

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ  
Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования  
«Национальный исследовательский Томский политехнический университет»  
Институт кибернетики  
Кафедра автоматизации и компьютерных систем

УТВЕРЖДАЮ:  
Зам. проректора-директора ИК  
\_\_\_\_\_ Гайворонский С.А.  
«25» марта 2012 г.

## **Командный интерфейс и политика безопасности ОС FreeBSD**

Методические указания к выполнению лабораторных работ  
по курсам: **«Администрирование в информационных системах»**  
и **«Системное программное обеспечение»**  
для студентов профилей «Управление и информатика в технических  
системах», «Программное обеспечение средств вычислительной техники и  
автоматизированных систем», «Информационные системы и технологии в бизнесе»  
и специализаций: «CALS-технологии» и «Информационное и программное  
обеспечение систем управления»

УДК 629.76

Командный интерфейс и политика безопасности ОС FreeBSD.

Методические указания к выполнению лабораторных работ по курсу: «Администрирование в информационных системах» для студентов профилей «Управление и информатика в технических системах», «Программное обеспечение средств вычислительной техники и автоматизированных систем», «Информационные системы и технологии в бизнесе» и специализаций: «CALS-технологии» и «Информационное и программное обеспечение систем управления» – Томск: изд. ТПУ, 2012. – 20 с.

Составитель: Фадеев А.С..

Рецензент: к.т.н. Дмитриева Е.А.

Методические указания обсуждены на заседании кафедры Автоматики и компьютерных систем протокол № 7 от 17 марта 2005г.

Зав. кафедрой,  
д.т.н., профессор

Г.П. Цапко

## ВВЕДЕНИЕ

На примере командной оболочки C-Shell ОС FreeBSD, пользователь получает прямое взаимодействие с операционной системой видя на экране терминала все сообщения ОС в текстовом представлении без сокрытия графической оболочкой мелких нюансов. Именно при помощи командного текстового интерфейса достигается наиболее гибкое управление работой и конфигурацией ОС семейства Unix. ОС FreeBSD является ярким представителем этого семейства операционных систем, наиболее часто используемая в качестве информационного Интернет-сервера во всем мире.

Стандартный набор программ и команд Unix позволяет не только запускать на выполнение как простые задания, так и большие программы, но и комбинировать выходы одних программ со входами других, перенаправлять стандартные входы-выходы программ в файлы и устройства, группировать файлы для выполнения определенных действий используя специальные символы и т.д.

Политика безопасности ОС Unix рассчитана на многозадачный, многопользовательский режим работы систем в сетях различного масштаба, что накладывает на ОС необходимость обеспечения защиты файлов, процессов и других ресурсов от неосторожного или умышленного вмешательства нежелательных пользователей (точнее, процессов ими запущенных). Чрезвычайно эффективная система безопасности ОС FreeBSD является при этом легко модифицируемой и очень гибкой: каждый пользователь может изменять права доступа к собственным объектам, как для индивидуальных пользователей, так и для пользователей объединенных в группы.

Целью данной работы является ознакомление с командным интерфейсом C-Shell и методами управления средствами защиты файлов.

# 1 Командный интерфейс C-Shell ОС FreeBSD

## 1.1 Вход в систему

Операционная система UNIX - это многопользовательская система с разделением времени. Начинать сеанс работы с ней нужно с сообщения о том, кто Вы. Это не зависит от того, работаете Вы за терминалом в своем кабинете или по коммутируемой линии связываетесь с большим узлом общего пользования. В этом заключается одно из отличий UNIX от DOS, Windows95/98 и Macintosh – операционных систем, в которых понятие «многопользовательский» отсутствует. UNIX должна знать, кто Вы, чтобы выделять Вас и Ваше хозяйство среди десятков, сотен и даже тысяч других пользователей.

Диалог при входе в UNIX примерно такой:

```
login: guser1
Password:
Last login: Sat Sep 7 17:16:35
ttc
%
```

На вопрос login вводите свое пользовательское имя, далее вводите, пароль, данный вам администратором системы. После появления % (или #, или \$) можно работать.

Пользовательское имя – это имя, с которым связан Ваш вход в систему; администратор системы присваивает его, регистрируя данного пользователя. Пароль позволяет подтвердить, что Вы действительно тот, за кого себя выдаете, и таким образом предотвратить незаконный вход в систему. Вводимый пароль не отображается на экране, чтобы никто его не увидел.

Знак % - это приглашение, свидетельствующее о том, что UNIX готова к приему команд. В Вашей системе приглашение может быть иным, очень часто приглашение UNIX включает имя компьютера.

Для завершения сеанса работы в UNIX дайте команду **logout** или **exit**.

## 1.2 Команды

Сначала мы покажем, как работают команды операционной системы UNIX, а затем приведем перечень наиболее важных команд.

### 1.2.1 Формат команды

Большинство команд UNIX имеют следующий формат:

```
% команда ключи параметры
```

Сначала идет имя команды, затем – ключи и параметры. Параметрами, как правило, являются имена файлов. Все ключи начинаются с дефиса. Например, приведенная ниже команда означает: «Выполнить команду **ls** с ключом **-l** для файла **a.out**»:

```
% ls -l a.out
```

Это 90 процентов того, что Вам нужно знать. Имена команд почти всегда приводятся строчными буквами; ключи обычно состоят из одной буквы (строчной или прописной). В отличие от многих других операционных систем, UNIX учитывает разницу между прописными и строчными буквами.

Если Вы используете два и более однобуквенных ключа, большинство команд позволяют объединять их. Например, две приведенные ниже команды идентичны:

```
% ls -lg a.out
```

```
% ls -l -g a.out
```

Некоторые ключи требуют наличия параметра. В этом случае параметр дается после ключа, в этом случае последний нельзя объединять с другим ключом. Такие команды в нашем учебнике не описываются.

Есть несколько команд, которые не соответствуют этим правилам, и одна из них — команда **tar**.

## 1.2.2 Основные команды

**cd** *каталог* — Перейти в *каталог*. Имя нового каталога (директории) задается относительно текущего каталога (без символа «/») или относительно корня дерева файлов «/»: **cd /usr/home/std** — перейти в директорию **std**, которая находится в **/usr/home**. **cd std** — перейти в директорию **std**, которая находится в текущем каталоге (**std** — дочерняя директория). **cd /** — перейти в корневой каталог. **cd ..** — перейти в родительский каталог (на один уровень вверх). **cd** (без параметров) — перейти в домашнюю директорию.

**chgrp group file** установить группу *group* группой владельцев для файла *file*.

**chown user file** установить пользователя *user* владельцем для файла *file*.

**compress файл** — Сжать файл так, чтобы он занимал меньше места в памяти. В результате получается двоичный файл с тем же именем, что и исходный, и суффиксом **.Z**. Исходный файл удаляется. Сжатый файл нельзя пересылать по электронной почте, т.к. он двоичный, но его можно преобразовать в текстовый командой **uuencode**. Подобную операцию в UNIX выполняет **gzip**.

**cp файл1 файл2** — Копировать *файл1* в новый файл с именем *файл2*. Если в качестве *файл2* указано имя директории, *файл1* будет скопирован в эту директорию с именем *файл1*.

**grep -i образец файл** — Аналогична предыдущей команде, но регистр не учитывается: прописные и строчные буквы воспринимаются одинаково.

**grep образец файл** — Показать все строки в файле, отвечающие образцу. Поиск осуществляется с учетом регистра. Команда **grep** используется для поиска конкретных текстовых строк в файлах. Например, если *phones* – перечень номеров телефонов и имен, то команда

```
% grep "John Johnson" phones
```

находит в файле **phones** строку, содержащую фразу **"John Johnson"**. Обратите внимание: мы взяли образец поиска в кавычки. Кавычки *никогда* не повредят, но когда Вы ищете строку с пробелом или звездочкой, кавычки необходимы. Однако, если ищется слово без пробелов и символов регулярных выражений, кавычки ставить не обязательно.

**kill** — завершить фоновый процесс с помощью идентификатора процесса (PID). Вы можете получить PID, запустив команду **ps**;

**logout** — Завершить сеанс работы с UNIX.

**ls** — (от слова list) Дать перечень файлов, находящихся в текущем каталоге. Эквивалент команды **DIR**, имеющейся во многих операционных системах. Команда **ls -l** (ключ **-l** — сокращение от слова long) дает более подробную информацию, включая размер файлов, их принадлежность и дату создания. Ключ **-a** (от слова all) дает информацию обо всех файлах текущей директории, включая скрытые. Для использования обоих ключей возможны записи: **ls -la**, **ls -al**, **ls -l -a** и **ls -a -l**.

**man команда** — Вывести на экран справку ОС UNIX по команде команда.

**mkdir каталог** — Создать новый каталог без файлов с именем каталог.

**more файл** — Вывести текстовый файл на экран в постраничном режиме. Для вывода следующей страницы нужно нажать клавишу пробел. Многие используют для этого команду **cat** и жалуются, что ее имя вводит в заблуждение. Это, однако, их вина. Команда **cat** не предназначалась для вывода файлов на экран; она срабатывает чисто случайно (и то не очень хорошо). Команда **more** – гораздо лучшее средство; не утруждайте себя возней с **cat**. Кроме того, **more** не совсем «интуитивна»; полезно запомнить фразу «show me more of this file» («покажи мне этот файл дальше»).

- mv** *старое-имя новое-имя* — Переименовать (переместить) файл из *старое-имя* в *новое-имя*. Многие жалуются, что эту команду трудно запомнить из-за несоответствия сокращенного имени смыслу (*mv* – *rename*). Правильное замечание. Пользователи UNIX, однако, предпочитают считать, что файлы «перемещаются» (*move*), а не переименовываются (*rename*). Если рассматривать эту операцию как перемещение, то работа в UNIX будет более осмысленной.
- pg** — отображает содержимое указанного файла на терминал постранично. После распечатки каждой страницы система делает паузу и ждет от вас подтверждения на продолжение вывода следующей страницы;
- pr** — форматирует и выдает файлы на стандартный вывод. Команда *pr* разбивает текст на страницы;
- ps** — отображает состояние и номер каждого процесса, запущенного пользователем, выполняющегося в данный момент. **ps -aux** — отобразить все процессы системы.
- pwd** — Показать текущий каталог;
- rm** — Стереть (удалить) один или несколько файлов. Например, команда **rm file1 file2 file3** удаляет три файла: *file1*, *file2*, *file3*. Команда **rm -i** перед удалением каждого файла просит Вас подтвердить свое намерение.
- rmdir** *каталог* — Стереть (удалить) каталог с именем *каталог*. Этот каталог должен быть пуст, т.е. в нем не должно быть файлов.
- tar** — Эта команда предназначалась для создания архивов магнитных лент (**tape archive**), но используется и для создания архивов файлов (объединений, включающих несколько файлов). Подобные архивы встречаются в Internet. Структура этой команды довольно необычна. Поэтому вместо ее описания ниже приведены три примера команды **tar**, которых Вам будет достаточно для работы. В этих примерах **file.tar** – архив, созданный командой **tar**. В первых двух командах мы будем работать с архивом, полученным извне. В последнем случае мы сами создадим архив.
- tar tf file.tar**  
**tar xf file.tar**  
**tar cf file.tar list**
- uname** — отобразить имя системы UNIX, в которой вы работаете;
- uncompress** *файл* — Получить исходный файл из сжатого файла. Подобную операцию в UNIX выполняет **gunzip**.
- wall** [-g *group*] [*file*] — отправить на терминалы пользователей (группы) содержимое файла *file*.
- wc** — подсчитать числа строк, слов и символов в указанном файле и отобразить результат на терминале;
- who** — отобразить регистрационные имена пользователей, в данный момент зарегистрированных в вашей системе UNIX;
- who am i** — узнать какое регистрационное имя вы использовали при входе в систему;
- write** *user*— отправить пользователю *user* текстовые сообщения с консоли.

### 1.3 Стандартный ввод и вывод

Одной из сильных сторон операционной системы UNIX является гибкость ее системы ввода-вывода. Многие команды посылают свою выходную информацию на терминал (экран). Вместо этого Вы можете путем переназначения записать выходную информацию любой команды в файл. Аналогичным образом, многие команды принимают входную информацию с клавиатуры, но Вы можете выполнить переназначение так, чтобы ввод производился из файла. ОС UNIX рассматривает все операции ввода-вывода одинаково – и входной и выходной информацией является файл, даже если он существует в системе микросекунды.

Рассмотрим примеры:

**Команда > файл** — Стандартный вывод

— Поместить выходную информацию в файл, а не посылать ее на экран. То, что находилось в файле раньше, будет уничтожено. Например, если Вы хотите, чтобы список файлов Вашего каталога не выводился на экран, а был записан в файл, необходимо дать следующую команду:

```
% ls -l > file
```

Чтобы в файл записать ваш логин, нужна команда

```
% whoami > file
```

**Команда >> файл — Стандартный вывод с добавлением.**

— Дописать выходную информацию в файл следом за его содержимым.

**Команда < файл — Стандартный ввод.**

— Взять входную информацию из файла, а не с клавиатуры.

**Команда | другая-команда — Конвейер.**

— Взять стандартный вывод одной программы и использовать как стандартный ввод другой. Это одна из самых богатых возможностей ОС UNIX; ее можно использовать для создания собственных команд. Предположим, Вам нужен список всех файлов, принадлежащих пользователю **edk**. Эту операцию нельзя выполнить с помощью ключей команды **ls**, но, используя конвейер, можно объединить команды **ls -l** и **grep edk**:

```
% ls -l | grep "edk"
```

Команда **ls -l** выдает список всех файлов, включая информацию о владельцах, а **grep** извлекает все пункты, содержащие строку **edk**. Если Вы работаете с UNIX редко, то сможете обойтись и без конвейеров. Но если Вы начнете выполнять нечто существенное с помощью конвейеров, то быстро поймете, насколько они полезны.

**Переназначения стандартного ввода-вывода (>, >> или | ) всегда помещаются в самом конце команды, после всех ключей и параметров.**

## 1.4 Структура каталогов

Как и в MS-DOS и Macintosh, в операционной системе UNIX иерархическая (или "древовидная") файловая система. Это означает, что каждый файл находится в *каталоге*, а каталоги могут включать другие каталоги. В системе Macintosh каталоги называются "папками". В системах DOS и Windows используется тот же файлово-каталоговый язык, что и в UNIX.

Для разделения имен каталогов в UNIX используется прямая косая черта (/). Например, */home/john/letters/mom.brt* означает: "файл *mom.txt* в каталоге *letters* в каталоге *john* в каталоге *home*". Можно также сказать, что *john* - это подкаталог каталога *home* и т.д.

В этом примере следует отметить еще несколько моментов:

- Перед именем стоит косая черта (/). Косая черта в начале имени обозначает "корневой каталог", который является, по сути, точкой, в которой "склеены" между собой все диски системы. В ОС UNIX никогда не обращаются к самому диску, а всегда - к подкаталогам корневого каталога.
- UNIX-системы являются многопользовательскими. Каждому пользователю назначается "домашний каталог", в котором он должен хранить свои файлы, даже если он - является единственным пользователем системы. */home/john* - это, вероятно, домашний каталог пользователя *john*.
- Пользователи могут создавать собственные каталоги так, как Джон создал каталог *letters*.

Взятое нами в качестве примера имя файла (*/home/john/letters/mom.txt*) называется полным именем, потому что оно показывает весь "путь" к файлу, начиная с корневого каталога. Такие имена не обязательно использовать постоянно. Применяется целый ряд сокращений:

- *Рабочий каталог* всегда в Вашем распоряжении. Можно указывать пути относительно текущего каталога, а не корневого. Например, если текущим является каталог */home/john/letters*, можно указать только имя файла - *mom.txt*. (Именно это Вы и делаете в большинстве случаев: указываете файл в текущем каталоге.) Команда **pwd** выдает имя текущего каталога; команда **cd** *каталог* делает текущим другой каталог. Так, если текущим является каталог */home/john*, то команда **cd letters** перенесет Вас в каталог *letters*. Команда **mkdir** *каталог* создает новый каталог, а команда **rmdir** *каталог* удаляет каталог при условии, что он не содержит файлов.

- Вы можете обозначать домашний каталог знаком *~*, а комбинацией *~имя* - "начальный каталог пользователя *имя*". Например, *~john/letters/mom.txt* - еще один способ указать файл Джона. Команда **cd** без параметров предназначена для возврата в начальный каталог, при этом не учитывается, откуда Вы начали работу.

- Символы «*..*» обозначают "родительский каталог". Чаще всего они используются с командами **cd**. Например, если текущим является каталог *~john/letters*, то команда **cd ..** перенесет Вас в каталог *~john*.

Вам следует знать, как UNIX организует файлы. В отличие от персональных компьютеров, где используются относительно небольшие диски, UNIX-системы обычно работают с большими дисками, причем в значительных количествах. Гигабайтные диски здесь не редкость, а многие системы используют несколько дисководов. В большой системе их может быть десяток и более. Чем больше объем дисковой памяти, тем большее значение приобретают каталоги, с помощью которых можно правильно ее организовать.

### 1.4.1 Каталоги системы UNIX

*/* — каталог *root*;

*/stand* — содержит программы и файлы данных, используемые в процессе загрузки;

*/dev* — содержит специальные файлы, представляющие собой устройства, такие как: *console* – консоль, *lp* - печатающее устройство, *term/\** - пользовательские терминалы.

*/etc* — содержит файлы конфигурации и базы данных;

*/home* — содержит домашние каталоги пользователей;

*/tmp* — содержит временные файлы, например, буферы для редактирования файла;

*/var* — поддерево для изменяемых файлов (например, файлы регистрации);

*/usr* — содержит другие каталоги, например, *bin*, *lib*;

*/usr/bin* — содержит основные исполняемые программы.

*/usr/lib* — содержит библиотеки для программ и языков программирования.

### 1.4.2 Допустимые имена файлов

В операционной системе UNIX правил использования имен файлов не так уж много. В новейших UNIX-системах имена файлов могут иметь любую длину и включать почти все символы, кроме косой черты, которая используется для разделения каталогов. Лучше, однако, ограничиться строчными и прописными буквами, цифрами, точками и запятыми. Пробелы и другие специальные символы требуют специальной обработки.

Не ставьте в начале имени файла точку; команда **ls** не найдет такой файл в списке, если Вы не укажете ее с ключом **-a**. Использование точки в качестве первого символа имени файла позволяет "прятать" определенные файлы, чтобы они не "загрязняли" списки каталогов. Однако если Вы новичок в UNIX, то эта особенность может ввести Вас в заблуждение.

## 1.5 Символы подстановки (регулярные выражения) в именах файлов



В операционной системе UNIX в качестве стандартных символов подстановки (шаблонов) для имен файлов приняты \*, ? и []. В настоящее время чаще всего используется звездочка. Эти универсальные символы имеют следующее значение:

- \* используется для обозначения любого количества любых символов, то есть, любого объекта. Например, просто \* обозначает любой файл каталога (или все файлы); \*.txt - все имена файлов с расширением **txt**; **gorilla\*** - все имена файлов, начинающиеся со слова **gorilla**.

- ? обозначает любой (но только один) символ. Например, **source.?** обозначает **source.h**, **source.c**, **source.y** и т.д.

- [...] обозначает любой один символ из указанных в скобках. Вы можете указывать одиночные символы (например, **[chyg]** обозначает с, h, y либо g); диапазон (**[a-z]** обозначает любую строчную букву, а **[a-z0-9A-Z]** - любую букву или цифру). Отметим, что в одном диапазоне нельзя сочетать строчные и прописные буквы или буквы и цифры. **[a-Z]** или **[A-9]** приведут к удивительным результатам - возможно, Вы достигнете цели, но лишь если Вам очень повезет.

*Например:* **ls /var/log/[0-9]\*[a-zA-Z]** осуществляет вывод на экран всех файлов, начинающихся на цифру и оканчивающихся на букву в директории **/var/log**.

К сожалению, символы подстановки нельзя применять в части "образец" команды **grep**. Вместо этого команда **grep** использует более сложное средство - "регулярные выражения". Конечно, в элементе файл команды **grep** символы подстановки использовать можно. Например, команда

```
% grep "John Johnson" *
```

осуществляет поиск строки с именем Джона во всех файлах текущего каталога.

## 1.6 ЗАДАНИЕ:

1. Чтобы открыть терминальное соединение с компьютером, на котором установлена ОС **UNIX**, необходимо запустить программу-эмулятор терминала **putty**:  
*//aics/server/student/for students/putty.exe* указав в качестве адреса сервера *fas.aics.ru*, протокол: *SSH*, *Character set: KOI8-R* (на вкладке *Translation*), *Font size: 12-14 points* (на вкладке *Appearance*). Остальные данные для входа в систему, а также логин и пароль выдаются преподавателем.
2. Проверьте ваше местоположение в дереве файловой системы используя команду **pwd**. Обратите внимание на то, что в файловой системе Unix нет понятия «имя диска», вместо традиционного DOS-приглашения *a:\*, *c:\*, ... *w:\*, мы видим всегда только *«/»* — корневой каталог (**root**).  
В любой момент можно перейти в «домашнюю» директорию командой **cd** без параметров; в корневую директорию (**root**) командой **cd /**.
3. Изучите работу команды **ls** с различными параметрами.
4. Создайте (если не создана) директорию с вашим именем. Теперь это ваша директория: все создаваемые и редактируемые вами файлы должны содержаться в ней.
5. Изучите работу команды **who**. Изучите работу стрелок перенаправления **>** и **>>**.  
**Создайте** файл *«myfile1»*, в котором бы содержался список всех находящихся на данный момент в системе, при помощи команды **who** и перенаправления ввода/вывода.
6. **Добавьте** в файл строку, содержащую информацию о пути к вашей текущей директории (той, в которой вы работаете).

7. Добавьте в файл строку, содержащую информацию о **количестве строк, слов и символов** в вашем файле, проверив работу команды **wc**.
8. Создайте **директорию** «labal» в вашей домашней директории.
9. **Скопируйте** в директорию «labal» созданный файл. Перейдите в директорию labal и убедитесь в том, что файл действительно скопирован.
10. Произведите **архивацию** файла в директории «labal» используя команду **compress**.
11. Добавьте в исходный файл, лежащий в вашей домашней директории, строку, содержащую информацию о количестве строк, слов и символов в заархивированном файле. Для чего вернитесь сначала в вашу директорию, добейтесь вывода результата команды **wc** на экран, и только после этого, примените механизм перенаправления.
12. Изучите работу команды **grep**, с ее помощью из вашего файла выведите на экран все строки содержащие слово «user» (либо «pts», либо другое, по указанию преподавателя).
13. Создайте конвейер, который бы подсчитал количество строк в списке файлов вашей директории. Результат добавьте в ваш файл.
14. Добавьте в ваш файл список файлов из директории «/dev», содержащих сочетание букв «in» при помощи **конвейера**.

### Контрольные вопросы:

1. Чем отличается работа команды ls с ключом -l от работы с ключом -a; -la; -l -a?
2. Что такое «Домашняя директория»? Где она расположена?
3. Что такое Конвейер? Чем работа конвейера отличается от работы «<>» и «<>>»?
4. Как одной командой просмотреть содержимое домашней директории пользователя guser1, не зная пути к ней?
5. Как произвести поиск строки содержащей слово независимо от регистра букв?
6. В какой директории содержатся исполняемые программы who и wc?

### Дополнительное задание 1

Составьте конвейер, выполняющий действия в одну строку.

1. Подсчитайте количество пользователей в системе, использующих терминал «pts».
2. Подсчитайте, сколько студентов из вашей группы зарегистрировались в системе.
3. Подсчитайте, сколько сообщений система добавила в файл **/var/log/messages** за сегодняшний день.
4. Подсчитайте, сколько файлов из директории **/usr/home/std/guser1** имеют в полном описании комбинацию букв **student**.
5. Подсчитайте, сколько процессов запущенных в системе имеют статус (STAT) «I».
6. Подсчитайте, сколько сообщений система добавила в файл **/var/log/messages** за сегодняшний день за последний (предпоследний) час.
7. Подсчитайте, сколько файлов из директории **/var/run** содержат в имени последовательность букв «pid».
8. Подсчитайте, сколько процессов запущенных пользователем root имеют статус (STAT) «I».
9. Подсчитайте, сколько файлов из директории **/usr/home/std/guser1** имеют в полном описании комбинацию букв **lrwxr** и число **13**.
10. Подсчитайте, сколько файлов из директории **/usr/home/std/guser1** имеют в полном описании комбинацию букв **drwxr** и число **12**.

## Дополнительное задание 2

При помощи регулярных выражений выведите на экран файлы из директории `/usr/lib`, имена которых удовлетворяют следующим требованиям (используйте для вывода на экран команду `ls` по аналогии с `ls /usr/lib/[0-9]*[a-zA-Z]`):

### 1. Вариант

- имя файла начинается с букв `lib`,
- далее следует три любых буквы,
- далее стоит точка (`.`),
- далее произвольное количество символов,
- далее еще одна точка (`.`).
- завершает имя любая цифра.

Например: `libABC.DEF.7`

### 2. Вариант

- в имени файла присутствует символ `<_>` (знак подчеркивания),
- в имени также присутствует заглавная буква (после знака `<_>`),
- между знаком подчеркивания и заглавной буквой должен быть как минимум один символ.
- до знака `<_>` и после буквы возможны другие символы и буквы,
- имя файла должно оканчиваться любой латинской буквой.

Например: `abc_defgH-7j`

### 3. Вариант

- имя файлов начинается с букв `s` или `c`,
- в имени также присутствует заглавная буква,
- между первой буквой имени и заглавной буквой возможны другие символы,
- имя файла должно оканчиваться прописной буквой,
- между заглавной буквой и последней прописной буквой должен быть только один символ.

Например: `cdefgH-j`

### 4. Вариант

- в имени файла присутствует гласная буква: `a`, `u`, `e`, `y`, `i` или `o`,
- до гласной буквы присутствует знак `<_>`,
- до и после знака `<_>` возможны другие символы,
- после гласной буквы присутствует еще один знак `<_>`,
- до и после знака `<_>` возможны другие символы.

Например: `libcom_err_p.a`

### 5. Вариант

- в имени файла последовательно встречаются буквы `p`, `a`, `p`, `a`,
- между буквами `p`, `a`, `p`, `a` возможны различные символы,
- далее следует две буквы `s` или `c`,
- через несколько символов следует символ `<.>`,
- далее три любых символа,
- оканчивается имя цифрой.

Например: `pa_passwdqc.so.3`

### 6. Вариант

Отправьте всем пользователям системы сообщение о том, что до перезагрузки сервера осталось 2 минуты.

### 7. Вариант

Устройте переписку между двумя терминалами, подобно чату сети Интернет.

## 2 Политика безопасности в ОС FreeBSD: права доступа к элементам файловой системы

### 2.1 Создание учетной записи пользователя

Учетная запись — один из основных элементов, которые определяют работу компьютера под управлением Unix, предоставляющего услуги пользователям. Главная задача многих подсистем, входящих в состав операционной системы, — обеспечить пользователям надежный доступ к ресурсам. Так, например, электронное письмо, отправленное в удаленный офис, может по каким-либо причинам опоздать на сутки, но вы не услышите жалоб от пользователей. В то же время если при подготовке этого же сообщения информация, вводимая с клавиатуры, станет запаздывать на экране на два-три символа, многие пользователи сочтут работу системы неудовлетворительной.

Создание учетной записи и управление ею происходят достаточно просто. Учетная запись состоит из одной-двух строк, расположенных в нескольких файлах, которые содержат данные о привилегиях пользователя, пароль для входа в систему, сведения об оболочке и рабочий каталог (часть файловой системы, выделенная в полное распоряжение пользователя). В круг обязанностей системного администратора входит не только настройка основных сервисов, но и работа с файлами, содержащими учетные записи.

В состав различных версий Unix входят утилиты, предназначенные для управления учетными записями. Некоторые из этих утилит предоставляют дружественный интерфейс. В разных версиях системы подобные утилиты организованы по-разному. Пытаясь выиграть в конкурентной борьбе, производители Unix стараются интегрировать данные утилиты в специальные оболочки администрирования (однако не все подобные попытки успешны). Некоторым администраторам такие оболочки нравятся, мы же считаем, что подобный подход к решению основных задач администрирования скрывает структуру, которая, по существу, одинакова для всех версий.

Если система с дружественным интерфейсом выходит из строя либо файлы, в которые эта система записывает данные оказываются поврежденными, начинающий администратор оказывается абсолютно беспомощным перед возникшими проблемами, кроме того, у него формируется ложное представление о Unix как о ненадежной системе, основанной на недостаточно проработанных стандартах. Поэтому мы не будем рассматривать оболочки и утилиты администрирования, предназначенные для управления учетными записями пользователей, а вместо этого сосредоточим внимание на подсистемах, с которыми взаимодействуют эти утилиты.

### 2.2 Файл паролей /etc/passwd

Один из самых важных системных файлов — это файл /etc/passwd, присутствующий в каждой версии Unix. Если данный файл будет поврежден или удален, ни один пользователь, в том числе root (эта учетная запись принадлежит администратору) не сможет зарегистрироваться в системе.

Для каждого из пользователей в файле /etc/passwd содержится запись, представленная в следующем формате:

**имя\_пользователя : пароль : UID : GID : комментарии : рабочий\_каталог : оболочка**

Назначение каждого из полей записи приведено ниже.

**Имя\_пользователя.** Имя учетной записи пользователя.

**Пароль.** Пароль, представленный в зашифрованном виде. Если в данном поле содержится символ "\*", то зарегистрироваться под этим именем в системе невозможно или сам пароль в зашифрованном виде вынесен в другой файл.

**UID.** Числовой идентификатор данного пользователя.

**GID.** Идентификатор группы, которой принадлежит пользователь. Поскольку каждый пользователь может быть членом нескольких групп, в файле `/etc/passwd` указывается идентификатор группы, принадлежность к которой подразумевается по умолчанию.

**Комментарии.** Дополнительные сведения о пользователе (обычно в этом поле указывается его полное имя).

**Рабочий\_каталог.** Рабочий каталог пользователя.

**Оболочка.** Интерпретатор команд, используемый в процессе работы.

Информация в файле `/etc/passwd` представлена в текстовом виде и доступна для чтения всем пользователям. Право изменять содержимое данного файла имеет только суперпользователь. Для редактирования файла `/etc/passwd`, суперпользователь может применять любой текстовый редактор, например `vi` или `joe`.

В последних версиях Unix информация о пароле удалена из файла `/etc/passwd` и хранится в файле `/etc/shadow` для того, чтобы предотвратить попытки кражи паролей. В отличие от `/etc/passwd`, файл `/etc/shadow` недоступен для чтения обычным пользователям.

Несмотря на то, что алгоритма расшифровки пароля не существует, файл, содержащий пароли в закодированном виде, можно использовать для взлома системы. Злоумышленнику не составляет труда запустить несложную программу, выполняющую шифрование каждого слова из словаря. Зашифрованные слова сравниваются с кодами паролей в файле, и, если будет обнаружено хотя бы одно совпадение, соответствующее слово может быть использовано для незаконного проникновения в систему.

Файл `/etc/shadow` содержит не только закодированные пароли, но и информацию о сроке действия паролей и дополнительные сведения для реализации политики безопасности узла. Подобно файлу `/etc/passwd`, записи в файле `/etc/shadow` представлены в текстовом формате, а для разделения полей используются двоеточия.

В некоторых системах хэш-функции паролей каждого пользователя, а также некоторая дополнительная информация, хранятся в файле `/etc/master.passwd`.

Файлы `master.passwd` и `shadow`, как правило, защищены от чтения и изменения обычными пользователями.

## 2.3 Файл групп `/etc/group`

Зарегистрированных пользователей можно объединять в группы. Это облегчает управление учетными записями, особенно если количество учетных записей велико. Механизм групп позволяет достаточно просто ограничивать доступ пользователей к данным и системным службам. Так, например, если вы захотите запретить пользователям из технического отдела обращаться к личным делам сотрудников предприятия, вы сможете сделать это, объединив пользователей в группы.

Файл групп (`/etc/group`) содержит сведения о группах, созданных в вашей системе. Информация в файле `/etc/group` также используется при выполнении некоторых команд, например `ls` или `find` позволяет заменить идентификатор группы (GID) текстовым описанием. Информация в файле `/etc/group` представлена в текстовом формате; каждой группе соответствует запись, представленная в отдельной строке. Формат записи имеет следующий вид:

**группа : пароль : GID : пользователи**

В полях записи содержится следующая информация:

**Группа.** Имя группы.

**Пароль.** Пароль в закодированном виде. Если данное поле не заполнено, пароль не нужен.

**GIID.** Числовой идентификатор группы.

**Пользователи.** Список имен учетных записей пользователей, принадлежащих группе. Имена пользователей разделяются запятыми.

Как видите, для создания учетной записи пользователя не надо прилагать много усилий. Достаточно открыть с помощью текстового редактора файл `/etc/passwd` и включить в него новую запись, а также создать для пользователя рабочий каталог, установив для этого каталога владельца и права доступа. После того как вы выполните указанные действия, пользователь получит возможность регистрироваться в системе.

Изменить владельца файла *file* можно с помощью команды **chown**. В большинстве случаев пользователю принадлежат как его рабочий каталог, так и все файлы в этом каталоге. Так, например, если для пользователя по имени Фред (регистрационное имя `fred`) создан рабочий каталог `/home/fred`, то для того, чтобы объявить Фреда владельцем этого каталога, администратору надо выполнить следующие команды:

```
cd /home/fred
chown fred ..??*
```

Выражение `..??*` в составе команды соответствует всем файлам, имена которых начинаются с точки и содержат, помимо этой точки, еще как минимум два символа. В данном случае мы не собираемся изменять владельца родительского каталога (`/home`). Если бы вместо `..??*` мы включили в состав команды последовательность символов `.*`, имя родительского каталога (`..`) также соответствовало бы этому выражению. После выполнения приведенных выше команд пользователь `fred` назначается владельцем файлов, используемых при запуске оболочки (например, `.cshrc`, `.login` или `.profile`).

## 2.4 Права доступа: кто может обращаться к файлу?

FreeBSD является прямым потомком BSD UNIX® и основывается на некоторых ключевых концепциях UNIX. В первую очередь это, конечно, тот факт, что FreeBSD - многопользовательская операционная система. Это означает, что несколько пользователей могут работать одновременно, решая различные задачи и совершенно не мешая друг другу. На системе лежит ответственность за правильное разделение и управление такими ресурсами как память, процессорное время, периферийные устройства и прочее.

Многопользовательская среда предполагает наличие механизма регулирования прав доступа к любому ресурсу в системе. Существует три типа прав доступа: на чтение, запись и исполнение. Права сгруппированы три по три, соответственно чтение/запись/исполнение, каждому элементу ресурсов ставятся в соответствие права доступа для владельца (конкретный пользователь), для группы (все пользователи в группе) и для "всего мира" (все остальные пользователи). Объединив всех сотрудников технического отдела в одну группу, а пользователей, которым необходимо предоставить доступ к личным делам персонала предприятия, — в другую группу, вы сможете установить для разных групп различные права доступа к файлам и лишить таким образом сотрудников технического отдела возможности работать с личными делами.

Численное представление:

Значение	Права доступа	Список файлов каталога
0	Ничего не разрешено	---
1	Нельзя читать и писать, разрешено исполнять	--x
2	Нельзя читать и исполнять, разрешено писать	-w-
3	Нельзя читать, разрешено писать и исполнять	-wx
4	Разрешено читать, нельзя писать и исполнять	r--
5	Разрешено читать и исполнять, нельзя писать	r-x
6	Разрешено читать и писать, нельзя исполнять	rw-
7	Разрешено все	rwx

Вы можете использовать опцию `-l` команды `ls(1)` для получения подробного листинга каталога, включающего колонку с информацией о правах на файл для владельца, группы и всех остальных. Например, команда `ls -l` в произвольном каталоге может вывести следующее:

```
% ls -l
total 530
-rw-r--r--  1 root  wheel      512 Sep  5 12:31 myfile
-rw-r--r--  1 root  wheel      512 Sep  5 12:31 otherfile
-rw-r--r--  1 root  wheel    7680 Sep  5 12:31 email.txt
...
```

Вот как выглядит первая колонка вывода `ls -l`:

```
-rw-r--r--
```

Первый (считая слева) символ говорит обычный ли это файл, каталог, символьное устройство, сокет или любое другое псевдо-файловое устройство. В нашем случае - указывает на обычный файл. Следующие три символа (в данном случае это `rw-`) задают права доступа владельца файла. Затем идут права группы, которой принадлежит файл (`r--`). Последняя тройка (`r--`) определяет права для всех остальных. Минус означает отсутствие каких-либо прав (т.е. нельзя ни читать, ни писать, ни выполнять). В данном случае права установлены таким образом, что владелец может читать и писать в файл, а группа и другие могут только читать. Таким образом, численное представление прав *644*, где каждая цифра представляет три части прав на файл.

Права на устройства контролируются аналогичным образом. В FreeBSD все устройства представлены в виде файлов, которые можно открывать, читать и писать в них. Эти специальные файлы содержатся в каталоге `/dev`.

Каталоги также являются файлами. К ним применимы те же права на чтение, запись и выполнение. Правда, в данном случае "выполнение" имеет несколько другой смысл. Когда каталог помечен как "исполнимый", это означает, что можно "зайти" в него (с помощью команды `cd`). Это также означает, что в данном каталоге можно получить доступ к файлам, имена которых известны (конечно, если собственные права на файл разрешают такой доступ).

Если же требуется получить список файлов в некотором каталоге, права доступа на него должны включать доступ на чтение. Для того, чтобы удалить из каталога какой-либо файл, имя которого известно, на этот каталог должны быть даны права на запись и на исполнение.

Для изменения прав доступа к файлу используется команда `chmod`:

```
chmod 641 personnel.txt,
```

где `personnel.txt` – файл, первое число (6) устанавливает права владельцу файла, второе число (4) – права членам группы владельца, третье число (1) – права всем прочим.

Существуют и другие права доступа, но они как правило используются в особых случаях, например, `setuid`-бит на выполняемые файлы и `sticky`-бит на каталоги.

### Символические обозначения прав

Символические обозначения, иногда называемые символическими выражениями, используют буквы вместо восьмеричных значений для назначения прав на файлы и каталоги. Символические выражения используют синтаксис **chmod (кто)(действие)(права) file**, где существуют следующие значения:

Опция	Буква	Значение
(кто)	u	Пользователь (User)
	g	Группа (Group)
	o	Другие (Other)
	a	Все (All)
(действие)	+	Добавление прав
	-	Удаление прав
	=	Явная установка прав
(права)	r	Чтение (Read)
	w	Запись (Write)
	x	Выполнение (Execute)
	t	Sticky бит
	s	SUID или SGID

Эти значения используются командой `chmod` так же как и раньше, но с буквами:

`chmod кто действие права имя_файла`

Например, вы можете использовать следующую команду для запрета доступа других пользователей к *FILE*:

```
% chmod go-rwx FILE или chmod go= FILE
```

В этом примере после знака «=» стоит пробел — «пустота», «ничего».

Для изменения более чем одного набора прав можно применить список, разделенный запятыми. Например, следующая команда удалит права группы и "всех остальных" на запись в *FILE*, а затем добавит права на выполнение для всех:

```
% chmod go-w,a+x FILE
```



## 2.5 Задание

### 1. Работа с файлом учетных записей пользователей `/etc/passwd`.

- Найдите в файле `/etc/passwd` строку, описывающую Вашу учетную запись. Создайте файл `users`, содержащий эту строку.
- Ответьте на вопросы: какую оболочку вы используете, какой идентификационный номер у Вас и какой идентификатор у вашей группы, что используется в качестве комментария к Вашей учетной записи?
- Подсчитайте, сколько зарегистрировано пользователей в той же системной группе, что и вы. Добавьте это число в файл `users`.
- Добавьте в файл `users` строку, описывающую учетную запись главного администратора системы `root`.

### 2. Работа с файлом системных групп `/etc/group`

- Просмотрите файл `/etc/group`. Чем запись с `GID=0` отличается от всех остальных? Что это за группа?
- Подсчитайте, сколько групп зарегистрировано в системе. Добавьте это значение к файлу `users`.
- Найдите запись о вашей группе по её **GID**.
- Добавьте в файл **users** строку с именем Вашей группы и её **GID**.

### 3. Работа с правами доступа

- Просмотрите содержимое вашей домашней директории с помощью команды `ls -la`. Имя какого пользователя указано, как имя владельца файлов? Какая группа является группой владельца этих файлов?
- Какие пользователи и группы являются владельцами текущего (.) и родительского (..) каталога?
- Какие права доступа установлены для ваших файлов и каталогов? Кто из пользователей системы может осуществлять операции чтения, записи и выполнения с файлами?
- Скопируйте в файл **users** строку, описывающую права доступа к файлу **users**.

### 4. Работа с разными пользователями.

- Зайдите в систему параллельно с существующим сеансом через две новые консоли под именами `guser1` (пароль – `guser1`) и `guser2` (пароль – `guser2`).
- Выясните, к какой группе принадлежат эти пользователи. Из консоли с Вашим пользователем добавьте строки о пользователях `guser1` и `guser2` в файл `users`.
- В каких отношениях эти пользователи состоят с Вами (члены Вашей группы или «все прочие»)?

### Ограничение прав доступа.

5. Для файла *users* полностью запретите доступ (для чтения, записи и выполнения) всем не принадлежащим вашей группе пользователям («всем прочим»). Но сохраните разрешения для владельца файла и членов его группы.
  - a. Из консоли с Вашим пользователем скопируйте в файл **users** строку, описывающую права доступа к файлу **users**.
  - b. Попробуйте прочесть содержимое файла *users* командой `more` под пользователями `guser1` и `guser2` (не забудьте указать правильный путь к файлу). Сделайте выводы.
6. Запретите доступ на чтение, запись и выполнение для файла *users* всем принадлежащим вашей группе пользователям, но разрешите полный доступ для «всех прочих».
  - a. Из консоли с Вашим пользователем скопируйте в файл **users** строку, описывающую права доступа к файлу **users**.
  - b. Попробуйте прочесть содержимое файла *users* командой `more` под пользователями `guser1` и `guser2` (не забудьте указать правильный путь к файлу). Сделайте выводы.

## 7. Копирование чужих файлов

- a. Для копирования используйте консоль пользователя, не принадлежащего вашей группе. Попробуйте «украсть» файл *users*, скопировав его в директорию `lab1`, созданную вами на прошлой работе (в Вашей домашней директории).
- b. Если копирование не удастся, проанализируйте имеющиеся запреты. Добейтесь копирования любым способом: изменяйте требуемые права доступа под любым пользователем и любым файлам.
- c. Убедитесь в завершении копирования: выведите список файлов целевой директории (`lab1`) с полным описанием. Имя владельца скопированного файла должно отличаться от имени владельца директории.

## 8. Работа с директориями

- a. Для пользователя, не принадлежащего Вашей группе проведите три эксперимента с директорией `lab1` (установите на эту директорию для этого пользователя персональные ограничения прав: сначала «---», затем «r--» и «--x»), в каждом из которых определите разрешено ли этому пользователю:
  - а) переходить в директорию командой `cd`;
  - б) просматривать список файлов директории (`ls`);
  - в) создавать новые файлы.
- b. Результаты представьте в виде таблицы.

## 9. Ответьте на контрольные вопросы.

## 10. Сделайте выводы по работе.

**Контрольные вопросы:**

- 1) Какое число в десятичной системе исчисления соответствует правам доступа к файлу **-rw-r-xr--**?
- 2) Какими правами доступа к файлу `file1`, лежащему в директории `dir1` обладает член группы создателя этого файла, если команда `ls -l` выводит на экран информацию:
 

```
drwx--xrw-    dir1
-r--rwxr-x    file1
```
- 3) Какой командой можно запретить изменения файла `file1` для всех пользователей системы, не принадлежащих группе создателя файла?

**Дополнительные задания:**

1. Определите системные имена (login) всех системных администраторов.
2. Определите всех пользователей группы `nogroup`.
3. Определите всех пользователей группы `mail`.
4. Определите всех пользователей группы `operator`.
5. Определите всех пользователей группы `wheel`.
6. Определите имя группы, членом которой является пользователь с именем `www`.
7. Определите имя группы, членом которой является пользователь `news`.
8. Определите имя группы, членом которой является пользователь `games`.
9. Найдите среди файлов других студентов Вашей студенческой группы, файлы или папки, к которым вам разрешен полный доступ.
10. Назовите UID и GID пользователя, являющегося владельцем директории `/dev`.
11. Назовите UID и GID пользователя, являющегося владельцем файла `./snat`.
12. Назовите UID и GID пользователя, являющегося владельцем директории `/usr/home/std/`.
13. Назовите UID и GID пользователя, являющегося владельцем директории `/usr/home/`.

УДК 629.76

Командный интерфейс и политика безопасности ОС FreeBSD.

Методические указания к выполнению лабораторных работ по курсу: «Администрирование в информационных системах» для студентов профилей «Управление и информатика в технических системах», «Программное обеспечение средств вычислительной техники и автоматизированных систем», «Информационные системы и технологии в бизнесе» и специализаций: «CALS-технологии» и «Информационное и программное обеспечение систем управления» – Томск: изд. ТПУ, 2012. – 20 с.

Составитель:

Фадеев А.С.

Подписано к печати

Формат 60x84.16. Бумага писчая №2

Плоская печать. Усл.Печ.Л. . Уч.-изд.л.

Тираж экз. Заказ Цена свободная

ИПФ ТПУ. Лицензия ЛТ №1 от 18.07.94.

Ротапринт ТПУ. 634034, Томск, пр. Ленина, 30.