



Полезные утилиты и программы

Содержание

- Текстовый редактор Vim
- Консольный оконный менеджер Screen
- Системная утилита Syslog
- Планировщики задач Cron, AT, Batch
- Анализ логов с помощью Less, head, tail, split утилит

Текстовый редактор Vim

Vim - это мощный редактор, имеющий множество команд с полной свободой настройки и автоматизации.

Режимы редактора

Normal mode – обычный режим (ввод команд)

Insert mode – режим вставки (редактирование текста)

Запуск редактора

```
# vim file.txt
```

Перемещение курсора по экрану

h – влево

^

j – вниз

k

k – вверх

< h

l >

l – вправо

j

Завершение работы с Vim

<Esc>	вернуться в обычный режим
:q	выйти из vim
:q!	выйти из vim без сохранения любых сделанных изменений
:wq	сохранить изменения и выйти
:wq!	принудительно сохранить изменения и выйти

Удаление текста

x	удалить символ выделенный курсором
dw	удалить текст от курсора до конца слова, включая завершающий пробел
d\$	удаление от курсора до конца строки
d^	удаление от курсора до начала строки
de	удаление от курсора до конца слова, НЕ включая завершающий пробел
dd	удаление всей строки

Ввод текста

i перейти в режим вставки
<Esc> вернуться в обычный режим

Команды и объекты

d[число]объект формат команды «удаление» d таков.

Где:

d	команда удаления.
число	сколько раз исполнить команду
объект	с чем команда должна быть выполнена

Краткий список объектов:

w	от курсора до конца слова, включая завершающий пробел.
e	от курсора до конца слова, НЕ включая завершающий пробел.
\$	от курсора до конца строки.
^	от курсора до начала строки.

Команды отмены и вставки

- u** отмена результата работы предыдущей команды
- U** отмена исправлений во всей строке
- p** вставка после курсора последнего удаленного (или скопированного). текста (если была удалена строка, то она будет помещена в строке под курсором).
- r** замена единичного символа под курсором (далее должен следовать символ на который он заменяется).
- c[<число>]<объект>** Изменение текста (удаляет объект и переводит в режим вставки
- o** создание пустой строки ПОД курсором и переход в режим вставки
- O** создание пустой строки НАД курсором и переход в режим вставки
- a** вставка (переход в режим вставки) текста ПОСЛЕ курсора

Информация о файле и расположение в нем

- Ctrl+g** месторасположение в файле и информация о нем
- <номер строки>Shift+g** перемещение к заданной строке в файле
- Shift+g** перемещение к концу файла

Команды поиска и замены

% – поиск парных скобок «)», «]» или «}» (переход на соответствующую парную скобку)

:s/<было>/<станет> замена ТОЛЬКО первого найденного вхождения в текущей строке

:s/<было>/<станет>/g подстановка ГЛОБАЛЬНО во всей строке (заменит все найденные в строке вхождения)

:s/<было>/<станет>/gc подстановка глобально во всей строке (заменит все найденные в строке вхождения) с подтверждением каждой замены

:#,#s/<было>/<станет>/g замена всех вхождений последовательности символов между двумя строками (#,# – номера этих строк)

:%s/<было>/<станет>/g замена всех вхождений во всем файле

/<текст>

поиск <текста> ВПЕРЕД

?<текст>

поиск <текста> НАЗАД

n – переход к следующему вхождению искомой строки

:set ic – игнорировать регистр при поиске или замене
hlsearch – подсветка поиска
incsearch – инкрементальный поиск

Выполнение внешних команд

:!<команда> исполнение внешней команды
:w FILENAME запись текущего редактируемого файла на диск под именем FILENAME
:#,#w FILENAME сохранение строк от # до # в файл FILENAME
:r FILENAME считывание с диска файл FILENAME и помещение его в текущий файл следом за позицией курсора.

Учебник по использованию Vim

[# vimtutor ru](http://vimtutor.ru)

Консольный оконный менеджер Screen

Screen – полноэкранный оконный менеджер (консольный). Его суть состоит в том, что он мультиплексирует один физический терминал между несколькими процессами (каждому предоставляет свой виртуальный терминал – virtual terminal) и реализует дополнительную функциональность по управлению этими терминалами.

Часто используемые аргументы командной строки

- | | |
|-----------------------|--|
| -S sessionname | опция может использоваться для задания имени сеансу при его создании. |
| -ls -list | выводит список идентифицирующих сеансы screen строк. |
| -d -D | не запускает новый сеанс, а отключает вместо этого уже запущенный ранее. |
| -x sessionname | подключиться к заданной сессии screen. |
| -R | возобновить работу первого попавшегося отключённого сеанса. |
| -X | отправить указанную команду в работающий сеанс screen. |
| -wipe [match] | удаляет файлы уничтоженных сеансов вместо того чтобы помечать их как "dead" (мёртвые). |

Часто используемые комбинации/последовательности клавиш

Ctrl-a w Показать список окон.

Ctrl-a ? (help) Показать привязки клавиш.

Ctrl-a c Создать новое окно с запущенным интерпретатором и переключиться в это окно.

Ctrl-a C Очистить экран.

Ctrl-a d Отключить screen от этого терминала.

Alt+q Переход по окнам влево

Alt+w Переход по окнам вправо

Ctrl-d Завершить окно скрина или завершить весь скрин если было создано одно окно.

Ctrl-a > Записать буфер обмена в файл.

Ctrl-a < Прочитать файл обмена (screen-exchange) в буфер обмена.

Ctrl-a] Записать содержимое буфера обмена в стандартный поток ввода текущего окна.

Области

C-a S - Разделить текущий регион на два новых.

C-a tab - Переключить фокус ввода на следующую область.

C-a X - Заккрыть текущую область.

Прокрутка, копирование, вставка в screen

C-a esc режим прокрутки.

<PageUp> и **<PageDown>** работают как механизм скроллинга.

Копирование работает при выделении необходимой области от начала и до конца. Это происходит путем отметки от текущей позиции курсора нажатием клавиш **<Space>**. Перемещения по содержимому между метками происходит клавишами **<h>**, **<j>**, **<k>**, **<l>**, как в редакторе vim (или клавишами-стрелками). Окончание копирования – повторное нажатие **<Space>**. Содержимое буфера обмена может быть вставлено в любое другое окно текущей сессии Screen командой **C-a]**.

Системная утилита rsyslog

Syslog – система регистрации системных сообщений.

Система syslog состоит из трех частей:

- демон **rsyslogd** и его конфигурационный файл **/etc/rsyslog.conf**;
- библиотечные программы **openlog**, **rsyslog**, **closelog**, которые используются разработчиками для обмена данными с **rsyslogd**;
- программа пользовательского уровня **logger** для записи сообщений.

Конфигурационный файл демона rsyslogd

Формат **/etc/rsyslog.conf** файла следующий:

<селектор> **<действие>**

Например:

mail.err	/var/log/mail.errors
-----------------	-----------------------------

Селектор – это правило отбора записей для журнала. Для прошедших отбор записей выполняется указанное **действие**. **Селектор** записывается в составном виде. В общем виде это выглядит, как:

<средство>.<уровень>

Примеры селекторов:

средство.уровень

средство1,средство2.уровень

средство1.уровень1;средство2.уровень2

***.уровень**

***.уровень;средство.none**

Далее в таблицах перечислены основные имена **средств** и **уровней (level)** серьезности системы syslog.

**Средство
(источник)****Программы и подсистемы, использующие его**

kern	Ядро системы
user	Пользовательские процессы
mail	Система электронной почты
daemon	Системные демоны
auth	Системы защиты и полномочий
authpriv	Системы защиты и полномочий
lpr	Система печати
news	Система новостей
cron	Демон cron
mark	Метка времени (каждые 20 минут)
security	Различные службы безопасности
local0-7	Восемь уровней локального сообщения
syslog	Внутренние сообщения системы syslog
ftp	Демон ftpd
*	Все возможные (вышеперечисленные) средства

Уровень	Значение уровня
emerg	Экстренные ситуации. Система в панике, чрезвычайно нестабильна. Продолжение работы невозможно
alert	Срочные ситуации. Система может продолжить работу, но эту ошибку следует устранить немедленно
crit	Критические ошибки. Проблемы с аппаратным обеспечением или серьезные нарушения работы программного обеспечения
err	Разнообразные ошибки
warning	Предупреждения
notice	Общая информация
info	Информационные сообщения
debug	Отладочная информация
none	Специальный уровень, означающий – «ничего не записывать в данной категории». Он обычно применяется для исключения информации из групповых записей
*	Все сообщения указанного средства

Пример:

mail.info /var/log/maillog

Использование операторов сравнения

Например:

mail.info /var/log/maillog
mail.=debug /var/log/maillog.debug

или

mail.info /var/log/maillog
mail.>info /var/log/maillog.debug

logrotate

Далее приведен пример файла настроек (etc/logrotate.conf).

```
"/home/<user-www>/logs/access.log"
{
    rotate 10                # кол-во хранимых сжатых фрагментов
    size=16M                 # максимальный размер несжатого файла; пока размер текущего
                             # файла журнала не превысит данный порог, файл не будет "ротирован"

    missingok                # отсутствие файла не является ошибкой
    nocopytruncate           # не сбрасывать файл журнала после копирования
    nocreate                  # не создавать пустой журнал
    nodelaycompress          # не откладывать сжатие файла на следующий цикл
    nomail                    # не отправлять содержимое удаляемых (старых) журналов по почте
    notifempty                # не обрабатывать пустые файлы
    noolddir                  # держать все файлы в одном и том же каталоге
    compress                  # сжимать
    postrotate
        /usr/bin/killall -HUP httpd
    endscript                # Между postrotate и endscript расположены команды
                             # интерпретатора sh(1), исполняемые непосредственно после ротации.
                             # В данном примере сюда помещена команда kill, перезапускающая
                             # httpd-сервер. Это необходимо для нормальной процедуры
}
```

Планировщики задач Cron, AT, Batch

cron

cron – демон-планировщик задач, использующийся для периодического выполнения заданий в заданное время.

crontab -e команда позволяет редактировать файл, содержащий список заданий в текстовом редакторе

Файл **/etc/crontab** имеет следующий формат значений :

#minute	hour	mday	month	wday	command
0,20,40	*/5	1-31	*	mon-fri	echo hello

Допустимые значения:

Minute минута от 0 до 59.

Hour час от 0 до 23.

Mday день месяца от 1 до 31.

Month месяц от 1 до 12 (можно три буквы из названия месяца, регистр не имеет значения от jan до dec).

Wday день недели от 0 до 7 (0 это воскресенье, можно писать от sun до sat).

at (batch)

at(batch) - планирование выполнения команд в определенное время.

Синтаксис:

at время [дата] [+задержка]

at -r идентификатор_задания ...

at -l [идентификатор_задания ...]

-r - Удалить задания, запланированные ранее с помощью **at** или **batch**, по идентификаторам_заданий.

-l - Вывести информацию о запланированных заданиях по идентификаторам_заданий.

Пример планирования задания:

```
# echo "sh sfile" | at 1900 thursday next week
```

**Анализ логов с помощью less, head, tail,
split утилит**

less

less — консольная программа в UNIX-подобных системах используемая для просмотра содержимого текстовых файлов на экране.

Синтаксис команды:

less [параметры] <имя_файла>

Основные команды:

PageUp	Переместиться на одну страницу вверх
PageDown	Переместиться на одну страницу вниз
SPACE	Переместиться на одну страницу вниз
ENTER	Переместиться на одну строку вниз
d	Переместиться на полстраницы вниз
b	Переместиться на одну страницу вверх
/образец/	Поиск по заданному образцу вперед
?образец?	Поиск по заданному образцу назад
h	Помощь
q	Выход

tail

tail — утилита в UNIX, выводящая несколько последних строк из файла.

Синтаксис:

tail [параметры] имя_файла

-n - <количество строк> (или просто - <количество строк>)

позволяет изменить количество выводимых строк.

-f утилита tail следит за файлом, а новые строки (добавляемые в конец файла другим процессом) автоматически выводятся на экран в реальном времени.

Пример:

tail -20 /var/log/messages

tail -f /var/log/messages

head

head — утилита в UNIX и UNIX-подобных системах, выводящая первые **n** строк из файла, по умолчанию **n** равно 10.

Синтаксис:

head [-строк] [файл...]

Зачастую используется в качестве элемента конвейера обработки текста различными утилитами, чтобы ограничить вывод информации:

```
# df | head -n 2 | tail -n 1  
/dev/da0s1a 496M 53M 403M 12% /
```

split

split — команда, копирующая файл и разбивающая его на отдельные файлы заданной длины.

Синтаксис:

split [-l строк] [-b байтов[km]] [файл [выходной_префикс]]

- l** В каждом выходном файле должно оказаться не более указанного количества строк
- b** В каждом выходном файле должно оказаться не более указанного количества байтов. Дополнительные спецификаторы обозначают:
 - k** килобайты,
 - m** мегабайты
- n частей** (не во всех версиях split) Разбивает файл на **n** равных по размеру частей



Полезные утилиты и программы

Содержание

- Текстовый редактор Vim
- Консольный оконный менеджер Screen
- Системная утилита Syslog
- Планировщики задач Cron, AT, Batch
- Анализ логов с помощью Less, head, tail, split утилит

**Текстовый редактор
Vim**

Vim - это мощный редактор, имеющий множество команд с полной свободой настройки и автоматизации.

Режимы редактора

Normal mode – обычный режим (ввод команд)

Insert mode – режим вставки (редактирование текста)

Запуск редактора

```
# vim file.txt
```

Перемещение курсора по экрану

h – влево	^
j – вниз	k
k – вверх	< h l >
l – вправо	j

Завершение работы с Vim

<Esc>	вернуться в обычный режим
:q	выйти из vim
:q!	выйти из vim без сохранения любых сделанных изменений
:wq	сохранить изменения и выйти
:wq!	принудительно сохранить изменения и выйти

Удаление текста

x	удалить символ выделенный курсором
dw	удалить текст от курсора до конца слова, включая завершающий пробел
d\$	удаление от курсора до конца строки
d^	удаление от курсора до начала строки
de	удаление от курсора до конца слова, НЕ включая завершающий пробел
dd	удаление всей строки

Ввод текста

i перейти в режим вставки
<Esc> вернуться в обычный режим

Команды и объекты

d[число]объект формат команды «удаление» d таков.

Где:

d	команда удаления.
число	сколько раз исполнить команду
объект	с чем команда должна быть выполнена

Краткий список объектов:

w	от курсора до конца слова, включая завершающий пробел.
e	от курсора до конца слова, НЕ включая завершающий пробел.
\$	от курсора до конца строки.
^	от курсора до начала строки.

Команды отмены и вставки

- u** отмена результата работы предыдущей команды
- U** отмена исправлений во всей строке
- p** вставка после курсора последнего удаленного (или скопированного). текста (если была удалена строка, то она будет помещена в строке под курсором).
- r** замена единичного символа под курсором (далее должен следовать символ на который он заменяется).

c[<число>]<объект> Изменение текста (удаляет объект и переводит в режим вставки

- o** создание пустой строки ПОД курсором и переход в режим вставки
- O** создание пустой строки НАД курсором и переход в режим вставки
- a** вставка (переход в режим вставки) текста ПОСЛЕ курсора

Информация о файле и расположение в нем

- Ctrl+g** месторасположение в файле и информация о нем
- <номер строки>Shift+g** перемещение к заданной строке в файле
- Shift+g** перемещение к концу файла

Команды поиска и замены

% – поиск парных скобок «)», «]» или «}» (переход на соответствующую парную скобку)

:s/<было>/<станет> замена ТОЛЬКО первого найденного вхождения в текущей строке

:s/<было>/<станет>/g подстановка ГЛОБАЛЬНО во всей строке (заменит все найденные в строке вхождения)

:s/<было>/<станет>/gc подстановка глобально во всей строке (заменит все найденные в строке вхождения) с подтверждением каждой замены

:#,#s/<было>/<станет>/g замена всех вхождений последовательности символов между двумя строками (#,# – номера этих строк)

:%s/<было>/<станет>/g замена всех вхождений во всем файле

/<текст> поиск <текста> ВПЕРЕД

?<текст> поиск <текста> НАЗАД

n – переход к следующему вхождению искомой строки

Замечание: Это очень удобно при отладке программ с пропущенными скобками!

Строковые команды (начинающиеся с «:», а также «/» и «?») должны завершаться нажатием <Enter>

:set ic – игнорировать регистр при поиске или замене
hlsearch – подсветка поиска
incsearch – инкрементальный поиск

Выполнение внешних команд

:!<команда> исполнение внешней команды
:w FILENAME запись текущего редактируемого файла на диск под именем FILENAME
:#,#w FILENAME сохранение строк от # до # в файл FILENAME
:r FILENAME считывание с диска файл FILENAME и помещение его в текущий файл следом за позицией курсора.

Учебник по использованию Vim

vimtutor ru

**Консольный оконный
менеджер Screen**

Screen – полноэкранный оконный менеджер (консольный). Его суть состоит в том, что он мультиплексирует один физический терминал между несколькими процессами (каждому предоставляет свой виртуальный терминал – virtual terminal) и реализует дополнительную функциональность по управлению этими терминалами.

Часто используемые аргументы командной строки

-S sessionname	опция может использоваться для задания имени сеансу при его создании.
-ls -list	выводит список идентифицирующих сеансы screen строк.
-d -D	не запускает новый сеанс, а отключает вместо этого уже запущенный ранее.
-x sessionname	подключиться к заданной сессии screen.
-R	возобновить работу первого попавшегося отключённого сеанса.
-X	отправить указанную команду в работающий сеанс screen.
-wipe [match]	удаляет файлы уничтоженных сеансов вместо того чтобы пометать их как "dead" (мёртвые).

Screen – полноэкранный оконный менеджер (консольный). Его суть состоит в том, что он мультиплексирует один физический терминал между несколькими процессами (каждому предоставляет свой виртуальный терминал – virtual terminal) и реализует дополнительную функциональность по управлению этими виртуальными терминалами.

Наиболее интересны две функции screen. Первая – это возможность запуска нескольких консольных программ (window, shell) в одном терминале. А таким образом, нет необходимости множество раз подключаться к удаленному серверу для открытия нескольких shell. Вторая – это возможность отключения (detach) сессии screen с продолжением выполнения всех запущенных под сессией программ. Возможно также и повторное подключение (reattach) для продолжения работы с сессией.

При запуске screen (без параметров), создается новая сессия (screen session) с одним полноэкранным окном (window), в котором уже запущена оболочка пользователя. Также, в зависимости от настроек screen, может отображаться вступительное сообщение (только при создании сессии, до запуска shell).

Далее, в рамках данной сессии, есть возможность создавать новые окна с запущенными программами в них, закрывать существующие окна, переключаться между ними, просматривать список окон, прокручивать буфер истории, копировать и вставлять текст, разбивать экран на области (в каждой может быть отображено любое окно) и многое другое. Причем, каждое окно содержит одну программу, которая выполняется в изолированной среде (в своем виртуальном терминале). Окно продолжает существовать до тех пор, пока запущена программа. Программа работает даже в том случае, когда окно не активно (не отображается) или сессия screen отключена (detached). Сессия screen существует до тех пор, пока существует хотя бы одно окно.

Простейший способ выйти, то есть завершить сессию, – это закрыть все окна, завершив все программы.

Поведение screen управляется командами. Различают несколько видов команд/опций для screen:
Аргументы командной строки – «command-line options». Используются при запуске screen (например: «-ls»).

Комбинации/последовательности клавиш – «key bindings». Используются внутри сессии screen'ов (например: «Ctrl+a+d»). Здесь вместо записи вида «Ctrl+a+c» используется запись вида «C-a c».
Команды управления/инициализации сессии. Набираются в командной строке внутри сессии screen'ов (например: «bufferfile /tmp/1»).

-ls

Сеансы, которые отмечены словом "detached" могут быть продолжены с помощью команды "screen -r". Сеансы, которые отмечены словом "attached", работают, и у них есть управляющий терминал. Если сеанс работает в многопользовательском режиме, он отмечен словом "multi". Сеансы, которые отмечены словом "unreachable" или работают на другом хосте, или умерли (dead). Недоступный (unreachable) сеанс считается мёртвым, если его имя соответствует или имени локального хоста, или указанному параметру (если такой есть). Как описывать строки для выбора рассказывается в описании ключа -r. Сеансы, отмеченные как "dead", нужно проверить и удалить. Если вы не уверены, нужно ли удалять какой-то сеанс, вы можете обратиться к системному администратору (если это не вы сами, иначе может возникнуть бесконечная рекурсия). Удалить сеансы можно с помощью опции -wipe.

-d|-D [pid.tty.host]

не запускает новый сеанс, а отключает вместо этого уже запущенный ранее. Достигается тот же эффект, что и в случае нажатия клавиш "C-a d" на управляющем терминале screen. Ключ -D эквивалентен ключу power detach. Если ни к одному сеансу нельзя обратиться, опция игнорируется.

Часто используемые комбинации/последовательности клавиш

Ctrl-a w Показать список окон.

Ctrl-a ? (help) Показать привязки клавиш.

Ctrl-a c Создать новое окно с запущенным интерпретатором и переключиться в это окно.

Ctrl-a C Очистить экран.

Ctrl-a d Отключить screen от этого терминала.

Alt+q Переход по окнам влево

Alt+w Переход по окнам вправо

Ctrl-d Завершить окно скрина или завершить весь скрин если было создано одно окно.

Ctrl-a > Записать буфер обмена в файл.

Ctrl-a < Прочитать файл обмена (screen-exchange) в буфер обмена.

Ctrl-a] Записать содержимое буфера обмена в стандартный поток ввода текущего окна.

bufferfile [*exchange-file*]

Изменить имя файла, использующегося для чтения и записи буфера обмена. Если имя файла не указано, реактивируется настройка по умолчанию (/tmp/screen-exchange). В этом примере системный файл passwd копируется в окно (при помощи буфера обмена, в котором остаётся копия):

C-a : bufferfile /etc/passwd C-a < C-a] C-a : bufferfile

Области

C-a S - Разделить текущий регион на два новых.

C-a tab - Переключить фокус ввода на следующую область.

C-a X - Закрыть текущую область.

Прокрутка, копирование, вставка в screen

C-a esc режим прокрутки.

<PageUp> и **<PageDown>** работают как механизм скроллинга.

Копирование работает при выделении необходимой области от начала и до конца. Это происходит путем отметки от текущей позиции курсора нажатием клавиш **<Space>**. Перемещения по содержимому между метками происходит клавишами **<h>**, **<j>**, **<k>**, **<l>**, как в редакторе vim (или клавишами-стрелками). Окончание копирования – повторное нажатие **<Space>**. Содержимое буфера обмена может быть вставлено в любое другое окно текущей сессии Screen командой **C-a]**.

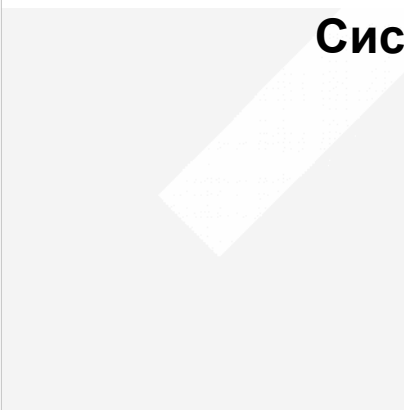
Экран screen можно поделить на области по горизонтали, в каждой области будет отображаться отдельный терминал. Новая область создается командой **C-a S**. Чтобы попасть во вновь созданную область нужно переключиться – **C-a tab**. Теперь можно циклически переключаться между терминалами с помощью команд **C-a space** или **C-a backspace**. Области закрывается командой **C-a X** (или **C-a Q**).

Размер текущей области может быть изменен командой **C-a +** или **C-a -**. По умолчанию высота меняется с шагом 3 строки.

C-a S - Разделить текущий регион на два новых.

C-a tab - Переключить фокус ввода на следующую область.

C-a X - Закрыть текущую область.



**Системная утилита
rsyslog**

Syslog – система регистрации системных сообщений.

Система syslog состоит из трех частей:

- демон **rsyslogd** и его конфигурационный файл **/etc/rsyslog.conf**;
- библиотечные программы **openlog**, **rsyslog**, **closelog**, которые используются разработчиками для обмена данными с **rsyslogd**;
- программа пользовательского уровня **logger** для записи сообщений.

Конфигурационный файл демона rsyslogd

Формат **/etc/rsyslog.conf** файла следующий:

<селектор> **<действие>**

Например:

mail.err	/var/log/mail.errors
----------	----------------------

Syslog – система регистрации системных сообщений

Система syslog в ОС FreeBSD представляет собой систему регистрации и управления информацией, которую генерируют ядро системы и системные утилиты. С ее помощью можно сортировать сообщения по их источникам, степени важности, а так же направлять в файлы, на терминалы или удаленные системы регистрации. Система syslog состоит из трех частей:

демон **syslogd** и его конфигурационный файл **/etc/syslog.conf**;

библиотечные программы **openlog**, **syslog**, **closelog**, которые используются разработчиками для обмена данными с **syslogd**;

программа пользовательского уровня **logger** для записи сообщений.

Демон **syslogd** запускается во время начальной загрузки и работает непрерывно. Если был изменен конфигурационный файл **/etc/syslog.conf**, нужно перезапустить **syslogd**, послав ему сигнал HUP (или 1). Например, таким образом:

```
# kill -1 `cat /var/run/syslog.pid`
```

Демон **syslogd** по этой команде закроет все файлы регистрации, перечитает файл **/etc/syslog.conf** и запустит регистрацию снова.

Конфигурационный файл демона syslogd

Действиями демона **syslogd** управляет файл конфигурации **/etc/syslog.conf**. Это простой текстовый файл, в котором пустые строки и строки со знаком «#» в первой позиции игнорируются (есть исключения!). Формат файла следующий:

<селектор> **<действие>**

Например:

mail.err **/var/log/mail.errors**

Эта строка обеспечит запись всех ошибок, связанных с доставкой почты в файл **/var/log/mail.errors**.

Селектор – это правило отбора записей для журнала. Для прошедших отбор записей выполняется указанное **действие**. **Селектор** записывается в составном виде. В общем виде это выглядит, как:

<средство>.<уровень>

Примеры селекторов:

средство.уровень
средство1,средство2.уровень
средство1.уровень1;средство2.уровень2
***.уровень**
***.уровень;средство.none**

Далее в таблицах перечислены основные имена **средств** и **уровней (level)** серьезности системы syslog.

Селектор – это правило отбора записей для журнала. Для прошедших отбор записей выполняется указанное **действие**.

Селектор записывается в составном виде. В общем виде это выглядит, как:

<средство>.<уровень>

К тому же в поле **<селектор>** может содержаться один или несколько **селекторов**, разделенных точкой с запятой. **Селектор** может содержать группу **средств**, разделенных запятыми. В общем случае, **средство (facility)** – это программа или подсистема, которая отправляет сообщение (источник записей для журналирования). Иногда, вместо названия **средство** используется название **категория**.

Примеры селекторов:

средство.уровень
средство1,средство2.уровень
средство1.уровень1;средство2.уровень2
***.уровень**
***.уровень;средство.none**

**Средство
(источник)**
Программы и подсистемы, использующие его

kern	Ядро системы
user	Пользовательские процессы
mail	Система электронной почты
daemon	Системные демоны
auth	Системы защиты и полномочий
authpriv	Системы защиты и полномочий
lpr	Система печати
news	Система новостей
cron	Демон cron
mark	Метка времени (каждые 20 минут)
security	Различные службы безопасности
local0-7	Восемь уровней локального сообщения
syslog	Внутренние сообщения системы syslog
ftp	Демон ftpd
*	Все возможные (вышеперечисленные) средства

Уровень	Значение уровня
emerg	Экстренные ситуации. Система в панике, чрезвычайно нестабильна. Продолжение работы невозможно
alert	Срочные ситуации. Система может продолжить работу, но эту ошибку следует устранить немедленно
crit	Критические ошибки. Проблемы с аппаратным обеспечением или серьезные нарушения работы программного обеспечения
err	Разнообразные ошибки
warning	Предупреждения
notice	Общая информация
info	Информационные сообщения
debug	Отладочная информация
none	Специальный уровень, означающий – «ничего не записывать в данной категории». Он обычно применяется для исключения информации из групповых записей
*	Все сообщения указанного средства

<http://www.k-max.name/linux/syslogd-and-logrotate/>

Пример:

mail.info /var/log/maillog

Использование операторов сравнения

Например:

mail.info /var/log/maillog
mail.=debug /var/log/maillog.debug

или

mail.info /var/log/maillog
mail.>info /var/log/maillog.debug

Описание правила отбора источника информации включает в себя **средство** и **уровень** детализации, разделенные точкой. Когда вы указываете **уровень**, по умолчанию в журнал записываются сообщения, **уровень** которых выше или равен указанному. В качестве примера рассмотрим эту запись из файла /etc/syslog.conf:

mail.info /var/log/maillog

В журнал /var/log/maillog будут записаны сообщения от почтовой системы, с **уровнем** выше или равным уровню info.

В журнал /var/log/maillog будут записаны сообщения от почтовой системы, с **уровнем** выше или равным уровню info.

В /etc/syslog.conf вы можете использовать операторы сравнения.

Таблица 3 – Операторы сравнения

Оператор Значение оператора
>= Больше или равно указанному уровню (по умолчанию)
< Менее чем указанный уровень
= Равно указанному уровню
> Больше указанного уровня
<= Менее или равно указанному уровню
!= Инвертировать смысл последующего оператора. Например: «!=», «!>»

Например:

mail.info /var/log/maillog
mail.=debug /var/log/maillog.debug

logrotate

Далее приведен пример файла настроек (etc/logrotate.conf).

```
"/home/<user-www>/logs/access.log"
{
    rotate 10                # кол-во хранимых сжатых фрагментов
    size=16M                # максимальный размер несжатого файла; пока размер текущего
                           # файла журнала не превысит данный порог, файл не будет "ротирован"
    missingok               # отсутствие файла не является ошибкой
    nocopytruncate          # не сбрасывать файл журнала после копирования
    nocreate                # не создавать пустой журнал
    nodelaycompress         # не откладывать сжатие файла на следующий цикл
    nomail                  # не отправлять содержимое удаляемых (старых) журналов по почте
    notifempty              # не обрабатывать пустые файлы
    noolddir                # держать все файлы в одном и том же каталоге
    compress                # сжимать
    postrotate
        /usr/bin/killall -HUP httpd
    endscrip                # Между postrotate и endscrip расположены команды
                           # интерпретатора sh(1), исполняемые непосредственно после ротации.
                           # В данном примере сюда помещена команда kill, перезапускающая
                           # httpd-сервер. Это необходимо для нормальной процедуры
}
```


Планировщики задач Cron, AT, Batch

Планировщик задач cron

cron – демон-планировщик задач в FreeBSD (и других UNIX-подобных операционных системах), использующийся для периодического выполнения заданий в заданное время. Позволяет неоднократно запуск заданий. Т.е. задание можно запустить в определенное время или через определенный промежуток времени.

При загрузке системы, запускается демон cron и проверяет очередь заданий at (однократный запуск в указанное время), заданий общесистемного файла crontab и заданий пользователей в пользовательских файлах crontab.

Демон cron проверяет каталог /var/cron/tabs на наличие файлов crontab, файлы crontab имеют имена пользователей, соответствующие именам пользователей из /etc/passwd. Каждый пользователь может иметь только один файл crontab, записей в файле может быть несколько.

Файлы crontab содержат инструкции для демона cron, который запустит задание(я) описанное в файле crontab в заданное время. По умолчанию общесистемный файл crontab (/etc/crontab) имеет следующий вид:

```
# /etc/crontab - root's crontab for FreeBSD
#
# $FreeBSD: src/etc/crontab,v 1.32 2002/11/22 16:13:39 tom Exp $
#
SHELL=/bin/sh
PATH=/etc:/sbin:/usr/bin:/usr/sbin
HOME=/var/log
#
#minute hour mday month wday who command
#
*/5 * * * * root /usr/libexec/atrun
#
# Save some entropy so that /dev/random can re-seed on boot.
*/11 * * * * operator /usr/libexec/save-entropy
#
# Rotate log files every hour, if necessary.
0 * * * * root newsyslog
#
# Perform daily/weekly/monthly maintenance.
1 3 * * * root periodic daily
15 4 * * 6 root periodic weekly
30 5 1 * * root periodic monthly
#
# Adjust the time zone if the CMOS clock keeps local time, as opposed to
# UTC time. See adjkerntz(8) for details.
1,31 0-5 * * * root adjkerntz -a
```

PORTAone Building VoIP Revenue

cron

cron – демон-планировщик задач, использующийся для периодического выполнения заданий в заданное время.

crontab -e команда позволяет редактировать файл, содержащий список заданий в текстовом редакторе

Файл /etc/crontab имеет следующий формат значений :

#minute	hour	mday	month	wday	command
0,20,40	*/5	1-31	*	mon-fri	echo hello

Допустимые значения:

Minute	минута от 0 до 59.
Hour	час от 0 до 23.
Mday	день месяца от 1 до 31.
Month	месяц от 1 до 12 (можно три буквы из названия месяца, регистр не имеет значения от jan до dec).
Wday	день недели от 0 до 7 (0 это воскресенье, можно писать от sun до sat).

Файла указывает, что:

/usr/libexec/atrun будет запускаться от имени пользователя root каждую 5-ю минуту часа;
/usr/libexec/save-entropy будет запускаться от имени пользователя operator каждую 11-ю минуту часа;
newsyslog будет запускаться от имени пользователя root каждый час (на нулевой минуте);
periodic daily будет запускаться от имени пользователя root каждый день на первой минуте третьего часа;
periodic weekly будет запускаться от имени пользователя root каждую субботу в 4:15;
periodic monthly будет запускаться от имени пользователя root каждый первый день месяца в 5:30.
Вначале crontab объявлены переменные окружения, используемые для выполнения заданий. Некоторые переменные автоматически устанавливаются демоном cron согласно конфигурации пользователя-владельца файла crontab.

SHELL=/bin/sh – использовать для запуска команд /bin/sh, если переменная не указана, то значение будет взято из /etc/passwd для пользователя являющимся владельцем файла (это более актуально для пользовательских файлов crontab).

HOME=/var/log/ – корневой каталог для пользователя (параметр не обязательный). При необходимости доступа к специальным свойствам интерпретатора, значения переменных SHELL и HOME можно изменить, не зависимо оттого что прописано в /etc/passwd.

После того, как демон cron запущен и прочел содержимое всех файлов crontab, он бездействует, просыпаясь каждую минуту и проверяя не требуется ли запуск какой-либо команды в данную минуту, или не появился ли новый файл crontab, который необходимо обработать. Демон cron определяет изменения по времени модификации файлов или каталогов, такое его свойство избавляет от необходимости перезапуска демона (после изменения файлов crontab).

Для использования файлов crontab пользователями (даже суперпользователем), нужно использовать команду crontab. Команда служит для создания, изменения и добавления файла для демона cron.

Доступ в каталог /var/cron/tabs непривилегированному пользователю закрыт, поэтому, чтобы посмотреть пользователем «user» есть ли у него файл crontab, достаточно набрать команду **crontab -l**, если файл существует – будет показано его содержимое.

Для удаления файла используется команда **crontab -r**.

Для редактирования – **crontab -e**.

Для управления чужими файлами crontab пользователем «root», также указывается имя пользователя – **crontab -u user_name**. Для создания crontab из файла (соответствующего формата) – **crontab file**.

Для редактирования используется редактор, заданный переменной окружения VISUAL или EDITOR.

Формат и значения полей

Каждая команда в пользовательском файле crontab занимает одну строку и состоит из шести полей. Пользовательские файлы crontab находятся в каталоге /var/cron/tabs/. Общесистемный crontab содержит на одно поле больше – пользователь (6 поле), от имени которого выполняется указанная команда (7 поле).

```
Общий формат команды:
#minute hour mday month wday command
0,20,40 */5 1-40 * mon-fri echo hello
```

Допустимые значения:

- Minute – минута от 0 до 59.
- Hour – час от 0 до 23.
- Mday – день месяца от 1 до 31.
- Month – месяц от 1 до 12 (можно три буквы из названия месяца, регистр не имеет значения от jan до dec).
- Wday – день недели от 0 до 7 (0 это воскресенье, можно писать от sun до sat).

at (batch)

at(batch) - планирование выполнения команд в определенное время.

Синтаксис:

at время [дата] [+задержка]

at -r идентификатор_задания ...

at -l [идентификатор_задания ...]

-r - Удалить задания, запланированные ранее с помощью **at** или **batch**, по идентификаторам_заданий.

-l - Вывести информацию о запланированных заданиях по идентификаторам_заданий.

Пример планирования задания:

```
# echo "sh sfile" | at 1900 thursday next week
```

at выполняет команды в указанное время.

batch выполняет команды, когда уровень загрузки системы позволяет, другими словами, когда средняя загрузка падает ниже 0,8, или значение, указанное в вызове **atrunp**.

НАЗВАНИЕ

at - планирование выполнения команд в определенное время

СИНТАКСИС

at время [дата] [+задержка] **at -r** идентификатор_задания ... **at -l** [идентификатор_задания ...] **ОПИСАНИЕ**

Команда **at** в первом из приведенных вариантов читает со стандартного ввода задание, выполнение которого планируется на указанное время. Смысл опций двух других вариантов команды **at** таков: **-r** удалить задания, запланированные ранее с помощью **at** или **batch**(1), по идентификаторам_заданий. Идентификаторы сообщаются командами **at** и **batch**. Их можно узнать также по команде **at -l**. Только суперпользователь может удалять чужие задания. **-l** вывести информацию о запланированных заданиях по идентификаторам_заданий. Если идентификаторы не указаны, выдается список всех заданий, запланированных пользователем и еще не выполненных. Если стандартный вывод и стандартный протокол не переназначены, то весь вывод запланированных команд пересылается пользователю по почте. Переменные окружения **shell'a**, текущий каталог, маска режима создания файлов и максимальный размер файлов [см. **umask(1)** и **ulimit(1)**] сохраняются, то есть задание выполняется в том же окружении, том же каталоге и т.д. Дескрипторы открытых файлов, прерывания и приоритет теряются.

Пользователю разрешается выполнять команду **at** только при условии, что его имя встречается в файле **/usr/lib/ cron/at.allow**. Если этого файла не существует, то проверяется файл **/usr/lib/cron/at.deny**, для того чтобы узнать, не запрещен ли пользователю доступ к **at**. Если оба файла отсутствуют, то только суперпользователю разрешено планировать выполнение задания. Если файл **at.deny** пуст, а **at.allow** отсутствует, то эти действия могут выполнять все. Файлы **at.allow** и **at.deny** содержат по одному имени в строке. Модифицировать эти файлы может только суперпользователь.

Время может быть указано 1, 2 или 4 цифрами. Если время состоит из одной или двух цифр, то оно обозначает часы; четырехзначное число обозначает часы и минуты. Время также может быть задано как два числа, разделенные двоеточием, что понимается как часы:минуты. Могут быть добавлены суффиксы **am** (до полудня) или **pm** (после полудня), в противном случае часы указываются от 0 до 23. Если необходимо указать время по Гринвичу, то можно добавить суффикс **zulu**. Распознаются специальные имена **noon** (полдень), **midnight** (полночь), **now** (сейчас), и **next** (следующий).

Дата может быть указана двумя способами: во-первых, в виде названия месяца, за которым следует число [и, может быть, год (через запятую)], а во-вторых, как день недели (полностью или сокращенный до 3 букв). Распознаются два специальных "дня" **today** (сегодня) и **tomorrow** (завтра). Если дата не задана, то предполагается сегодняшняя дата, если указанное время больше, чем текущее, и завтрашняя, если меньше. Если заданный месяц меньше, чем текущий и год явно не задан, то предполагается, что имеется в виду следующий год.

Дополнительная задержка представляет собой просто число, за которым следует одно из следующих слов: **minutes** (минуты), **hours** (часы), **days** (дни), **weeks** (недели), **months** (месяцы), или **years** (годы). Можно указывать единицу измерения и без числа, например **at now +minutes**.

Далее приведены примеры корректных команд:

at 0815am Jan 16 at 8:15am Jan 16 at now +1 day at 5 pm Friday Команда **at** выдает идентификатор задания и запланированное время его выполнения в стандартный протокол.

ПРИМЕР

Чтобы задание могло снова себя запланировать, следует вызвать **at** из **shell-файла** (назовем его **sfile**), включив в файл текст такого вида:

```
echo "sh sfile" | at 1900 thursday next week
```

ФАЙЛЫ
/usr/lib/cron

Основной каталог команд, связанных со временем.

/usr/lib/cron/at.allow

Список пользователей, которым разрешено выполнять команды **at** и **batch**.

/usr/lib/cron/at.deny

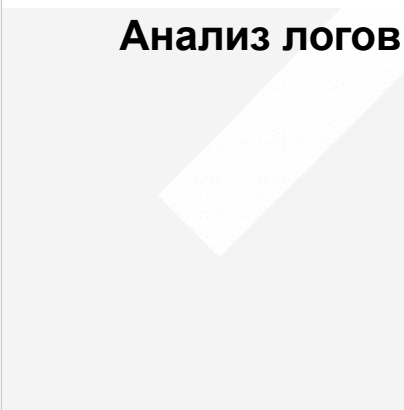
Список пользователей, которым запрещено выполнять команды **at** и **batch**.

/usr/lib/cron/queuedefs

Информация о планировании.

/usr/spool/cron/atjobs

Область накопления вывода.



**Анализ логов с помощью less, head, tail,
split утилит**

less

less — консольная программа в UNIX-подобных системах используемая для просмотра содержимого текстовых файлов на экране.

Синтаксис команды:

less [параметры] <имя_файла>

Основные команды:

PageUp	Переместиться на одну страницу вверх
PageDown	Переместиться на одну страницу вниз
SPACE	Переместиться на одну страницу вниз
ENTER	Переместиться на одну строку вниз
d	Переместиться на полстраницы вниз
b	Переместиться на одну страницу вверх
/образец/	Поиск по заданному образцу вперед
?образец?	Поиск по заданному образцу назад
h	Помощь
q	Выход

tail

tail — утилита в UNIX, выводящая несколько последних строк из файла.

Синтаксис:

tail [параметры] имя_файла

-n - <количество строк> (или просто - <количество строк>)

позволяет изменить количество выводимых строк.

-f утилита tail следит за файлом, а новые строки (добавляемые в конец файла другим процессом) автоматически выводятся на экран в реальном времени.

Пример:

tail -20 /var/log/messages

tail -f /var/log/messages

head

head — утилита в UNIX и UNIX-подобных системах, выводящая первые **n** строк из файла, по умолчанию **n** равно 10.

Синтаксис:

head [-строк] [файл...]

Зачастую используется в качестве элемента конвейера обработки текста различными утилитами, чтобы ограничить вывод информации:

```
# df | head -n 2 | tail -n 1  
/dev/da0s1a 496M 53M 403M 12% /
```

Зачастую используется в качестве элемента конвейера обработки текста различными утилитами, чтобы ограничить вывод информации:

split

split — команда, копирующая файл и разбивающая его на отдельные файлы заданной длины.

Синтаксис:

split [-l *строк*] [-b *байтов*[*km*]] [*файл* [*выходной_префикс*]]

-l В каждом выходном файле должно оказаться не более указанного количества строк

-b В каждом выходном файле должно оказаться не более указанного количества байтов. Дополнительные спецификаторы обозначают:

k килобайты,

m мегабайты

-n *частей* (не во всех версиях split) Разбивает файл на *n* равных по размеру частей

В качестве аргументов ей надо указать имя исходного файла и префикс имен выходных файлов. Имена выходных файлов будут состоять из этого префикса и двух дополнительных букв *aa*, *ab*, *ac* и т. д. (без пробелов и точек между префиксом и буквами). Если префикс имен файлов не задан, то по умолчанию используется *x*, так что выходные файлы будут называться *хаа*, *хаb* и т. д.

Разбить *файл* или поток на файлы указанного размера в строках или байтах. В результате операции создается набор файлов. Файлы получают имена, начинающиеся с указанного *выходного_префикса* (по умолчанию — *'x'*) и заканчивающиеся набором букв в соответствующем лексикографическом порядке.