по SSH, для этого пишем команду «ifconfig» и смотрим его IP адрес (с условием, что в сети есть DHCP).
9. Удаленно подключаемся к серверу. Для этого нам понадобится Putty в ОС Windows или же консоль в Linux (кодировка консоли Ubuntu — UTF-8).
10. Для Linux пишем в консоли «ssh administrator@айпи-адрес». Соглашаемся на принятие ключа «yes».
11. Повышаем привилегии «sudo -s», локально уже можно выйти.
12. Запускаем «mc» и начнём «Приручать пингвина».

НАСТРОЙКА SAMBA

Находим файл «/etc/samba/smb.conf» и приводим его к виду:

[global]	# Глобальные настройки
netbios name = Lit-Server-P	# Имя Машины в сети Windows
workgroup = BUDOVA	# Рабочая группа
server string = Printer	# Комментарий
wins support = no	# Не использовать систему имён от Windows
dns proxy = yes	# Пытаться разрешать имена с помощью nslookup
name resolve order = bcast	# Каким образом искать имена в сети
domain master = no	# Не как Мастер Браузер
local master = no	# Не участвовать в выборе Мастер Браузера
map to guest = Bad Password	# Любой вход расценивать как Гостя
log file = /var/log/samba/log.%m	# Куда писать лог файл
max log size = 1000	# Максимальный размер лог файла
security = share	# Политика работы сервера
guest account = printer	# Кто в системе является Гостем
printing = cups	# Какой демон отвечает за печать
printcap name = cups	# Какой демон отвечает за печать
max connections = 0	# Неограниченное число подключений
max open files = 10000	# Максимальное число открытых файлов
max print jobs = 1000	# Максимальное число заданий на принтер
load printers = yes	# Показывать принтеры в шарах

[printers]	# Принтеры
comment = All Printers	# Комментарий
browseable = yes	# Видимый в шарах
path = /var/spool/samba	# Путь к спулеру
printable = yes	# Печатаемый
guest ok = yes	# Пускать гостя
read only = yes	# Только для чтения
create mask = 0700	# Права доступа
public = yes	# Публичный
writable = no	# Запись запрещена
use client driver = yes	# Использовать драйвера клиента

[print$]	#Драйвера для автоматической подгрузки
comment = Printer Drivers	# Комментарий
path = /var/lib/samba/printers	# Где находятся драйвера
browseable = yes	# Видимый
read only = yes	# Только для чтения
guest ok = yes	# Вход гостям разрешён

[drv-printers]	#Драйвера для настройки вручную
comment = Drivers For Printer	# Комментарий
path = /home/userall/drv	# Где находятся драйвера
read only = no	# Только для чтения
create mask = 0700	# Права доступа
directory mask = 0700	# Права доступа
guest ok = Yes	# Доступ Гостям
browseable = yes	# Видимый

[Users-All]	# Для разной информации Настройка CUPS
comment = For All Users	# Комментарий
path = /home/userall/usersall	# Расположение
browseable = yes	# Видимый
read only = no	# Только для чтения
create mask = 0700	# Права доступа
directory mask 0700	# Права доступа
guest ok = Yes	# Доступ Гостям

НАСТРОЙКА CUPS

Открываем файл «/etc/cups/cupsd.conf» и приводим его к виду:

LogLevel warn	# Уровень логирования
MaxLogSize 0	# Размер логов
SystemGroup lpadmin	# Системная группа
Listen *:631	# Слушать сеть
Browsing On	# Видимость
BrowseOrder allow,deny	# Политика просмотра
BrowseAllow all	# Политика разрешения
BrowseLocal Protocols CUPS dnssd	# Протоколы для печати
BrowseAddress @ LOCAL	# Вещать на все адреса
DefaultAuthType Basic	# Тип авторизации по умолчанию

<Location />	# Доступ к службе по HTTP
Order deny,allow	# Последовательность политик
Allow From 192.168.4.*	# Разрешить с нашей сети доступ
</Location>	

<Location /admin>	# Доступ к администрированию
Order deny,allow	# Последовательность политик
</Location>	

<Location /admin/conf>	# Доступ к файлу конфигурации
AuthType Default	# Тип авторизации по умолчанию
Require user @ SYSTEM	# Потребовать пользователя системы
Order deny,allow	# Последовательность политик
</Location>	
<Location /admin/conf>	# Доступ к файлу конфигурации
AuthType Default	# Тип авторизации по умолчанию
Require user @ SYSTEM	# Потребовать пользователя системы
Order deny,allow	# Последовательность политик
</Location>	

<Policy default>	# Политики для принтеров
<Limit Send-Document Send-URI Hold-Job Release-Job Restart-Job Purge-Jobs Set-Job-Attributes Create-Job-Subscription Renew-Subscription Cancel-Subscription Get-Notifications Reprocess-Job Cancel-Current-Job Suspend-Current-Job Resume-Job CUPS-Move-Job CUPS-Get-Document>	
Require user @OWNER @SYSTEM	# Запросить пользователей
Order deny,allow	# Последовательность политик
</Limit>	

<Limit CUPS-Add-Modify-Printer CUPS-Delete-Printer CUPS-Add-Modify-Class CUPS-Delete-Class CUPS-Set-Default CUPS-Get-Devices>	
AuthType Default	# Тип авторизации по умолчанию
Require user @ SYSTEM	# Запросить пользователя
Order deny,allow	# Последовательность политик
</Limit>	

<Limit Pause-Printer Resume-Printer Enable-Printer Disable-Printer Pause-Printer-After-Current-Job Hold-New-Jobs Release-Held-New-Jobs Deactivate-Printer Activate-Printer Restart-Printer Shutdown-Printer Startup-Printer Promote-Job Schedule-Job-After CUPS-Accept-Jobs CUPS-Reject-Jobs>	
AuthType Default	# Тип авторизации по умолчанию
Require user @SYSTEM	# Запросить пользователя
Order deny,allow	# Последовательность политик
</Limit>	

<Limit Cancel-Job CUPS-Authenticate-Job>	
Require user @OWNER @SYSTEM	# Запросить пользователей
Order deny,allow	# Последовательность политик
</Limit>	

<Limit All>	
Order deny,allow	# Последовательность политик
</Limit>	

</Policy>	
<Policy authenticated>	# Политика аутентификации
<Limit Create-Job Print-Job Print-URI>	
AuthType Default	# Тип аторизации по умолчанию
Order deny,allow	# Последовательность политик
</Limit>	

<Limit Send-Document Send-URI Hold-Job Release-Job Restart-Job Purge-Jobs Set-Job-Attributes Create-Job-Subscription Renew-Subscription Cancel-Subscription Get-Notifications Reprocess-Job Cancel-Current-Job Suspend-Current-Job Resume-Job CUPS-Move-Job CUPS-Get-Document>	
AuthType Default	# Тип авторизации по умолчанию
Require user @OWNER @SYSTEM	# Запросить пользователей
Order deny,allow	# Последовательность политик
</Limit>	

<Limit CUPS-Add-Modify-Printer CUPS-Delete-Printer CUPS-Add-Modify-Class CUPS-Delete-Class CUPS-Set-Default>	
AuthType Default	# Тип авторизации по умолчанию
Require user @ SYSTEM	# Запросить пользователей
Order deny,allow	# Последовательность политик
</Limit>	

<Limit Pause-Printer Resume-Printer Enable-Printer Disable-Printer Pause-Printer-After-Current-Job Hold-New-Jobs Release-Held-New-Jobs Deactivate-Printer Activate-Printer Restart-Printer Shutdown-Printer Startup-Printer Promote-Job Schedule-Job-After CUPS-Accept-Jobs CUPS-Reject-Jobs>	
AuthType Default	# Тип авторизации по умолчанию
Require user @ SYSTEM	# Запросить пользователя
Order deny,allow	# Последовательность политик
</Limit>	

<Limit Cancel-Job CUPS-Authenticate-Job>	
AuthType Default	# Тип авторизации по умолчанию
Require user @OWNER @SYSTEM	# Запросить пользователей
Order deny,allow	# Последовательность политик
</Limit>	

<Limit All>	
Order deny,allow	# Последовательность политик
</Limit>	
</Policy>	

1. Для управления принтерами, заходим на наш принт-сервер по адресу: «http://его айпи:631/».

2. Теперь добавим пользователя, который будет выступать в роли Гостя «useradd printer», регистрируем его в «Samba», пишем в консоли «smbpasswd -a printer».

3. Теперь у нас все могут печатать из сети 192.168.4.0/24, регистрируются как пользователь Printer.

Олег Деордиев
г.Одесса

Доступ к удалённым файлам: SSHFS

Методов доступа к файлам, расположенным на удалённой системе, придумано великое множество: FTP, NFS, SMB/CIFS и много других. Среди всего многообразия протоколов, ориентированных на передачу файлов, думаю, многим известен SFTP — FTP-подобный протокол поверх SSH. Удобная штука, учитывая тот факт, что на большинстве хостов сегодня чаще всего встретишь поднятый OpenSSH-сервер, нежели FTP с поддержкой SSL или, тем более, VPN. Итак, SFTP безопасен, доступен, удобен. Хотя, с последним утверждением вряд ли согласятся те, кто хоть раз пробовал скопировать пару десятков файлов при помощи «родного» sftp-клиента. Чего только стоит отсутствие автодополнения в командной оболочке! Да, есть FTP-клиенты, избавляющие от этого геморроя, вроде yafc, однако согласитесь, куда удобней смонтировать удалённую ФС, а-ля NFS и, запустив Midnight Commander, получать удовольствие. Итак, сегодня речь пойдёт об SSHFS — FUSE-ФС, использующей SFTP в качестве транспорта.

УСТАНОВКА

Для корректной работы SSHFS вам понадобится присутствие в системе модуля ядра FUSE, обеспечивающего в целом возможность работы виртуальных файловых систем в пользовательском пространстве. В большинстве стандартных поставок дистрибутивов этот модуль присутствует и, если вы только не собирали ядро сами, то беспокоиться не о чем — модуль должен быть. Если пересобирали ядро — убедитесь, что не забыли о FUSE, без него никак.

Далее, необходимо установить саму SSHFS. В моей Ubuntu 10.10 пакет имеет одноимённое название и легко установился командой:

```
$sudo apt-get install sshfs
```

Думаю, в вашем дистрибутиве пакет должен называться так же или примерно так.

МОНТИРОВАНИЕ

Допустим, вам необходимо смонтировать удалённый каталог /home/vasya/documents, находящийся на сервере pupkin.com. Точкой монтирования на локальной системе будет каталог /home/vasya/mounts/documents. Имя пользователя, от имени которого подключаемся к удалённой системе — vasya (неожиданно). Итак, монтируем:

```
$ sshfs vasya@pupkin.com:/home/vasya/documents /home/vasya/mounts/documents
vasya@pupkin.com's password:
```

На приглашение ввести пароль отвечаем, естественно, вводом пароля и если всё правильно — вуаля, получаем прозрачный доступ к удалённой ФС.

ОТКЛЮЧЕНИЕ

Для того, чтобы отмонтировать FUSE-ФС, umount не подойдёт, поскольку вам понадобятся привилегии root (можно конечно sudo, если вы администратор системы, однако, на мой взгляд, сие идеологически неверно). Пользуемся специально предназначенной для этого утилитой (предварительно, естественно, закрыв все используемые файлы на отключаемой ФС):

```
$ fusermount -u /home/vasya/mounts/documents
```

ОПЦИИ

Опций у sshfs довольно много, чтобы их все здесь рассматривать. Что частенько встречается, так это нестандартный порт SSH, который многие администраторы выбирают из соображений безопасности системы. Если это ваш случай, то переопределить номер порта для подключения к SSH можно при помощи опции -p:

```
$ sshfs vasya@pupkin.com:/home/vasya/documents /home/vasya/mounts/documents -p 12345
```

Также можно и, чаще всего, нужно включить сжатие (если оно не включается у вас по умолчанию настройками SSH-клиента):

```
$ sshfs vasya@pupkin.com:/home/vasya/documents /home/vasya/mounts/documents -C
```

При ненадёжных соединениях бывает полезен синхронный режим записи:

```
$ sshfs vasya@pupkin.com:/home/vasya/documents /home/vasya/mounts/documents -o ssh_sync
```

ЛИТЕРАТУРА

• Полный перечень опций sshfs, а также FUSE читайте на man-странице SSHFS www.linux.die.net;

• немножко о том, что такое виртуальные ФС www.ru.wikipedia.org;

• статья на Хакере: «Побег за пределы ядра. Обзор файловых систем, основанных на fuse» www.xakep.ru;

• статья на Rus-Linux.Net: «Обмен файлами с помощью wdfs и FUSE» www.rus-linux.net .

www.ashep.org

Настройка GRUB2

В этой статье мы рассмотрим, как настроить загрузчик GRUB2 с помощью его конфигурационных файлов. Ознакомиться с данной информацией будет очень полезно, если раньше вы пользовались только первой версией GRUB: в GRUB2 вся структура конфигурационных файлов претерпела большие изменения. Однако для уверенного использования GRUB2 достаточно запомнить новое расположение конфигурационных файлов и их смысл.

ПРИНЦИПИАЛЬНЫЕ ОТЛИЧИЯ GRUB1 ОТ GRUB2

В первой версии GRUB все настройки и пункты загрузки хранились в простом файле /boot/grub/menu.lst, но в GRUB2 этого файла просто не существует. Однако имеется файл /boot/grub/grub.cfg и он действительно очень напоминает menu.lst, но редактировать его не рекомендуется. Конечно, никто не запрещает этого делать, но проблема в том, что после обновления меню загрузчика этот файл будет создан заново и ваши изменения исчезнут. Для решения этой проблемы нужно работать с другими файлами настройки, которые, в итоге, и создают файл grub.cfg.

ОБНОВЛЕНИЕ ЗАГРУЗОЧНОГО МЕНЮ И ПРИМЕНЕНИЕ НОВЫХ НАСТРОЕК

Допустим, вы собрали новое ядро и вам нужно, чтобы оно прописалось в меню загрузчика или вы просто изменили настройки GRUB2 и хотите, чтобы они вступили в силу. Для этого используется команда (от суперпользователя):

```
#update-grub
```

В ходе выполнения этой команды, обновляется файл /boot/grub/grub.cfg, в который вносятся новые пункты меню или/и настройки. Также в процессе выполнения этой команды в терминале вы увидите найденные операционные системы и ядра. Например, вот ход выполнения обновления в моей системе:

```
  Generating grub.cfg …
  Found linux image: /boot/
vmlinuz-2.6.35-19-generic
  Found initrd image: /
boot/initrd.img-2.6.35-19-
generic
  Found linux image: /boot/
vmlinuz-2.6.32-25-generic
  Found initrd image: /
boot/initrd.img-2.6.32-25-
generic
  Found linux image: /boot/
vmlinuz-2.6.32-21-generic
  Found initrd image: /
boot/initrd.img-2.6.32-21-
generic
  Found memtest86+ image: /
boot/memtest86+.bin
  done
```

Это означает, что теперь у меня в меню загрузки будет дистрибутив с различными версиями ядра linux и утилита memtest для проверки оперативной памяти.

/etc/default/grub – основные настройки GRUB2

В файле /etc/default/grub хранятся основные настройки GRUB2: пункт загрузки по умолчанию, время отображения меню загрузчика (при включении), параметры загрузки ядра по умолчанию, название дистрибутива, разрешение меню GRUB2. Есть ещё несколько параметров, но они вряд ли вам понадобятся. Давайте рассмотрим изменения этих параметров непосредственно в /etc/default/grub. Для понимания структуры файла, я приведу листинг этого файла в моей системе.

```
  GRUB_DEFAULT=0
  GRUB_HIDDEN_TIMEOUT=0
  GRUB_HIDDEN_TIMEOUT_
QUIET=true
  GRUB_TIMEOUT=»0»
  GRUB_DISTRIBUTOR=`lsb_
release -i -s 2> /dev/null
|| echo Debian`
  GRUB_CMDLINE_LINUX_
DEFAULT=»»
  GRUB_CMDLINE_LINUX=»»
  #GRUB_TERMINAL=console
  #GRUB_GFXMODE=640x480
```

Рассмотрим каждую стоку, содержащую значение параметра (переменной), отдельно.

GRUB_DEFAULT=0 – этот параметр указывает на строку загрузочного меню по умолчанию. Например, значение 0 указывает на самую верхнюю строку, а значение 1 на вторую строку сверху. Если вы хотите, чтобы по умолчанию выбирался не верхняя строка, то укажите ее номер, считая, что верхняя строка – 0. Указанная строка будет выделена в меню загрузчика и будет автоматически загружена, если пользователь не выберет другую строку.

GRUB_HIDDEN_TIMEOUT=0 – параметр, указывающий на время отображения меню выбора загрузки, если на компьютере установлена одна операционная система. Значение указывается в секундах. Если вам нужно будет попасть в меню, то вы должны указать положительное значение, например 5, и при старте компьютера нажать «escape» для входа в меню GRUB2 (где можно будет выбрать версию ядра или запустить memtest).

GRUB_TIMEOUT=»0» – параметр, указывающий время отображения меню загрузчика. Значение указываются в секундах и число должно быть заключено в двойные кавычки. Отрицательное значение отключит таймер полностью и меню будет «висеть», пока пользователь вручную не выберет пункт загрузки. При положительном значении, меню GRUB будет ждать действий пользователя заданное время, а если действия не будет, то автоматически будет загружен пункт меню по умолчанию.

GRUB_DISTRIBUTOR=`lsb_release -i -s 2> /dev/null || echo Debian` – параметр, указывающий на название дистрибутива, которое будет отображаться в GRUB. ▶

Как видим из данного значения, будет запущена команда lsb_release -is, в результате успешного выполнения которой получим название дистрибутива, а при неудачном выполнении выведется слово «Debian». Чтобы изменить значение, пропишите свою строку, например echo MyBestLinux (echo – команда bash, указывающая, что нужно вывести строку). Обратите внимание, что параметр указывается в обратных одинарных кавычках.

`GRUB_CMDLINE_LINUX_DEFAULT=»» и GRUB_CMDLINE_LINUX=»»` – параметры, которые нужно передать ядру. Если оставить пустым, как у меня, то при загрузке будут выводится все сообщения ядра, а splash (заставка загрузки) включена не будет. В кавычках можете передать нужные вам параметры ядру, например `GRUB_CMDLINE_LINUX_DEFAULT=»quiet splash»`

`#GRUB_TERMINAL=console` – включает консольный режим (по умолчанию строка закомментирована, что включает графический режим). Имеет смысл раскомментировать строку для увеличения производительности.

`#GRUB_GFXMODE=640x480` – параметр, задающий разрешение меню. Напомню, что список доступных режимов можно узнать командой GRUB vbeinfo

`/etc/grub.d/` – скрипты, формирующие grub.cfg

Довольно сложные shell скрипты настройки GRUB2 находятся в папке /etc/grub.d/. Нужно быть довольно опытном программистом и знать язык программирования shell, чтобы разобраться в этих скриптах и уметь изменять их. К счастью, это вряд ли может понадобиться. В этой статье я не стану рассматривать эти скрипты – они слишком сложны. Тем более, все основные настройки можно выполнить без них, а обычному пользователю вряд ли нужно будет углубляться в работу скриптов, формирующих grub.cfg. Если все же я в них разберусь, то это станет темой для отдельной статьи, так сказать, для продвинутых пользователей. Однако, некоторые файлы мы рассмотрим в следующих статьях. В следующей статье я напишу о том, как изменить внешний вид меню GRUB2.

www.linuxnow.ru

COMMUNIGATE

Установка, базовая настройка, ввод в эксплуатацию

часть 2

ВВЕДЕНИЕ

В первой части цикла о Communigate (CommuniGate Pro) мы изучали вопросы общей организации и назначения программного комплекса Communigate. Были рассмотрены основные компоненты, образующие логический состав сервера и составляющие его структуру. Также были описаны нюансы, возникающие при выборе различных платформ для развертывания Communigate.

Во второй части цикла будут рассмотрены вопросы установки и базовой настройки Communigate, достаточные для ввода программы в эксплуатацию. Причем, в процессе описания будет предпринята попытка объединить эти рекомендации для версии 5.0.5 Communigate и более поздних.

ПЛАН УСТАНОВКИ И ВВОДА В ЭКСПЛУАТАЦИЮ COMMUNIGATE

Независимо версии UNIX/Linux, общий план развертывания системы Communigate на сервере имеет следующий вид:

1. установить программу Communigate;
2. отключить (деинсталлировать) имеющиеся в системе службы pop, imap;
3. выполнить скрипт /etc/rc.d/init.d/CommuniGate start;
4. уточнить пароль для учетной записи postmaster в каталоге: /var/CommuniGate/Accounts/postmaster.macnt/account.settings;
5. изменить настройки системного firewall для открытия на нем необходимых портов. Пока нам будут жизненно необходимы протоколы http с портом 8010 и https с портом 9010;
6. зайти на сервер под учетной записью postmaster, ввести полученный пароль и приступить к базовым настройкам системы по адресу: http://server_address:8010 или https://server_address:9010.

Теперь рассмотрим более подробно установку Communigate в системах Linux/UNIX.

УСТАНОВКА COMMUNIGATE В LINUX СИСТЕМЕ

Нужно зайти в систему как суперпользователь (root).

- При использовании менеджера пакетов Red Hat (или подобных ему систем – с расширением файла .rpm) нужно исполнить команду:

```
rpm -i CGatePro-Linux-version.rpm
```

- при использовании других систем (файл .tgz):

```
tar -xzf CGatePro-Linux-version.tgz
cd CGateProSoftware
sh install.sh
```

CommuniGate Pro будет установлен в директории /opt.

- программа-установщик создаст файл со сценарием автоматического запуска /etc/rc.d/init.d/CommuniGate. Для того, чтобы сервер CommuniGate Pro начинал и прекращал работу автоматически вместе с системой, установщик добавит ссылку на этот файл в /etc/rc.d/rcn.d (где n – номер уровня загрузки);
- если в системе запущен отдельно SMTP сервер/MTA (такой, как sendmail или postfix), его нужно остановить (например, командой /sbin/chkconfig sendmail off) или деинсталлировать совсем;
- если в системе запущены POP, IMAP, или другие службы, то их тоже желательно остановить и деинсталлировать, если они больше использоваться не будут. Если есть какие-то сомнения по этому поводу, то удалять их не нужно. В случае последующего удаления CommuniGate Pro, ранее используемая почтовая программа будет переименована обратно в /bin/mail;
- программа-установщик создаст новое приложение /bin/mail – взамен существующей программы mail;
- программа-установщик также создаст поддиректорию /var/CommuniGate, используемую сервером по умолчанию. Ее можно переместить в любое другое место. При этом

откройте сценарий запуска /etc/init.d/CommuniGate и внесите в него соответствующие изменения;

• перезапустите систему или запустите сценарий запуска вручную:

```
/etc/rc.d/init.d/CommuniGate start
```

• затем можно продолжить настройку базовой конфигурации Communigate.

Нужно обратить внимание на то, что некоторые ранние (но ныне все еще широко используемые) версии Linux (такие как RedHat 9.0, SuSE 9.1) используют нестабильно работающую версию NPTL библиотеки.

Для того, чтобы решить проблему для этих версий Linux, сценарий запуска CommuniGate Pro использует команду LD_ASSUME_KERNEL=2.4.1. При этом компоновщик использует более стабильную версию этой библиотеки. Кроме этого, когда используется старая NTPL библиотека, системными утилитами ps и top каждая нить CommuniGate Pro отображается как отдельный процесс. В этом случае все эти «процессы» в действительности являются нитями сервера CommuniGate Pro, и они совместно используют все свои ресурсы – VRAM, дескрипторы файлов и тд. Не помешает помнить и тот факт, что ядра Linux до версии 2.6.13 имеют критическую уязвимость в реализации NFS клиента. При этом ядро нужно «пропатчить» или использовать сразу систему с более свежим ядром. Кроме этого, некоторые ядра Linux некорректно поддерживают параллельные вычисления на системах x86, поэтому лучше отключить функцию hyperthreading в BIOS сервера, имеющего x86 архитектуру.

УСТАНОВКА COMMUNIGATE НА БАЗЕ FREEBSD

Имеется два пакета CommuniGate Pro для установки на систему FreeBSD: один под FreeBSD 4.x (поддерживающий версии FreeBSD 4.x), другой – поддерживающий FreeBSD 5.3 и более поздние версии. Независимо от используемой платформы, порядок действий будет практически одинаков.

Необходимо войти в систему как суперпользователь (root).

Установить соответствующий пакет CommuniGate Pro. FreeBSD 4.x:

```
pkg_add CGatePro-FreeBSD4-version.tgz
```

FreeBSD 5.x:

```
pkg_add CGatePro-FreeBSD-version.tgz
```

CommuniGate Pro будет установлен в директории /usr/local/sbin.

Если в системе запущен sendmail или любой другой SMTP сервер, его нужно остановить и изменить сценарий запуска операционной системы таким образом, чтобы он не стартовал автоматически. Если не планируется обратный переход, то можно деинсталлировать имеющуюся SMTP службу.

Если в системе запущены POP, IMAP сервера, то нужно удалить их из тех конфигурационных файлов, которые обеспечивают их старт. Как правило, это /etc/inetd.conf файл.

Программа-установщик создаст сценарий /usr/local/etc/rc.d/CommuniGate.sh. Таким образом, Communigate будет запускаться автоматически, одновременно со стартом системы FreeBSD.

Программа-установщик создаст символьную ссылку /bin/cgmail на программу mail, работающую в режиме командной строки для взаимодействия с Communigate.

Программа-установщик создаст поддиректорию /var/CommuniGate, используемую сервером по умолчанию. Можно переместить этот каталог в любое другое место. В этом случае нужно внести соответствующие изменения в сценарий запуска.

Выполнить программу запуска сервера вручную:

```
/usr/local/etc/rc.d/CommuniGate.sh start
```

• Теперь можно продолжить базовую настройку сервера в веб-интерфейсе администартора.

УСТАНОВКА COMMUNIGATE НА БАЗЕ NETBSD

убедитесь, что вы используете NetBSD версии 2.0 или выше;

зарегистрируйтесь как суперпользователь (root);

установите пакет CommuniGate Pro:

```
pkg_add CGatePro-NetBSD-version.tgz
```

CommuniGate Pro будет установлен в директории /usr/pkg.

Если в системе запущен sendmail или любой другой SMTP сервер, остановите его и измените сценарий запуска операционной системы таким образом, чтобы он не стартовал автоматически или деинсталлируйте его;

если на вашей системе запущены POP, IMAP или подобный сервер, то удалите его из файла, отвечающего за его старт. Как правило, это файл /etc/inetd.conf для BSD систем;

установщик при этом создаст сценарий запуска /etc/rc.d/CommuniGate и сервер будет запускаться автоматически, одновременно со стартом системы NetBSD;

программа-установщик создаст символьную ссылку /bin/cgmail на программу mail, работающую в режиме командной строки для взаимодействия с CommuniGate Pro;

программа-установщик создаст подкаталог /var/CommuniGate, используемый сервером по умолчанию. Его можно переместить в любое другое место. В этом случае нужно внести соответствующие изменения в сценарий запуска;

перезапустите систему или запустите сценарий запуска вручную:

```
/etc/rc.d/CommuniGate start
```

продолжите настройку системы из веб-интерфейса администратора.

УСТАНОВКА COMMUNIGATE НА БАЗЕ OPENBSD

необходимо использовать версию OpenBSD 3.4 или выше;

нужно войти в систему как суперпользователь (root);

установите пакет CommuniGate Pro:

```
pkg_add CGatePro-OpenBSD-version.tgz
```

CommuniGate Pro будет установлен в директории /usr/local/sbin.

Если в системе запущен sendmail или любой другой SMTP сервер, остановите его и измените сценарий запуска операционной системы таким образом, чтобы он не стартовал вместе с системой. Можно просто удалить его.

Если на вашей системе запущены POP, IMAP сервера, то удалите строки, в которых содержится описание этих серверов из файла /etc/inetd.conf.

Программа-установщик добавит ссылку на сценарий запуска CommuniGate Pro в файл /etc/rc.local, таким образом сервер CommuniGate Pro будет запускаться автоматически, одновременно со стартом системы OpenBSD.

• Программа-установщик создаст группу mail (если она отсутствовала).

• Программа-установщик создаст символьную ссылку /bin/cgmail на программу mail, работающую в режиме командной строки для взаимодействия с CommuniGate Pro.

• Программа-установщик создаст подкаталог /var/CommuniGate, ис-

пользуемый сервером по умолчанию. Его можно переместить в любое другое место. В этом случае нужно внести соответствующие изменения в сценарий запуска Communigate.

• Перезапустите систему или запустите сценарий запуска вручную:

```
/usr/local/sbin/CommuniGate/
Startup start
```

• Продолжите настройку системы из административного веб-интерфейса.

НАЧАЛЬНЫЕ ДЕЙСТВИЯ В СИСТЕМЕ

После того, как CommuniGate Pro установлен, нужно:

• перезапустить операционную систему или перезапустить сервер CommuniGate Pro вручную;

• затем, в течение 10 минут зайти на сервер через веб-интерфейс администратора, порт 8010, используя любой веб-браузер. Введите следующий URL в вашем браузере:

```
http://your.server.
domain:8010
```

где your.server.domain is доменное имя или IP адрес компьютера, на котором запущен сервер CommuniGate Pro или используйте порт 9010, тогда адрес измениться на https://your.server.domain:9010;

• прочитайте лицензионное соглашение, введите желаемый пароль для пользователя postmaster, затем щёлкните по кнопке «Accept»;

• после этого произойдет перенаправление на страницу быстрого старта, – используйте имя postmaster и введённый ранее пароль для открытия страницы;

• продолжите настройку в административном веб-интерфейсе.

Если новый пароль для пользователя postmaster не будет введен в течение 10 минут, сервер отключиться. Причем это новшество касается версий выше, чем 5.0.5.

ПЕРЕХОД НА БОЛЕЕ НОВУЮ ВЕРСИЮ COMMUNIGATE

При установке новой версии сервера старые системные файлы замещаются новыми, но пользовательские данные при этом изменениям не подвергаются. Директория данных и файлы, находящиеся в ней, не меняются при переходе на новую версию CommuniGate Pro. Таким образом, все пользователи, папки, сообщения, настройки, файлы в хранилище файлов, лицензии, изменённые виды веб-интерфейса и файлы приложений реального времени остаются в неизменном виде и будут работать под новой версией. Но, в любом случае, необходимо сделать резервную копию и подстраховаться от внеплановых неожиданностей.

Для обновления имеющейся версии Communigate выполните следующие действия:

• загрузите новую версию CommuniGate Pro;

• остановите загруженный пакет;

• удалите предыдущую версию сервера, использую ту же самую утилиту, которая использовалась для установки (директория данных не будет удалена). Это нужно делать только в том случае, если установщик операционной системы не может переустанавливать новую версию «поверх» старой (как например в Solaris или FreeBSD, Linux).

Стоит обратить внимание на то, что если используется менеджер пакетов Linux rpm, то его опцию «update» использовать не надо. Лучше деинсталлировать старую версию, а затем установить новую как в примере:

```
rpm -e CGatePro-Linux
rpm -i CGatePro-Linux-
version.rpm
```

• установите новую версию Communigate;

• запустите Communigate как описывалось раньше.

ПЕРЕХОД НА НОВОЕ ОБОРУДОВАНИЕ

При необходимости перевода Communigate на другой компьютер, работающий под той же или другой операционной системой, все настройки модулей, настройки пользователей и доменов, папки, сообщения, лицензии и другие данные могут быть сохранены. CommuniGate Pro хранит их в папке для данных. Это единственная папка, требующая резервного копирования при переводе системы на новое оборудование. CommuniGate Pro использует одинаковый формат файлов на любой операционной системе и аппаратной платформе, так что обычно бывает достаточно упаковать всю директорию данных CommuniGate Pro в один архив (используя tar или gzip в системах Unix) с последующей распаковкой на другом сервере. Дополнительная обработка данных может потребоваться в случае, если сервер переносится с компьютера под управлением любой из операционных систем семейства MS Windows на компьютер под управлением Unix/Linux, или наоборот. Файлы CommuniGate Pro – это текстовые файлы, а текстовые файлы в MS Windows и в Unix используют различные EOL (окончание строки) символы: CR-LF (возврат каретки-перевод строки) в MS Windows и один LF символ (перевод строки) в системах Unix. Для корректного копирования файлов между ними можно использовать FTP протокол: когда FTP клиент передаёт файлы в режиме ASCII, он автоматически конвертирует EOL символы, а именно это нам и нужно.

Стоит обратить внимание на то, что папка с данными в CommuniGate Pro может содержать не только текстовые (двоичные) файлы внутри директорий WebSkins и PBXApps, а также внутри директорий Accounts и Domains. Там могут храниться графические, аудио и видео – файлы, используемые для работы в приложениях реального времени и в настройках внешнего вида веб-интерфейса. В хранилище файлов у пользователей тоже могут находиться двоичные файлы. Они хранятся в директориях account.web внутри директории Accounts.

Если директория данных Communigate перемещается между системами, в которых используются различные правила, касаемые EOL, то нужно убедиться, что они копируются в режиме BINARY (при этом не будет преобразований символа EOL). Если новый аппаратный сервер работает под управлением Unix, то следует проверить его на предмет соответствия прав доступа у скопированной директории и всех ее файлов и подкаталогов правам доступа на той системе, где они находились ранее. После копирования папки с данными можно загрузить новую версию Communigate и установить ее на новом оборудовании. Не нужно копировать содержание папок с программами. Причем эта рекомендация касается случая работы обоих серверов под одинаковыми системами. Затем можно проверить, настроена ли установленная копия Communigate (и сценарий запуска, при его наличии) на использование скопированной поддиректории данных. После этого можно запустить Communigate на новом оборудовании. Через административный веб-интерфейс нужно изменить на новом сервере все аппаратные настройки, связанные с правильным функционированием системы. При этом может понадобиться изменить таблицу IP адресов клиентов сервера или переназначить IP адреса доменов, которые обслуживает Communigate.

ВЫВОДЫ

В статье рассмотрены основные вопросы, связанные с установкой и стартом системы Communigate, входом в интерфейс администратора и начала работы по конфигурированию системы. Процесс установки и начальной настройки системы описан для различных систем – от Linux до FreeBSD, причем таким образом, чтобы в описании нашли отражение как более ранние версии Communigate, так и новейшие. В продолжении цикла мы поговорим о настройке почтового домена и учетных записей пользователей.

Александр Деревянко
www.ibm.com

Установка Ubuntu Desktop на LVM

Классические разделы, на которые чаще всего разбивается жёсткий диск для установки системы и хранения данных, имею ряд существенных недостатков. Их размер очень сложно изменять, они находятся в строгой последовательности и просто взять кусочек от первого раздела и добавить к последнему не получится, если между ними есть ещё разделы. Поэтому очень часто при начальном разбиении винчестера пользователи ломают себе голову – сколько места выделить под тот или иной раздел. И почти всегда в процессе использования системы приходят к выводу, что они сделали неправильный выбор.

Решить большинство этих проблем может технология LVM. Она создаёт дополнительную абстракцию – логические тома, которые видны в системе, как обычные разделы, однако ими не являются. Она имеет ряд преимуществ:

- Логические тома LVM больше не привязаны к физическому местоположению. В рамках LVM вообще не существует такого понятия, как порядок логических томов.
- Размер логических томов можно увеличивать прямо на лету, а у отмонтированных томов можно, кроме того, легко уменьшать размер, не выходя из системы.
- При необходимости можно «размазать» логические тома по нескольким физическим жёстким дискам, увеличив таким образом доступное место. При этом система всё так же будет видеть только один логический том, хотя размер его будет превышать доступные размеры жёстких дисков. Можно провести и обратную операцию, удалив жёсткий диск из LVM, таким образом освободив его для другого использования.
- LVM поддерживает механизм снапшотов – мгновенных копий файловой системы тома. Это может очень пригодиться для создания бекапов.
- Есть ещё масса плюсов, о которых можно почитать в специализированных статьях про LVM.

Ubuntu полностью поддерживает LVM, однако из Desktop версии установочного диска убраны необходимые для работы с LVM утилиты. Кроме того, установщик Desktop версии не умеет изменять LVM тома. Поэтому, если вы хотите использовать LVM, то вам придётся либо поставить систему с Alternate диска, либо немного помухлевать с обычным LiveCD. Установка с Alternate неудобна и у многих вызывает дискомфорт и, кроме того, чаще всего Alternate диска нет под рукой, поэтому давайте рассмотрим вариант с LiveCD.

Учтите, что управление LVM осуществляется через терминал, поэтому, дабы ничего не сломать, вам необходимо сначала научиться более-менее комфортно с ним работать. Кроме того, ознакомьтесь с принципами работы и основными концепциями LVM, дабы понимать, что это такое. Статья рассчитана не на новичков, а на тех, кто уже разобрался в основах Ubuntu.

ПРЕДВАРИТЕЛЬНАЯ ПОДГОТОВКА

Вам необходимо запустить систему с LiveCD и подключить компьютер к интернету. Далее откройте терминал и установите прямо в LiveCD сессию необходимые утилиты командой:

```
sudo apt-get install lvm2
```

Всё, теперь можно начинать работу с LVM. Но сначала необходимо выделить место, на котором мы будем создавать LVM. Для этого вам потребуется редактор разделов Gparted, который находится в меню СистемаАдминистрирование (SystemAdministration). Создайте с его помощью раздел, поверх которого вы будете разворачивать LVM. Если вы готовы выделить под LVM весь жёсткий диск, то создайте раздел на весь жёсткий диск. Не выбирайте никакой файловой системы для раздела – просто пустой раздел (unformatted). Не забудьте, для применения всех внесённых вами через Gparted изменений, нужно нажать на зелёную галочку на верхней панели или выбрать в меню Edit пункт Apply.

После того, как изменения будут успешно внесены в разметку, необходимо поменять тип нужного вам раздела на Linux LVM. Для этого нажмите правой клавишей на разделе и выберите пункт «Управление флагами» (Manage flags). В открывшемся окне поставьте галочку в поле с именем lvm, дождитесь окончания применения всех изменений и закройте Gparted. На этом подготовительный этап закончен.

СОЗДАНИЕ ЛОГИЧЕСКИХ ТОМОВ LVM

Теперь пора приступить к непосредственному созданию LVM. Для примера будем считать, что LVM мы создаём поверх раздела /dev/sda1. В этом случае сначала необходимо инициализировать физический раздел командой:

```
sudo pvcreate /dev/sda1
```

Далее создать группу томов (о том, что это такое, читайте в статьях про LVM), командой:

```
sudo vgcreate local /dev/sda1
```

И, наконец, создать необходимые вам логические тома. Тут стоит заметить, что так как увеличивать размер LVM томов можно легко прямо в работающей системе, то лучше всего выделять для логических томов минимальный необходимый размер. Не бойтесь, что большой объём останется нераспределённым в рамках группы томов, он не пропадёт. Как только вам понадобится дополнительное место, вы сможете его легко добавить к любому логическому тому. А вот уменьшить размер логического тома значительно сложней.

Обычно для установки системы требуется раздел под корень, раздел под /home, раздел под swap и иногда – раздел под данные. Создать логические тома под все эти четыре задачи можно командами:

```
sudo lvcreate -L 7G -n root local
sudo lvcreate -L 5G -n
```

```
home local
  sudo lvcreate -L 3G -n
swap local
  sudo lvcreate -L 10G -n
data local
```

Параметр -n, если вы ещё не поняли, задаёт имя логического тома, -L - его размер.

Теперь необходимо создать файловые системы на созданных логических томах. Сделать это можно примерно следующими командами:

```
  sudo mkfs.ext4 /dev/
local/root
  sudo mkfs.ext4 /dev/
local/home
  sudo mkswap -f /dev/
local/swap
  sudo mkfs.ext4 /dev/
local/data
```

Обратите внимание, что имена логических томов LVM в системе выглядят как /dev/{имя_группы_томов}/{имя_тома}. Кроме того, файловая система ext4 по умолчанию резервирует часть места для системных данных. Поскольку на /home и, уж тем более, на разделе с пользовательскими файлами, никаких системных данных никогда не будет, то лучше отменить это резервирование, дабы освободить пропадающее зря место. Для это пригодятся команды:

```
  sudo tune2fs -r 0 /dev/
local/home
  sudo tune2fs -r 0 /dev/
local/data
```

Ни в коем случае не отменяйте резервирование для корневого раздела, иначе система может вообще перестать работать!

Ну и наконец вы возможно захотите присвоить разделу с данными нормальную метку, дабы он красиво отображался в установленной системе. Сделать это можно примерно такой командой:

```
  sudo tune2fs -L Data /
dev/local/data
```

Теперь можно приступать непосредственно к установке системы на созданные нами логические тома.

УСТАНОВКА СИСТЕМЫ

Собственно, сама установка стандартна, однако, когда вам предложат выбрать способ разбиения диска, выберите ручной режим. В открывшемся окне вы увидите все созданные вами тома LVM. Укажите для каждого соответствующую точку монтирования, однако не ставьте галочку форматирования. Для раздела под данные можете указать точку монтирования /media/data.

Дождитесь завершения установки, но компьютер не перезагружайте!

Если вы перезагрузите компьютер, то ваша новая система не запустится. В этом случае необходимо будет снова зайти в LiveCD, установить утилиты работы с LVM, а далее выполнить команду:

```
  sudo vgchange -a y
```

После чего выполнить инструкции, приведённые ниже.

АКТИВИРОВАНИЕ LVM В УСТАНОВЛЕННОЙ СИСТЕМЕ

Систему вы поставили, однако осталась одна небольшая проблема – в установленной Ubuntu нет утилит работы с LVM, а значит она просто не запустится. Это достаточно легко исправить. Для начала, не выходя с LiveCD, смонтируйте логический том, который вы выделили под корень, в папку /mnt. Сделать это можно командой:

```
  sudo mount /dev/local/
root /mnt
```

Теперь необходимо будет войти в установленную систему с помощью chroot, однако, предварительно надо временно подключить к ней некоторые важные системные ресурсы. Для этого выполните команды:

```
  sudo mount --bind /dev /
mnt/dev
  sudo mount --bind /proc /
mnt/proc
  sudo mount --bind /sys  /
mnt/sys
```

Теперь перейдите в установленную систему командой:

```
  sudo chroot /mnt /bin/
bash
```

И установите необходимые утилиты командой:

```
  apt-get install lvm2
```

Всё, установка завершена. Закройте терминал, нажмите Alt+Ctrl+Del и перезагрузите компьютер. Не забудьте достать LiveCD из привода. Компьютер должен загрузиться в только что установленную систему.

ДАЛЬНЕЙШАЯ РАБОТА

Допустим, в какой-то момент вам перестало хватать 5 гигабайт, которые вы выделили под раздел /home. Не беда. Посмотрите, сколько неиспользованного места осталось в группе томов командой:

```
  sudo vgdisplay local
```

Теперь увеличьте размер логического тома /dev/local/home до нужного командой:

```
  sudo lvresize -L 15G /
dev/local/home
```

Учтите, что в параметре -L указывается полный желаемый размер, а не его приращение. После увеличения размера логического тома останется лишь растянуть файловую систему на весь новый объём. Сделать это можно командой:

```
  sudo resize2fs /dev/
local/home
```

Всё, размер логического тома увеличен.

Кроме увеличения размера логических томов на лету, LVM умеет ещё много чего полезного. Например, создавать мгновенные снапшоты. Однако обо всех тонкостях работы с этой технологией читайте в специализированных статьях.

Вадим Неворотин
www.help.ubuntu.ru

Аутентификация с помощью USB Flash

В данной статье описывается способ, как использовать USB флеш-накопитель для аутентификации пользователя вместо традиционного пароля. Для этого используется Pluggable Authentication Modules (PAM) и USB флеш-накопитель.

УСТАНОВКА PAM USB

pam_usb доступен в большинстве репозиториев Linux дистрибутивов.

```
sudo apt-get install pamusb-tools libpam-usb
```

Добавляем USB флеш-накопитель в конфигурацию PAM

```
sudo pamusb-conf --add-device my-usb
Please select the device you wish to add.
* Using «flash-drive» (only option)
Which volume would you like to use for storing data ?
0) /dev/sdb2 (UUID: 1234-1234)
1) /dev/sdb1 (UUID: 2314-1234)
[0-1]: 0
Name       : my-usb
Vendor     : noname
Model      : no_model
Serial     : flash-drive
UUID       : 1234-1234
Save to /etc/pamusb.conf ?
[Y/n] Y
Done.
```

Вместо my-usb можете использовать любое, удобное для вас, название.

Этой командой мы добавили в /etc/pamusb.conf блок кода XML, чтобы в дальнейшем определять наше usb-устройство.

```
<pre>
<device id=»my-usb»>
<vendor>
noname
</vendor>
<model>
no_model
</model>
<serial>
flash-drive
</serial>
<volume_uuid>
1234-1234
</volume_uuid>
</device>
</pre>
```

ДОБАВЛЕНИЕ ПОЛЬЗОВАТЕЛЯ

Мы можем добавить несколько usb-устройств в конфигурацию PAM, в то же время мы можем назначить нескольким пользователям одно usb-устройство. Здесь будет описана простая конфигурация: один пользователь – одна флешка.

```
sudo pamusb-conf --add-user flash-user
Which device would you like to use for authentication ?
* Using «my-usb» (only option)
User      : flash-user
Device    : my-usb
Save to /etc/pamusb.conf ?
[Y/n] y
Done.
```

Вместо flash-user прописывайте пользователя, для которого собственно и хотите сделать авторизацию с помощью флешки.

НАСТРОЙКА PAM ДЛЯ ИСПОЛЬЗОВАНИЯ БИБЛИОТЕКИ PAM_USB

На данный момент, мы определили устройство USB «my-usb» для проверки подлинности пользователя «flash-user». Теперь нужно настроить систему на использование модуля pam_usb. Чтобы добавить pam_usb в процесс проверки подлинности, необходимо изменить файл /etc/pam.d/common-auth.

Это текущая конфигурация для аутентификации пользователя.

```
auth required pam_unix.so nullok_secure
```

Изменим конфигурацию:

```
auth sufficient pam_usb.so
auth required pam_unix.so nullok_secure
```

чтобы для аутентификации было достаточно usb-устройства.

```
su flash-user
*pam_usb v0.4.2
* Authentication request for user «flash-user» (su)
* Device «my-usb» is connected (good).
* Performing one time pad verification...
* Regenerating new pads...
•  Access granted.
```

Если флешки, определенной для flash-user, в системе нет, пользователю необходимо ввести пароль. Для усиления безопасности меняем в /etc/pam.d/common-auth “sufficient” (достаточно) на “required” (необходимо).

```
auth required pam_usb.so
auth required pam_unix.so nullok_secure
```

Теперь пользователю flash-user для аутентификации необходимо иметь флешку и правильный пароль.

```
su flash-user
* pam_usb v0.4.2
* Authentication request for user «flash-user» (su)
* Device «my-usb» is connected (good).
* Performing one time pad verification...
* Access granted.
Password:
```

СОБЫТИЯ ПРИ ОТКЛЮЧЕНИИ ФЛЕШКИ

Можно настроить pam_usb так, что к примеру при отключении флешки блокируется экран.

Для этого правится /etc/pamusb.conf

```
<user id=»flash-user»>
<device>
my-usb
</device>
<agent event=»lock»>gnome-screensaver-command -l</agent>
<agent event=»unlock»>gnome-screensaver-command -d</agent>
</user>
```

Примечание: Все это производилось в ubuntu, но подходит для большинства других дистрибутивов.

www.thelinux.org.ru

Чистим хомяк Linux'а

Ну вот так всегда... Файл весит 700 мегабайт, а свободно лишь 500... И удалить ничего не могу — все нужно. Что делать – удалять нужное или бежать в круглосуточный магазин дисков?» Знакомая ситуация? А как же! Вот только не нужно спешить удалять нужные файлы. Сегодня я вам расскажу, как можно безболезненно освободить немного свободного места! Дело вот в чем. Линукс — он как живое существо. В процессе жизни он оставляет много продуктов своей жизнедеятельности. И вы, как настоящий заботливый хозяин, должны за ним убирать :-) Конечно, это только метафора, но в ней намек на то, чем мы сегодня будем заниматься. Наша цель — научится безопасно удалять ненужные файлы, при этом не повредить систему и освободить место на жестком диске. Могу сказать только одно — не бойтесь делать то же самое — все абсолютно безопасно. Итак, поехали!

УДАЛЕНИЕ НАСТРОЕК НЕСУЩЕСТВУЮЩИХ ПРОГРАММ

Программы приходят и уходят, а мусор от них, т.е. устаревшие настройки, остаются. Обычно конфиги хранятся в домашней папке, в которой создаются файлы и папки, названые по такому принципу: [.]+[имя_программы]. Как вы знаете, файлы, название которых начинаются на точку, в UNIX, а соответственно и в Linux, являются скрытыми. Для их просмотра в Nautilus нажмите «Ctrl»+«H». Еще раз нажмете — они исчезнут. Их скрывают по понятным причинам — рядовому пользователю они не интересны. Посмотрите на эти папки — может каких то программ уже давно нет на компьютере. Только тут будьте осторожны — думайте, что удаляете. Полагаюсь на вашу интуицию :) Также есть смысл заглянуть в папки ~/.config и ~/.gnome2 — некоторые программы там тоже настройки хранят.

ВРЕМЕННЫЕ ФАЙЛЫ FILEROLLER

Эта программа служит стандартным менеджером архивов в GNOME. Когда мы с его помощью открываем архив, и прямо из окна архиватора открываем файл, то сначала создается папка ~/.cache/.fr-* (или ~/.fr-*). В нее то и распаковывается выбранный файл, который потом открывается в нужной программе. Сейчас у меня там килобайты, но если вы открывали какой-нибудь «массивный» файл, туда стоит заглянуть.

ВРЕМЕННЫЕ ФАЙЛЫ MC.

Этот отличный консольный файловый менеджер поддерживает множество разных форматов архивов. Распаковывать их очень просто — просто выделяем курсором архив и жмем «ENTER». А вот убирать после себя он не всегда может. Временная папка для распакованных файлов может быть как /tmp/mc-[username], так и ~/tmp/mc-[username]. Давайте заглянем туда на моей машине:

```
$ du -h $TMP/mc-keed/
17M /home/keed/tmp/mc-keed/
```

Это немного, но помню один раз было почти 800 МБ — просматривал iso-образ и mc рухнул. Поэтому время от времени туда заглядывать было бы неплохо:

```
$ rm -Rf $TMP/mc-keed/*
$ du -h $TMP/mc-keed/
0 /home/keed/tmp/mc-keed/
```

ЭСКИЗЫ (THUMBNAILS)

Мы уже привыкли к тому, что в nautilus отображаются эскизы для разных типов файлов — jpeg, png, xpm, pdf, djvu, avi, mpeg4... И вот как эта технология работает. Возьмем, к примеру, документ ~/document.pdf. Сначала наутилус определяет имя файла и его тип MIME. Затем он дает запрос программе evince-thumbnailer, которая обрабатывает первую страницу документа и создает ее уменьшенную png-копию. Далее миниатюра помещается в папку ~/.thumbnails/large или ~ /.thumbnails/normal. Затем имя файла эскиза передается обратно наутилусу, а он ставит его вместо стандартного значка. И так для многих файлов, только принцип создания может быть другим. И когда файл удаляется, эскиз для него еще остается. Давайте заглянем в эту папку с эскизами:

```
$ du -h ~/.thumbnails/
85K ./.thumbnails/fail/
gnome-thumbnail-factory
85K ./.thumbnails/fail
6,5M ./.thumbnails/large
41M ./.thumbnails/normal
47M ./.thumbnails/
```

Как видим, почти 100 МБ занимают эскизы. Для чистки их можно запустить специальный скрипт clean, который можно скачать здесь: http://thumb-clean.narod.ru. Он и почистит, и выведет подробную информацию. Clean также удобно ставить на задания в cron. Посмотрим на его работу:

```
$ clean
Видалено 6165 файлів.
Звільнено місця: 95M /home/
keed/.thumbnails/
Використовуй -h або --help
для отримання довідки,
а також -v чи --version для
отримання версії
```

ВРЕМЕННЫЕ ФАЙЛЫ EVINCE

Эта программа просмотра документов также создает временные файлы. Они находятся в папках /tmp/evince-* или ~/tmp/evince-*. В них лежат документы PDF, сканированные версии которых могут занимать очень много. Посмотрим, что у меня там:

```
$ du -h $TMP/evince-*
400K /home/keed/tmp/
evince-3123
8,9M /home/keed/tmp/
evince-3825
$ rm -Rf $TMP/evince-*
```

Вот еще 9 МБ освободили. Ура!

КЭШ БРАУЗЕРОВ

Тут тоже много всякой всячины. Можно поудалять временные файлы и cookie вручную. Кэш оперы находится тут: ~/.opera/cache. Он у меня занимает пока что 326 МБ, недавно чистил. Удаляем! Кэш Mozilla Firefox хранится здесь: ~/.mozilla/firefox/[имя_пофиля]/Cache. у меня это папка /home/keed/.mozilla/firefox/47b0yujw.default/Cache и там 741 Мб! Тоже удаляем:

```
$ rm -Rf ~/.opera/cache/*
~/.mozilla/firefox/47b0yujw.
default/Cache/*
```

Освободили 326+741=1067 Мб, чуть больше 1ГБ. Вот так. Только нужно помнить, что потом все сохраненные пароли будет нужно вводить заново.

ИНДЕКСЫ ПОИСКОВИКА BEAGLE

Те, кто пользуется этим замечательным поисковиком, должны остаться довольны его возможностями и скоростью поиска. Однако такое удобство идет в ущерб занимаемому месту на жестком диске. Чем больше размер диска и больше файлов на нем, тем больше места займут индексные файлы, и это логично. Они хранятся по адресу ~/.beagle/Indexes. Так как жесткий у меня не маленький и Beagle сканирует все разделы и папки кроме /boot, /dev, /proc и /sys, то и индексы у мня занимают аж 2,3 Гб. Ими можно смело пожертвовать при нехватке места, что сейчас и сделаем:

```
$ rm -Rf ~/.beagle/
Indexes/*
```

ЗАКЛЮЧЕНИЕ

В заключение давайте подсчитаем, сколько же мы сегодня места освободили: 17+95+9+1067+2300=3488 мегабайт, что составляет аж 34% моего 10 гигабайтного домашнего раздела. Хотя эта цифра могла быть и большей. Был ли смысл в этой статье? Цифры говорят, что был. Время от времени повторяйте описанные мной действия. И вам это надоест. И вы напишите скрипт, который автоматизирует эти действия и повысит знания языка shell. Be free. With Linux!

Андрей 'keedhost' Кондратьев
www.keedhost.blogspot.com