

## Часть 4. Обработка текста

Влад 'mend0za' Шахов  
Linux & Embedded Team Leader

Linux & Embedded Department



## Введение в Unix-Way

Заповедь номер 3:

Всё есть текст<sup>1</sup>



---

<sup>1</sup>см статью Дениса Смирнова 'Классический Unix-way'

## Введение в Unix-Way

Заповедь номер 3:

Всё есть текст<sup>1</sup>

Заповедь номер 4:

Пускайте данные по трубам

---

<sup>1</sup>см статью Дениса Смирнова 'Классический Unix-way'



## Введение в Unix-Way

Заповедь номер 3:

Всё есть текст<sup>1</sup>

Заповедь номер 4:

Пускайте данные по трубам

Заповедь номер 5:

Всё есть файл

---

<sup>1</sup>см статью Дениса Смирнова 'Классический Unix-way'



## Текст в Unix-Way

В Unix (и Linux) в виде обычного текста или **plain text** представлены:

---

<sup>2</sup> в каталоге \$HOME

<sup>3</sup> в каталоге /etc

<sup>4</sup> справедливо для **syslog** и совместимых систем

<sup>5</sup> 'Unix is toolbox' - 'Unix это ящик с инструментами'



## Текст в Unix-Way

В Unix (и Linux) в виде обычного текста или **plain text** представлены:

- **конфигурационные файлы**, как локальные<sup>2</sup>, так и общесистемные<sup>3</sup>

---

<sup>2</sup> в каталоге \$HOME

<sup>3</sup> в каталоге /etc

<sup>4</sup> справедливо для **syslog** и совместимых систем

<sup>5</sup> 'Unix is toolbox' - 'Unix это ящик с инструментами'



## Текст в Unix-Way

В Unix (и Linux) в виде обычного текста или **plain text** представлены:

- **конфигурационные файлы**, как локальные<sup>2</sup>, так и общесистемные<sup>3</sup>
- **системные логи**<sup>4</sup>
- **исходные тексты программ**, включая скрипты на Shell
- **основной формат ввода и (или) вывода данных** для множества программ и утилит

---

<sup>2</sup>в каталоге \$HOME

<sup>3</sup>в каталоге /etc

<sup>4</sup>справедливо для **syslog** и совместимых систем

<sup>5</sup>'Unix is toolbox' - 'Unix это ящик с инструментами'



## Текст в Unix-Way

В Unix (и Linux) в виде обычного текста или **plain text** представлены:

- **конфигурационные файлы**, как локальные<sup>2</sup>, так и общесистемные<sup>3</sup>
- **системные логи**<sup>4</sup>
- **исходные тексты программ**, включая скрипты на Shell
- **основной формат ввода и (или) вывода данных** для множества программ и утилит

Богатый выбор изодранных острогаченных инструментов для работы с текстом во всех представлениях<sup>5</sup>

---

<sup>2</sup>в каталоге \$HOME

<sup>3</sup>в каталоге /etc

<sup>4</sup>справедливо для **syslog** и совместимых систем

<sup>5</sup>'Unix is toolbox' - 'Unix это ящик с инструментами'





## Просмотр обычных файлов

- Просмотр текста:
  - `cat` - вывести на stdout<sup>6</sup>

---

<sup>6</sup> Для двоичных файлов: чревато порчей настроек терминала  
<sup>7</sup> может отсутствовать в стандартной поставке



## Просмотр обычных файлов

- Просмотр текста:
  - **cat** - вывести на stdout<sup>6</sup>
  - **more** - вывести, разбив на страницы
  - **less**<sup>7</sup> - **more** на стероидах, с прокруткой, поиском

---

<sup>6</sup> Для двоичных файлов: чревато порчей настроек терминала  
<sup>7</sup> может отсутствовать в стандартной поставке



## Просмотр обычных файлов

- Просмотр текста:
  - **cat** - вывести на stdout<sup>6</sup>
  - **more** - вывести, разбив на страницы
  - **less**<sup>7</sup> - **more** на стероидах, с прокруткой, поиском
- Просмотр двоичных данных:
  - **od** - дамп файла в не-текстовых форматах

```
~$ od -c .bashrc
0000000  #      ~  /  .  b  a  s  h  r  c  :  e  x  e
0000020  c  u  t  e  d  b  y  b  a  s  h  (  1  )

~$ od -x .bashrc
0000000  2023 2f7e 622e 7361 7268 3a63 6520 6578
0000020  7563 6574 2064 7962 6220 7361 2868 2931

~$ od -f .bashrc
0000000  2.311258e-10  1.785672e+31  8.676411e-04  7.331327e+22
0000020  7.215143e+22  7.338225e+34  1.785670e+31  3.933694e-14
```

- **strings** - извлечь текстовые строки из двоичных файлов

<sup>6</sup> Для двоичных файлов: чревато порчей настроек терминала  
<sup>7</sup> может отсутствовать в стандартной поставке



## Редакторы

- Любой редактор, с которым вы можете справиться.

---

<sup>8</sup>В этом качестве он внесен в стандарт Single Unix Specification. Существует множество реализаций редакторов, совместимых с vi: vim, elvis, nvi, vi-mode в Emacs, Sublime Text 2, vi из busybox и т.д.

<sup>9</sup>С получением и сохранением данных по протоколу FTP и SSH



## Редакторы

- Любой редактор, с которым вы можете справиться. Но его может не быть в вашей системе...

---

<sup>8</sup>В этом качестве он внесен в стандарт Single Unix Specification. Существует множество реализаций редакторов, совместимых с vi: vim, elvis, nvi, vi-mode в Emacs, Sublime Text 2, vi из busybox и т.д.

<sup>9</sup>С получением и сохранением данных по протоколу FTP и SSH



# Редакторы

- Любой редактор, с которым вы можете справиться. Но его может не быть в вашей системе...
- Редактор **vi** присутствует как стандартный в любой Unix-подобной системе<sup>8</sup>

---

<sup>8</sup>В этом качестве он внесен в стандарт Single Unix Specification. Существует множество реализаций редакторов, совместимых с vi: vim, elvis, nvi, vi-mode в Emacs, Sublime Text 2, vi из busybox и т.д.

<sup>9</sup>С получением и сохранением данных по протоколу FTP и SSH



## Редакторы

- Любой редактор, с которым вы можете справиться. Но его может не быть в вашей системе...
- Редактор **vi** присутствует как стандартный в любой Unix-подобной системе<sup>8</sup>
- Не обязательно редактировать локально: В **mc**, **vim**, **emacs** есть возможность удалённого редактирования файлов<sup>9</sup>.

---

<sup>8</sup>В этом качестве он внесен в стандарт Single Unix Specification. Существует множество реализаций редакторов, совместимых с vi: vim, elv, nvi, vi-mode в Emacs, Sublime Text 2, vi из busybox и т.д.

<sup>9</sup>С получением и сохранением данных по протоколу FTP и SSH



## vi и vim

Перед стартом:

- 1 Редактор vi изначально создавался как универсальный и переносимый<sup>10</sup>. Все действия можно осуществить на алфавитно-цифровой части клавиатуры, без мыши.

---

<sup>10</sup>Обязан работать на любых типах терминалов и виртуальных консолей

<sup>11</sup>Командного стиля, а не меню-ориентированный

<sup>12</sup>сохранение, открытие файлов, выход, вставка файла в текущий и т.д.

<sup>13</sup>`export LANG='ru_RU.UTF-8'` - на русском языке





## vi и vim

Перед стартом:

- 1 Редактор vi изначально создавался как универсальный и переносимый<sup>10</sup>. Все действия можно осуществить на алфавитно-цифровой части клавиатуры, без мыши.
- 2 Редактор командного стиля<sup>11</sup>. Действия подачей прямых управляющих команд.

---

<sup>10</sup>Обязан работать на любых типах терминалов и виртуальных консолей

<sup>11</sup>Командного стиля, а не меню-ориентированный

<sup>12</sup>сохранение, открытие файлов, выход, вставка файла в текущий и т.д.

<sup>13</sup>`export LANG='ru_RU.UTF-8'` - на русском языке



## vi и vim

Перед стартом:

- 1 Редактор vi изначально создавался как универсальный и переносимый<sup>10</sup>. Все действия можно осуществить на алфавитно-цифровой части клавиатуры, без мыши.
- 2 Редактор командного стиля<sup>11</sup>. Действия подачей прямых управляющих команд.

3 основных режима: ~~портить текст и противно библиковать~~

- Командный режим (Normal mode) - по умолчанию при запуске.
- Режим изменения текста (Edit mode)
- Режим построчного редактирования (Ex mode) - операции над файлом целиком<sup>12</sup>.

---

<sup>10</sup>Обязан работать на любых типах терминалов и виртуальных консолей

<sup>11</sup>Командного стиля, а не меню-ориентированный

<sup>12</sup>сохранение, открытие файлов, выход, вставка файла в текущий и т.д.

<sup>13</sup>export LANG='ru\_RU.UTF-8' - на русском языке



## vi и vim

Перед стартом:

- 1 Редактор vi изначально создавался как универсальный и переносимый<sup>10</sup>. Все действия можно осуществить на алфавитно-цифровой части клавиатуры, без мыши.
- 2 Редактор командного стиля<sup>11</sup>. Действия подачей прямых управляющих команд.

3 основных режима: ~~портить текст и противно библиковать~~

- Командный режим (Normal mode) - по умолчанию при запуске.
- Режим изменения текста (Edit mode)
- Режим построчного редактирования (Ex mode) - операции над файлом целиком<sup>12</sup>.

Упражнение: проходим **vimtutor**, встроенный в vim учебник<sup>13</sup>

<sup>10</sup> Обязан работать на любых типах терминалов и виртуальных консолей

<sup>11</sup> Командного стиля, а не меню-ориентированный

<sup>12</sup> сохранение, открытие файлов, выход, вставка файла в текущий и т.д.

<sup>13</sup> export LANG='ru\_RU.UTF-8' - на русском языке



# Текстовый фильтр

Определение:

**Текстовый фильтр** - программа, обрабатывающая и преобразующая текст.

Примеры: **sort**, **cat**, **tac**, **rev**



## Текстовый фильтр

Определение:

**Текстовый фильтр** - программа, обрабатывающая и преобразующая текст.

Примеры: **sort**, **cat**, **tac**, **rev**

- Фильтр, запущенный без параметров - читает стандартный ввод.
- Параметры фильтра - интерпретируются как имена файлов
- Ключи фильтра - управляют режимами работы



## Текстовый фильтр

Определение:

**Текстовый фильтр** - программа, обрабатывающая и преобразующая текст.

Примеры: **sort**, **cat**, **tac**, **rev**

- Фильтр, запущенный без параметров - читает стандартный ввод.
- Параметры фильтра - интерпретируются как имена файлов
- Ключи фильтра - управляют режимами работы

Фильтр почти всегда используется совместно с перенаправлением ввода-вывода Shell (особенно '|', pipes).



## Простые текстовые фильтры

Соглашения о параметрах: '-' как имя файла обозначает стандартный ввод.

- **cat** и **tac** - вывести файл целиком



## Простые текстовые фильтры

Соглашения о параметрах: '-' как имя файла обозначает стандартный ввод.

- **cat** и **tac** - вывести файл целиком
- **head** и **tail** - вывести начало и конец файла





# Простые текстовые фильтры

Соглашения о параметрах: '-' как имя файла обозначает стандартный ввод.

- **cat** и **tac** - вывести файл целиком
- **head** и **tail** - вывести начало и конец файла
- **sort** и **uniq** - сортировка и убрать повторы в отсортированном



## Простые текстовые фильтры

Соглашения о параметрах: '-' как имя файла обозначает стандартный ввод.

- **cat** и **tac** - вывести файл целиком
- **head** и **tail** - вывести начало и конец файла
- **sort** и **uniq** - сортировка и убрать повторы в отсортированном
- **grep** - поиск по образцу



## Простые текстовые фильтры

Соглашения о параметрах: '-' как имя файла обозначает стандартный ввод.

- **cat** и **tac** - вывести файл целиком
- **head** и **tail** - вывести начало и конец файла
- **sort** и **uniq** - сортировка и убрать повторы в отсортированном
- **grep** - поиск по образцу
- **paste** - объединить файлы построчно



## Простые текстовые фильтры

Соглашения о параметрах: '-' как имя файла обозначает стандартный ввод.

- **cat** и **tac** - вывести файл целиком
- **head** и **tail** - вывести начало и конец файла
- **sort** и **uniq** - сортировка и убрать повторы в отсортированном
- **grep** - поиск по образцу
- **paste** - объединить файлы построчно
- **wc** - счётчик строк, слов и байт в тексте



## Простые текстовые фильтры

Соглашения о параметрах: '-' как имя файла обозначает стандартный ввод.

- **cat** и **tac** - вывести файл целиком
- **head** и **tail** - вывести начало и конец файла
- **sort** и **uniq** - сортировка и убрать повторы в отсортированном
- **grep** - поиск по образцу
- **paste** - объединить файлы построчно
- **wc** - счётчик строк, слов и байт в тексте
- **tee** - копирует стандартный ввод в файл и на экран



## Простые фильтры - упражнения

**Упражнение 1:** посчитать сколько файлов в папке `/bin`<sup>14</sup>

**Упражнение 2:** сколько слов в первых 15 строках `.bashrc`<sup>15</sup>

**Упражнение 3:** найти, в каких файлах (и сколько их вообще) в области системных логов (каталог `/var/log`) была записана информация о входе вашего пользователя в систему.<sup>16</sup>

**Упражнение 4:** сколько было входов в систему от имени вашего пользователя? в какое время был первый? последний?<sup>17</sup>

**Упражнение 5:** то же, что и 4. Только для выходов.  
Дополнительно сохранить записи о ваших выходах в отдельный файл.<sup>18</sup>

---

<sup>14</sup> подсказка - `wc` и `ls`

<sup>15</sup> подсказка - `head` и `ls`

<sup>16</sup> подсказка - `grep` и, возможно, но необязательно, `uniq`

<sup>17</sup> подсказка - `grep`, `head`, `tail`

<sup>18</sup> подсказка: + `tee`



## Изожрённые фильтры

- **diff** (и **diff -u**<sup>19</sup>) - сравнить 2 файла и получить описание разницы (изменения между ними)
- **patch** - утилита применения изменений от diff

---

<sup>19</sup> Формат **diff -u** : индустриальный стандарт пересылки списка изменений между версиями текстовых данных. Активно используется в частности в различных системах хранения и управления исходным кодом



## Изожрённые фильтры

- **diff** (и **diff -u**<sup>19</sup>) - сравнить 2 файла и получить описание разницы (изменения между ними)
- **patch** - утилита применения изменений от diff
- **sed** - не-интерактивный поточный редактор текста
- **awk** - язык и утилита сканирования и обработки текста

---

<sup>19</sup> Формат **diff -u** : индустриальный стандарт пересылки списка изменений между версиями текстовых данных. Активно используется в частности в различных системах хранения и управления исходным кодом





## Изоощрённые фильтры

- **diff** (и **diff -u**<sup>19</sup>) - сравнить 2 файла и получить описание разницы (изменения между ними)
- **patch** - утилита применения изменений от diff
- **sed** - не-интерактивный поточный редактор текста
- **awk** - язык и утилита сканирования и обработки текста

**Упражнение 1:** с помощью **diff** получить разницу между своим профайлом `.bashrc` и пользователя "user" (`/home/user/.bashrc`)

**Упражнение 2:** создать файл патча в unified формате, на основе Упражнения 1.

**Упражнение 3:** создать Unified-патч между домашним каталогом другого пользователя (например user) и вашим домашним каталогом. Включить в патч все подкаталоги и файлы.

<sup>19</sup> Формат **diff -u** : индустриальный стандарт пересылки списка изменений между версиями текстовых данных. Активно используется в частности в различных системах хранения и управления исходным кодом



## Описываем текст

### Как описать текст?

Необходим инструмент и формат описания текста



## Описываем текст

### Как описать текст?

Необходим инструмент и формат описания текста

### Регулярные выражения

**Регулярные выражения (regular expression или regexp)** - специальные строки символов, которые задаются для поиска совпадающих фрагментов. Иначе говоря это способ описания наборов букв.



## Описываем текст

### Как описать текст?

Необходим инструмент и формат описания текста

### Регулярные выражения

**Регулярные выражения** (regular expression или regexp) - специальные строки символов, которые задаются для поиска совпадающих фрагментов. Иначе говоря это способ описания наборов букв.

### Универсальный язык описания текста

Все Unix-программы, осуществляющие поиск в тексте, используют регулярные выражения.



## Элементы регулярных выражений

- **литералы** - обычные символы (буквы и цифры)



## Элементы регулярных выражений

- **литералы** - обычные символы (буквы и цифры)
- **метасимволы** - спецсимволы (количество повторов, группировка фрагментов, позиция в тексте).



## Элементы регулярных выражений

- **литералы** - обычные символы (буквы и цифры)
- **метасимволы** - спецсимволы (количество повторов, группировка фрагментов, позиция в тексте).

Примеры регулярных выражений:

```
~$ file /bin/*|grep symbolic  
~$ grep -o 'user[0-9]*' /var/log/auth.log
```



## Класс на 1 символ

- `.` (точка) - заменяет любой символ

Пример: `us.r.` = 'user0', 'us rX', 'us9r ' и т.д.





## Класс на 1 символ

- `.` (точка) - заменяет любой символ  
Пример: `us.r.` = 'user0', 'us rX', 'us9r ' и т.д.
- `[]` символьный класс - заменяет любой символ из перечисленных в скобках
  - 1 `user[0-9]` = 'user0', 'user5', но не равно 'user'
  - 2 `-[abc-]` = '-', '-a', '-b', '-c', но не равно '-a'
  - 3 `[^abc]1`<sup>20</sup> = 'd1', '11', но не равно 'a1'



<sup>20</sup> инвертировать символьный класс

## Класс на 1 символ

- **.** (точка) - заменяет любой символ  
Пример: `「us.r.」` = 'user0', 'us rX', 'us9r' и т.д.
- **[ ]** символьный класс - заменяет любой символ из перечисленных в скобках
  - ① `「user[0-9]」` = 'user0', 'user5', но не равно 'user'
  - ② `「-[abc-]」` = '-', 'a', 'b', 'c', но не равно '-a'
  - ③ `「[^abc]1」`<sup>20</sup> = 'd1', '11', но не равно 'a1'
- **[ :class: ]** - дополнительные POSIX-классы для символов, **внутри символьного класса**<sup>21</sup>  
Примеры классов: `[ :alnum: ]`, `[ :alpha: ]` `[ :digit: ]` `[ :space: ]`  
`[ :lower: ]`, `[ :upper: ]`, `[ :print: ]`  
Примеры regex с POSIX классами: `「[ы[:digit:]]」`

**Упражнение 1:** Из `/var/log/messages` вывести все слова, состоящие из одного символа(любого). Без повторов.

**Упражнение 2:** То же, что 1, но вывести слова из двух букв.

**Упражнение 3:** Извлечь все метки времени из файла

<sup>20</sup> инвертировать символьный класс



## квантификаторы - регулируем повторы

**Квантификаторы** указывают, сколько раз может повторяться символ или выражение, после которого указаны. Не являются шаблонами текста.

- **?<sup>22</sup>** - необязательный символ

пример: «**a.b**» - совпадёт с 'ab', 'a9b', 'a b'



## квантификаторы - регулируем повторы

**Квантификаторы** указывают, сколько раз может повторяться символ или выражение, после которого указаны. Не являются шаблонами текста.

- **?<sup>22</sup>** - необязательный символ  
пример: «**a.b**» - совпадёт с 'ab', 'a9b', 'a b'
- **\*** - любое количество символов, включая нулевое  
примеры: «**.\***», «**[:digit:]\***»



## квантификаторы - регулируем повторы

**Квантификаторы** указывают, сколько раз может повторяться символ или выражение, после которого указаны. Не являются шаблонами текста.

- **?<sup>22</sup>** - необязательный символ  
пример: «**a.b**» - совпадёт с 'ab', 'a9b', 'a b'
- **\*** - любое количество символов, включая нулевое  
примеры: «**.\***», «**[[:digit:]]\***»
- **+<sup>22</sup>** - не менее одного символа  
примеры: «**[a-d]+**», «**(02:)+**»

**Упражнение 1** - извлечь все слова из 1 или 2 букв из файла

**Упражнение 2** - все слова, начинающиеся на a из  
/var/log/messages

**Упражнение 3** - все числа из /var/log/messages



## Интервалы (интервальные квантификаторы)

- **{число}** - количество повторов выражения перед
- **{число1,число2}**
- **{число,}**

**Упражнение:** Найти все строки, длинее 30 символов в `$HOME/.bashrc`.



## Группировка и обратные ссылки

**Группировка** - поместить выражение в скобки `(выражение)`

22



## Группировка и обратные ссылки

**Группировка** - поместить выражение в скобки `(выражение)`  
22

**Обратные ссылки**<sup>22</sup> - способ обратиться к уже найденному тексту по регулярному выражению в круглых скобках.

Пример: `([a-z])([:digit:])\2\1` - совпадёт с 'a11a', но не с 'a12b'

Вычисление для подстановки и сравнения значения на месте обратной ссылки - производится прямо во время сравнения с регулярным выражением





## Группировка и обратные ссылки

**Группировка** - поместить выражение в скобки `(выражение)`  
22

**Обратные ссылки**<sup>22</sup> - способ обратиться к уже найденному тексту по регулярному выражению в круглых скобках.

Пример: `([a-z])([:digit:])\2\1` - совпадёт с 'a11a', но не с 'a12b'

Вычисление для подстановки и сравнения значения на месте обратной ссылки - производится прямо во время сравнения с регулярным выражением

**Нумерация групп** - слева-направо, от 1.

Широко применяется в операциях замены текста.

Упражнение: найти все повторяющиеся (идущие подряд)



## Якоря (anchors) или указатели позиции

- ^
- \$

Пример:



## Якоря (anchors) или указатели позиции

- ^
- \$

Пример:

- \< и \>

**Упражнение 1:** Найти все действующие определения псевдонимов (alias) в .bashrc, с учётом синтаксиса объявления alias.

**Упражнение 2:** Найти все закомментированные определения псевдонимов (alias) в .bashrc

