Тема: Текстовые редакторы. Редактирование текстов. Печать в ОС Linux. Работа с архиваторами.

Содержание :

1. Средствами редактирования текстов в Linux.
2. Редактор текстов vi.
3. Редактор текстов joe.
4. Редактор текстов nano.
5. Подсистема печати в Linux. Вывод текста на принтер.
6. Упаковка файлов в ОС Linux. Связка упаковщиков tar+gzip.

Стоит различать текстовые редакторы и текстовые процессоры. Текстовые процессоры, например OpenOffice Writer или Microsoft Word, предназначены для создания **документов**, в которых, помимо собственно текста, содержится и различная **метаинформация** (информация об оформлении): размещение текста на странице, шрифт и т. п. Поскольку в текстовом формате не предусмотрено средств для сохранения информации об оформлении (там есть только символы и строки), текстовые процессоры используют собственные форматы для хранения данных. Текст, в котором нет никакой метаинформации об оформлении, называют "plain text" (только текст, "плоский", простой текст).

Важнейшее условие для текстового редактора в Linux – возможность работать в терминале, так как это основной способ управления системой. Ввод данных и редактирование должны полностью осуществляться средствами терминала, т. е. алфавитно-цифровыми и некоторыми функциональными клавишами.

Редактор Vi. В любой системе Linux, даже при самой минимальной конфигурации, всегда присутствует текстовый редактор. Первая версия была написана в 1976г.

В то время наиболее распространённым был редактор **ed**. Поскольку он был довольно сложным для «простого смертного», затем был разработан редактор **em** (editor for mortals — редактор для смертных). После этого он был модифицирован и получил название **ex**, на котором и основан **vi**.

В отличие от многих привычных редакторов, vi имеет **модальный** интерфейс. Это означает, что одни и те же клавиши в разных режимах работы выполняют разные действия. В редакторе vi есть два основных режима: **командный режим** и **режим вставки**. По умолчанию, работа начинается в командном режиме.

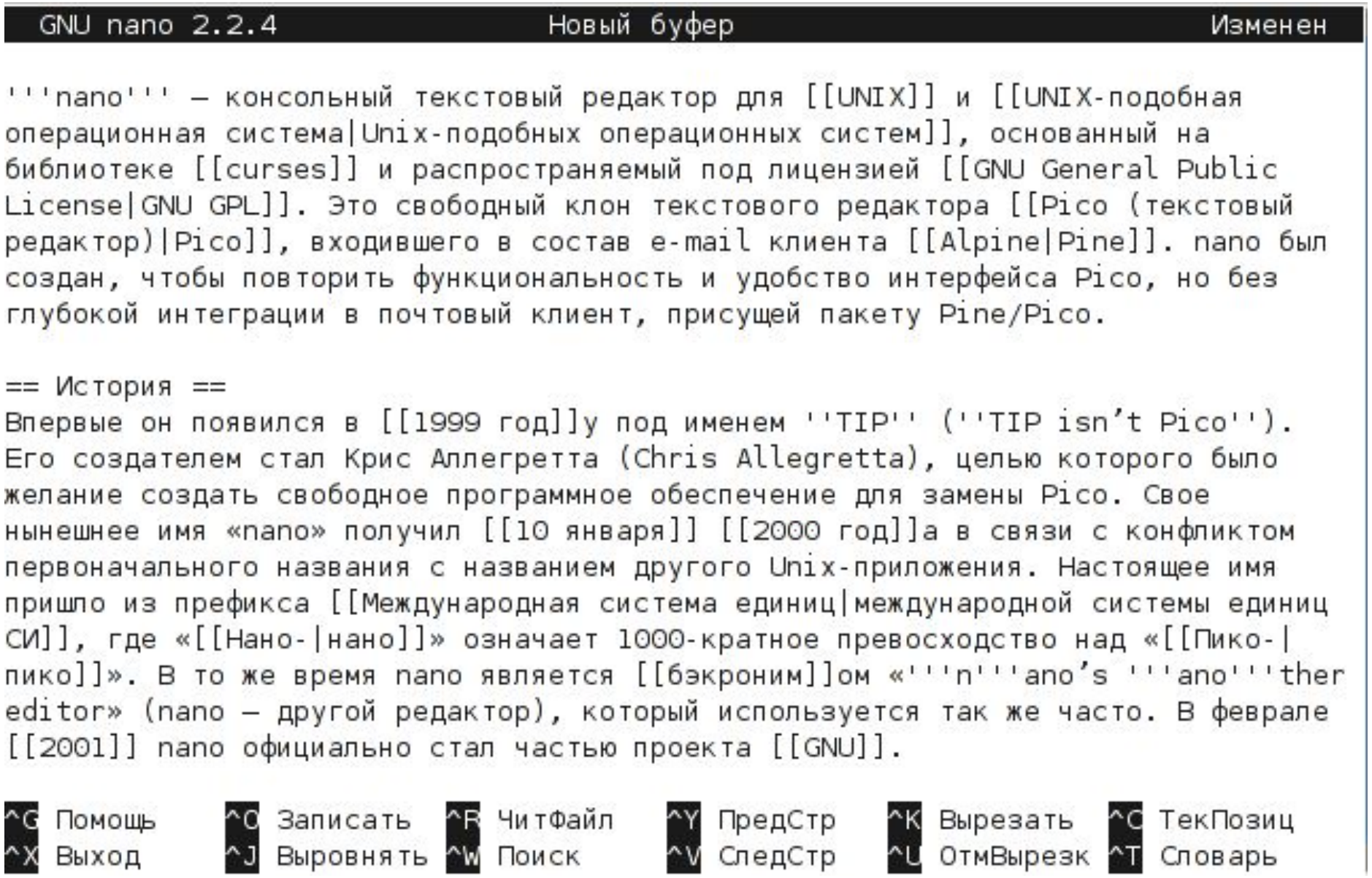
В ***режиме вставки*** клавиатура используется для набора текста. Для выхода в командный режим используется клавиша **Esc** или комбинация **Ctrl + c** .

В ***командном режиме*** алфавитные клавиши соответствуют командам перемещения и изменения текста. Так, **команды h, j, k, l** перемещают курсор на одну позицию влево, вниз, вверх, вправо соответственно, команда x удаляет один символ и т. д. Это позволяет работать без необходимости использования дополнительной клавиатуры и клавиш-модификаторов, таких, как **Ctrl , Alt** и т. д. Более сложные операции редактирования получаются комбинацией простых, например, **2dw** удаляет два слова. Для полнотекстовых операций имеется возможность задавать команды в подобии командной строки

Часто используемые команды

/str — Поиск строки str вперед. str может быть регулярным выражением
?str — Поиск строки str назад
n — Повторить поиск в том же направлении
N — Повторить поиск в обратном направлении
:e! — перезагрузить текущий файл
:К — перепрыгнуть на К-ую строку текстового файла
i — перейти в режим редактирования
a — перейти в режим редактирования после текущего символа
u — отменить последнее действие
. — повторить последнее действие
x — удалить символ под курсором
yy — копировать строку
dd — вырезать строку
p — вставить
J — склеить две строки
:w — сохранить файл на диске
:wq — выход с сохранением файла (shift + ZZ)
:q — выход
:q! — выход без сохранения файла
:r — вставить в документ другой файл

Чтобы получить подробную помощь по редактору vi, выполните команду man vi в Unix shell (q — выход из справки)


GNU nano 2.2.4 Новый буфер Изменен
'''nano''' — консольный текстовый редактор для [[UNIX]] и [[UNIX-подобная операционная система|Unix-подобных операционных систем]], основанный на библиотеке [[curses]] и распространяемый под лицензией [[GNU General Public License|GNU GPL]]. Это свободный клон текстового редактора [[Pico (текстовый редактор)|Pico]], входившего в состав e-mail клиента [[Alpine|Pine]]. nano был создан, чтобы повторить функциональность и удобство интерфейса Pico, но без глубокой интеграции в почтовый клиент, присущей пакету Pine/Pico.
== История ==
Впервые он появился в [[1999 год]]у под именем ''TIP'' (''TIP isn't Pico''). Его создателем стал Крис Аллегретта (Chris Allegretta), целью которого было желание создать свободное программное обеспечение для замены Pico. Свое нынешнее имя «nano» получил [[10 января]] [[2000 год]]а в связи с конфликтом первоначального названия с названием другого Unix-приложения. Настоящее имя пришло из префикса [[Международная система единиц|международной системы единиц СИ]], где «[[Нано-|нано]]» означает 1000-кратное превосходство над «[[Пико-|пико]]». В то же время nano является [[бэкроним]]ом «'''n'''ano's '''ano'''ther editor» (nano — другой редактор), который используется так же часто. В феврале [[2001]] nano официально стал частью проекта [[GNU]].
^G Помощь ^O Записать ^R ЧитФайл ^Y ПредСтр ^K Вырезать ^C ТекПозиц
^X Выход ^J Выровнять ^W Поиск ^V СледСтр ^U ОтмВырезк ^T Словарь

РЕДАКТОР joe

Ознакомиться практически со всеми возможностями редактора joe можно посредством его системы помощи, она выводится на экран нажатием комбинации клавиш **Control+K-H** и насчитывает семь секций, каждая из которых занимает собственный экран, перемещение между которыми осуществляется комбинациями клавиш **Meta+(>)**. (вперед) и **Meta-(<)**, (назад).

Первая секция, Basic, описывает действия наиболее общего плана: перемещения курсора (субсекция CURSOR), переходы по тексту (субсекция GO TO), операции с текстовыми блоками (субсекция BLOCK), команды удаления символов и текстовых фрагментов (субсекция DELETE), команды поиска, проверки орфографии, форматирования (субсекции SEARCH, SPELL, MISC), операции с файлами (субсекция FILE), а также выход из редактора.

Вторая секция посвящена описанию манипуляций с окнами - расщеплению (split) экрана, скрытию и показу открытых окон, переходу между окнами, изменению их размера.

В третьей секции собрано описание расширенных возможностей для редактирования текстов.

Четвертая секция - расширенные возможности для программистов (команды перехода к регулярным выражениям, компилирования и отладки).

В пятой секции дано описание сложных регулярных выражений.

Шестая секция - операции с командной строкой встроенной в редактор командной оболочки.

Подсистема печати. Печать в Linux. Наиболее простой путь для печати в операционной системе Linux это посылка файла для печати прямо на устройство печати. Для того чтобы сделать это используется команда cat. Как пользователь root, наберите командной строке:

```
# cat for_print.txt > /dev/lp
```

В этом случае, /dev/lp это символическая ссылка на устройство печати – это может быть матричный или лазерный принтер, typesetter, или плоттер.
В целях безопасности, только пользователь root и пользователи в той же группе что и демон принтера могут писать прямо на принтер. Поэтому такие команды как lpr, lprm, и lpq должны использоваться для доступа к принтеру.

Просмотр очереди печати с помощью lpq. Для просмотра содержимого очереди печати используется команда lpq. Команда, запущенная без аргументов, она возвращает содержимое очереди печати принтера по умолчанию.

```
$ lpq
lp is ready and printing
Rank   Owner      Job  Files                    Total Size
active mwf        31   thesis.txt               682048 bytes
```

Отмена задания печати с использованием команды lprm.

```
$ lprm -
```

Показанная выше команда отменяет все задания печати владельцем, которых является пользователь, выполнивший эту команду. Для того чтобы отменить одиночное задание печати, надо сначала получить номер задания с помощью команды lpq и затем сообщить полученный номер команде lprm.

Например,

```
$ lprm 31
```

отменит задание номер 31 (thesis.txt) на принтере по умолчанию.

Контроль демона lpd с помощью lpc. Программа lpc используется для контроля принтеров, которые обслуживает демон lpd. Вы можете разрешать или запрещать использование принтера или их очередей, перераспределять задания внутри очереди, и получать информацию о состоянии принтеров и их очередей. Lpc наиболее часто используется в системах, где несколько принтеров установлено на одну машину.

$ lpc

Команда, показанная выше, запускает программу lpc. По умолчанию, она входит в интерактивный режим, и вы можете начинать вводить команды. Другие опции используются для запуска команды lpc в командной строке.

$ lpc status all

Список всех возможных команд перечислен на странице руководства команды lpd, но здесь перечислено несколько главных команд, которые вы должны знать. Любые команды, обозначенные как option, могут быть либо именем принтера (lp, print, etc) или ключевым словом all, которое обозначает все принтера.

disable option - запрещает добавление любых новых заданий печати
down option - запрещает все задания на принтере
enable option - разрешает ввод новых заданий в очередь печати
quit (or exit) - покинуть lpc
restart option - перезагрузить lpd для данного принтера
status option - статус печати принтера
up option - разрешить все и стартовать новый демон lpd

Упаковка файлов. Связка tar+gzip. tar (англ. tape archive) — формат битового потока или файла архива, а также название традиционной для Unix программы для работы с такими архивами. Программа tar была стандартизирована в POSIX.1-1998, а также позднее в POSIX.1-2001. Первоначально программа tar использовалась для создания архивов на магнитной ленте, а в настоящее время **tar** используется для хранения нескольких файлов внутри одного файла, для распространения программного обеспечения, а также по прямому назначению — для создания архива файловой системы. Одним из преимуществ формата tar при создании архивов является то, что в архиве содержится информацию о структуре каталогов, о владельце и группе отдельных файлов, а также временные метки файлов.
Как и другие утилиты Unix, **tar** — специализированная программа, которая следует философии Unix: «делать только одну вещь» (работать с архивами), «но делать её хорошо». Поэтому tar не создаёт сжатых архивов, а использует для сжатия внешние утилиты, такие как **gzip** и **bzip2**. Ранее для сжатия использовалась также утилита compress, которая в настоящее время не используется.
gzip (сокращение от GNU zip) — утилита сжатия и восстановления (декомпрессии) файлов, использующая алгоритм Лемпеля — Зива (LZW). Используется в основном в UNIX-системах, в ряде которых является стандартом де-факто для сжатия данных. Была создана Jean-Loup Gailly и Марком Адлером (Mark Adler). Версия 0.1 была впервые выпущена 31 октября 1992 г., а версия 1.0 — в феврале 1993 г.

bzip2 сжимает большинство файлов эффективнее, но медленнее, чем более традиционные **gzip** или **ZIP**. В этом отношении он похож на другие современные алгоритмы сжатия.

В некоторых случаях **bzip2** проигрывает форматам **7z** и **RAR** по абсолютной эффективности сжатия. Согласно открытой информации, **bzip2** проигрывает от 10 до 15 процентов наилучшему классу алгоритмов сжатия данных, известных на данный момент, но при этом в два раза быстрее при сжатии и в 6 раз быстрее при распаковке.

bzip2 использует преобразование Барроуза-Уилера (англ. Burrows-Wheeler transform, сортировка блоков) для превращения последовательностей многократно чередующихся символов в строки одинаковых символов, затем применяет преобразование MTF (англ. move-to-front), и в конце кодирование Хаффмана. Блоки в **bzip2** имеют одинаковый размер в несжатом потоке. Размер блока можно выбрать при помощи аргумента командной строки, и он помечается в сжатом тексте произвольно выбранной последовательностью битов представления числа Пи.

Пример обычного использования **gzip** :

$ gzip infile

Выходной файл будет назван `infile`.gz и почти всегда будет меньше входного. Обратите внимание, что `infile`.gz заменит `infile`. Это значит, что `infile` прекратит своё существование. Останется только его сжатая копия. Приведённый выше пример, это нечто среднее между качеством сжатия и затраченным временем. Максимальное сжатие может быть получено при помощи такой команды:

$ gzip -9 infile

Это займёт больше времени, но выходной файл будет настолько сжатым, насколько **gzip** вообще может его сжать. Использование меньших значений займёт меньше времени, но соответственно и качество компрессии будет хуже.

Распаковывание gzipped (запакованных GNU zip) файлов может быть выполнено при помощи двух команд, которые на самом деле являются одной и той же программой. **gzip** распакует любой файл с узнаваемым им расширением. Вот список расширений, которые узнаёт команда: .gz, -gz, .z, -z, .Z, или -Z. Первый метод - применить команду **gunzip**(1) к файлу:

$ gunzip infile.gz

Выполнение этой команды приведёт к тому, что вместо указанного файла в этом же каталоге появится его распакованная версия и .gz часть его имени исчезнет.

Чаще всего **tar** используется для распаковки и раз-архивирования пакетов, скачанных с вэб или ftp сайтов. Большинство файлов будут иметь .tar.gz расширение. Это так называемый "tarball". Это означает, что несколько файлов были помещены в архив при помощи **tar** и затем этот архив был сжат при помощи **gzip**. Иногда они так же имеют расширение .tar.Z. Это означает то же самое, но обычно встречается на более старых Unix системах.

.tar.bz2 файлы. Исходный текст ядра поставляется в таком виде, потому что так вам придётся скачивать меньший файл. Это несколько файлов, объединённых в архив при помощи **tar** и сжатых при помощи **bzip2**.

Вы можете получить файлы из таких архивов при помощи **tar** команды с определёнными аргументами командной строки. Разархивирование tarbal-а требует указания ключа **-z**, что фактически вызовет вначале выполнение **gunzip**, для распаковки файла. Обычно tarbal-ы распаковываются такой командой:

$ tar -xvzf linux-3.10.10.tar.gz

"-x" значит извлечь (extract), этот параметр говорит **tar**-у, что именно делать с входным файлом,

"-v" скажет программе быть "многословной" (verbose). Указание этого ключа приведёт к тому, что в процессе извлечения будет выводиться список файлов, которые извлекаются,

"-z" говорит **tar**-у, вначале пропустить файл hejaz.tar.gz через **gunzip**,

"-f" опция указывает, что далее в командной строке будет указанно имя файла, с которым надо работать.