Оглавление

Введение
Модуль 1. Основы работы в Astra Linux
Упражнение 1. Основы работы в системе. Рабочий стол Fly
Модуль 2. Основные команды и утилиты для работы в системе
Упражнение 2. Основы командной строки ..... —If **9**
Модуль 3. Работа с текстовым редактором Vi/Vim ....'.....1**3**
Упражнение 3. Использование редактора vi 14
Модуль 4. Команды поиска. Конвейеры и перенаправление ввода-вывода..... 15
Упражнение 4. Команды поиска. Конвейеры и перенаправление ввода-вывода.....18
Модуль 5. Архивирование и сжатие файлов. :.**1**.....*X;*,.....*1*. . * 19
Упражнение 5. Архивирование и сжатие Файлов 19
Модуль **6**. Права доступа к файлам 21
Упражнение **6**. Права доступа к файлам .....(..*".тШШЙ*1I!.....22
Модуль 7. Работа с п о л ь з о в а т е л я м и : . 1 X $ ..... ,.r.,k.W.24
Упражнение 7.L ...*.....^ *J.*
Упражнение 7.2..... л; .... *‘:Ъ* ... .26
Упражнение 7.3. Работа с учетными записями пользователей 30
Модуль **8**. Процессы..... ....I—.34
Задание 8.1. Изучение и анализ отображаемой информации о процессах 36
Задание 8.2. Управление процессами ..... . 1.....1 .1. 37
Модуль 9. Файловая система Astra Linux. LVM 40
Упражнение 9.1. Символические и жесткие ссылки 44
Упражнение 9.2. Файловая система Astra Linux. Добавление и форматирование нового диска.
Модуль 10. Файловая система Astra Linux. 47
Упражнение 10. Файловая система Astra Linux. LVM 48
Модуль 11. Управление программными пакетами 50
Упражнение 11.1.' Набор команд dpkg.... ■>* - • у
Упражнение Л 1.2. Программа apt-get.. 53
Модуль 12. Процесс запуска и инициализации системы..*к.* . — . . -- —— 54
Загрузчик GRUB. .....* ... .54
Упражнение 12.11 Загрузка ОС Astra Linux.....^.^^^.^ ... .54
Упражнение 12.2. GRUB..... ... .54
Упражнение 12.3. Установка пароля на GRUB.-AJll..... 55
Модуль 13. Базовая диагностика системы, файлы журналов 56
Упражнение 13. Базовая диагностика системы, файлы журналов..... 57
Модуль 14. Управление процессами. Автоматизация заданий.....J.....58
Упражнение 14. Управление процессами. Автоматизация заданий..... 60
Модуль 15. Настройка swap ....
Упражнение 15. Настройка swap 62
Модуль 16. ACL (Access Control List)., ...
Упражнение 16. ACL (Access Control List) **66**
Модуль 17. Конфигурирование сетевых подключений 69
Упражнение 17. Конфигурирование сетевых подключений 69
Модуль 18. Служба удаленного администрирования SSH.... 73
Упражнение 18.1. Общая принципы работы SSH... 73
Упражнение 18.2. Настройка SSH сервера .. .. ,.,,,.....,,.74
Упражнение 18.3. Настройка SSH клиента..... . ...•«,.« . 79
Упражнение 18.4. Генерация ключей..... ....^.^^.... 80
Упражнение 18.5. Подключение с использовнием открытого ключа ... .81

Модуль 19. Протокол передачи файлов FTP ^
Упражнение 19.1. Установка по умолчанию
Упражнение 19.2. Простая конфигурация сервера FTP
Упражнение 19.3. Сложная конфигурация сервера FTP
Упражнение 19.4. Конфигурация сервера FTP с шифрованием SSL
Модуль 20. Шифрование дисков LUKS 90
Упражнение 20. Шифрование дисков LUKS 90
Дополнительно: 91
Модуль 21. Комплекс средств защиты (КСЗ) 93
Упражнение 21.1. Комплекс средств защиты (КСЗ) 95
Модуль 22. Идентификация и аутентификация 96
Упражнение 22. Идентификация и аутентификация 102
Модуль 23. Мандатное разграничение доступа 104
Модуль 24. Очистка памяти 114
Модуль 25. Защита ввода-вывода информации на отчуждаемые физические носители 115
Модуль 26. Печать и маркировка документов 116
Модуль 27. Регистрация событий 132
Модуль 28. Средства восстановления 139
Модуль 29. Контроль целостности 141
Модуль 30. Служба Astra Linux Directory (ALD) 146
Модуль 31. Основы работы с СУБД PostgreSQL 162
Совместная работа сервера СУБД PostgreSQL с ALD 2 ^
Модуль 32. Web-сервер Apache2
РЕШЕНИЯ К УШАЖНЕНИЯМ

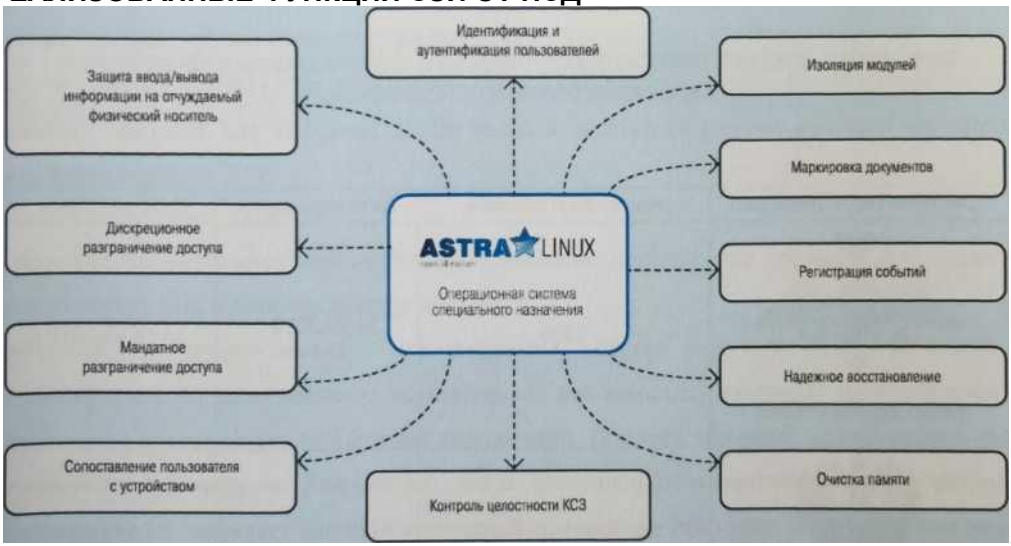
Введение

Операционные системы Astra Linux Common Edition и Astra Linux Special Edition созданы коллективом открытого акционерного общества «Научно-производственное объединение Русские базовые информационные технологии» и основаны на свободном программном обеспечении.

Astra Linux Common Edition предназначена для автоматизации предприятий малого и среднего бизнеса, органов государственного управления.

Astra Linux Special Edition предназначена для применения в автоматизированных системах в защищенном исполнении, обрабатывающих информацию ограниченного распространения, включая государственную тайну до степени секретности "совершенно секретно".

РЕАЛИЗОВАННЫЕ ФУНКЦИИ СЗИ ОТ НСД



КЛАСС ЗАЩИЩЕННОСТИ АВТОМАТИЗИРОВАННЫХ ■ cm

Средства защиты информации от несанкционированного доступа из состава ОС СН позволяют обеспечить соответствие автоматизированных систем, в которой применяется ОС СН, требованиям по защите информации от несанкционированного доступа, предъявляемых к следующим классам

Группа АС	3 группа Однопользовательская				2 группа Многопользовательская с равными полномочиями		1 группа Многопользовательская с разными полномочиями			
Класс АС	ЗБ		ЗА		2Б	2А	1Д	1Г	1В	1Б
Применение модулей доверенной загрузки	-	-	-	-	- чХТЧ\		-			+ V
Класс межсетевых экранов	5	3	2	1	5	3 2 1	5	4	3	2
Класс СВТ										
Уровень контроля отсутствия НДВ	5 ЦЦЦЦЦЦЦЦЬЮЦЦц 4 г -Ш 14 :Жг						4#4-		3	2
ОС СН «Astra Linux Special Edition»	KXW АПМДЗ «Максим-МЬ					яш утитиМШИА				

автоматизированных систем:

Сертификаты соответствия Систем сертификации средств зашиты информации по требованиям безопасности информации

Система сертификации	Номер сертификата	Дата получения	Срок действия
Минобороны России	1339	24.09.2010 г.	до 15.09.2018 г.
ФСТЭК России	2557	27.01.2012 г.	До 27.01.2018 г.
ФСБ России	СФ/014-2578 (вер. 1.2) СФ/014-2579 (вер. 1.4)	20.03.2015 г.	д° 31.12.2018 г.

Модуль 1. Основы работы в Astra Linux

Astra Linux **SE** 1.4 — это многозадачная, многопользовательская операционная система. В Astra Linux пользователи должны себя идентифицировать при входе: ввести имя (имя, по которому вас идентифицирует система) и пароль.

В любых **ОС** Linux/Unix существует, как минимум, один пользователь - root. Пользователь root - это пользователь, имеющий административный доступ к вашей системе. Обычные пользователи не имеют этого доступа по соображениям безопасности.

Поэтому, прежде, чем вы сможете использовать систему, вы должны создать еще один аккаунт в системе (если он не создан). Это необходимо потому, что неразумно использовать имя суперпользователя **(root)** для обычных нужд. Пользователь **root** нужен для выполнения привилегированных команд и сопровождения системы. Неосторожное поведение пользователя с такими правами легко может привести к печальным последствиям, вплоть до полного краха системы. Поэтому обычно под этим именем входят в систему только для выполнения административных задач.

Чтобы зарегистрировать себя, необходимо зайти в систему под именем **root** и использовать команду **useradd** или **adduser.** После входа в систему (в режиме консоли) вы видите подобную строку:

[rootQlocalhost /root]#

Такая строка называется приглашением. Появление приглашения означает, что система готова воспринять и выполнить вашу команду.

Оболочка, или просто **shell** - это программа, которая осуществляет все общение с пользователем. Именно оболочка воспринимает все команды, вводимые пользователем с клавиатуры, и организует исполнение этих команд. Поэтому оболочку можно назвать еще командным процессором. Каждый раз, когда очередной пользователь входит в систему, команда **login** запускает для него командный процессор - оболочку. **У** пользователя **root** приглашение оканчивается символом #, а у всех остальных пользователей - символом $. Для Linux систем разработано несколько альтернативных оболочек. Их можно использовать и в Astra Linux, но по умолчанию запускается именно bash.

Оболочка **bash** является не только командным процессором, но и мощным языком программирования. В ней имеется целый ряд встроенных (внутренних) команд и операторов, а, кроме того, в качестве команды может использоваться любая программа, хранящаяся в виде файла на диске.

Упражнение 1. Основы работы в системе. Рабочий стол Fly.

1. Определите какой мандатный уровень установлен в вашей системе по умолчанию
2. Измените настройки рабочего стола с помощью утилиты **fly-admin-theme**
3. Запустите **firefox** из командной строки, любым известным вам способом. Например, запустите fly-run, и в приглашении для ввода команды наберите **firefox.**
4. Посмотрите информацию об устройствах в вашем компьютере с помощью утилиты **fly-admin-device-manager**
5. Запустите менеджер файлов **f ly-fm,** посмотрите какие файлы и папки есть в домашней директории пользователя **student**
6. С помощью утилиты поиска (запуск из командной строки: # fly-find -d) найдите файлы содержащие аккаунты пользователей и группы: **passwd** и **group**
7. Установите корректное время на вашей системе с помощью утилиты **fly-admin-date**
8. Поменяйте пароль для пользователя root (утилита **fly-passwd)** и задайте новый пароль пользователю **student** (# **fly-passwd student)**
9. Запланируйте периодический запуск браузера **firefox** (каждый понедельник в 7 утра). Для этого воспользуйтесь утилитой планировщика заданий **fly-admin-cron**
10. Посмотрите настройки сети, при необходимости измените их. Утилита **fly-admin- wicd**
11. Выполните: а) перезагрузку; Ь) смену пользователя; с) выключение; d) блокировку экрана.

Основные каталоги ОС Linux

Имя	Описание1
/	Корневой каталог
/bin	Наиболее важные команды и программы
/boot	Все, что необходимо для загрузки операционной системы, ядро Linux
/dev	Файлы устройств
/etc	Системные конфигурационные файлы
/home	Домашние каталоги пользователей
/lib	Общие библиотеки, модули ядра
/mnt	Это каталог для монтирования локальных и удаленных файловых систем
/opt	Дополнительные программные пакеты
/proc	Информация, касающаяся ядра; управление процессами
/root	Домашний каталог пользователя root
/sbin	Системные команды
/tmp	Временные файлы
/usr	Иерархия вторичных программных файлов
/var	Переменные данные (например, регистрационные журналы); файлы спула

Навигация и поиск с помощью командного интерпретатора. Для навигации по файловой системе пользуйтесь командой cd (она встроена в командный интерпретатор).

Некоторые команды и утилиты для работы в системы

pwd вывод полного имени текущей рабочей директории вывод

Is списка файлов в каталоге вывод списка файлов в каталоге

Is -1 и их атрибутов вывод списка файлов в каталоге, включая

Is -a скрытые

Is -lat вывод списка файлов с их атрибутами, отсортированного по времени последнего изменения переход по дереву каталогов

cd отображение статистики использования диска (покажет список всех каталогов и их размер)

du копирование файла переименование/перемещение файла создание каталога удаление файла

unlink удаление файла (воспринимает те же аргументы, что и команда rm)

rm -d удаление пустого каталога удаление пустого каталога

cp

rmdir удаление каталога, всех файлов и подкаталогов в нем

mv

mkdir создание жесткой ссылки на файл создание

rm

ln символической ссылки на файл монтирование файловой

In -s системы демонтирование файловой системы

mount быстрый поиск файла по имени; осуществляется в базе проиндексированных системой файлов

umount поиск файла

locate

find

uname — вывод информации об операционной системе (прежде всего - ее имя)

alias — назначение командам псевдонимов

unalias — Уничтожает синоним команды.

ps — просмотр списка текущих процессов системы

man — обращение к страницам справочного руководства

bg — Запускает прерванный процесс в "фоновом" режиме.

fg — Переводит процесс из "фонового" режима в "активный" режим.

cat — Выводит на экран содержимое текстового файла.

finger — узнать, кто из пользователей телекоммуникационной сети работает на линии в данный момент.

ftp — Позволяет получать файлы со специалицированных файл-серверов сети "Интернет".

gopher — Позволяет пользоваться одноименным информационным сервисом сети "Интернет Выдает на экран первые десять строк текстового файла.

head — Показывает список использованных комманд.

history — Показывает идентификатор пользователя системы.

id — Дает возможность пользоваться электронной почтой.

mail — Показывает наличие связи с определенным сервером сети "Интернет".

ping — Информирует о версии операционной системы на сервере телекоммуникационной сети.

uname — Дает возможность узнать, кто из пользователей телекоммуникационной сети работает на линии в данный момент.

who

Упражнение 2. Основы командной строки

Что из приведенного ниже является абсолютным путем, относительным путем или простым именем файла или директории?

a) millc_co _____

b) **correspond/business/milk_co** _____

c) /home/alex _____

d) /home/alex/literature/promo _____

e) . . _____

f) letter.0610 _____

1. Напишите команды, с помощью которых можно:

   a) Сделать свой, домашний каталог рабочим _____ _

   b) Узнать свой рабочий каталог_____ _

   Проверьте правильность написания, запустив их в командной строке

2. Пусть в вашем рабочем каталоге /home/student имеется подкаталог Desktop. Приведите три набора команд, с помощью которых в подкаталоге Desktop можно создать каталог с именем classics.

3. Приведите также несколько наборов команд, с помощью которых можно удалить каталог classics вместе с его содержимым._____

4. Какая команда позволяет отобразить информацию по системе?

5. Введите команду uname -s. Какая информация появляется?

_____

6. Введите команду uname -r. Какая информация появляется?

7. Посмотрите текущее время и дату на вашей системе. _____

8. Посмотрите календарь этого месяца.

9. Какой специальный символ используется, чтобы разделить команды в командной строке?

10. Очистите окно терминала _____

11. Используйте опцию **-k** выбор с командой man; найдите страницы помощи, которые описывают, как очистить консольное окно (используйте ключевое слово `clear`) .

12. Посмотрите в какой директории вы сейчас находитесь? _____

13. Перейдите в Вашу домашнюю директорию _____

14. Посмотрите содержимое вашего рабочего каталога _____

В чем разница между этими двумя командами?

15. Посмотрите все файлы в текущей директории, включая скрытые_____

16. Отобразите длинный список содержимого текущего каталога:

17. Посмотритетип файлов в вашей рабочей директории:

18. Перейдите из текущей директории в директорию

19. Перейдите в директорию **/etc/xil**

20. Вернитесь в домашнюю директорию пользователя root

21. Вернитесь в директорию /etc

22. Посмотрите содержимое файла group (данный файл находится в директории /etc)

23. Посмотрите содержимое файлов **passwd** и **group** одной командой

24. Посмотрите первые **5** строк файла **/etc/passwd**

25. Посмотрите последние 3 строки файла **/etc/passwd**

26. Сосчитайте количество строк в файле **/etc/passwd**

27. Перейдите в домашнюю директорию пользователя **student**

28. Создайте три новых файла **filel, file2** и **f±le3**

29. Создайте три новых директории **dirl, dir 2** и **dir3**

30. Скопируйте файл filel в директорию dirl и посмотрите содержимое директории dirl

31. Скопируйте содержимое директории /etc/ssh в созданную директорию dir**2** и посмотрите содержимое директории dir**2**

32. Создайте директорию с именем **vegetables** в директории dir3.

33. Перенесите файл **dir2/ssh/sshd_config** в директорию **dir3/vegetables**

34. Создайте в вашей домашней директории каталог **practical**

35. Скопируйте файлы **filel** и **file2** в директорию **practice!**

36. В Вашей домашней директории одной командой создайте директорию с названием **house** и с поддиректорий **furniture**

37. Создайте пустой файл называющийся chairs в новой директории furniture

38. Используя одну команду, создайте три директории, называющиеся records, memos, и misc в Вашей домашней директории

39. Создайте новый файл carrot, а затем переименуйте его в celery.

40. Одной командой удалите директории memos и misc из домашней директории

41. Попробуйте удалить директории, **house/furniture** командой **rm** без опций, что произойдет?

42. Как удалить командой **rm** директорию **house/furniture,** если она не пуста?

43. Создайте новою директорию newname, и перенесите ее в **dir3/vegetables**

44. Создайте символическую ссылку **mycontents,** которая ссылается на файл **/var/log/messages**

45. Создайте жесткую ссылку **hard_link** на файл **mycontents**

46. Поэкспериментируйте с вызовами утилиты **file** с именами файлов в **/usr/bin.** Какие типы файлов встречаются в данном каталоге чаще всего?

47. Что получится в результате вызова утилиты **which** с именем команды, находящейся в каталоге, который не указан в вашем пути поиска?

Модуль 3. Работа с текстовым редактором Vi/Vim

Vim - улучшенная версия текстового редактора **vi** и совместимая с ним. **vi** - один из первых редакторов, разработанных для операционных систем UNIX. Он и по сей день остается одним из самых мощных редакторов и стандартно поставляется практически с каждой операционной системой типа UNIX. В текстовом редакторе **vi** отсутствуют меню, и все действия осуществляются с помощью клавиш и их специальных комбинаций.

Чтобы запустить редактор **vi,** необходимо ввести **vi.** Если за ним последует имя файла, он откроется для редактирования, а если вы зададите имя несуществующего файла, то **vi** будет считать, что это новый файл.

```
$ vi file.txt
```

Редактор **vi** обеспечивает, как режим вставки (режим редактирования), так и режим просмотра (его называют командным режимом). Сразу после запуска редактор **vi** будет находиться в командном режиме. В этом режиме нажатия клавиш интерпретируются как команды редактору, а не как текст, вводимый в документ. Чтобы переключиться в режим ввода текста необходимо нажать одну из клавиш **a, i** или **o.** Клавиша **a** обозначает append (присоединить). В этом режиме вводимый текст вставляется после символа, на котором находится курсор. Клавиша **i** обозначает insert (вставить). В этом режиме вводимый текст вставляется после символа, на котором находится курсор.

Если из режима ввода текста вы хотите вернуться в командный режим, нажмите клавишу Escape. При переходе в командный режим редактор vi по умолчанию выдает звуковой сигнал.

Перемещение по тексту в редакторе vi

Чтобы воспользоваться клавишами перемещения нажмите клавишу Escape и перейдите в командный режим. В этом режиме можно пользоваться такими клавишами h, j, к и i для перемещения курсора влево, вниз, вверх и вправо, соответственно. В командном режиме имеется еще несколько клавиш перемещения. В таблице 2 приведены различные клавиши и их функции:

Клавиша	Действие
W	Перемещает курсор вперед на одно слово.
B	Перемещает курсор назад на одно слово
E	Перемещает курсор в конец следующего слова.

0	Перемещает курсор в начало строки
$	Перемещает курсор в конец строки
)	Перемещает курсор в начало следующего предложения
(	Перемещает курсор в начало предыдущего предложения
}	Перемещает курсор в начало следующего абзаца
{	Перемещает курсор в начало предыдущего абзаца
G	Перемещает курсор в конец текущего документа
H	Перемещает курсор на первую строку на экране
L	Перемещает курсор на последнюю строку на экране

Чтобы работать с текстом понадобится ещё несколько команд. Вот они:

x Удаляет символ в позиции курсора и сдвигает следующие за ним символы влево.

X Удаляет символ перед курсором и сдвигает следующие за ним символы влево.

Заменяет букву в позиции курсора той же буквой другого регистра.

D Удаляет текст от позиции курсора до конца строки, dd Удаляет всю текущую строку целиком.

ndd Здесь n — число удаляемых строк. Например, команда 5dd удаляет текущую строку и четыре строки, следующие за ней.

:q Завершает работу редактора.

:q! Завершает работу программы без сохранения сделанных изменений.

:sh Запустить командный интерпретатор (чтобы вернуться в редактор, введите ‘exit’).

:w Сохранить редактируемый файл.

/ поиск текста.

Упражнение 3. Использование редактора vi

1. Запустите команду **`vimtutor`.**
2. Выполните уроки с 1-го по 4-ый.

Модуль 4. Команды поиска. Конвейеры и перенаправление ввода-вывода

Потоки входных и выходных данных для команды называются *ввод* и *вывод*. Потоков ввода и вывода у каждой программы может быть несколько. В Linux каждый процесс, при создании в обязательном порядке получает так называемые стандартный ввод (standard input, stdin) и стандартный вывод (standard output, stdout) и стандартный вывод ошибок (standard error, stderr).

Стандартные потоки ввода/вывода предназначены для обмена текстовой информацией. При работе с командной строкой, стандартный ввод командной оболочки связан с клавиатурой, а стандартный вывод и вывод ошибок^- с экраном монитора. На примере команды **cat** рассмотрим работу потока входных и выходных данных. Обычно команда **cat** читает данные из всех файлов, которые указаны в качестве её параметров, и посылает считанное непосредственно в стандартный вывод (stdout). Следовательно, команда

```
# cat passwd group
```

выведет на экран сначала содержимое файла **passwd**, а затем — файла **group**.

Однако если имя файла не указано, программа cat читает входные данные из stdin и сразу возвращает их в stdout, при этом не изменяя. Например:

```
# cat
Hello, tbSresl "'
Hello there.
Bye,
Bye|№f Ctrl+D
#
```

Каждую строку, вводимую с клавиатуры, программа **cat** возвращает на экран. При вводе информации со стандартного ввода конец текста сигнализируется вводом специальной комбинации клавиш, как правило **Ctrl+D.**

Рассмотрим еще один пример. Команда **sort** читает строки вводимого текста (также из stdin, если не указано ни одного имени файла) и выдаёт набор этих строк в упорядоченном виде на stdout.

```
# sort
bananas
 carnets
 apples
 Ctrl+D
 apples
 hah#nas
 ca^rets
```

Как видно, после нажатия **Ctrl+D, sort** вывела строки упорядоченными в алфавитном порядке.

Перенаправление ввода и вывода. Допустим, вы хотите направить вывод команды **sort** в некоторый файл, чтобы сохранить упорядоченный по алфавиту список на диске. Командная оболочка

позволяет перенаправить стандартный вывод команды в файл, используя символ >.

```
# sort > shopping-list
bananas
carrots
apples
Ctrl+D#
```

Видим, что результат работы команды **sort** не выводится на экран, но сохраняется в файле с именем **shopping-list.** Посмотрим содержимое данного файла:

```
# cat shopping-list
apples
bananas
carrots
```

Создадим файл **items** с неупорядоченным списком. Этот список можно упорядочить с помощью команды **sort**, если указать ей, что она должна читать из данного файла, а не из своего стандартного ввода. Также мы можем перенаправить стандартный вывод в файл.

```
# sort items shopping-list
# cat shopping-list
apples
bananas
carrots
```

Можно поступить иначе, перенаправив не только стандартный вывод, но и стандартный ввод утилиты из файла, используя для этого символ *<1*

```
# sort < items apples
bananas carrots
```

Результат команды **sort < items** эквивалентен команде **sort items,** но в первом случае при выдаче команды **sort < items** система ведёт себя так, как если бы данные, которые содержатся в файле **items,** были введены со стандартного ввода. Перенаправление осуществляется командной оболочкой. Команде **sort** не сообщалось имя файла **items:** эта команда читала данные из своего стандартного ввода, как если бы мы вводили их с клавиатуры.

Конвейер. Теперь попробуем сортировать данные в обратном алфавитном порядке; используя опцию -г команды sort. Если вы хотите перечислить файлы в текущем каталоге в обратном алфавитном порядке, один из способов будет таким. Применим сначала команду Is:

```
# Is
english-list
history-final
masters-thesis notes
```

Теперь перенаправляем выход команды Is в файл с именем `file-list`

```
# Is > file-list
# sort -r file-list notes
```

```
masters-thesis
history-final
english-list
```

Здесь выход команды `Is` сохранен в файле, а после этого этот файл был обработан командой `sort -r`. Однако, этот путь требует использования временного файла для хранения выходных данных программы is.

Решением в данной ситуации может служить создание состыкованных команд `(pipelines)`. Стыковку осуществляет командная оболочка, которая stdout первой команды направляет на stdin второй команды. `B` данном случае мы хотим направить stdout команды `Is` на stdin команды `sort`. Для стыковки используется символ |, как это показано в следующем примере:

```
#  Is | sort -r
notes
masters-thesis
history-final
english-list
```

Недеструктивное перенаправление вывода. Эффект от использования символа > для перенаправления вывода файла является деструктивным; т.е. команда

```
#  Is > file-list
```

уничтожит содержимое файла **file-list,** если этот файл ранее существовал, и создаст на его месте новый файл. Если вместо этого перенаправление будет сделано с помощью символов », то вывод будет приписан в конец указанного файла, при этом исходное содержимое файла не будет уничтожено. Например, команда

```
#  Is » file-list
```

приписывает вывод команды **Is** в конец файла **file-list.**

Упражнение 4. Команды поиска. Конвейеры и перенаправление ввода-вывода

1. Найдите строку, содержащую слово operator в файле /etc/group
2. Отобразите все строки, содержащие слово root из файлов /etc/passwd и /etc/group
3. Подсчитайте количество строк в которых содержится пользователь student в файле /etc/group
4. Подсчитайте количество строк в которых НЕ содержится пользователь student в файле /etc/group
5. Используя команду grep, отобразите все строки, содержащие слово List или слово System в файле /etc/passwd
6. Найдите в домашней директории пользователя student все файлы и директории, имя которых заканчивается на **1**
7. Найдите в директории /usr все файлы и директории, имя которых заканчивается на In
8. Найдите в домашней директории пользователя student только файлы
9. Найдите в домашней директории пользователя student только директории
10. У кого из пользователей в качестве оболочки используется bash
11. Найдите все файлы в директории /etc с соответствующими правами доступа: владелец и группа имеют полные права и ограничение на запись имеют остальные пользователи
12. Найдите все файлы принадлежащие пользователю student
13. Найдите все файлы принадлежащие группе ftp
14. Найдите все файлы несуществующих пользователей или групп
15. Найдите все файлы являющиеся символическими ссылками в директории /etc
16. Найдите все файлы в системе размером более **200M**
17. Найдите все файлы с расширением .html в директории /usr/share/doc
18. Найдите в директории /trap файлы нулевого размера, а затем удалите не нужные из них

Модуль 5. Архивирование и сжатие файлов

Существует различие между архивированием и сжатием файла. Архивный файл — это набор файлов и каталогов, помещенных в один файл. Архивный файл занимает такое же дисковое пространство, какое занимают все файлы и каталоги, входящие в него. Сжатый файл — это набор файлов и каталогов, помещенных в один файл таким образом, что он занимает меньше дискового пространства, чем занимают файлы и каталоги, входящие в него. Если в вашем компьютере не слишком много свободного дискового пространства, вы можете сжать некоторые из файлов. Также вы можете создать архивный файл и затем сжать его, чтобы сэкономить дисковое пространство.

Сжатые файлы занимают меньше дискового пространства и могут быть загружены быстрее, чем большие несжатые файлы. Для сжатия файлов используются следующие программы: bzip**2**, gzip или zip.

bzip**2** - обеспечивает наибольшее сжатие и поддерживается большинством UNIX-подобных операционных систем. Программа сжатия gzip также поддерживается большинством UNIX-подобных операционных систем. При необходимости переноса файлов между Linux и другими операционными системами, такими как MS Windows, лучше использовать программу сжатия zip, так как она наиболее совместима с программами, используемыми для сжатия файлов в Windows.

Программа сжатия	Расширение файла	Программа декомпрессии
bzip**2**	.bz**2**	bunzip**2**
gzip	•gz	gunzip
zip	.zip	unzip

Файлам, сжатым с помощью программы bzip**2**, принято давать расширение .bz**2**, файлам, сжатым с помощью программы gzip, принято давать расширение .gz и файлам, сжатым с помощью программы zip, принято давать расширение .zip.

Для декомпрессии файлов, сжатых с помощью программы gzip, используется программа gunzip, файлов, сжатых с помощью программы bzip**2**, используется программа bunzip**2** и файлов, сжатых с помощью программы zip, используется программа unzip.

Упражнение 5. Архивирование и сжатие файлов

1. Используйте команду find, чтобы определить местонахождение всех файлов принадлежащих пользователю root в домашней директории пользователя student, а

затем заархивируйте их с помощью команды tar. Архив должен находиться в каталоге / tmp

2. Добавьте содержимое каталога /etc в соседний архив в директорию /tmp
3. Посмотрите размер полученных файлов:
4. Используйте gzip для сжатия ваших архивов и снова посмотрите получившийся размер файлов
5. Отмените сжатие существующих файлов и сожмите их снова с помощью утилиты bzip**2**, и снова посмотрите размер

Модуль 6. Права доступа к файлам

В Linux существует три права доступа: чтение (r - от read), запись (w - от write), выполнение (x - от eXecute). Право выполнения для файла означает, что его можно запустить на выполнение. Обычно право выполнения устанавливается для программ. А для каталога право выполнения означает право просматривать оглавление каталога.

Вы можете установить разные наборы прав доступа для владельца файла, группы владельца и для всех остальных пользователей. Сейчас продемонстрирую, как это выглядит на практике. Откройте терминал и введите команду:

```
# Is -1 <имя файла>
```

```
-rw-r—r— 1 student users 6051 Ноя 28 14:44 <имя_файла>
```

Строка -rw-r--r— права доступа. Первый символ (-) означает, что это файл, а не каталог (для обозначения директорий используется символ "d" (от directory)).

Первые три символа (после тире) означают права доступа для владельца. Владельцу (пользователю student) разрешено читать (r), изменять файл (w), но запрещено запускать его на выполнение (третий символ - тире, а не "x").

Следующие два набора означают права доступа для членов группы (группа users) первый набор, и всех остальных пользователей (которые не являются членами группы и владельцем файла) второй набор. И те, и другие имеют право только читать файл (r-).

Для установки прав доступа к файлу и каталогу используется команда chmod. Сначала указываются права доступа, а затем имя файла/каталога. Помните, что вы можете установить права доступа только к "своим" файлам, т.е. к файлам, владельцем которых вы являетесь. Пользователь root имеет право изменить права доступа абсолютно любого файла/каталога. Перед выполнением команды необходимо выбрать класс пользователей, для кого мы хотим изменить права (ugo, User, Group, Others). Есть так же дополнительная группа a (all, все). Затем выбираем оператор: + (дать права), - (убрать права) и = (присвоить права). И в конце выбрать сами права (rwx, r-x и т.д.).

```
# chmod u+x skrypt.sh
```

Пользователю (и) добавляется (+) право на выполнение (x) файла skrypt. sh.

```
# chmod go-r filel
```

У группы и остальных (go) отбирается (-) право на чтение (r).

```
# chmod a=w filel
```

Всем (a) присваивается (=) право на запись (w), а остальные права стираются (=).

```
# chmod u+rwx,g+rwx,o+x filel
```

Пользователю прибавляются права на все, группе тоже, а остальным прибавляется право на выполнение.

Для упрощения записи команды chmod можно использовать числовое представление прав. У каждого типа доступа есть числовое представление, для этого используется двоичное представление. Единица означает - есть право, 0 - нет права. Таким образом запись правила rwx r-x r-x в бинарном виде будет выглядить следующим образом: 111 101 101. Но двоичное представление не очень удобно, поэтому используют десятичное представлени. 111 в двоичной системе - это 7 в десятичной, а 101 - это 5, таким образом 111 101 101 - это 755. Итак получаем:

-х 1 —w- 2 —wx 3 r- 4 r-x 5 rw- 6 rwx 7 Таким образом вы можете записать как

```
# chmod u=rwx,g=rwx,o=x filel
```

так и

```
# chmod 771 filel
```

Суммарное представление прав доступа показано на рисунке ниже:

Упражнение 6. Права доступа к файлам

1. Отключитесь от всех терминалов и переключитесь на 1-ый терминал (ttyl): <CTRL- ALT-F1>
2. Войдите в систему как обычный пользователь (не root)
3. Получите информацию о данном логине:
4. Получите информацию обо всех параллельных логинах на локальной рабочей станции (должен только быть один пользователь, зарегистрированный в системе).
5. Переключитесь в виртуальный терминал 2 (tty2): <CTRL-ALT-F2>
6. Войдите с данного терминала как пользователь student с паролем student
7. Получите информацию о данном логине:
8. Получите информацию обо всех параллельных логинах на локальной рабочей станции
9. Находясь в системе, под пользователем student посмотрите маску:
10. Создайте пару файлов и директорий, не изменяя прав доступа к ним:

11. Измените вашу маску на более безопасную, а затем создайте новый файл У[10] директорию:
12. Как изменились права доступа у новых файлов? ___ —---------- ------- --- - *

Модуль 7. Работа с пользователями

Linux —многопользовательская ОС, потому пользователь — ключевое понятие для организации всей системы доступа в Linux. Когда пользователь регистрируется в системе (проходит процедуру авторизации, например, вводя системное имя и пароль), он идентифицируется с учётной записью, в которой система хранит информацию о каждом пользователе: его системное имя и некоторые другие сведения, необходимые для работы с ним. Именно с учётными записями, а не с самими пользователями, и работает система.

Linux связывает системное имя с идентификатором пользователя в системе — UID (User ID). UID — это положительное целое число, по которому система и отслеживает пользователей. Обычно это число выбирается автоматически при регистрации учётной записи. В Linux есть некоторые соглашения относительно того, каким типам пользователей могут быть выданы идентификаторы из того или иного диапазона.

Кроме идентификационного номера пользователя с учётной записью связан идентификатор группы. Группы пользователей применяются для организации доступа нескольких пользователей к некоторым ресурсам. У группы, так же, как и у пользователя, есть имя и идентификационный номер— GID (Group ID). В Linux каждый пользователь должен принадлежать как минимум к одной группе — группе по умолчанию. При создании учётной записи пользователя обычно создаётся и группа, имя которой совпадает с системным именем, именно эта группа будет использоваться как группа по умолчанию для этого пользователя. Пользователь может входить более чем в одну группу, но в учётной записи указывается только номер группы по умолчанию. Группы позволяют регулировать доступ нескольких пользователей к различным ресурсам.

Файлы всех пользователей в Linux хранятся раздельно, у каждого пользователя есть собственный домашний каталог, в котором он может хранить свои данные. Доступ других пользователей к домашнему каталогу пользователя может быть ограничен. Информация о домашнем каталоге обязательно должна присутствовать в учётной записи, потому что именно с него начинает работу пользователь, зарегистрировавшийся в системе.

Все перечисленные данные об учётных записях хранятся в файле **/etc/passwd.** Файл представляет собой таблицу, каждая строка которой представляет отдельную учетную запись, состоящую из 7 полей. Обратите внимание на характерные разделители полей (регистрационное имя, признак пароля, идентификатор пользователя, идентификатор группы, дополни тельная информация, "домашний" каталог, имя командного процессора) в виде двоеточия. Символ x вместо пароля указывает на то, что хэшированный пароль находится в другом файле - **/etc/shadow.** Он также представляет собой таблицу, каждая

строка которой состоит из 9 полей, разделенных двоеточиями (регистрационное имя, хэшированный пароль, контрольные сроки в днях, среди которых:

- число дней с 1.01.70 г. до дня последнего изменения пароля,
- минимальное число дней действия пароля со дня его последнего изменения,
- максимальное число дней действия пароля,
- за сколько дней до устаревания пароля система начнет выдавать предупреждения,
- число дней со времени обязательной смены пароля до блокировки учетной записи,
- день блокировки учетной записи.
- Последнее, девятое поле зарезервировано и не используется.

Также отдельно хранится информация обо всех группах пользователей в системе, для этого предназначен файл /etc/group. В файле /etc/group: сначала идёт имя группы (как и имя учётной записи, потом поле для пароля (здесь опять “x”, но пароли для группы используются очень редко), GID, список через запятую названий учётных записей (имён пользователей), входящих в данную группу. Любой пользователь может получить список названий групп, в которых он состоит командой groups, а более подробные сведения о своей или чужой учётной записи командой id имя_пользователя. Принадлежность к группе существенна только в одном отношении— прав доступа, поскольку для каждого файла определён не только пользователь-владелец, но и группа-владелец.

Добавить учётную запись можно с помощью команды useradd. Утилита passwd, вызванная с правами суперпользователя, позволяет назначить данному пользователю любой пароль. При этом сведения о предшествующем пароле данного пользователя (если таковой был), будут полностью утрачены, passwd, вызванная обычным пользователем (без параметров), позволяет ему сменить свой собственный пароль, но для этого потребуется ввести текущий пароль пользователя.

Существуют аналогичные useradd утилиты для модификации параметров уже существующей учётной записи (usermod) и для удаления пользователей (userdel). Пользователь также может изменить некоторые некритичные сведения в своей учётной записи самостоятельно.

Для управления группами существует комплект утилит groupadd, groupdel, groupmod, подробности о работе с ними можно найти в соответствующих руководствах.

Упражнение 7.1.

1. У вас должно быть право на чтение файла /etc/passwd. Для ответа на следующие вопросы воспользуйтесь для вывода на экран /etc/passwd утилитой cat или less; Просмотрите в файле /etc/passwd поля с информацией о пользователях вашей системы.

   a) Какой символ используется для разделения полей в /etc/passwd? _____

   b) Сколько полей используется для описания каждого пользователя?_____

   c) Сколько пользователей в вашей системе?_____

2. Сколько различных входных оболочек используется в вашей системе? (Совет: посмотрите на последнее поле.)

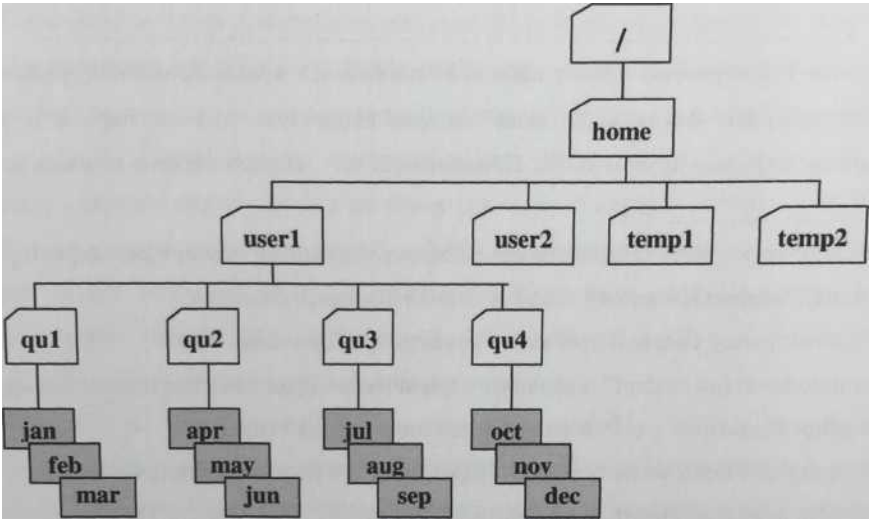
3. Второе поле в /etc/passwd хранит пароли пользователей в закодированной форме. Если поле пароля содержит букву x, то ваша система использует теневые пароли и хранит закодированные пароли в другом месте. Применяются ли в вашей системе теневые пароли?

4. Добавьте трех новых пользователей с соответствующими домашними директориями: student7, students, student9. Залайте пароли для каждого из них.
5. Создайте группу course и добавьте в нее всех трех пользователей.
6. Для пользователя student7 выставите ограничение: срок действия пароля 5 месяцев и предупреждение об окончании срока действия пароля 7 дней
7. Заблокируйте пользователя student8. Проверьте что блокировка подействовала.
8. Войдите в ситсему под пользователем student9
9. Создайте два новых файла filel и file2 и две новых директории dirl и dir2
10. Войдите в систему под пользователем root
11. Сделайте filel исполняемым. Запускаться файл должен от имени владельца и группы файла
12. Установите в качестве владельца файла filel пользователя student7 и группу course
13. Войдите в систему под пользователем student7. Попробуйте перейти в директорию пользователя student9 и сделать запись в файле filel.
14. Войдите в систему под пользователем root. Разблокируйте пользователя student8. Проверьте что блокировка снята

Упражнение 7.2.

1. Создайте в ОС Linux двух пользователей (userl и user2) и задайте их пароли. Зарегистрируйтесь в первой консоли как userl.

3. С помощью Ctrl+Alt+F2 (Alt+F2) откройте второй текстовый терминал и зарегистрируйтесь как user**2**.
4. Аналогично откройте третий текстовый терминал и зарегистрируйтесь в нем с правами суперпользователя.
5. Нажатием Ctrl+Alt-Fl (Alt-Fl) вернитесь в первую консоль. Теперь, переключая консоль, вы можете работать с объектами операционной системы от имени двух разных пользователей и администратора системы. Основная часть задания выполняется с правами обычного пользователя. Переходите в третью консоль и используйте права root только при выполнении соответствующих пунктов задания.



6. С правами userl попробуйте войти в каталог /root. Объясните результат. С помощью команды Is -la / просмотрите список основных каталогов и укажите, каких прав доступа вам недостает для входа в каждый из каталогов
7. Переключитесь в консоль администратора и создайте два новых временных каталога mkdir -m 777 /home/templ и mkdir -m 1777 /home/temp2. Проверьте права доступа к каталогам /bome/user **1** и /home/user**2:** они должны быть установлены в 755. Вернитесь в консоль userl.
   Пользуясь командой mkdir, создайте в домашнем каталоге пользователя /home/userl четыре каталога с именами: qul, qu2, qu3, qu4. При создании каталогов объявите следующие права доступа к ним: (qul - 777, qu2 - 404, qu3 - 1333, qu4 - 505. Пример: cd; mkdir -m 777 qul). С помощью команды Is /home/userl убедитесь в том, что каталоги созданы. Какие из предоставленных прав кажутся Вам лишенными смысла? Почему?

8. Задайте права доступа к файлам "по умолчанию". Для этого установите umask 022. Поясните, какие права к вновь создаваемым файлам и каталогам будут предоставляться пользователю, членам его группы и остальным.
9. В каждом из каталогов создайте по три текстовых файла с именами (jan, feb, mar), (apr, may, jun), (jul, aug, sep), (oct, nov, dec). В каждый файл запишите календарь на определенный месяц текущего года. Например, команда cal 12010 >jan создает в текущем каталоге файл jan и записывает в него календарь на январь 2010 года. Не забывайте, что использование относительного (короткого) имени файла требует, чтобы Вы находились в нужном ката логе. В противном случае следует указывать полный путь к создаваемому файлу. Для навигации по каталогам используйте команды cd и pwd. В каком случае создание файлов не удалось? Почему?
10. С помощью команды chmod измените нужные права доступа в "недоступные" каталоги qu2, qu4 и создайте там указанные файлы. После этого верни те каталогам прежние права доступа.
11. С помощью команд cd и Is войдите в каждый из созданных каталогов и просмотрите список созданных файлов. Для просмотра каталога необходимо по следовательно ввести две команды: cd и Is. При просмотре используйте два режима: Is без аргументов и Is -I. В каких случаях не удалось войти в ката лог? В каких случаях не удалось посмотреть список файлов? Почему?
12. Прочитайте содержимое одного из файлов в "темном" каталоге (например, cd /home/userl/qu3; cat aug). Сделайте выводы.
13. Перейдите во 2-ю консоль и с правами пользователя user2 войдите в каталог /home/user 1/qul. Создайте в каталоге /home/user2 новый файл quartl путем конкатенации нескольких имеющихся (cat jan feb mar >/home/user2/quartl). С помощью команды file определите тип созданного файла. Попробуйте вывести его на экран командой cat. Что представляет собой данный файл?
14. С помощью команды chmod установите права доступа 077 на созданный файл quartl. Вновь попробуйте прочесть его. Ответьте, почему владельцу файла запрещается доступ, если файл доступен для всех? Что необходимо сделать, чтобы вернуть владельцу права на доступ?
15. Установите для файла quatrl права на доступ 4700. Кому и какие права вы при этом предоставили? Как воспользоваться этими правами? Какие из предоставленных прав не имеют смысла?
16. Перейдите в консоль администратора и передайте право владения на файлы may и aug

пользователю user2 (команда chown). Поочередно из консолей userl и user2 проверьте, как изменились права владения файлами после его передачи. Может ли пользователь user2 воспользоваться предоставленными правами?

17. С правами пользователя userl из каталогов /home/templ и /home/temp2 с по мощью команды In создайте две "жесткие" ссылки на файл dec с именами dec_hl и dec_h2 (пример: In /home/userl/qu4/dec /home/templ/dec_hl). Чем созданные ссылки отличаются от исходного файла? На сколько байт уменьшилось дисковое пространство после создания этих ссылок?
18. С помощью команды In -s создайте из каталогов /home/templ и /home/temp2 две символические ссылки на файл dec с именами dec_sl и dec_s2. Чем отличаются созданные ссылки от исходного файла? Попытайтесь прочитать содержимое файлов символических ссылок. Что они собой представляют?
19. С правами пользователя user2 с помощью команды ср создайте в каталогах /home/templ и /home/temp2 по одной копии файла dec с другим именем (dec_copyl). Чем отличаются исходный файл и его копия (обратите внимание на то, кто является владельцем исходного файла и его копии)? Чем отличаются права доступа на эти файлы? Вернитесь в консоль userl.
20. С помощью команды rm удалите файл dec. Что произошло с "жесткими" и символическими ссылками на данный файл? Что произошло с его копиями? Что нужно сделать для того, чтобы файл перестал существовать (на логическом уровне)?
21. С правами userl удалите файлы из каталогов /home/templ и /home/temp2. Какие файлы не удалось удалить? Почему? Попробуйте удалить оставшиеся файлы правами пользователя user2. Объясните результат.
22. Попробуйте удалить любой из каталогов qul, qu2, qu3, qu4 с помощью команды rmdir (не удаляя предварительно из них файлов). Объясните результат.
23. Войдите в консоль администратора и с правами root, пользуясь командой chattr, заблокируйте файл feb от любых изменений. Установите параметр запрета любых операций, кроме добавления данных для файла mar. Вернитесь в консоль userl. С помощью команды lsattr -1 проверьте наличие дополнительных атрибутов у файлов.
24. С правами пользователя userl добавьте одну строку finish в конец файлов feb и mar (воспользуйтесь для этого командой echo finish » filejname). Убедитесь в успешном завершении операции, объясните результат.
25. С правами пользователя userl с помощью команды rm -rf последовательно удалите

г

ранее созданные каталоги qu2, qu3, qu4 вместе с файлами. Объясните результат.

26. С помощью команды md5sum вычислите и запишите контрольную сумму для одного из файлов в каталоге /home/userl/qul. Добавьте один символ в этот файл с помощью команды echo (например, echo a» /home/userl/qul/jan). Вновь вычислите контрольную сумму файла и сравните два результата.
27. С помощью команды cat /dev/fdO попробуйте прочесть специальный файл устройства. Объясните результат? Для чего служат специальные файлы?
28. С помощью команды find с правами администратора найдите в корневом каталоге файлы:
    - имеющие атрибуты SUID (find / -type f -perm -4000);
    - файлы, которые разрешено модифицировать всем (find / -type f - perm -2);
    - файлы, не имеющие владельца (find / -nouser);
    - объясните, какой интерес могут представлять для администратора указанные категории файлов?
29. Зарегистрируйтесь в системе в консольном режиме с правами root.

Упражнение 7.3. Работа с учетными записями пользователей

30. Используя команду cat с правами root, просмотрите содержимое файла /etc/passwd.
31. Аналогично изучите содержимое файла /etc/shadow.

Вам необходимо создать учетные записи и определить права доступа для десяти (10) сотрудников: w gromov, n_kalinina, e_ivanova, r_klinova, b_rebrov, kjbeglov, i_frolov, d_lavrov, m_kruglov, t_uporov, работающих в одном подразделении и занятых созданием и редактированием текстовых документов различного уровня конфиденциальности. Разграничение доступа к информации должно быть произведено на основании следующих требований:

- допуск к секретным сведениям имеют четыре пользователя: w_gromov, n_kalinina, b_rebrov, kjbeglov;
- три пользователя: n_kalinina, b_rebrov, kjbeglov работают над созданием секретных документов, каждый по своему профилю. Их домашние каталоги и файлы должны быть полностью недоступными как друг для друга, так и для всех остальных, исключая w gromov;
- три пользователя: i_frolov, djavrov, ejvanova имеют допуск к конфиденциальной информации и работают над документами с соответствующим грифом. Они имеют право читать файлы с конфиденциальной информацией, созданные своими коллегами, без права их модификации;

- все секретоносители имеют право знакомиться с конфиденциальными файлами;
- три пользователя: r_klinova, m_kruglov, t_uporov могут работать только с открытой информацией. Их файлы должны быть доступны для чтения каждому сотруднику подразделения (без права модификации);
- w_gromov является редактором подразделения и имеет право читать и модифицировать файлы всех сотрудников и всех уровней конфиденциальности. Завершенные документы копируются пользователем w_gromov в его домашний каталог, который должен быть недоступен для всех остальных сотрудников подразделения.

32. Укажите в отчете, какие коллизии вы усматриваете в сформулированных требованиях? Как реализовать указанные требования таким образом, чтобы пользователи не могли по своему усмотрению изменять установленный по рядок?
33. С помощью команды groupadd создайте четыре пользовательских группы: alfa, beta, nabla, sigma. Формат команды groupadd -g GID groupname. Идентификатор группы GID можно назначать произвольно, начиная с номера 100 (например, groupadd -g **101** alfa).
34. Создайте учетные записи для вышеуказанных десяти новых пользователей. Регистрационные данные (кроме паролей и групп) сведены в таблицу. Пароли назначайте произвольно, длиной не менее **8** символов, не забывая фиксировать их в черновике отчета. Для пользователей e_ivanova, r_klinova задайте одинаковые пароли. Распределите сотрудников по группам таким образом, чтобы удовлетворить вышеперечисленным требованиям. Изобразите в отчете схему, поясняющую разграничение доступа сотрудников подразделения к компьютерной информации.
35. Пять первых пользователей (w_gromov, n_kalinina, e_ivanova, r_klinova, b_rebrov) зарегистрируйте с помощью команды useradd. Синтаксис коман ды: useradd -и UID -g groupname -d dirhome - m -p password -e date_del_user user name. Например, useradd -u 501 -g sigma -d /home/n_kalinina -p v5g7K2S4 -e 2011-01-07 n_kalinina. Параметр -m обеспечивает создание домашнего каталога пользователя, если он еще не существует. Прочие параметры команды можно не указывать. *Помните, имя пользователя не должно начинаться с цифры и содержать заглавных и русских букв, символов типа* *#%ª.... *Идентификаторы пользователей UID назначаются, начиная с 500.* Дата удаления учетной записи пользователя вводится в формате ГГГГ-ММ-ДЦ.

36. Пять последних пользователей зарегистрируйте с помощью командного файла adduser, которая запрашивает значения в интерактивном режиме. При вводе данных ориентируйтесь на подсказки системы [в квадратных скобках]. Все параметры, кроме имени пользователя, его идентификатора, имени группы, пароля и домашнего каталога можно игнорировать. Для ввода параметра по умолчанию вводить Enter.

Пользователи	UID	Пароль	Группа	Домашний каталог	Дата удаления учетной записи
i_frolov	501			/home/ifrolov	T+ 10 дней
mkruglov	502			/home/m_kruglov	T+ 30 дней
b_rebrov	503			/home/b_rebrov	T+ 12 дней
d_lavrov	504			/home/dlavrov	T+ 60 дней
e_ivanova	505			/home/e_ivanova	T+ 30 дней
tuporov	506			/home/tuporov	T+ 15 дней
kbeglov	507			/home/k_beglov	T+ 45 дней
n_kalinina	508			/home/n_kalinina	T+ 30 дней
rjklinova	509			/home/rklinova	T+ 90 дней
w gromov	510			/home/w_gromov	не удалять

37. Переключаясь во вторую консоль, отслеживайте изменения, происходящие в файле /etc/passwd по мере ввода новых учетных записей.
38. Попробуйте с правами пользователя посмотреть файл /etc/shadow. Повторите попытку просмотра с консоли суперпользователя. Почему поля, отведенные для хэшированных паролей у пользователей e_ivanova и r_klinova различаются?
39. Из первой консоли с помощью команды su измените права администратора на права пользователя w_gromov. Почему система не запрашивает пароль? С помощью команды exit верните себе права администратора. Был ли запрошен пароль? (В различных дистрибутивах Linux возврат полномочий адми нистратора организован различным образом).
40. Запустите оболочку Midnight Commander в режиме редактирования (F4) файла паролей /etc/passwd и удалите в учетной записи пользователя n_kalinina символ признака пароля (между первым и вторым двоеточием), включая пробел. Сохраните изменения в файле, завершите сеанс в Midnight Commander, с помощью Ctrl+Alt+F2 (Alt+F2) откройте второй текстовый терминал и зарегистрируйтесь пользователем n_kalinina, но теперь с «пустым» паролем. Сделайте вывод относительно опасности предоставления прав на запись в этот файл. Завершите сеанс для пользователя n_kalinina с помощью команды exit.
41. Пользователь d_lavrov уволен за дисциплинарный проступок. С помощью команды

userdel -r user_name удалите его учетную запись вместе с домашним каталогом. В реальных условиях необходимо вначале скопировать в другую директорию файлы пользователя, представляющие ценность для организации.

42 Зарегистрируйте вместо уволенного пользователя нового сотрудника f_mironov с предоставлением ему аналогичных прав (пароль должен быть новым!).

43. Пользователь rjklinova убыла в командировку сроком на две недели. Заблокируйте ее учетную запись, для чего с правами администратора войдите в режим редактирования файла паролей и вставьте во второе поле (между первым и вторым двоеточием) любой символ, который не разрешено использовать для пароля. Попытайтесь зарегистрироваться во второй консоли с правами r_k!inova и убедитесь в том, что для этого пользователя система не доступна.

44. Зарегистрируйтесь во второй консоли с правами пользователя kjbeglov, вызовите команду passwd и измените свой пароль. В качестве нового пароля введите qwerty.

45. Перейдите в консоль администратора и назначьте пользователю kjbeglov новый пароль zxcvbnm. Затем с помощью команды chage (change aging - изменить информацию об устаревании) установите для этого пользователя минимальное время действия паролей, равное 5 дням. С какой целью устанавливается минимальный срок действия пароля?

46. Просмотрите электронную справку по файлу /etc/sudoers. Отредактируйте его таким образом, чтобы предоставить следующим пользователям дополни тельные права за счет использования команды sudo:
    - пользователю e_ivanova - право монтировать файловые системы,
    - пользователю b_rebrov - право изменения владельца файлов.

    Ответьте, чем отличается предоставление прав пользователям с помощью sudo от использования эффективных идентификаторов SUID?

47. Из второй консоли с правами пользователя fjmironov создайте файл cal 2010 > /home/f_mironov/cal2010. С помощью команды su переключите консоль на пользователя b_rebrov и с помощью временно предоставленных ему привилегий передайте права на созданный f_mironov файл другому владельцу n_kalinina. Каким еще путем можно предоставить подобные права пользователям, не передавая им "опасных" полномочий администратора?

48. Просмотрите с правами администратора системные журналы в каталоге /var/log и убедитесь, что система зафиксировала факты присвоения полномочий администратора.

Модуль 8. Процессы

Процесс (process) — блок адресного пространства в котором выполняются одна или более нитей, экземпляр выполняемой программы. Любой процесс может запускать другие процессы.

Таким образом, процессы в среде UNIX образуют иерархическую структуру. На вершине этой структуры находится процесс init, являющийся предком всех остальных процессов.

С каждым процессом связан набор атрибутов, которые помогают системе контролировать выполнение процессов и распределять между ними ресурсы системы.

- **Идентификатор процесса (process ID)** это целое число однозначно

  идентифицирующее процесс. Процесс с идентификатором 1 это процесс init.
- **Идентификатор родительского процесса (parent process ID)** указывает на родительский процесс.
- **Идентификатор группы процессов (process group ID).**

  Процессы могут объединяться в группы. Каждая группа обозначается идентификатором группы. Процесс, идентификатор которого совпадает с идентификатором группы, называется лидером группы.
- **Идентификатор сеанса (session ID).**

  Каждая группа процессов принадлежит к сеансу. Сеанс связывает процессы с управляющим терминалом. Когда пользователь входит в систему, все создаваемые им процессы будут принадлежать сеансу, связанному с его текущим терминалом.
- **Программное окружение (programm environment)** это просто набор строк, заканчивающихся нулевым символом. Строки называются переменными окружения и имеют следующий формат: имя переменной = значение переменной
- **Дескрипторы открытых файлов.**

  Дескриптор файла — некоторое число, которое используется для обращения к файлу.

  При запуске процесс наследует дескрипторы от родительского процесса.
- **Текущий рабочий каталог** это каталог от которого система производит разрешение относительных имен.
- **Текущий корневой каталог** это каталог от которого производится разрешение абсолютных имен. Процесс не имеет доступа к файлам находящимся выше корневого каталога.
- **Идентификаторы пользователя и группы.**

  С каждым процессом связаны действительные идентификаторы пользователя (real userlD) и группы (real group ID), совпадающие с соответствующими

идентификаторами пользователя, запустившего процесс. Кроме того, с процессом связаны эффективные идентификаторы пользователя (effective user ID) и группы, определяющие права процесса в системе. Обычно, действительные и эффективные идентификаторы совпадают.

- **Приоритет (nice).**

  Значение nice ("дружелюбность") показывает готовность процесса уступить свое процессорное время другим процессам. Чем больше значение nice, тем ниже приоритет процесса.

Команды для работы с процессами

`ps [-axewjlu] [-o формат] [-U пользователь] [-p pid]`

Выводит список и статус процессов работающих в системе. Без аргументов выводит список процессов текущего пользователя, подключенных к терминалу. Значения параметров следующие:

-a вывести информацию о процессах всех пользователей.

-x вывести информацию о процессах не подключенных к терминалу.

-e вывести значения переменных окружения процесса.

-w использовать строки длиной 132 символа. Если указан несколько раз, то строки не обрезаются совсем.

-j, -l, -и - меняют формат вывода информации.

-o *формат* вывести информацию в указанном формате.

-U *пользователь* вывести информацию о процессах указанного пользователя.

-p pid вывести информацию о процессе с указанным идентификатором.

Значение формата для параметра **`-o`** является списком из следующих ключевых слов разделенных запятыми (без пробелов):

Command командная строка и аргументы.

nice уровень nice (приоритет).

pgid идентификатор группы процессов.

pid идентификатор процесса.

ppid идентификатор родительского процесса.

rgid, ruid реальные идентификаторы группы и пользователя.

uid реальный идентификатор пользователя.

tty управляющий терминал

top выводит сведения о запущеных процессах в динамике. Прекратить работу команды можно нажав клавишу q

Задание 8.1. Изучение и анализ отображаемой информации о процессах

1. Из консоли пользователя командой ps -ef 1 | more выведите расширенный поэкранный список исполняемых процессов (перечень параметров для расширенного вывода информации можно уточнить с помощью электронного справочника man ps). Разберитесь с выводимой информацией. Определите процессы:

- по типу: системные, демоны, пользовательские (тип процесса определяется по косвенным признакам, в частности, по имени),
- по состоянию **S:** (исполняющиеся - **R или O,** ожидающие записи на диск - **D,** ожидающие событий - **s,** приостановленные - T, зомби - Z и т.д.),
- по текущему динамическому приоритету **PRI** (наименьшее значение у высокоприоритетных процессов),
- по относительному приоритету N**1**.

2. **B** консоли суперпользователя запустите утилиту **top** для текущего контроля процессов. Утилита позволяет отобразить наиболее активные процессы (столько, сколько их помещается на экран) с достаточно полной информацией о них (для пользователя утилита представляет ограниченный набор выводимых параметров).
3. Из первой консоли создайте процесс **od /dev/zero** > **/dev/null. B** соответствии с введенной командой утилита **od** читает и выводит непрерывный поток нулевых байт из «рога изобилия» в нулевое устройство. Переключившись в другую консоль, с помощью команды **top** просмотрите список наиболее активных процессов. Найдите и идентифицируйте запущенный процесс, найдите по идентификатору **PPID** его «родителя», определите его приоритет (возможно это - величина переменная), долю загрузки централь ного процессора **%CPU** и оперативной памяти **%MEM.**
4. Поочередно из первой и второй консолей с правами администратора и пользователя с помощью команды **od /dev/zero** > **/dev/null** & создайте по **2-3** одинаковых фоновых процесса.

5. По мере создания новых процессов отслеживайте в третьей консоли их текущий приоритет, загрузку процессора и памяти. Имеются ли различия в приоритете процессов, выполняемых от имени администратора и пользователя?
6. С консоли пользователя userl измените приоритет одного из принадлежащих ему процессов. Для этого воспользуйтесь командой renice -10 PID. Изменился ли относительный приоритет процесса?
7. Повторите предыдущий пункт с правами администратора.
8. Переключитесь в консоль пользователя и измените приоритет одного из принадлежащих ему процессов командой renice 5 PID. Произошло ли изменение приоритета?
9. Проконтролируйте из третьей консоли изменение приоритетов запущенных процессов.
10. Удалите созданные процессы командой kill.

Задание 8.2. Управление процессами

1. С правами пользователя создайте в своей директории сценарий с именем abed. Сценарий можно создать с помощью команды cat:

```
cat: >abcd
#!
/bin/bash
while : rem обратите внимание на пробел перед
двоеточием! do
echo HELLO!
done
Ctrl+d
```

2. Используя команду **chmod,** присвойте пользователю полные права на чтение, запись и исполнение данного сценария. Запустите сценарий на исполнение (на экран должны непрерывно выводиться приветствия HELLO!)
3. Перейдите в третью консоль, с помощью команды **top** просмотрите список процессов и найдите в нем «зависший» процесс, запущенный пользователем (на самом деле это только имитация зависания, которое пользователь легко может прекратить сам). Прочитайте идентификатор процесса PID.
4. Нажатием **Ctrl+C** из второй консоли остановите процесс. Как изменилось при этом состояние процесса?

5. Повторно запустите из второй консоли процесс, перейдите в первую консоль и отправьте "зависшему" процессу сигнал на останов (команда kill -15 PIDprocess).
6. Перейдите в другую консоль и отправьте «непослушному» процессу сигнал kill -20 РШ. Как реагирует процесс на данный сигнал? Посмотрите в электронном справочнике, что означает данный сигнал.
7. С помощью команды kill -9 PID отправьте этому процессу сигнал принудительного завершения. С другой консоли проконтролируйте выполнение команды. Остановился ли процесс? Остался ли он в списке процессов? Какая программа на самом деле перехватывает и исполняет команду kill -9 PID?
8. С помощью команды echo SPATH поочередно из консоли администратора и пользователя userl выведите список директорий, в которых производится поиск исполняемых файлов, заданных только по имени. В чем заключается различие выведенных списков? Почему в списке PATH администратора отсутствует текущий каталог (.)? Почему в списке PATH пользователя отсутствует каталог /sbin? Имеет ли пользователь возможность изменить порядок проверки каталогов для администратора?
9. Попробуйте запустить несколько утилит из второй консоли с правами пользователя (например, renice -10 PID, date -s O). Как реагирует система на ваши попытки?
10. Перейдите в административную консоль и повторите запуск утилит с правами суперпользователя.
11. Убедитесь в том, что пользователю разрешен запуск указанных утилит. Объясните, почему пользователь не может запустить утилиты с некоторыми «критичными» параметрами? Где, по вашему мнению, расположен механизм контроля за ходом запуска (в ядре операционной системы, в командной оболочке, в самой утилите?). Ответ обоснуйте.
12. С правами пользователя скопируйте в свой рабочий каталог один из исполняемых файлов с параметром SUID каталога /bin (исполняемые файлы вы делены цветом и символом *, а параметр SUID отмечен символом «s» в правах владельца на исполнение). Как изменились права доступа к файлу после его копирования?
13. Из второй консоли с правами пользователя скопируйте в свой домашний каталог утилиту, которую разрешено запускать только администратору (например, **chattr).** Копирование производите с параметрами, гарантирующими переход копии во

владение пользователю. Убедитесь, что пользователь имеет на скопированный файл все необходимые права. Попробуйте использовать свою копию утилиты по ее назначению (в случае копирования утилиты chattr установите дополнительный атрибут +i одному из своих файлов). Сделайте выводы.

Модуль 9. Файловая система Astra Linux.

Файловая система — это структура, с помощью которой ядро операционной системы организует и представляет пользователям ресурсы памяти системы. Сюда относится память на различного рода носителях информации. Емкость и количество носителей различно в разных системах. Ядро объединяет эти ресурсы в единую иерархическую структуру, которая начинается в каталоге / и разветвляется, охватывая произвольное число подкаталогов. Цепочка имен каталогов, через которые необходимо пройти для доступа к заданному файлу, вместе с именем этого файла называется путевым именем файла (pathname). Путевые имена могут быть полными или относительными. В любой момент каждый процесс привязан к некоторому текущему каталогу. Относительные имена интерпретируются с текущего каталога.

Файловое дерево может быть произвольного размера. Однако существуют определенные ограничения зависящие от конкретной операционной системы. Как правило имя каталога не должно содержать более 256 символов, а в определении одного пути не должно быть более 1023 символов.

В ОС Linux существует восемь типов файлов:

Обычный файл— это просто последовательность байтов. Обычный файл может содержать выполняемую программу, главу книги, графическое изображение и т.п.

Каталоги -могут содержать файлы любых типов в любых сочетаниях. Специальные имена . и ..обозначают соответственно сам каталог и его родительский каталог.

Файлы устройств - позволяют программам взаимодействовать с аппаратными средствами и периферийными устройствами системы. При конфигурировании ядра к нему добавляются те модули, которые знают, как взаимодействовать с каждым из устройств системы. За всю работупо управлению конкретным устройством отвечает специальная программа, называемая драйвером устройства.

Драйверы устройств образуют стандартный коммуникационный интерфейс, который выглядит как обычный файл. Когда ядро получает запрос к байт-ориентированному или блок-ориентированному файлу устройства, оно просто передает этот запрос соответствующему драйверу устройства.

Каждому типу устройств системы может соответствовать несколько файлов устройств.

Поэтому файлы устройств характеризуются двумя номерами: старшим и млад- шим.Старший определяет драйвер, а младший конкретное устройство.

Доменные гнезда (sokets) Linux — это соединения между процессами, которые позволяют им взаимодействовать, не подвергаясь влиянию других процессов. Доменные гнезда Linux

локальны для конкретного хост-компьютера. Обращение к ним осуществляется через объект файловой системы, а не через сетевой порт.

Именованные каналы - также как и доменные гнезда обеспечивают взаимодействие двух несвязанных процессов, выполняемых на одной машине.

Жесткие ссылки — это скорее не тип файла, а его дополнительное имя. У каждого файла имеется как минимум одна ссылка. Как правило, это имя, под которым он был создан. Добавлением

ссылки создается псевдоним файла. Ссылку невозможно отличить от имени файла, к которому она присоединена: в ОС Linux они идентичны. Linux подсчитывает количество ссылок, указывающих на каждый файл, и не освобождает блоки данных файла до тех пор, пока не удалит его последнюю ссылку.

Символические ссылки— обеспечивают возможность указывать вместо путевого имени файла имя ссылки. Символическая ссылка содержит путевое имя файла, на который она ссылается. Имена файлов могут состоять из любых символов, за исключением слэша и символа с кодом ноль. Максимальная длина имени файла определяется конкретной системой. Для каждого файла определен владелец этого файла и группа владелец данного файла. Для каждого файла определяются права доступа владельца файла, группы, всех остальных. Есть три типа прав доступа: чтение, запись, выполнение/поиск. Изменить права доступа к файлу может только владелец и суперпользователь (root).

Для того, чтобы создать символическую ссылку, используется уже упоминавшаяся команда In с дополнительной опцией -s:

```
# In -s имя_файла_или_каталога имя__символической_ссылки
```

Аналогично создается жесткая ссылка, используется команда In без дополнительных опций:

```
# In имя_файла_или_каталога имя_жесткой_ссылки
```

Для Linux существует огромное количество файловых систем. Рассмотим основные файловые системы, используемые в Astra Linux.

ext2 (second extended file system) — была разработана Remy Card в 1993 году. Не журналируемая файловая система, это был основной её недостаток, который исправит ext3. ext3 (third extended filesystem) — по сути расширение исконной для Linux ext2, способное к журналированию. Разработана Стивеном Твиди (Stephen Tweedie) в 1999 году, включена в основное ядро Linux в ноябре 2001 года. На фоне других своих сослуживцев обладает более скромным размером пространства, до 4 тебибайт (4*240 байт) для 32-х разрядных систем. На данный момент является наиболее стабильной и поддерживаемой файловой системой в среде Linux.

ext4 — попытка создать 64-х битную ext3 способную поддерживать больший размер файловой системы (1 эксбибайт). Позже добавились возможности — непрерывные области дискового пространства, задержка выделения пространства, онлайн дефрагментация и прочие. Обеспечивается прямая совместимость с системой ext3 и ограниченная обратная совместимость при недоступной способности к непрерывным областям дискового пространства.

Информация о разделах хранится на диске в таблице разделов. В этой таблице содержится информация о начале и конце каждого раздела, о его типе, а также о том, является ли раздел загрузочным или нет. Чтобы создать или удалить раздел, необходимо отредактировать таблицу разделов с помощью специальной программы.

Команда fdisk с опцией **-1** выводит на экран список разделов. Если вам требуется просмотреть все разделы на определенном диске, то добавьте имя устройства, например, /dev/ sda.

```
# fdisk -1 /dev/sda
```

Эта команда также содержит систему меню для редактирования таблицы разделов (т. е. для создания или удаления разделов). Для запуска команды fdisk в интерактивном режиме просто укажите в качестве параметра имя диска, например /dev/hda или /dev/sdb.

```
# fdisk /dev/sda
```

После ввода этой команды, вы попадете в подменю команды fdisk. Чтобы вывести список доступных односимвольных команд, наберите **т.** Для вывода информации о разделах выбранного диска используйте команду р.

Создание нового раздела осуществляется командой п.

```
Command (m for help): n Command action
   1  logical (5 or over)
   p  primary partition (1-4)
P
Selected partition 4
First cylinder (9112-121601, default 9112):
Using default value 9112

Last cylinder, +cylinders or +size{K,M,G) (9112-9633, default 9633): +521 Command (m

for help): p
Disk /dev/sda: 1000.2 GB, 1000204886016 bytes 255
heads, 63 sectors/track, 121601 cylinders Units
= cylinders of 16065 * 512 = 8225280 bytes
```

```
Disk identifier: 0x000de20f
   DeviceBoot       Start          End       Blocks   Id  System

/dev/sdal   ★            1         9111     73184076     7 HPFS/NTFS
/dev/sda2             9634         9730        779152+83  Linux

/dev/sda3             9731       116679     859067842+   5 Extended


/dev/sda4             9112         9633      4192965   83  Linux

/dev/sda5             9731        20917     89859546   83  Linux

/dev/sda6            20918        39644    150424596   83  Linux

/dev/sda7            39645        53905    114551451   83  Linux
Partitiontable entries are  not in disk  order

Command (m for   help) :
```

Берем значение по умолчанию для первого цилиндра и указываем значение +521 (количество цилиндров). Из листинга видно, что размер нашего раздела составляет приблизительно 4 ГБ. Поскольку это основной раздел, он должен иметь номер от 1 до 4.

Новый раздел имеет тип 83, т. е. является Linux-разделом для хранения данных. Тип раздела можно рассматривать как метку, которая указывает операционной системе на его предполагаемое назначение. Также при вводе подкоманды п для создания нового раздела можно было выбрать один из двух вариантов: Т для создания логического раздела и 'р' — для создания основного. Это обусловлено тем, что в списке присутствуют только оставшиеся доступные типы разделов. Если бы диск еще не имел расширенного раздела, вы могли бы увидеть в списке опцию 'e', предназначенную для создания раздела этого типа. Также обратите внимание на то, что наш расширенный раздел (/dev/sda3) имеет тип 5.

Все что мы делали до этого шага - просто вносили изменения в таблицу разделов в оперативной памяти. На данном этапе в любой момент можно завершить работу без сохранения изменений с помощью команды q. Если необходимо что-то изменить, вы можете выполнить команду d для удаления одного или нескольких разделов, а затем создать их заново. Если конфигурация устраивает, то выполните команду v для контрольной проверки конфигурации, а затем команду w для записи изменений в таблицу разделов и выхода из программы.

Для создания файловых систем в Linux используется команда mkfs. Фактически команда mkfs представляет собой внешний интерфейс к нескольким командам для работы с файловой системой, таким как mkfs.ext3 для ext3 и mkfs.

Давайте отформатируем раздел **/dev/sda4** в файловой системе ext3 с помощью команды **mkfs**

```
# mkfs -t ext.3 /dev/sda4
```

В момент создания, для файловой системы ext3 был создан журнал. Если вы хотите добавить журнал к существующей файловой системе ext2, то используйте для этого команду tune2fs с опцией -j.

Полезная опция, используемая для файловых систем ext2 и ext3 - это опция -L с именем, которое будет присвоено разделу в качестве метки. Эту метку можно использовать вместо имени устройства при монтировании файловой системы; при внесении определенных изменений, которые также необходимо отражать в управляющих файлах, метка позволяет изолировать имя устройства, которому она фактически соответствует. Для просмотра существующей метки файловой системы ext2 или ext3 или создания новой используется команда **e2iabei.** Длина метки ограничена 16 символами.

Одним из последних нововведений является универсальный уникальный идентификатор (Universally Unique Identifier, UUID), который используется вместо метки. UUID — это 128-битовый идентификатор, обычно отображаемый в виде 32 шестнадцатеричных цифр, разделенных дефисами. Для большинства файловых систем Linux UUID генерируется автоматически при их форматировании. Для просмотра UUID раздела, который мы только что отформатировали, используйте команду bikid (ее можно запустить без прав суперпользователя), как показано в листинге 13. Идентификаторы UUID обладают большей уникальностью по сравнению с метками и особенно полезны при использовании устройств с горячей заменой, например, USB -накопителей.

Просмотреть идентификатор UUID можно с помощью bikid

```
#    bikid /dev/sda4
```

По окончанию процедуры форматирования, примонтируйте новый диск, предварительно создав соответствующую директорию (точку монтирования):

```
#    mkdir /newdisk
#    mount: /dev/sda4 /newdisk
```

Проверьте, подмонтировался ли диск:

```
# mount | grep sda4
```

Упражнение 9.1. Символические и жесткие ссылки

1. Скопируйте файл passwd из каталога /etc в домашнюю директорию пользователя student
2. Создайте символическую ссыжу symlink на скопированный файл passwd
3. Создайте две жестких ссылки hardlink и hardlinkl на скопированный файл passwd
4. Проверьте что ссылки работают

5. Удалите оригинал /home/student/passwd
6. Что изменилось?

7. Посмотрите номер инода оставшихся жестких ссылок
8. Как найти все жесткие ссылки, если они находятся в разных директориях и вы не знаете в каких. Покажите это на примере ваших жестких ссылок

Упражнение 9.2. Файловая система Astra Linux. Добавление и форматирование нового диска.

1. Используйте fdisk **-1**, чтобы посмотреть информации о дисках и разделах на вашем компьютере
2. Какие из ваших дисков в системе имеют разметку?

3. Посмотрите свободное место на этих разделах с помощью команды df
4. Добавьте 2 новых жестких диска в в виртуальную машину (SCSI и ГОЕ). Для добавления новых дисков виртуальную машину нужно выключить. После добавления новых дисков загрузите Astra Linux.
5. После загрузки проверьте что новые диски появились в системе
6. Например, один из наших новых дисков определился в системе как /dev/sdc. Используя команду fdisk, создайте два новых раздела (/dev/sdc 1 и /dev/sdc2) на данном диске, размером 512 М.
7. Перезагрузитесь, чтобы удостовериться, что вы корректно изменили разделы на жестком диске.
8. Используя команду **mke2fs** или команду **mkfs.ext2,** создайте новую файловую систему **ext2** на новом логическом разделе **/dev/sdcl.**
9. На разделе **/dev/sdc2** создайте файловую систему **ext4**
10. Создайте директорию **/data,** в которую вы будете монтировать новый логический том **/dev/sdc2.**
11. Используйте команду **mount,** чтобы смонтировать новый логический том в директорию **/data.**
12. Скопируйте **/etc/passwd** в директорию **/data** и проверьте, что копирование было успешно.
13. Затем размонтируйте директорию:
14. Еще раз проверьте содержимое каталога /data

15. Добавьте метку /data к новому разделу с помощью команды e2label:
16. Отредактируйте файл /etc/fstab так, чтобы новый раздел монтировался при загрузке системы
17. Проверьте, что вы правильно прописали данную строку в файл fstab
18. Перезагрузите систему и убедитесь что новый раздел монтируется автоматически
19. Посмотрите свойства файловой системы для раздела /dev/sdcl. Включено ли журналирование данной файловой системы?
20. Включите журналирование для раздела /dev/sdcl
21. Создайте для данного раздела точку монтирования /datal и смонтируйте туда этот слайс
22. Перейдите в раздел /datal и попробуйте размонтировать его
23. Почему вы не можете размонтировать данный раздел? ....,
24. Посмотрите, кто из пользователей и какими процессами занял раздел /datal
25. Завершите все процессы на разделе /datal
26. Попробуйте размонтировать раздел /datal снова. Что произошло?
27. Увеличьте резервируемое место (minfree) файловой системой до 10% на разделе **/dev/sdcl**
28. Осуществите проверку файловой системы для раздела **/dev/sdcl**

Linux Volume Manager (LVM) - это мощная система управления томами с данными для Linux. Она позволяет создавать поверх физических разделов (или жестких дисков) логические тома, которые в самой системе будут видны как обычные блочные устройства с данными - разделы.

Основные преимущества LVM:

- одну группу логических томов можно создавать поверх любого количества физических разделов;
- размер логических томов можно легко менять прямо во время работы;
- LVM поддерживает механизм снапшотов, копирование разделов «на лету» и зеркалирование, подобное RAID-1.

Создание и удаление. LVM строится на основе разделов жёсткого диска и/или целых жёстких дисков. На каждом из дисков/разделов должен быть создан **физический том** (physical volume). Возьмем для LVM диск sda и раздел sdb2:

```
# pvcreate /dev/sda
# pvcreate /dev/sdb2
```

На этих физических томах создаём **группу томов,** назовем ее vgl:

```
# vgcreate -s 32M vgl /dev/sda /dev/sdb2
```

Посмотрим информацию о нашей группе томов:

```
# vgdisplay vgl
```

Групп можно создать несколько, каждую со своим набором томов. Теперь в группе томов можно создать **логические тома** lvl и lv2 размером 20 Гбайт и 30 Гбайт соответственно:

```
# lvcreate -n lvl -L 20G vgl
# lvcreate -n lv2 -L 30G vgl
```

У нас появились блочные устройства /dev/vgl/lvl и /dev/vgl/lv2. Осталось создать на них файловую систему:

```
# mkfs.ext4 /dev/vgl/lvl
# mkfs.reiserfs /dev/vgl/lv2
```

Удаление LVM (или отдельных его частей: логических томов или групп томов) происходит в обратном порядке - сначала нужно отмонтировать разделы, затем удалить логические тома (lvremove), затем можно удалить группы томов (vgremove) и ненужные физические тома (pvremove).

Добавление физических томов. Чтобы добавить новый жесткий диск sdc в группу томов, создадим физический том:

```
# pvcreate /dev/sdc
```

И добавим его в нашу группу:

```
# vgextend vgl /dev/sdc
```

После этого появляется возможность создать ещё один логический диск Overeat*) или увеличить размер существующего (lvresize).

Удаление физических томов Чтобы убрать из работающей группы томов винчестер «da сначала перенесём

все данные с него на другие диски:

```
# pvmove /dev/sda
```

Затем удалим его из группы томов:

```
# vgreduce vg /dev/sda
```
Теперь

можно удалить физический том:

```
# pvremove /dev/sda
```

Последняя команда просто убирает отметку о том, что диск является членом Ivm,. После удаления из LVM для дальнейшего использования диск необходимо переразбить.

Изменение размеров. LVM позволяет легко изменять размер логических томов. Для этого нужно сначала изменить сам логический том:

```
# lvresize -L 40G
```
vgl/lv2 а затем файловую систему

на нём:

```
# res±ze2fs /dev/vgl/lv2
# resize_reiserfs /dev/vgl/lv2
```

Упражнение 10. Файловая система Astra Linux. LVM

1. Для данного упражнения нужно установить пакет lvm2 (если он у вас не установлен)
2. Создайте три новых раздела на диске по 512 М. При необходимости добавьте в систему новый диск (если вы работаете на виртуальной машине). Предположим, что в нашем случае мы добавили новый диск и он был распознан системой как /dev/sda (соответственно, новые разделы будут /dev/sdal, /dev/sda2 и /dev/ sda3)

   Разделы на диске должны иметь тип LVM
3. LVM строится на основе разделов жёсткого диска и/или целых жёстких дисков. Поэтому на первых двух из созданных нами разделов (/dev/sdal, /dev/sda**2**) создайте физический том (physical volume).
4. Проверьте, что физические тома созданы корректно
5. На первых двух физических томах (/dev/sdal и /dev/sda**2**) создаём группу томов, которая будет называться, скажем, vgl
6. Проверьте, что группа vgl создана корректно

Групп можно создать несколько, каждая со своим набором томов. Но обычно это не требуется.

7. Теперь в группе томов создадите логический том lvl размером 200 Мб и lv**2** размером 300 Мб
8. Проверьте, что том создан корректно
9. Теперь у нас есть блочные устройства /dev/vgl/lvl и /dev/vgl/lv**2** . Создайте на них файловую систему: на lvl - ext4 и на lv2 - ext3

   Удаление LVM (или отдельных его частей, например, логических томов или групп томов) происходит в обратном порядке - сначала нужно отмонтировать разделы, затем удалить логические тома (lvremove), после этого можно удалить группы томов (vgremove) и ненужные физические тома (pvremove). На данном шаге удалять ничего не нужно!
10. Смонтируйте созданные логические тома lvl и lv**2** в директории /lvdirl и /lvdir**2** (предварительно их создав)
11. Посмотрите информацию по смонтированным томам
12. Выполните настройки для автоматического монтирования данных томов во время запуска системы. Для этого внесите изменения в файл /etc/f stab
13. Проверьте корректность правок в файле /etc/f stab без перезагрузки
14. Чтобы добавить раздел /dev/sda3 в группу томов, создайте физический том для этого раздела
15. Теперь добавьте его в нашу группу
16. Теперь можем увеличим размер существующего тома lv**2** за счет физического тома, добавленного в группу. Для этого размонтируйте том lv**2** и проверьте его на наличие ошибок
17. Увеличьте lv**2** на 512M. После увеличения выполните проверку файловой системы на наличие ошибок
18. Подмонтируйте измененный том и проверьте изменился ли его размер
19. Чтобы убрать из работающей группы томов раздел /dev/sda**2** сначала перенесите все данные с него на другие диски
20. Затем удалите его из группы томов
21. После этого удалите физический том

Рассмотрим основные утилиты для работы с пакетами в Astra Linux.

Dpkg - пакетный менеджер для Debian систем. Он может устанавливать, удалять и создавать пакеты, но, в отличие от других систем управления пакетами, не может автоматически загружать и устанавливать пакеты или их зависимости.

Для получения списка пакетов, установленных в системе, в терминале наберите:

```
# dpkg -1
```

В зависимости от количества пакетов в вашей системе, вы можете получить большой объем вывода. Поэтому очень удобно использовать совместно с командой grep, чтобы определить установлен ли определенный пакет:

```
# dpkg -1 | grep apache2
```

Для получения списка файлов, установленных пакетом, введите:

```
# dpkg -L ufw
```

Если вы не уверены из какого пакета установлен файл, используйте ключ -S:

```
# dpkg -S /etc/host.conf
 base-files: /etc/host.conf
```

В выводе видим, что /etc/host.conf принадлежит пакету base-files. Вы можете установить локальный .deb файл, набрав:

```
# dpkg -i zip_3.0-4_i386.deb
```

Удаление пакета выполненяется с использованием ключа -r:

```
# dpkg -r zip
```

APT — интерфейс к dpkg в Debian. APT обладает рядом уникальных возможностей, таких как установка пакетов в нужном порядке и использование нескольких источников- хранилищ пакетов для установки и обновления. Программа **apt-get** использует собственную базу данных для слежения за установленными, не установленными и доступными для установки пакетами.

Скачанные файлы программ не удаляются после установки и хранятся в каталоге **/var/cache/apt/archives.** При необходимости их можно удалить командой **apt-get clean** или **apt-get autoclean.**

Основные команды apt-get при работе с пакетами.

apt-cdrom add	Добавление установочного диска в репозиторий
apt-get update	обновление информации о пакетах из репозитариев. Список репозиториев находится в файле **/etc/apt/sources. list**
apt-get upgrade	Обновляет уже установленные программы.

apt-get dist-upgrade	обновление системы в целом
apt-get clean	очистка локального хранилища, кроме файлов кэша
apt-get autoclean	очистка локального хранилища, с удалением кэш-файлов
apt-get check	обновляет кэш и проверяет неудовлетворенные зависимости
apt-get autoremove	удаление ранее скачаных, но ненужных пакетов
apt-get remove	удаление пакета с сохранением конфигурационных файлов
программа	
apt-get purge *программа*	удаление пакета со всеми зависимостями
apt-get install	Установка программы. Если устанавливается сразу несколько
программа	пакетов, то они зазделяются пробелами.
apt-cdrom install	Устанавливает либо обновляет пакет с cdrom
программа	
apt-get -f install	Исправляет найденные поломанные зависимости.
apt-get build-dep	устанавливает все для сборки исходных пакетов
apt-get source	скачивает пакеты с исходниками
apt-cache search	Поиск программы в кэше.
программа	
apt-cache show *программа*	Показывает информацию и описание пакета программы.
apt-cache showpkg	Покажет зависимости пакета программы.
программа	
	Параметры:
-h, —help	справка
-q, —quiet	скрыть индикатор процесса

qq — не показывать ничего кроме ошибок

d, —download-only — только получить пакеты и выйти

-s, —simulate — выполнить симуляцию событий

-y, —yes — автоматически отвечать "Да" на все вопросы

—reinstall — переустановить пакеты

-f, —fix-broken — исправить нарушенные зависимости

-т, —ignore-missing — игнорировать отсутствующие пакеты

-и, —show-upgraded — показать обновленные пакеты

—no-upgrade — не производить обновления пакетов

-Ъ, —compile, —build — собрать пакет после получения

-D — при удалении, удалить зависящие компоненты

-V — подробно показывать номера версий пакетов

—no-remove — если пакеты отмеч. к удален., то apt-get выкл.

—force-yes
принудительное выполнение заданной операции

Aptitude является интерактивной программой, которая используется в консоли в полуграфическом режиме. Вы можете пролистать список установленных и доступных пакетов, найти всю необходимую информацию и выбрать пакеты для установки или удаления. Эта программа предназначена специально для администраторов, поэтому её действия по умолчанию более интеллектуальны, чем у **apt-get,** и её интерфейс более понятен.

Сразу после запуска **aptitude** показывает список пакетов, отсортированный по их состоянию (установленные, не установленные или установленные, но недоступные на зеркалах - прочие секции отображают задачи, виртуальные пакеты и новые пакеты, которые недавно появились на зеркалах). Для упрощения тематической навигации существуют и другие режимы просмотра. В любом случае **aptitude** выводит на экран комбинированный список категорий и пакетов. Категории организованы в виде древовидной структуры, ветви которой можно развернуть или закрыть клавишами **Enter,**

[или]. Клавишу + следует использовать для отметки пакета на установку, - для отметки на удаление и _ для полной очистки (следует отметить, что эти же клавиши можно использовать и для категорий, в случае чего выбранное действие будет применено ко всем пакетам в данной категории). Нажатие и обновляет список доступных пакетов и **Shif t+u** подготавливает общее обновление системы.

Synaptic - это графический менеджер пакетов, имеющий простой и эффективный графический интерфейс, основанный на GTK+/GNOME. Множество готовых фильтров, содержащихся в нём, предоставляют быстрый доступ к новым пакетам; установленным пакетам; пакетам, для которых доступно обновление; устаревшим пакетам и так далее. При навигации по этим спискам пакетов вы можете выбрать действие, которое требуется совершить с данными пакетами (установка, обновление, удаление, полная очистка); эти операции не выполняются немедленно, а заносятся в список задач. Простое нажатие одной кнопки подтверждает операции и они все выполняются за один раз.

Упражнение 11.1. Набор команд dp kg

1. Посмотрите краткую помощь по утилите **dpkg**
2. Посмотрите все установленные пакеты в системе. Сколько всего пакетов установлено?
3. Установлен ли на вашем компьютере Midnight Commander?
4. Какие файлы входят в пакет Midnight Commander?

5. посмотрите к какому пакету относится файл tc?
6. Выведите полную информацию по пакету tc
7. Подмонтируйте установочный диск с Astra Linux (если не смонтирован)
8. Посмотрите информацию по пакету smc-users с помощью утилиты dpkg
9. Установите пакет smc-users (или любой другой) с помощью утилиты dpkg

   В случае, если неудовлетворенные зависимости пакета отсутствуют, он будет установлен. В случае нарушения зависимостей dpkg выдаст сообщение об ошибке, в котором будут перечислены все необходимые пакеты, которые следует установить, чтобы разрешить зависимости.

   Для удаления ненужного пакета можно использовать ключ -г

   Для удаления пакета и очистки системы от всех его компонентов (в случае, если пакет не связан зависимостями с другими установленными пакетами) следует использовать ключ **-p**

Упражнение 11.2. Программа apt-get

1. Посмотрите краткую помощь по утилите **dpkg**
2. Чтобы обновить базу данных необходимо использовать команду **apt-get update.** Эта команда просматривает списки пакетов в архивах, указанных в файле **/etc/apt/sources.list.**
3. Найдите доступные для установки пакеты: **mutt, nano** или какой-либо другой на ваш выбор
4. Установите пакет **nano, mutt** или любой другой на ваш выбор с помощью утилиты **apt-get**
5. Удалите установленный ранее пакет
6. Изучите дополнительные ключи apt-get:
7. Запустите менеджер **synaptic** из командной строки и попробуйте с помощью него установить/удалить пакеты, с которыми работали ранее

Модуль 12. Процесс запуска и инициализации системы.

Загрузчик GRUB.

GRUB (англ. GRand Unified Bootloader) — загрузчик операционной системы. GRUB позволяет пользователю иметь несколько установленных операционных систем и при включении компьютера выбирать одну из них для загрузки.

После настройки GRUB пользователь при включении компьютера видит список операционных систем, которые установлены на его компьютер и которые можно загрузить, выбрав подходящую и нажав Enter.

GRUB 2 — следующая версия GRUB. Разработчики писали GRUB 2 «с нуля», чтобы добиться переносимости и модульности. В связи с существованием GRUB 2 разработка GRUB 1 прекращена.

Если сравнивать GRUB2 со старым GRUB самое большое различие - измененная структура файлов конфигурации. Настройка GRUB2 производится принципиально иным способом. В GRUB2 отсутствует "/boot/grub/menu.lst", в котором были прописаны пункты загрузки, таймаут меню и прочие настройки. Где это теперь? В grub2 основным файлом конфигурации является "/boot/grub/grub.cfg".

"grub.cfg" генерируется автоматически с использованием нескольких скриптов, поэтому после следующего обновления grub2 ваш "grub.cfg" будет создан заново Главным файлом, отвечающим за загрузочное меню, является "/boot/grub/grub.cfg"; кроме него имеются файл "/etc/default/grub" и директория "/etc/grub.d".

Упражнение 12.1. Загрузка ОС Astra Linux

1. Чем отличается однопользовательский режим от многопользовательского?

2. Какой уровень загрузки по умолчанию используется в Astra Linux?
3. Настройте систему так чтобы уровень загрузки по умолчанию был 3
4. Перезагрузите систему используя команду init.
5. Во время загрузки, в меню GRUB введите пароль и выберите загрузку в однопользовательском режиме. Назовите основные отличия данного вида загрузки

Упражнение 12.2. GRUB

Отключите графическую заставку и "молчаливый режим". Замените

```
GRUB_CMDLINE_LINUX_DEFAULT="quiet splash"
```

на

GRUB_CMDLINE_LINUX_DEFAULT=""

Теперь все наши пункты загрузки linux-ядер будут преобразованы в verbose mode, то есть режим загрузки без графической заставки и с выводом на экран текстовой информации о загрузке компонентов системы, что позволяет следить за процессом загрузки и выявлять неполадки.

Обратите внимание, чтобы ядру не передавались лишние параметры через GRUB_CMDLINE_LINUX, если он не пустой. Все вышеописанное работает для

GRUB_CMDLINE_LINUX="|

Упражнение 12.3. Установка пароля на GRUB

1. Сделайте бэкап конфигурационного файла **grub. cf g**
2. В **терминале от пользователя root создайте пароль для GRUB**

   # grub-mkpasswd-pbkdf2

В результате выполнения данной команды получаем хеш пароля, который нужен, чтобы GRUB при запуске распознавал наш пароль.

3. Полученный хеш необходимо добавить в скрипт, на основе которого будет генерироваться конфигурация GRUB2. Чтобы не возникло проблем с конфигурационным файлом, удобнее всего создать отдельный файл **/etc/grub. d/05_password**

   # touch /etc/grub.d/05_password #!/bin/sh

   set -e

   ```
   cat « EOF
   set superusers-"root"
   password_pbkdf2 root grub.pbkdf2.sha512.10QQ0.E7...* 4
   EOF
   ```

4. Устанавливаем необходимые права доступа на наш файл:

   # chown root:root /etc/grub.d/05_password

   # chmod 600 /etc/grub.d/05_password

   # update-grub

Модуль 13. Базовая диагностика системы, файлы журналов

Для работы с подсистемой протоколированием имеется ряд графических утилит, которые могут быть использованы для настройки параметров регистрации событий и просмотра протоколов:

- **fly-admin-smc** («Управление политикой безопасности») — управление протоколированием, привилегиями и мандатными атрибутами пользователей, работа с пользователями и группами;
- fly-admin-view audit («Журнал безопасности») — выборочный просмотр протоколов аудита.

Более подробное описание утилит см. в электронной справке.

Далее рассмотрены средства управления протоколированием в режиме командной строки.

Rsyslog отвечает за методы сбора и анализа файлов журналов. С помощью rsyslog можно, к примеру, организовать централизованный сбор и удобное представление логов с множества серверов и других устройств.

Сбор журналов на одном сервере дает ряд преимуществ:

- Отпадает необходимость ежедневно просматривать файлы журналов на каждом сервере (особенно актуально при большом количестве серверов). Вся информация находится в одной контрольной точке.
- Запись журналов идет и на локальную машину и на лог-сервер. В случае полного разрушения системы хранения данных сервера или в случае взлома, мы не теряем информацию, т.к. у нас есть копия на лог-сервере.

`B` старых версиях ОС Linux в качестве сервера журналов используется `syslog`. Задача отправки логов на удаленный сервер и задача сбора логов (функция лог-сервера) легко реализуема возможностями старого и надежного `syslogd` демона. Запуск с ключом -г заставляет службу прослушивать 514 порт UDP протокола и принимать сообщения от других машин (клиентов). Отправка же логов на удаленный сервер осуществляется, в простейшем случае, одной строкой в конфигурационном файле `/etc/syslog.conf` и перезапуском демона журналирования. Конфигруация `syslog` позволяет отфильтровать сообщения по полю facility.

Однако собрав все журналы в одном месте, становится очень трудно их разбирать. Здесь начинаются преимущества `rsyslog`.

Rsyslog позицианирует себя как расширеный модуль **syslogd** для систем Linux и Unix, обладающий продвинутой многопоточностью и акцентированный на безопасности и надежности.

Основные функции **RSyslog:**

- буферизация на жестком диске;
- **удаленная передача журналов на другой сервер (используя TCP, SSL, TLS и RELP);**
- запись журналов сразу в базу данных, а не только в файл (MySQL, Posrgres SQL, Oracle и множество других, через подключаемые расширения);
- **нотификация используя email** (smtp);
- полностью конфигурируемый и изменяемый формат сообщений (включая высокоточные временные метки);
- расширенные возможности фильтровки сообщений, дополняющие типичные facility.serverity (основанные на регулярных выражениях);
- возможность сжатия сообщений «на лету»;
- возможность чтения журналов с преобразованием к формату syslog;
- возможность легкого перехода с syslogd, благодаря совместимости конфигурацинных файлов.

Упражнение 13. Базовая диагностика системы, файлы журналов

Щ Как настроить (настройте) rsyslog, чтобы отсылать все лог-файлы на удаленный сервер без сохранения локально

2. Подкрректируйте настройи так, чтобы rsyslog оставлял копию логов на данном компьютере.
3. Не забудьте перезапустить rsyslog, чтобы изменения вступили в силу
4. Насколько безопасен обмен лог-файлами по сети между компьютерами/серверами?

Модуль 14. Управление процессами. Автоматизация заданий.

cron — это демон, отвечающий за запуск запланированных и повторяющихся команд (каждый день, каждую неделю и т. д.); atd — демон, работающий **с** командами, которые должны запускаться однократно, но в конкретный момент времени в будущем.

В системах Linux многие задачи планируются для регулярного запуска:

- ротация журналов;
- обновление базы данных для программы locate;
- резервное копирование;
- сценарии обслуживания (такие как удаление временных файлов).

По умолчанию пользователи могут планировать запуск задач. У пользователя есть свой собственный crontab. Его можно отредактировать, запустив **crontab -e** (его содержимое хранится в файле /var/spool/cron/crontabs/пользователь).

Доступ к cron можно ограничить, создав файл авторизации (белый список) в **/etc/cron, allow,** где указываются только пользователи, которым разрешено планировать задачи. Все остальные автоматически будут лишены такой возможности. Наоборот, чтобы заблокировать только одного-двух бедокуров, можно вписать их имена в файл блокировки (чёрный список), **/etc/cron.deny.** Такая же возможность есть и для atd, соответствующие файлы называются **/etc/at. allow** и **/etc/at. deny.**

У пользователя root есть свой собственный crontab, но он также может использовать файл /etc/crontab или создавать дополнительные файлы crontab в каталоге **/etc/cron.d. У** последних двух решений есть то преимущество, что можно указать пользователя, от имени которого запускается команда.

Пакет cron по умолчанию содержит некоторые запланированные задачи, которые выло ляются:

- программы из каталога **/etc/cron. hourly/** — каждый час;
- программы из каталога /**etc/cron. daily/** — каждый день;
- программы из каталога **/etc/cron. weekly/** — каждую неделю;
- программы из каталога **/etc/cron. monthly/** — раз в месяц.

Каждая значащая строка crontab описывает запланированную команду в следующих полях:

- значение минуты (число от 0 до 59);
- значение часа (от 0 до 23);
- значение числа месяца (от **1** до 31);
- значение месяца (от **1** до **12**);

- значение дня недели (от 0 до 7, где 1 соответствует понедельнику, а воскресенье может быть представлено как 0 или 7; также можно использовать первые три буквы англоязычного названия дня недели, например Sun, Mon и т. д.);
- имя пользователя, от имени которого должна выполняться команда (в файле /etc/crontab и в фрагментах, расположенных в /etc/cron. d/, но не в пользовательских файлах crontab);
- команда, которая должна быть запущена.

Все эти подробности описаны на странице man crontab(5).

Каждое значение может быть представлено в виде списка возможных значений (разделённого запятыми). Синтаксис a-b описывает интервал всех значений между а и Ь. Синтаксис a-b/c описывает интервал с промежутками с (пример: 0-10/2 означает 0,2,4,6,8,10). Звёздочка * является шаблоном, означающим все допустимые значения.

Пример файла crontab:

```
#Формат
#мин час чис мес день команда
# Загружать данные каждую ночь в 7:25
  25  19   *   *   *    $HOME/bin/get.pl
# 8:00 утра, по будням (с понедельника по пятницу)
  00  08   *   *   1-5  $HOME/bin/dosomething
# Перезапуск IRC-прокси после каждой перезагрузки Sreboot /usr/bin/dircproxy
```

Использование команды at. at запускает команду в заданный момент времени в будущем. Она принимает желаемые время и дату как параметры командной строки, а команду, которую необходимо запустить, через стандартный ввод. Команда будет запущена, как если бы она была введена в текущей оболочке, at беспокоится даже о том, чтобы сохранить текущее окружение, чтобы воспроизвести те же условия при вызове команды. Время указывается в соответствии со следующими соглашениями: 16:12 или 4:12рш означают 4:12 после полудня. Дата может быть задана в разных европейских и западных форматах, в том числе ДД.ММ.ГГ (27.07.12 означает 27 июля 2012 года), ГГГГ-ММ-ДД (та же дата выражается как 2012-07-27), ММ/ДД/[ГГ]ГГ (то есть 12/25/12 или 12/25/2012 будут 25 декабря 2012 года), или просто ММДД[ГГ]ГГ (так что 122512 или 12252012 будет, точно так же, соответствовать 25 декабря 2012 года). Без неё команда будет запущена, как только часы подойдут к указанному времени (в тот же день или на следующий день, если сегодня это время уже прошло). Можно также просто написать «today» или «tomorrow» — сегодня или завтра, соответственно.

```
$ at 09:00 27.07.14 «END
> echo "Не забудь поздравить Иванова с днём рождения! " \
>     | mail user0examle.com
```

> END

```
warning: commands will be executed using /bin/sh job
31 at Fri Jul 27 09:00:00 2014
```

Альтернативный синтаксис откладывает запуск на заданный промежуток времени: **at now + число период.** Значение период может быть в минутах, часах, днях или неделях. Значение число указывает число указанных единиц, которое должно пройти перед запуском программы.

Для отмены задачи, запланированной cron, нужно просто запустить **crontab -e** и удалить соответствующую строку в файле crontab. Для задач at это почти так же легко: надо запустить atrm номер-задачи. Номер задачи указывается командой at при её планировании, а также её можно найти с помощью команды atq, выводящей текущий список запланированных задач.

Планирование асинхронных задач: anacron. Апасгоп — это демон, дополняющий cron на компьютерах, которые не включены всё время. Поскольку регулярные задачи обычно планируются на середину ночи, они никогда не будут запускаться, если компьютер в это время выключен. Назначение апасгоп — запустить их, принимая во внимание периоды, в которые компьютер не работает.

Упражнение 14. Управление процессами. Автоматизация заданий.

1. Какие основные три процесса наиболее потребляют ресурсы на вашем компьютере?

2. Присутствуют ли на вашей машине процессы-зомби?_____
3. Вы должны однократно запустить некоторую задачу в определенное время. Вы также должны наметить другую задачу, которая стартовала бы каждые десять минут (между 8:00 и 17:00). Можете самостоятельно выбрать задачи и настроить их на запуск или продолжать по учебнику:
4. Вы хотите послать себе напоминание, чтобы пойти завтракать с Джо в 11:00 утра, сегодня.

   Войдите в систему как root и введите следующие команды:

   at noon; *<Нажмите Enter >*

   echo "Time for lunch with Joe. " *<Нажмите Enter >*

   <После завершения нажмите ctrl-D >
5. Используйте команду **atq,** чтобы проверить очередь заданий
6. К примеру, вы хотите получать информацию о статусе системы каждые десять минут. Войдя в систему как **root,** используйте команду **crontab -e,** чтобы редактировать ваш **cron** файл.
7. Введите следующую строку в ваш **crontab** файл:

```
*/10 8-17 * * * /usr/bin/free; /usr/bin/iostat
```

8. Проверьте ваш почтовый ящик, после того как задание отработает
9. Посмотрите, запущен ли на вашей системе **sendmail:**

   $ ps -ax | grep sendmail

10. Пошлите сигнал, требующий завершения процесса:

    $ kill 2057

11. Проведем небольшой эксперимент. Вначале убедимся, что каталог, где сохраняются задания at, пуст:

    # Is /var/spool/cron/atjobs/

12. Проверим время и дату:

    date

    ```
    Сбт 15 Май 2006 19:10:29
    ```

13. Запланируем создание файла в корневом каталоге через две минуты:

    ```
    at 1912
    at> touch /tuition at> <Ctrl-D>
    commands will be executed using /sbin/sh job 1084633920.a
    at Сбт Май 15 19:12:00 2006
    ```

14. В каталоге, где собираются задания в очереди на выполнение, появился новый файл:

    ```
    Is /var/at/jobs/
    1084633920.a
    ```

15. И по команде at **-1** получим информацию о поставленном в очередь задании (указывается имя пользователя, инициировавшего задание, идентификатор задания, по которому его можно удалить, и время планируемого запуска):

    ```
    at -1
    user = root 1084633920.a Сбт Май 15 19:12:00 2004
    ```

Модуль 15. Настройка swap

Под swap область рекомендуется выделять не более одного раздела (предпочтительнее самого первого) на каждом жестком диске.

В случае срочной необходимости под swap можно выделить и обычный файл. Создайте файл большого размера, а затем подключите его под swapping. Пример: выделяем 20 МЪ- овый файл

```
# dd if=/dev/zero of=/var/swapfile bs=lk count=20480
```

Чтоб swap подключался автоматически при начальной загрузке, надо вписать строку в /etc/fstab:

/dev/hdb**2** none swap sw

Приоритет SWAP-файлов

Создавать и использовать swap-файлов в Linux можно любое количество. При этом можно указать приоритет подключаемого swap-файла или раздела. Приоритет является целым числом от 0 до 32767.

Очистка swap-пространства после ресурсоёмких приложений

Командой **swapoff -a,** запущенной от имени root, можно отключить использование всех разделов и файлов подкачки. После ввода команды содержимое свопа за несколько минут загружается обратно в оперативную память, а сам раздел подкачки отключается. После загрузки содержимого свопа в оперативную память включем своп обратно командой **swapon** -a.

Системные настройки использования своппинга - Linux

За интенсивность обращения системы к swap-файлам и swap-разделам отвечает параметр **swappiness,** равный по умолчанию 60. Значение параметра может быть в пределах от 0 - наименьшее использование подкачки, до 100 - подкачка используется часто.

Насчёт оптимального значения **swappiness** существуют разные мнения. Например, считается, что для десктопа лучше ставить большое значение, чтобы всякое bloatware скинуть в своп и использовать оперативную память для чего-то нужного.

Но чрезмерное значение здесь приведёт к интенсивному использованию swap-файла, что нежелательно. Слишком маленькое значение может привести к тому, что при заполнении памяти будет принудительно запущен OOMkiller (процесс, запускающийся при исчерпании памяти и убивающий наиболее ресурсоёмкие задачи).

Упражнение 15. Настройка swap

1. Посмотрите текущие параметры swap-а в системе

```
# swapon -s
```

2. На одном из дисков (например, /dev/sdc) выделите отдельный раздел, например /dev/sdc3 (будем использовать под swap). Размер раздела можно задать произвольно. Не забудьте изменить id раздела (Linux swap).

```
# fdisk /dev/sdc
```

(В подменю команды fdisk: n - новый раздел; w-сохранение разбиения на диск; t — id раздела (для swap ставим 82))

3. Создайте swap из нового раздела

```
# mkswap /dev/sdc3
```

4. Подключите созданный swap-раздел:

```
# swapon /dev/sdc3
```

5. Проверьте, что swap-раздел был добавлен к основному swap

```
# swapon -s
```

6. Если на диске нет места под создание новых разделов, можно создать файл для использования его в качестве swap. Создайте файл определенного размера, например 500M

```
# dd if=/dev/zero of=/swapfile bs=lM count=500
```

7. Создайте swap из нового файле

```
# mkswap /swapfile
```

8. Подключите созданный swap-файл:

```
# swapon /swapfile
```

9. В выводе команды top или команды free должно появиться упоминание, что свопинга в системе поприбавилось. Проверьте это

```
# top
# free
```

10. Отключив файл подкачки, пишем

```
# swapoff /swapfile
```

11. Чтобы не подключать swap-файл или swap-раздел каждый раз, добавьте соответствующую запись в /etc/f stab

```
# vi /etc/fstab /dev/sdc3 none swap sw 0
0 /swapfile none swap sw 0 0
```

12. Установите, наивысший приоритет для файла подкачки

```
# swapon -p 1 /swapfile
```

13. Произведите очистку swap-пространства на вашем компьютере
14. Измените параметр **swappiness** временно (до перезагрузки системы):

 # echo 50 > /proc/sys/vm/swappiness

15. Также измените значение по умолчанию, для этого необходимо изменить параметр **vm**. **swappiness** в файле /etc/**sysctl**. conf vm.swappiness=50

    Следует отметить, что при больших значениях система потеряет в отзывчивости (будет вытеснять память, с которой работают приложения, в своп, хотя оперативной памяти ещё много). При малых значениях система работает отзывчивей, но когда оперативная память заканчивается, система начинает активно свопиться и притормаживать.

16. Также можно попробовать увеличить\уменынить объём потребляемой системой памяти за счёт изменения размеров дискового кеша. Уровень выделяемой под кеш памяти хранится в

    /proc/ sy s / vm/ vf s_cache_pres sure

    Значение по умолчанию: 100. Чтобы использовать меньше памяти под дисковые кеши (не желательно), поставьте значение 50. Если, наоборот, хочется больше отзывчивости системы, увеличьте размер кеша:

    # echo 1000 > /proc/sys/vm/vfs_cache_pressure

    продолжаем менять параметры вплоть до полного удовлетворения. Для того, чтобы настройки стали постоянными, занесите нужный параметр в файл **/etc/sysctl. conf # vi /etc/sysctl.conf**

    vm.vfs_cache_pressure = 1000

Модуль 16. ACL (Access Control List)

Для реализации сложных структур прав доступа используются расширенные права - ACL (Access control list - списки контроля доступа). Списки контроля доступом (ACL) дают большую гибкость, чем стандартный набор полномочий «пользователь/группа/ остальные».

Существуют два типа ACL:

- ACL для доступа;
- ACL по умолчанию.

ACL для доступа — это список управления доступом для заданного файла или каталога. Это сами права на объект, которые будут контролировать доступ к этому объекту.

ACL по умолчанию - может быть связан только с каталогом, и, если файл в этом каталоге не имеет ACL для доступа, он использует правила, определённые в ACL по умолчанию, связанном с каталогом. ACL по умолчанию являются необязательными.

Управления ACL списками осуществляется двумя командами:

setf acl - используется для назначения, модификации и удаления ACL прав.

getf acl - используется для просмотра установленных ACL.

Чтобы посмотреть, установлены ли ACL на объектах, достаточно воспользоваться командой Is -1.

Символ "+" в конце списка стандартных прав сообщает о наличии установленных прав ACL:

 # Is -1

```
drwxrwxrwt 2 root root 4096 2009-07-24 21:20 allex
 -rwxr-x --+ 1 root root 19 2009-07-25 14:45 qwert
```

Для модификации или добавления правила используется параметр -т.

```
-т user:[пользователь]:права[,user:пользователь:права]
-ш group:[группа]:права[,group:группа:права]
```

Если *пользователь* пропущен, то права назначаются владельцу файла. Если *группа* пропущена, то права назначаются группе-владельцу файла.

Добавить право на чтение/запись файла secretinfo пользователям **iivanov** и **a petrov:**

 # setfacl -m user:iivanov:rw,u:apetrov:rw secretinfo Добавить право на чтение/выполнение файла **runit** группе **manager:**

 # setfacl -m group:manager:rx runit

Добавить для файла **qwerty** право чтения для пользователя **iivanov,** право чтение/запись для **apetrov,** чтение выполнение для группы **legist,** запретить группе **workers** выполнять какие либо действия.

`setfacl —modify u:iivanov:r,u:apetrov:rw,g:workers:-,g:logist:rx qwerty`

Для директорий можно указать ACL права, которые будут автоматически добавляться для файлов и директорий, создаваемых в ней. Для этого используется идентификатор **default** или параметр **-d.** На саму директорию права указанные в **default** не распространяются.

Для файлов и директорий, создаваемых в директории **managerdata,** добавить право чтение/записи для пользователя **iivanov.**

`setfacl -m default:user:iivanov:rw managerdata`

Для установки прав для всех файлов и директорий внутри директории используется параметр **-R.**

Для удаления правила для пользователя или группы используется параметр -x.

`setfacl -x u:iivanov secretfile`

Упражнение 16. ACL (Access Control List)

1. Создайте новый файл **file2** в вашей рабочей директории
2. Посмотрите ACL для файла **file2.** Совпадают ли права доступа с установленной маской?
   $ getfacl file2
3. Измените права доступа на чтение запись и выполнение для группы файла **file2.**

   $ chmod g=rwx file2
4. Посмотрите права на **file2** с помощью команды **Is -1** а также посмотрите ACL для этого файла.

   $ getfacl file2 $ Is -1 file2
5. Установите маску для файла file2 - только чтение.

   $ setfacl -m mask::r— file2
6. Посмотрите права на **file2** с помощью команды **Is** -1 а также посмотрите ACL для этого файла.

   $ getfacl file2 $ Is -1 file2

7. Если группа groupl не существует в вашей системе, создайте эту группу с ID равным **101.**

8. Добавьте в ACL группу groupl для файла f ile**2**. Установите только права на чтение и выполнение для данной группы.
   $ setfacl -m g:groupl:r-x file**2**

9. Добавьте в ACL пользователя userlO для файла file**2**. Добавьте для этого пользователя доступ по исполнению файла.

   $ setfacl -m u:userl0:x file2

   $ getfacl f±le2

10. Попробуем удалить и воссоздать ACL для файла f ile**2**:

    $ setfacl -b file2

    $ setfacl -m u:nobody:rw-,g:wheel:rw- file2

Если указанные пользователь и группа отсутствуют, можно выбрать любых других из соответствующих файлов

Опция "-Ь" удаляет все ACL, кроме стандартных полномочий пользователя, группы и остальных. Опция "-т" модифицирует ACL указанным элементом (или несколькими элементами, разделенными запятой). Можете использовать **setfacl** для модификации "традиционных" полномочий; установка элемента **user**: : **rw-** эквивалентна запуску **chmod u=rw** для изменения прав на файл.

11. Удалите ACL для конкретного пользователя и группы:
    $ setfacl -x u:nobody,g:wheel file2

12. Рассмотрим усложненный пример. Вы хотите сделать ваш каталог **cool_widgets** доступным для пользователя **Bob,** кроме остальных.

13. Для этого необходимо добавить ACL-элемент. Хотя, когда вы добавляете файлы в этот каталог, они не будут автоматически приобретать ACL каталога. Вам необходимо установить для него ACL по умолчанию. Любые файлы, созданные в каталоге, будут наследовать ACL по умолчанию.
    Используя опцию **"-d"** при вызове **getfacl** или **setfacl** эффект будет виден на ACL каталога, а не на самом каталоге.

    $ mkdir cool_widgets $ chmod o-rwx

    cool widgets

unux special mention 1.4 $

```
Is -1
drwxr-x---  2 rob rob 512 Apr 19 21:21 cool_widgets
```

14 $ getfacl -d cool_widgets #file:cool_widgets

♦owner:1000 #group:1000

Добавляем ACL по умолчанию:

$ setfacl -d -m u:bob:rw- cool_widgets

Замечание: *В большинстве Linux/Unix ACL по умолчанию работают не так, как обычные ACL (если это Ваш случай - после предыдущей команды Вы увидите сообщение об ошибке). Вы не можете установить определенный элемент ACL по умолчанию, пока не добавите общие элементы user::, group:: и other::.*

$ setfacl -d -m u: :rw-,g: :г—,o: :----------,u:bob:rw- cool_widgets

$ setfacl -m u:bob:r-x cool_widgets

Обратите внимание на необычный элемент "r-x" для пользователя bob в записях для каталога: ACL по умолчанию заметен в свойствах файлов, созданных в каталоге, а не в свойствах самого каталога. ACL-элемент "u:bob:rw-" будет добавлен к ACL любого файла, созданного в каталоге cool_widgets.

Теперь у Вас имеется каталог cool_widgets, файлы в котором доступны для чтения и записи пользователям rob и bob без использования группы Если в дальнейшем Вы решите избавиться от ACL по умолчанию, используйте опцию **"-к"** команды setfacl, которая работает с ACL по умолчанию так же, как опция Ь" - с ACL файлов.

Настройка интерфейса Ethernet. Основную настройку сети можно выполнить, редактируя конфигурационный файл interfaces, который располагается в **/etc/network/interfaces.** Здесь Вы можете задать **IP** адресе сетевой карты (или использовать **DHCP),** настроить маршрутизацию, **IP** masquerading, установить маршрут по умолчанию и многое другое.

Если Вы хотите использовать DHCP вам необходимо написать следующие:

```
auto ethO
iface ethO inet dhep
```

Если Вы хотите сконфигурировать вручную, например задать шлюз по умолчанию (так же опционально можно задать: сеть, широковещательный адрес или шлюз): `auto ethO`

```
iface ethO inet static
address 192.168.0.7
netmask 255.255.255.0
gateway 192.168.0.254
```

/etc/hosts — список ip-адресов и назначенных им имен.

/etc/resolv. conf - указываются DNS сервера (например, nameserver 192.168.1.1) **/proc/sys/net/ipv4/ip_forward -** включение маршрутизации для между интерфейсами.

Включение осуществляется передачей "1" в этот файл:

 # echo *1' > /proc/sys/net/ipv4/ip_forward

Перезапуск всех интерфейсов:

 # /etc/init.d/networking restart

Статический маршрут для интерфейса (сохраняется при перезагрузке ОС):

В **/etc/network/interface,** дописать в конец файла: up ip routeadd 192.168.0.0/16 via 192.168.1.1

Статический маршрут для интерфейса (не сохраняется при перезагрузке ОС):

 # route add -net 192.168.0.0/16 gw 192.168.1.1 Просмотр информации о маршрутах:

 # netstat -r

Упражнение 17. Конфигурирование сетевых подключений

1. Убедитесь, что сетевые карты установлены, а если они интегрированы в материнскую плату, то включены в BlOSe, а так же подключены сетевые кабели. Проверьте видит ли система ваши сетевые карты (интерфейсы). Для этого посмотрите сообщения ядра

```
# dmesg | grep eth
```

2. Так же выведите список устройств с помощью команды **lspci:**

```
# lspci | grep Ether
```

Если на этом этапе вы получаете пустые строки или сообщения об ошибках, значит оборудование: либо не подключено, либо не исправно, либо не совместимо. Если проблема возникла после сборки своего ядра — проверяем ядро.

3. Затем проверьте вывод команды ifconfig:

```
# ifconfig
```

Отредактируйт файлы конфигураций. Перед редактированием конфигурационных файлов для сети следует остановить сетевой интерфейс.

```
# ifdown ethO
```

4. Настройте ваш сетевой интерфейс с помощью команды ifconfig

```
# ifconfig ethO 128.138.240.1 up netmask 255.255.255.0 broadcast
128.138.240.255
```

5. Проверьте доступность вашего сетевого интерфейса

```
# ping 192.168.2.131
```

6. Посмотрите текующую таблицу маршрутизации

```
# netstat -r -n
```

7 . Чтобы новые настройки сетевого интерфейса сохранялись при перезагрузке системы, добавьте их в конфигурационный файл **/etc/network/interf aces**

```
# vi /etc/network/interfaces
# The loopback network interface
auto lo
iface lo inet loopback
auto ethO iface ethO inet static # интерфейс (iface) сетевой карты (ethO)
                                 # со статическим ip (static).
                                                  # находится в диапазоне адресов
ipv4 (inet)
                          address 192.168.2.131
                          netmask 255.255.255.0
                          network 192.168.2.0
                        broadcast 192.168.2.255
                          gateway 192.168.2.1

# dns-* options are implemented by the resolvconf package, if installed
dns-nameservers 192.168.1.1     192.168.2.1
dns-search example.com
```

Если вы хотите чтобы при загрузке у сетевой карты **ethO** был другой MAC-адрес, для этого нужно дописать строчку **hwaddress ether 00:e0:4c:d0:99:28:**

Для настройки DHCP и получения сетевых параметров автоматически, нужно добавить две строчки:

vi /etc/network/interfaces auto eth0

iface eth0 inet dhcp

8. Каково назначение файла /вtc/hosts? **С помощью каких служб можно заменить или дополнить файл** hosts? _____

9. Отредактируйте файл /etc/hosts. **Поставьте в соответствие ГР адрес и имя компьютера:**

127.0.0.1 localhost.locAomain localhost

_IP_Address_ your-hostname.your-domain your-hostname

10. Установите доменное имя системы:

$ hostname your-hostname.your-domain

11. Отредактируйте файл **/etc/resolv.conf**:

 # vi

/etc/resolv.conf

search example.com

nameserver 192.168.2.1

nameserver 192.168.2.2

В первой строке указываем название своей рабочей группы или домена после слова search. Во второй и третьей строках после слов nameserver пишем ip-адреса DNS- серверов вашей сети.

12. Активируйте сетевой интерфейс:

 # ifup eth0

13. Перезапустите сетевые службы:

 # /etc/init.d/networking restart

14. Проверьте что получилось:

 # ifconfig

15. Настройте виртуальный сетевой интерфейс (алиас):

 # ifconfig eth0:1 192.168.2.135 netmask 255.255.255.0 broadcast 192.168.2.255 up

16. Перед тем, как добавить маршруты посмотрим таблицу маршрутизации:

 # route

17. Для локального интерфейса выполняем команду:

 # route add -net 127.0.0.0 netmask 255.0.0.0 lo

18. А для сетевого интерфейса выполняем сначала:

route add -net 192.168.2.0 netroask 255.255.255.0 ethO

19. А затем добавим основной шлюз:

route add default gw 192.168.2.1 ethO

20. Посмотрим опять таблицу маршрутизации:

route

21. Опишите сходства и различия перечисленных ниже утилит:

	и	ftp
scp		
ssh	**и**	telnet
	и	ssh
rsh		

Модуль 18. Служба удаленного администрирования SSH

SSH - это сетевой протокол для TCP-соединений, позволяет удалённо управлять операционной системой. SSH -аббревиатура от Secure SHell (безопасная оболочка). Передаваемая по этому протоколу информация сжимается и шифруется, а так же производится проверка модификации пакетов. Кроме администрирования, через SSH можно передавать другие сетевые протоколы, создавая защищённые туннели

Существует несколько реализаций программного обеспечения для SSH. Наиболее популярная — OpenSSH (Open BSD Secure SHell -изначально был разработан для BSD систем).

В пакет OpenSSH входит несколько программ: scp (Secure CoPy), ssh (Secure SHell) и sftp (Secure File Transfer Protocol). Scp нужна для защищённого копирования файлов с одного компьютера на другой, ssh предоставляет защищённый доступ к оболочке. В основном именно она используется для удалённого администрирования, sftp - для защищённой передачи файлов по протоколу FTP.

SSH использует технологию клиент/сервер Серверная часть **sshd.** Конфигурируется с помощью файла **/etc/ssh/sshd_config.** Клиентская часть **ssh.** Конфигурируется с помощью файла **/etc/ssh/ssh_config**

Упражнение 18.1. Общая принципы работы SSH

1. Проверьте, установлены ли в системе какие либо пакеты openssh:
   $ aptitude search openssh
2. Если серверная или клиентская часть не установлены, установите их
   # aptitude install openssh-cllent
   # aptitude install openssh-server
3. Проверьте, запущен ли **sshd:**
   # ps ax | grep sshd
4. Если не запущен, запустите его
   # service ssh start
5. Проверьте состояние сервера **ssh:**
   # /etc/init.d/ssh status

При подключении по SSH используются три вида аутентификации: аутентификация по паролю, по ключу хоста и по открытому ключу. Аутентификация по паролю в чистом виде, является наименее безопасной. Рассмотрим разницу между аутентификацией по ключу хоста

и аутентификацией по открытому ключу. При первом подключении с использованием аутентификации по ключу хоста открытый ключ хоста (сервера) копируется на компьютер - клиент в профиль пользователя, инициировавшего удалённое подключение, в файл /. ssh/known_hosts. При последующих подключениях копия открытого ключа на клиенте сравнивается с открытым ключом на сервере и далее следует запрос пароля для пользователя. При использовании аутентификации по открытому ключу, пара ключей генерируется не на хосте (сервере), а на клиенте и полностью его идентифицирует. Открытый ключ шифруется и копируется на сервер. При подключении происходит сравнение копии открытого ключа и оригинала и если ключи совпали, происходит вход на удалённый компьютер.

6. Осуществить подключение к серверу SSH можно с помощью команды ssh:

```
$ ssh 192.168.2.131 $ ssh
-1 user 192.168.2.231 $
ssh user@example.com
```

При первом подключении будет выдан следующий запрос:

```
The authenticity of host 'example (192.168.2.231)' can't be established.
RSA key fingerprint is
ee:ba:b8:48:ef:96:93:a8:3d:7b:3f:fa:85:8c:a7:ad.
Are you sure you want to continue connecting (yes/no)?
```

Предлагается принять открытый ключ удалённого сервера ssh и продолжить подключение. Для обеспечения безопасности ssh генерирует пару ключей: открытый и закрытый. Закрытый ключ никому не показывается, открытыми ключами компьютеры обмениваются между собой. Соглашаемся. Получаем сообщение о добавлении сервера example. com 192.168.2.231 в список известных хостов и приглашение для ввода пароля пользователя. Вводим пароль и получаем доступ к удалённому компьютеру. В результате, открытый ключ удалённого сервера example.com был скопирован на локальный компьютер в файл known_hosts, в директорию ~/. ssh (домашнюю директорию пользователя). Теперь при последующих подключениях к серверу example.com не будут выдаваться предупреждения, будет выводиться только запрос пароля.

Упражнение 18.2. Настройка SSH сервера

Настройки sshd находятся в файле /etc/ssh/sshd_conf ig.

1. Откройте этот файл для редактирования и отредактируйте его:

Port - по умолчанию используется 22 порт. Изменим его на нестандартный порт 2203 - это избавит сервер от сетевых роботов, которые автоматически сканируют интернет в поиске открытых портов и пытаются через них подключиться.

Port 2203

2. **Перезапустите службу ssh и попробуйте подключиться к серверу (нужно явно указать порт).**

   $ ssh -1 user -p 2203 192.168.2.231 или

   $ ssh user@exainple.com -p 2203

3. По умолчанию сервер «слушает» (принимает подключения) на всех сетевых интерфейсах. Если это устраивает, оставьте значение по умолчанию.

   ```
   #ListenAddress ::
   #ListenAddress 0.0.0.0
   ```

   Если нужно оставить подключение только через внешний интерфейс, то раскомментировав строку укажите:

   ```
   ListenAddress 192.168.2.0
   ```

4. Следующий параметр отвечает за версию протокола SSH. Значение по умолчанию 2. Не меняйте на первую версию — она не безопасна

   ```
   Protocol 2
   ```

5. Строки HostKey необходимы для второй версии протокола SSH и отвечают за названия файлов ключей и их расположение. Первая строка отвечает за пару ключей RSA, вторая за пару ключей DSA. К названиям открытых (публичных) ключей добавляется .pub. Эти ключи используются при аутентификации с ключом хоста. Отключите ключи DSA, будем пользоваться RSA:

   ```
   HostKey /etc/ssh/ssh_host_rsa_key
   #HostKey /etc/ssh/ssh_sunup_dsa_key
   ```

6. Privilege Separation указывает, должен ли демон sshd разделять привилегии. Если да - сначала будет создан непривилегированный дочерний процесс для входящего сетевого трафика. После успешной авторизации будет создан другой процесс с привилегиями вошедшего пользователя. Основная цель разделения привилегий - предотвращение превышения прав доступа. Оставляем yes.

   ```
   UsePrivilegeSeparation yes
   ```

7. Следующие строки отвечают за временный ключ и его длину при работе с первой версией протокола SSH. Не используем SSH-1, поэтому закомментируем строки.

   ```
   # KeyRegenerationInterval 3600
   # ServerKeyBits 768
   ```

8. Группа параметров, отвечающая за журналирование. События, связанные с доступом по SSH записываются в **/var/log/auth.** Первый параметр определяет, список событий в журнале. Доступны значения: DAEMON, USER, AUTH, LOCALO, LOCAL 1, LOCAL2, LOCAL3, LOCAL4, LOCAL5, LOCAL6, LOCAL7. Нас интересует авторизация, поэтому оставляем AUTH.

```
SyslogFacility AUTH
```

9. Второй параметр определяет уровень детализации событий. Доступны: SILENT , QUIET, FATAL, ERROR, INFO, VERBOSE, DEBUG, DEBUG1, DEBUG2, DEBUG3. Оставляем уровень детализации по умолчанию.

```
LogLevel INFO
```

10. Следующая группа параметров относится к аутентификации. Первый означает, что соединение будет разорвано через указанное количество секунд, если пользователь не войдёт в систему. Уменьшим это время в два раза.

```
LoginGraceTime 60
```

11. Второй параметр разрешает/запрещает вход под root-ом. Запрещаем вход суперпользователю.

```
PermitRootLogin no
```

12. Третий параметр включает проверку демоном ssh прав и владение домашним каталогом пользователя, который пытается получить удалённый доступ к компьютеру. Оставляем yes.

```
StrictModes yes
```

13. Добавляем параметр AllowUsers, которого нет в конфигурационном файле по умолчанию. Этот параметр разрешает доступ к серверу по протоколу SSH только для перечисленных пользователей.

```
AllowUsers student
```

В примере разрешение только у пользователя student. Значениями параметра могут быть имена пользователей, отделённые друг от друга пробелами. Можно использовать запись пользователя в виде student@example.com - доступ разрешён пользователю student с компьютера example.com. Существует параметр - DenyUsers, который запрещает доступ пользователям, перечисленным в значении этого параметра. Кроме параметров связанных с доступом отдельных пользователей существуют соответствующие параметры для групп: AllowGroups и DenyGroups.

14. Оставляем включенной аутентификацию RSA, в параметре RSAAuthentication:

```
RSAAuthentication yes
```

15. Оставляем включенной аутентификацию по открытому ключу, в дальнейшем настроим аутентификацию таким способом.

    PubkeyAuthentication yes

16. Параметр AuthorizedKeysFile определяет файл, в котором содержатся публичные ключи, используемые для аутентификации пользователей по открытому ключу. В записи могут присутствовать переменные, например %h означает домашний каталог пользователя, а %и - имя пользователя. В дальнейшем мы планируем использование аутентификации по открытому ключу - раскомментируем эту строку.

    AuthorizedKeysFile . ssh/authorized_keys ■

17. Следующие два параметра отвечают за включение совместимости с программой rhosts. Нам такая совместимость только повредит, поэтому оставляем значения по умолчанию. IgnoreRhosts yes

    RhostsRSAAutbentieation no

18. Аутентификация hostbased не нужна и она уже отключена, поэтому оставляем существующее значение

    HostbasedAuthentication по

19. Следующая строка нужна для совместимости с rhost - оставляем закомментированной.

    #IgnoreUserKnovmHosts yes

20. Параметр PermitEmptyPasswords разрешает или запрещает вход с пустым паролем. Естественно, запрещаем вход с пустым паролем — оставляем по.

    BermitEmptyPasswords по

21. ChallengeResponseAuthentication включает PAM интерфейс.Если yes, для всех типов аутентификации помимо обработки модуля сессии и аккаунта PAM, будет использоваться аутентификация на основе запроса-ответа (ChaUengeResponseAuthentication и PasswordAuthentication) Т.к. аутентификация запросов-ответов в PAM обычно выполняет ту же роль, что и парольная аутентификация, следует отключить, либо PasswordAuthentication, либо ChallengeResponseAuthentication. ChallengeResponseAuthentication no

22. Следующий параметр отвечает за аутентификацию по паролю. Сейчас используется аутентификация по ключу хоста - просто раскомментируем строку.

    PasswordSuthentig-ation;, yes

23. Группу из четырёх параметров, отвечающих за аутентификацию Kerberos, оставляем без изменений — не раскомментированными.

    #Kerber©sAutfo.enfeication до ■„
    #KerberosGetAFSToJ«en ..др #
    Kerb ero s;0rjio c,a 1P as s wd yes

```
#KerberosTicketCleanup yes
```

24. Следующие два закомментированных параметра отвечают за то, разрешена ли аутентификация пользователя на основе GSSAPI или нет. обычно GSSAPI не нужна — оставляем без изменений.

```
#GSSAPIAuthentication no
#GSSAPICleanupCredentials yes
```

25. Параметры, начинающиеся с X11, отвечают за проброс иксов через ssh туннель. Если сервер не имеет иксов, комментируем эти опции

```
# X11Forwarding yes
# X11DisplayOffset 10
```

26. Опция PrintMotd выводит при подключении к sshd так называемое сообщение дня, что на самом деле является содержимым файла /etc/motd. Опция PrintLastLog очень полезна, так как она включает отображение информации о том, когда вы последний раз и с какого компьютера заходили на сервер.

```
PrintMotd no
PrintLastLog yes
```

27. Установим параметру TCPKeepAlive значение no. Этот параметр важен для поддержания соединения со стороны сервера, но мы реализуем те же функции, но более безопасней.

```
TCPKeepAlive no
```

Для этого добавим два следующих параметра:

```
ClientAliveCountMax 3
ClientAliveInterval 20
```

Первый из параметров определяет количество запросов, которое ssh-сервер отправляет ssh-клиентам через определённые промежутки времени. Как только установленное значение запросов достигнуто, а клиент так и не ответил, соединение будет разорвано. Если не посылать такие запросы, то сессии на сервере придётся закрывать вручную, так как они будут длиться бесконечно, отбирая часть ресурсов. Второй параметр определяет интервал отправки запросов в секундах. В нашем примере, соединение будет разорвано, если между клиентом и сервером не будет связи в течение одной минуты.

28. Параметр UseLogin оставляем закомментированным. Его значение по умолчанию и так no.

```
#UseLogin no
```

29. Раскомментируем параметр MaxStartups и выставим ему следующее значение: `MaxStartups 3:30:9`

По умолчанию значение параметра 10 - количество неавторизованных подключений к серверу ssh. Длительность такого подключения определяется параметром

LoginGraceTime. Перенесём параметр MaxStartups под LoginGraceTime и запишем в виде start:rate:full, где start -имеющееся количество неавторизованных подключений, rate - процент вероятности отклонения попытки подключения, full - максимальное количество неавторизованных соединений. Т.е. если имеется 3 неавторизованных подключения, то следующее подключение будет отклонено с вероятностью 30%, а если количество неавторизованных подключений достигнет 9, все последующие попытки подключения - отвергнуты.

30. Опция Banner определяет место положения файла-баннера, который будет выведен на экран, при попытке подключиться к sshd. Предлагается файл /etc/issue.net, но можно использовать свой баннер, поместив его, например в /etc/ssh. По умолчанию выводится содержимое файла /etc/motd.tail.

    ```
    #Banner /etc/issue.net
    ```

31. Параметр AcceptEnv указывает, какие переменные окружения, переданные клиентом, будут приняты.

    ```
    AcceptEnv LANG LC_*
    ```

32. Следующий параметр включает внешнюю подсистему (например, FTP). В качестве параметров понимает, имя подсистемы и команду, которая будет выполнена при запросе подсистемы. Команда sftp-server, реализует протокол передачи файлов через SSH - SFTP. `Subsystem sftp /usr/lib/openssh/sftp-server`

33. Параметр UsePAM оставляем без изменений. Если директива UsePAM включена, то запустить sshd можно будет только от имени root.

    ```
    UsePAM yes
    ```

34. Сохраняем изменения и перезапускаем sshd. Подключаемся, проверяем, если всё в порядке, то настройка сервера ssh с аутентификацией оп ключу хоста закончена. Для разрешения проблем и вывода отладочной информации можно подключаться с ключами: **`-V, -w, -vw`**

Упражнение 18.3. Настройка SSH клиента

1. Глобальные клиентские настройки находятся в файле **`/etc/ssh/ssh_config`** и применяются ко всем пользователям. Пользовательские настройки могут находиться в домашнем каталоге пользователя, в ~/. **`ssh/config`** и применяются к одному пользователю. Файл пользовательских настроек не создаётся автоматически в отличие от файла глобальных настроек клиентской части **`ssh.`**

2. Параметр **Host.** Ограничивает множество хостов, к которым применяются последующие (до ближайшей новой директивы Host) директивы, по указанным шаблонам (хост должен соответствовать хотя бы одному шаблону). Шаблон, состоящий из одного символа *, соответствует любому хосту. Под хостом в данном контексте понимается аргумент имя_хоста передаваемый в командной строке (т.е. никаких преобразований перед сравнением не выполняется).
3. Параметр **HostName.** Устанавливает соответствие между псевдонимами, сокращениями и настоящими именами хостов. По умолчанию используется имя, передаваемое в командной строке. Допустимо непосредственное указание IP-адресов.
4. Параметр **Port.** Порт на удалённой машине, к которому следует подключаться. По умолчанию - **22**
5. Параметр **User.** Имя пользователя, которое следует использовать при регистрации в удалённой системе.
6. В качестве примера создадим файл пользовательских настроек **/home/selifan/. ssh/config** следующего содержания:

```
Host example HostName
example.com Port 2203
User student Host hostl
HostName hostl.example.com
Port 2280
User user2
Host 212.177.65.1
HostName 212.177.65.1
 Port 2222
User user5
```

Теперь при подключении к перечисленным компьютерам не нужно вспоминать имя пользователя или ни ssh порт подключения, достаточно после ssh набрать имя сервера.

Упражнение 18.4. Генерация ключей

1. При подключении с использованием аутентификации с ключом хоста открытый ключ сервера копируется на компьютер-клиент. На сервере ключи лежат в директории **/etc/ssh.** При установке OpenSSH создаются две пары ключей RSA и DSA, которые представляют собой отдельные файлы: **ssh_host_rsa_key, ssh_host_rsa_key.pub, ssh_host_dsa_key** и **ssh_host_dsa_key.pub.** Файлы, которые оканчиваются на .pub являются открытыми (публичными) ключами, а те, что оканчиваются на key - закрытыми. В процессе администрирования сервера может понадобиться сгенерировать

новые ключи. Рассмотрим генерацию ключей на следующем примере. Изменим параметр HostKey:
`HostKey /etc/ssh/ssh_sunup_rsa_key`

2. Удалим старые ключи:

```
# rm /etc/ssh/ssh_host*
```

3. Сгенерируем новую пару RSA ключей, для чего выполним следующую команду:

```
# ssh-keygen -t rsa -f /etc/ssh/ssh_sunup_rsa_key
```

В запросе о пароле оставляем поля пустыми. В результате будет создана новая пара RSA ключей с длиной шифрования 2048 бит.

4. Перезапускаем sshd:

```
# /etc/init.d/ssh restart
```

5. Теперь, если мы попробуем подключиться к серверу со старым ключом, то получим следующее предупреждение:

```
WARNING: HOST IDENTIFICATION HAS CHANGED! @
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that the RSA host key has just been changed.
The fingerprint for the RSA key sent by the remote host is
2b:f3:c6:42:29:f2:26:6c:6f:28:dc:16:39:9f:bb:e7 .
Please contact your system administrator.

Host key verification failed.
```

6. При попытке подключения не совпали слепки ключей (fingerprint). Решение проблемы содержится в сообщении - нужно просто удалить указанную строку, в нашем случае 3, в файле known_hosts, снова подключиться и принять новый ключ сервера.

```
$ vi ~/.ssh/known_hosts
```

Упражнение 18.5. Подключение с использовнием открытого ключа

1. Для подключения с авторизацией по открытому ключу нужно сгенерировать секретный ключ на стороне клиента. Делаем это с правами обычного пользователя:

```
$ ssh-keygen -t rsa
```

В процессе генерации пары ключей сначала будет предложено ввести желаемое название файла ключа, а так же ввести и подтвердить пароль. Вместо имени файла нажимаем Enter - будут созданы файлы ключей с именами по умолчанию. Секретную фразу не вводим, особенно если планируем удалённо запускать скрипты. В противном случае нужно будет каждый раз вводить секретную фразу или работать с помощью **ssh-agent.** Пароль для файла ключа может оказаться полезным в случае попадания ключа в чужие руки, но

такие вещи как секретные ключи надо беречь, даже если они защищены хорошим

паролем. Файлы закрытых ключей должны быть недоступны *для* чтения/записи всем кроме их владельца (режим 600). Открытые ключи для чтения должны быть доступны всем, но право на запись должен иметь только владелец (режим 644). В идеале лучше хранить ключи на Tokene.

2. **И** так, были сгенерированы два RSA ключа с именами **id_rsa** и **id._rsa.pub.** Теперь нужно скопировать на сервер открытый ключ в директорию, указанную в файле **/etc/ssh/sshd_config,** в параметре AuthorizedKeysFile, но сначала на сервере в домашней директории нужно создать скрытую поддиректорию .ssh, если её нет:

   $ mkdir ~/.ssh

3. Теперь с клиента копируем сгенерированные ключи на сервер, воспользовавшись утилитой scp, входящей в пакет OpenS SH:

   $ scp -P 2203 ~/.ssh/id_rsa.pub students example. com: . ssh/authorized_keys

4. Совсем отключаем аутентификацию по паролю. Для этого на сервере в файле **/etc/ssh/sshd_conf ig** меняем параметр PasswordAuthentication:

   PasswordAuthentication no

5. Подключаемся к серверу:

   $ ssh student@example.com -p 2203

6. Теперь можно копировать открытый ключ на сервер

Модуль 19. Протокол передачи файлов FTP

Предназначен обеспечения передачи файлов по сети по клиент-серверной технологии. Будем использовать vsftpd. VSFTPD (Very Secure FTP Daemon) — как следует из названия, очень защищённый демон FTP, с хорошей производительностью, поддерживаются: IPv**6**, SSL, виртуальные пользователи, есть контроль скорости полосы пропускания. Клиентская часть lftp. Серверная часть vsftpd. Конфигурация сервера осуществляется с помощью файла /etc/vsftpd. conf или с помощью утилиты fly-admin-ftp

Упражнение 19.1. Установка по умолчанию

1. Установите клиентскую часть lftp (если не установлена)

2. # aptitude search lftp

aptitude install lftp

3. Установите серверную часть vsftpd (если не установлена)

aptitude search vsftpd

aptitude install vsftpd

4. Созадайте пользователей для входа на ftp сервер (если нет пользователей в системе)

useradd userl

passwd userl

5. Выполнить с помощью клиентской части вход на сервер

FTP **`$ lftp 192.168.2.131`**

6. Выполнить команды

a) pwd - получение текущей директории

b) dir — получение списка файлов

c) get - получение фала (выберите файл самостоятельно)

d) put — поместить файл в выбранную директорию (выберите файл самостоятельно)

Упражнение 19.2. Простая конфигурация сервера FTP

1. Установите серверную часть vsftpd (если не установлена)

aptitude install vsftpd При

этом создаются:

- Системный пользователь **ftp,** который добавляется в ipyimy nogroup
- Домашняя директория пользователя **ftp - /home/ftp.**
- Создаётся файл **/etc/ftpusers. B** нём перечислены пользователи, которым запрещён доступ к FTP.

Пользователь ftp не будет работать с FTP сервером, но он нужен для его корректной работы.

2. Так как директория FTP сервера будет располагаться в **/var/f tp**, то изменим домашнюю директорию для пользователя ftp:

```
# usermod -d /var/ftp ftp
```

3. Удалим прежнюю директорию этого пользователя:

```
# rmdir /home/ftp
```

4. Теперь настроим пользователя, у которого будут права записи в корневую директорию FTP сервера. Создаём группу f tpuser

```
# addgroup ftpuser
```

5. Создаём пользователя ftpuser, добаляем его в группу ftpuser и устанавливаем домашним каталогом директорию FTP сервера, а так же меняем пароль

```
# useradd -d /var/ftp -g ftpuser ftpuser
# passwd ftpuser
```

6. Создаём директорию FTP сервера и устанавливаем права:

```
# mkdir /var/ftp
# chmod 555 /var/ftp
# chown root:ftpuser /var/ftp
```

7. Создаём публичный каталог.

```
# mkdir /var/ftp/pub
# chown ftpuser:ftpuser /var/ftp/pub
```

8. Конфигурационный файл **vsftpd** располагается в **/etc/vsftpd.conf**. Сделаем его резервную копию.

```
# cp /etc/vsftpd.conf /etc/vsftpd.conf_old
```

9. Теперь очистим /etc/vsftpd.conf, откроем текстовым редактором:

```
# cat /dev/null > /etc/vsftpd.conf
# vim /etc/vsftpd.conf и
```

напишем:

```
# Запускать vsftpd в независимом режиме
listen=YES
# Делаем анонимный доступ
anonymous ____ enable=YES
# Анонимные входят без пароля
no_anon_password=YES
# Анонимные будут попадать в публичную директорию
anon_root=/var/ftp/pub
anon_umask=022
# Разрешаем вход локальным пользователям с правом записи в домашних директориях
local enable=YES
```

```
local_umask=022
# Локальные пользователи будут входить только в свои домашние каталоги chroot_local_user=YES
chroot_lis t_enable=NO
# Сообщения будут записываться в собственный журнал xferlog_enable=YES
xferlog_file=/var/log/vsftpd.log
```

10. Теперь перезапустим `vsf tpd`:

```
# /etc/init.d/vsftpd restart
```

11. Если всё работает как надо и сервер будет использоваться не только локально, то добавим ещё несколько строк после listen:

```
# Установки публичного FTP
max_clients=100 max_per_ip=lO
hide_ids=YES
idle_ses s ion_timeout=60 0
data_connection_timeout=120
dirmessage_enable=YES f
tpd_banner=Welcome!
```

Упражнение 19.3. Сложная конфигурация сервера FTP

1. За основу возьмём предыдущую конфигурацию. Установим MySQL и поддержку авторизации PAM:

```
# aptitude install mysql-server mysql-client libpam-mysql
```

Во время установки будет создан суперпользователь root для управления MySQL. Нужно будет задать пароль для этого пользователя.

2. Авторизуемся в MySQL:

```
# mysql -u root -p
```

3. Создаём базу данных для хранения информации о пользователях FTP:

```
CREATE DATABASE vsftpd;
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP ON vsftpd.* TO 'vsftpd'@'localhost'
IDENTIFIED BY 'passftpd';
FLUSH PRIVILEGES;
```

4. Создаём таблицу для хранения пользователей и паролей. Таблица называется users и содержит три поля: **id** — целочисленный идентификатор пользователя, **login** — имя пользователя и **pass** — пароль пользователя. Под пароль отведено **60** символов, т. к. пароли хранятся в зашифрованном виде и длина шифра на много превышает длину секретного слова. Логины пользователей в таблице будут уникальными:

```
USE vsftpd;
CREATE TABLE users (
id INT NOT NULL AUTO_INCREMENT PRIMARY KEY , login VARCHAR(60) NOT NULL UNIQUE ,pass
VARCHAR(60) NOT NULL) ;
```

5. **Заводим в нашей базе данных пользователя для работы.**

```
INSERT INTO users (login, pass) VALUES('firstuser',
PASSWORD('topsecret') ) ;
```

6. **Выходим из** MySQL: `quit;`

7. **Открываем файл** `/etc/vsf tpd. conf` **и редактируем его, чтобы получилось**

следующее:

```
# Запускать vsftpd в независимом режиме
l±sten=YES
# Установки для публичного FTP
max_clients=lOO max_jper_ip=4 O
hide_ids=YES
idle_session_timeout=600
data_connection_timeout=120
dirmessage_enable=YES
f tpd_banner=We1come!
# Делаем анонимный доступ
anonymous_enable=YES
# Анонимные входят без пароля no_anon_password=YES
# Анонимные будут попадать в публичную директорию
anon_root=/var/f tp/pub
anon_umask=022
# Гостевой вход. Не анонимный вход рассматривается как гостевой guest_enable=YES
guest_username=ftpuser
nopriy_user=ftpuser
# Исползуем PAM
pam_service_name=vsf tpd
# Виртуальные пользователи будут иметь права локальных пользователей
virtual_use_local_privs=YES
local_root=/var/f tp/$USER
user_sub_token=$USER
# Специальные настройки пользователей находятся в этой директории
user_config_dir=/etc/vsftpd/user_conf
 # Разрешаем вход для теперь уже виртуальных пользователей local_enable=YES
write_enable=YES
 local_umask=022
 # Виртуальные пользователи будут входить в свои «домашние» каталоги
 chroot_local_user=YES
 chroot_list_enable=NO
 # Сообщения будут записываться в собственный журнал xferlog_enable=YES
 xferlog_f ile=/var/log/vsftpd.log
```

8. Создаём директорию, в которой будут располагается конфигурационные файлы пользователей FTP сервера:

```
# mkdir -p /etc/vsftpd/user_conf
```

9. Настроим PAM авторизацию. Сделаем резервную копию файла /etc/pam.d/vsfitpd

```
# cp /etc/pam.d/vsftpd /etc/pam.d/vsftpd_old
```

10. Очистим его:

```
# cat /dev/null > /etc/pam.d/vsftpd
```

11. И отредактируем следующим образом:

```
# vim /etc/pam.d/vsftpd
auth required pam_mysql.so user=vsftpd passwd=passftpd host=localhost db=vsftpd table=users
usercolumn=login passwdcolumn=pass crypt=2
account required pam_mysql.so user=vsftpd passwd=passftpd host=localhost db=vsftpd
                          table=users         usercolumn=login
passwdcolumn=pass crypt=2
```

12. Перезапустим vsf tpd:

```
# /etc/init.d/vsftpd restart
```

13. Создаём для пользователя firstuser директорию:

```
# mkdir /var/ftp/firstuser
```

14. Меняем владельца:

```
# chown ftpuser:nogroup /var/ftp/firstuser
```

15. Создаём файл со специальными настройками для firstuser. Для этого, в папке /etc/vsf tpd_user_conf создадим файл с именем пользователя

```
# touch /etc/vsftpd/user_conf/firstuser
```

и подредактируем его:

```
# vim /etc/vsftpd/user_conf/firstuser
```

16. В данном файле можете зададим следующие параметры:

```
# Домашний каталог для пользователя firstuser. Можно задать какой #угодно путь к домашнему
каталогу. Например:/var/ftp/ local_root=/var/ftp/firstuser/
# Ограничим скорость для данного пользователя
local_max_rate=15000
```

и так далее для для разных параметров.

17. Сохраняем файл и перезапускаем vsf tpd:

```
/etc/init.d/vsftpd restart
```

18. Создадим ещё одного пользователя: seconduser.

```
Mysql -u root -p USE
vsftpd;
 INSERT INTO users (login, pass) VALUES('seconduser',
 PASSWORD('topsecret')) ;
 quit;
```

19. Создаём для пользователя seconduser директорию:

```
# mkdir /var/ftp/seconduser
```

20. Меняем владельца:

```
# chown ftpuser:nogroup /var/ftp/seconduser
```

21. Создаём файл со специальными настройками для seconduser. Для этого, в папке /etc/vsf tpd_user conf создадим файл с именем пользователя

```
# touch /etc/vsftpd/user_conf/seconduser
```

22. Редактируем его:

```
# vim /etc/vsftpd/user_conf/seconduser
```

23. Нарпример, запретим запись:

```
write_enable=NO
```

24. Перезапустим `vsf tpd`:

```
# /etc/init.d/vsftpd restart
```

Не забываем, что владельцем корневого каталога `/var/ftp` должен быть пользователь ftpuser с правами 755. Если пользователю не определены персональные настройки, он будет иметь права пользователя ftpuser и попадать в его домашний каталог.

Упражнение 19.4. Конфигурация сервера FTP с шифрованием SSL

Теперь изменим нашу конфигурацию следующим образом: анонимные пользователи не используются. Используются только виртуальные пользователи, учётные записи которых хранятся в MySQL. Для каждого пользователя можно задать свою конфигурацию. Обеспечено шифрование SSL.

1. Установим OpenSSL:

```
# aptitude install openssl
```

2. Создаём директорию для сертификата

```
# mkdir /etc/vsftpd/ssl/
```

3. Создадим самоподписанный сертификат, срок действия которого 2 года:

```
# openssl req -x509 -nodes -days 730 -newkey rsa:1024 -keyout /etc/vsftpd/ssl/vsftpd.pem -out /etc/vsftpd/ssl/vsftpd.pem
```

В процессе создания сертификата заполняем предлагаемые поля:

Country Name: (пишем двухбуквенное название своей страны, например RU)
State or Province Name: (пишем название своего региона)
Locality Name: (пишем название своего города)
Organization Name: (пишем название своей организации)
Organization Unit Name: (пишем название подразделения организации)
Common Name: (пишем доменное имя организации)
Email Address: (пишем адрес электронной почты)

4. Открываем файл **/etc/vsf tpd. conf** и редактируем его, чтобы получилось следующее:

```
# Запускать vsftpd в независимом режиме
listen=YES
max clients=100
max_pe r _____ ip=10
hide_ids=YES idle_session_timeout=600
data_connection_timeout=l20
dirmessage_enable=YES
f tpd_banner=We1come!
# Убираем анонимный доступ
anonymous_enable=NO
```

```
# Гостевой вход. Не анонимный вход рассматривается как гостевой gues t_enable=YES
guest_username=ftpuser nopriv_user=ftpuser
# Виртуальные пользователи будут иметь права локальных пользователей virtual_use_local_jprivs=YES
local_root=/var/ftp/$USER user_sub_token=$USER
# Специальные настройки пользователей находятся в этой директории user_conf
ig_dir=/etc/vsftpd/user_conf
# Разрешаем вход для теперь уже виртуальных пользователей local_enable=YES
write_enable=YES
local_umask=022
# Виртуальные пользователи будут входить в свои «домашние» каталоги chroot_local_user=YES
chroot_list_enable=NO
# Сообщения будут записываться в собственный журнал
xferlog_enable=YES
xferlog_file=/var/log/vsftpd.log
# настройки SSL ssl_enable=YES allow_anon_ssl=NO
force_local_data_ssl=YES
force_local_logins_ssl=YES
ssl_tlsvl=YES
ssl_sslv2=YES
ssl_sslv3=YES
rsa_cert_file=/etc/vsf tpd/ ssl/vsf tpd. pern
```

5. Перезапускаем сервис

```
# /etc/inxt.d/vsftpd restart
```

Необходимо помнить, что теперь для подключения к нашему серверу следует использовать ftp — клиент, который поддерживает шифрование. Например File Zilla.

Модуль 20. Шифрование дисков LUKS

LUKS (Linux Unified Key Setup) - новый стандарт пришедший на смену dm-crypt и явно упрощающий работу с зашифрованными разделами. В этом случае в начале зашифрованного диска размещается информация о типе шифрования, размере ключа, а также зашифрованные парольными фразами ключи шифрования.

Преимущества:

- Работа с несколькими ключами
- Быстрое добавление/удаление ключей
- Возможность определить зашифрованный раздел

LUKS Минусы:

- Шифрование раздела возможно только с переносом данных

Упражнение 20. Шифрование дисков LUKS

1. Установите **cryptsetup-luks** если не установлен

   # aptitude install cryptsetup-luks

2. Создайте новый раздел на диске (можно использовать весь диск)

   # fdisk /dev/sdc

3. Отформатируйте раздел (диск) LUKS, например /dev/sdcl.

   # cryptsetup luksFormat /dev/sdcl

4. Подключите зашифрованный диск:

   # cryptsetup luksOpen /dev/sdcl lkfs

5. Создайте файловую систему на подключенном диске:

   # mkfs.ext3 /dev/mapper/lkfs

6. Создайте директорию для монтирования зашифрованного раздела и смонтируйте зашифрованный раздел в нее

   # mkdir /mnt/lkfs

   # mount /dev/mapper/lkfs /mnt/lkfs

7. Просмотрите список используемых ключей. Сколько свободных слотов для ключей присутствует?

   # cryptsetup luksDump /dev/sdcl

8. Добавьте ключевую фразу к слоту

   # cryptsetup luksAddKey /dev/sdcl

9. Добавьте ключевой файл

   # dd if=/dev/random of=/tmp/lkfs.key bs=l count=256

   # cryptsetup luksAddKey /dev/sdcl /tmp/lkfs.key

10. Разблокируйте зашифрованный раздел диска при помощи ключевого файла.

```
# oryptsetup -d /tmp/lkfs.key luksOpen /dev/sdcl lkfs
```

11. Удалите один из ключей

```
# cryptsetup luksDelKey /dev/sdcl 1
```

12. Чтобы операционная система сама научилась подключать и монтировать нужные криптованные устройства во время загрузки, а затем корректно отключать их во время останова системы, добавьте по одной строке в файлы **/etc/crypttab** и **/etc/fstab:**

```
# vi /etc/crypttab
lkfs /dev/sda5 none luks,cipher=aes-cbc-essiv:sha256"
# vi /etc/fstab
/dev/mapper/lkfs /mnt/lkfs ext4 defaults 0 0
```

Теперь во время каждой загрузки ОС будет спрашивать пароль для доступа к криптованному разделу, если он будет указан неправильно - загрузка остановится. Шифрование домашнего каталога производится по точно такой же схеме с тем лишь исключением, что перед добавлением новой записи в **/etc/fstab** следует удалить старую запись, ссылающуюся на **/home**. При создании шифрованной флешки специальные записи в **/etc/crypttab** и **/etc/fstab** не требуются. Подсистема HAL сама определит наличие на устройстве хранения LUKS-раздела и передаст эту информацию среде рабочего стола (Gnome, KDE, XFCE), которая, в свою очередь, выведет на экран окно с просьбой ввести пароль. Единственное, что необходимо сделать - при первом монтировании изменить права доступа на ее корневой каталог:

```
$ sudo chown -R student:student /media/usb name $ sudo chmod g+s /media/usb_name
```

Дополнительно:

LUKS/dm-crypt вполне подходит для создания образов файловых систем, которые можно использовать для хранения важной информации:

```
# dd if=/dev/urandom of=luks.img bs=lM count=100
# losetup /dev/loopO luks.img
# cryptsetup luksFormat /dev/loopO -c aes-cbc-essiv:sha256 -s 256
# cryptsetup luksOpen /dev/loopO luks
# mkfs.ext2 /dev/mapper/luks
# mkdir /rnt/luks
# mount /dev/mapper/luks /luks
```

LUKS/dm-crypt также умеет использовать одноразовые ключи, это полезно при шифровании swap-разделов:

```
# swapoff -a
# cryptsetup -d /dev/urandom create cryptoswap /dev/sdal
# mkswap /dev/mapper/cryptoswap -L accessisdenied -vl
# echo "cryptoswap /dev/sdal /dev/urandom swap" » /etc/crypttab
```

```
# echo "/dev/mapper/cryptoswap none swap sw 0 0" » /etc/fstab
# swapon -a
```

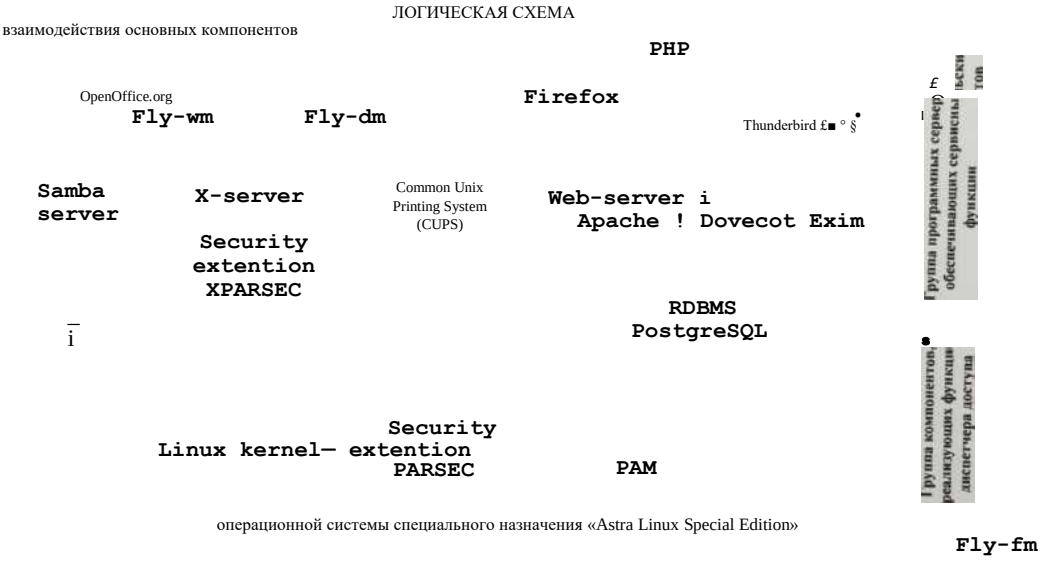
Программные средства, входящие в состав ОС СН, могут быть поделены на 3 основных группы:

- группа компонентов, реализующих функции диспетчера доступа - программные средства, реализующие диспетчеры доступа (модули ядра, защищенная СУБД);

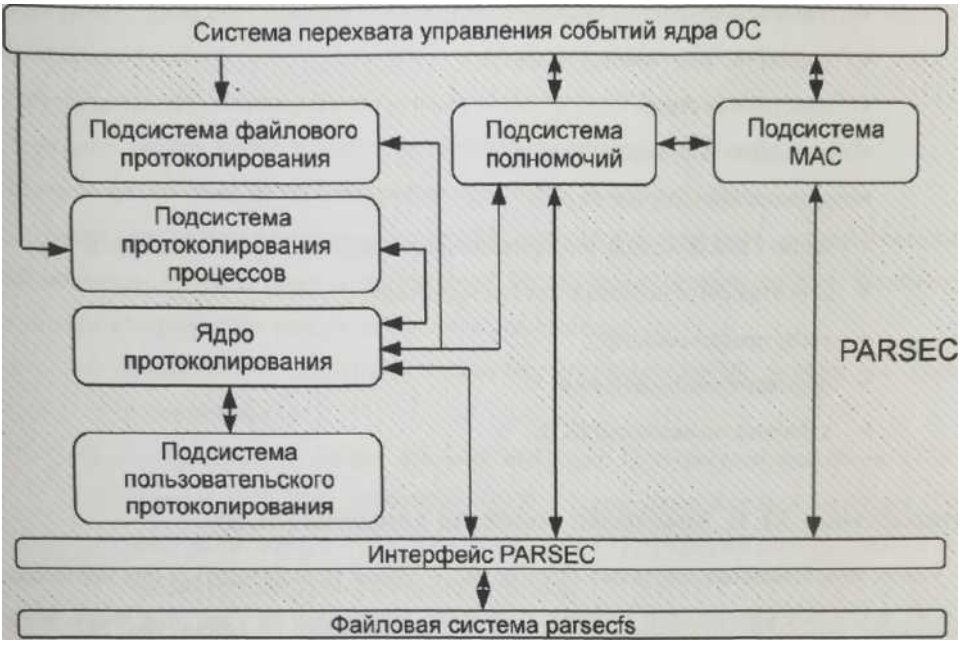
- группа программных серверов, обеспечивающих сервисные функции - программные средства, реализующие запуск процессов в контексте безопасности пользователя (сервисы, осуществляющие обработку запросов пользователей, поступающих из процессов, запущенных в различных мандатных контекстах, и средства инициализации графической сессии пользователя);

- группа «пользовательских» компонентов - программные средства, реализующие функциональные задачи пользователя (офисные средства, Web-браузер и т.д.).

Классификация программных средств приведена ниже.



Логическая схема информационных связей подсистемы безопасности PARSEC показана ниже.



КСЗ (подсистема безопасности PARSEC) предназначен для реализации функций по защите информации от несанкционированных действий (НСД) и предоставления администратору средств управления функциями КСЗ.

В состав КСЗ входят следующие основные подсистемы:

— модуль подсистемы безопасности PARSEC, входящий в состав ядра ОС;
— библиотеки;
— утилиты безопасности;
— подсистема протоколирования (регистрации);
— модули аутентификации;
— графическая подсистема;
— консольный вход в систему;
— средства контроля целостности;
— средства восстановления;
— средства разграничения доступа к подключаемым устройствам.

КСЗ обеспечивает реализацию следующих функций ОС по защите информации от НСД:

- идентификацию и аутентификацию;
- дискреционное разграничение доступа;

- мандатное разграничение доступа;
- очистку памяти;
- изоляцию модулей;
- маркировку документов;
- защиту ввода-вывода информации на отчуждаемый физический носитель;
- сопоставление пользователя с устройством;
- регистрацию событий;
- надежное восстановление;
- контроль целостности КСЗ.

Упражнение 21.1. Комплекс средств защиты (КСЗ)

1. Режим «Расширенные настройки безопасности» может устанавливаться при инсталляции ОС или в уже работающей системе. Проверьте, установлен ли у вас этот пакет, если не установлен, установите его.

   ```
   # aptitude search astra-safeplicy
   ```

   Расширенные настройки безопасности предназначены для определения, настройки и установки общесистемных параметров и их ограничений для следующих подсистем ОС:
   - файловой - ограничивается максимальный размер файла и максимальное число открытых файлов;
   - управления процессами - максимальное число процессов в системе;
   - сетевой — настраивается сетевой фильтр iptables;
   - безопасности — определяется минимальная длина паролей для пользователей в системе, настраиваются режимы запрета создания исполняемых файлов пользователем (режим nochmodx) и безопасного удаления файлов (режим secdel).

2. Запустите утилиту `Управление политикой безопасности`, через меню `Пуск -> Настройки`
3. Выберите в левом окне открывшейся утилиты `Настройки безопасности -> Глобальный профиль`. Попробуйте изменить о настройки (все настройки изучались ранее через командную строку)
4. Выберите в левом окне открывшейся утилиты `Политики паролей -> Глобальная политика`

Попробуйте изменить о настройки (все настройки изучались ранее через командную строку) Выберите в левом окне открывшейся утилиты **Группы,** а затем **Пользователи.** Измените настройки

5. Запустите утилиту **Сетевой фильтр,** через меню **Пуск -> Настройки**

Разрешите работу служб, которые Вы настраивали ранее в упражнениях к курсу Фактически при этом настраивается фильтр сетевых пакетов iptables «Уровень безопасности брандмауэра: низкий, средний или высокий» — предоставляется возможность выбрать уровень ограничений на входящий сетевой трафик. Фактически при этом настраивается фильтр сетевых пакетов iptables:

i. «Низкий» — брандмауэр отключен, разрешен любой трафик по любым интерфейсам;

ii. — «Средний» — запрет для всех входящих SYN-пакетов протокола TCP на порты 0-1023, 2049, 6000-6009, 71000, а также запрет для всех входящих UDP- пакетов на порты 0-1023,2049. Входящий TCP-трафик на порт 22 (SSH) и весь трафик по интерфейсу loopback разрешен;

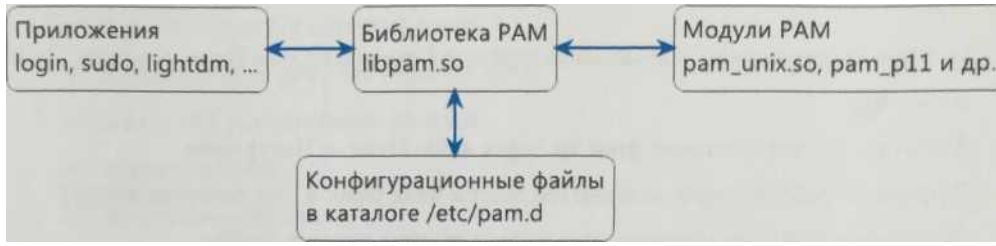
ш. «Высокий» — запрет всего TCP- и UDP-трафика, за исключением трафика по интерфейсу loopback и прохождения UDP-пакетов с адреса сервера имен на порт 52 (domain).

Модуль 22. Идентификация и аутентификация

Функция идентификации и аутентификации пользователей в ОС основывается на использовании механизма PAM. PAM представляют собой набор разделяемых библиотек - «модулей», с помощью которых системный администратор может организовать процедуру аутентификации пользователей прикладными программами. Каждый модуль реализует свой собственный механизм аутентификации. Изменяя набор и порядок следования модулей, можно построить сценарий аутентификации. Подобный подход позволяет изменять процедуру аутентификации без изменения исходного кода и повторного компилирования PAM.

Сценарии аутентификации (т.е. работа этих функций) описываются в конфигурационном файле **/etc/pam.conf** и в ряде конфигурационных файлов, расположенных в каталоге **/etc/pam.d/.** Сама аутентификация выполняется с помощью PAM. Модули располагаются в каталоге **/lib/security** в виде динамически загружаемых объектных файлов.

Общая схема работы PAM



Сильно упрощенная схема аутентификации в приложении, использующем PAM, выглядит следующим образом:

- Приложение инициализирует библиотеку PAM (libpam.so)
- PAM в соответствии с конфигурационным файлом для приложения обращается к требуемым модулям
- Модули выполняют возложенные на них действия
- Приложению возвращается результат операции

Модули PAM классифицируются по типу модуля. Каждый модуль должен выполнять функции хотя бы одного из четырех типов:

- **Модуль аутентификации** используется для аутентификации пользователей или создания и удаления учетных данных.
- **Модуль управления учетными записями** выполняет действия, связанные с доступом, истечением учетных данных или записей, правилами и ограничениями для паролей и т. д.
- **Модуль управления сеансами** используется для создания и завершения сеансов.
- **Модуль управления паролями** выполняет действия, связанные с изменением и обновлением пароля.

PAM обеспечивает различные функциональные возможности, такие как: аутентификация с однократной регистрацией, управление доступом и другие. Их реализация обеспечивается различными модулями:

pam_access обеспечивает управление входом в систему в виде протоколируемой службы при помощи имени пользователя и домена в зависимости от правил, указанных заранее в файле /etc/security/access.conf.

pam_cracklib проверяет пароли на соответствие правилам для паролей.

pam_env sets/unsets устанавливает и сбрасывает переменные среды из файла /etc/security/pam_env_conf.

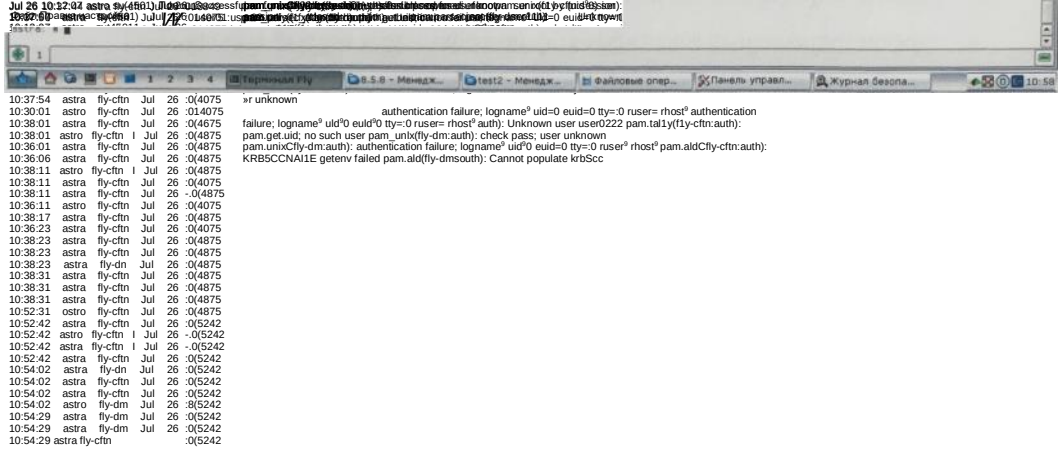
pam jichug выполняет отладку PAM. pam_deny блокирует модули PAM. pam_echo выводит сообщения. раш_схес выполняет внешнюю команду. pam_ftp модуль для анонимного доступа. pam_localuser проверяет наличие имени пользователя в файле /etc/passwd.
pain unix выполняет обычную аутентификацию на основе пароля из файла /etc/passwd.

Проверка реализации идентификации и аутентификации пользователей Для проведения данной проверки под учетными записями пользователей произведены попытки регистрации на АРМ используя (Рисунок 1.):

- незарегистрированный идентификатор пользователя;
- зарегистрированный идентификатор и неверный пароль;
- зарегистрированный идентификатор пользователя и верный пароль.

При проведении проверки получены следующие результаты:

- после ввода пользователем подлинного (зарегистрированного) идентификатора и пароля ему предоставляется доступ к защищаемым ресурсам в соответствии с установленными правилами разграничения доступа;
- после ввода пользователем незарегистрированных идентификаторов и/или паролей ему отказывается в доступе к защищаемым ресурсам.



Рисунок

Проверка фиксации в регистрационном протоколе всех попытки ложной аутентификации

При проведении проверки по данному пункту были выполнены действия, указанные далее по тексту.

неверный пароль.

2. Под учетной записью root выполнена команда userlog. (Рисунок 2.).

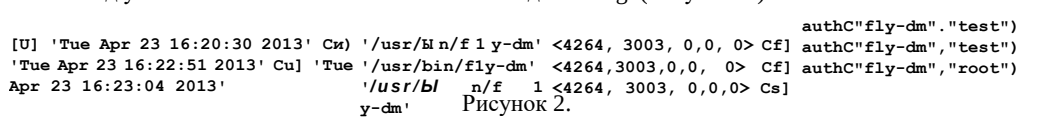


Рисунок 2.

1. Произведена попытка входа на АРМ, используя зарегистрированный идентификатор и

При проведении проверки получены следующие результаты:

- после ввода пользователем подлинного (зарегистрированного) идентификатора и неверного пароля ему отказано в предоставлении доступа к защищаемым ресурсам;
- в регистрационном журнале, вызванном командой userlog, отображаются все попытки неверной аутентификации.

Проверка возможности установки порога на максимальное число неверных попыток аутентификации для локальных пользователей

При проведении проверки по данному пункту были выполнены действия, указанные далее по тексту.

1. Выполнена регистрация на АРМ 1 с использованием учетной записи администратора (root).
2. На рабочем столе выполнен запуск приложения «FLY терминал».
3. Изменен параметр pam.tallyso per user deny=10 на pam.tally_so per_user deny=3 (Рисунок 3).

```
   here are the per-package modules (the "Primary" block)
auth    [success=ignore default=diel      pam_tally,so per_user deny=3|
auth    [success=l default=ignorel        pam_unix.so nul1ok_secure try_first_pass
   here's the fallback if no module succeeds
auth requisite pam_parsec_adv.so deny fail_____________________ -
```

Рисунок 3

4. Произведена попытка аутентификации под учетной записью зарегистрированного пользователя, используя неверный пароль более 3 раз (Рисунок 4).

Рисунок 4

□ после 3 попыток аутентификации с использованием неверного пароля учетная запись пользователя блокируется (при вводе логина и верного пароля появляется сообщение «Ошибка аутентификации»).

Проверка возможности установки порога на максимальное число неверных попыток

аутентификации для пользователей ALD

При проведении проверки по данному пункту были выполнены действия, указанные далее по тексту.

1. Выполнена регистрация на АРМ 1 с использованием учетной записи администратора (root).
2. На рабочем столе выполнен запуск приложения «FLY терминал».
3. Изменен параметр pam.tally_so per user deny=10 на pam.tally_so per_user deny=3 (Рисунок 5).

```
# here are the per-package modules   (the "Primary" block)
auth     [success=ignore default=die ] pam_tally.so per_user deriy=3|
auth     lsuccess=1 default=ignorel       pam_unix.so nul1ok_secure try_first_pass
M here'' s the fallback if no module :succeeds
         requisite pam_parsec_adv. so deny fai1
```

Рисунок 5

4. Произведена попытка аутентификации под учетной записью зарегистрированного пользователя, используя неверный пароль более 3 раз (Рисунок **6**).

Рисунок **6**

При проведении проверки получены следующие результаты:

□ после 3 попыток аутентификации с использованием неверного пароля учетная запись пользователя блокируется (при вводе логина и верного пароля появляется сообщение «Ошибка аутентификации»).

Проверка автоматического инициирования периодической смены паролей пользователями.

При проведении проверки по данному пункту были выполнены действия, указанные далее по тексту.

1. Выполнена регистрация на АРМ 1 с использованием учетной записи администратора (root).
2. На рабочем столе выполнен запуск приложения «FLY терминал».
3. Изменено число дней работоспособности пароля для тестового пользователя test посредством редактирования файла /etc/shadow.
4. Изменено системное время. Выставлено системное время, при котором пароль пользователя test является просроченным.
5. Выполнена попытка входа в систему с учетной записью пользователя test (Рисунок 7).

```
Rstra:~# login Login: test Пароль:
Вам необходимо немедленно сменить пароль (пароль устарел)
Смена пароля для test.
(текущий) пароль UNIX:
НоВый пароль UNIX :
Повторите ВВод нового пароля UNIX :
Последний Вход В систему:Срд Июн 19 11:48:57 MSD 2013на pts/3
Linux Astra 2.6.34-3-generic #10astra8 SMP Thu Mar 21 15:53:47 HSK 2013
   x86_64
```

Рисунок 7

При проведении проверки получены следующие результаты:

После прохождения аутентификации пользователем test ОС СН автоматически генерирует запрос о смене пароля пользователя.

Упражнение 22. Идентификация и аутентификация

1. Посмотрите список стандартных модулей в системе:

   % find / -name pam_*.so

2. Для примера, посмотрите на абстрактный файл конфигурации для приложения login.

   Попробуйте поменять параметры и посмотрите изменения

   # PAM configuration for login

auth requisite pam_securetty.so auth required pam_nologin.so auth required pam_env.so auth required pam_unix.so nullok account required pam_unix.so session required pam_unix.so session optional pam_lastlog.so

password required pam_unix.so nullok obscure min=:4 max=8 Каждая строчка конфига

записывается в виде

<тип модуля> <управляющий флаг> <путь к библиотеке> <параметры>
Тип модуля соответствует обозначениям самих модулей (т.е.

auth/account/session/passwd)

Управляющий флаг указывает критичность модуля для успешного выполнения операции. Флаг может принимать следующие значения: requisite (необходимый), required (требуемый), sufficient (достаточный) и optional (необязательный).

Путь к библиотеке задает путь до файла модуля. По умолчанию они ищутся в /lib/security/

Параметры задают список аргументов, которые будут переданы модулю.

Таким образом, мы получаем стек модулей, каждый из которых выполняет свое действие. PAM при этом разбирает стек как и положено - сверху вниз. В соответствии с управляющим флагом задаются следующие требования к успешности операции: requisite (необходимый): если модуль стека вернет отрицательный ответ, то запрос сразу же отвергается. Другие модули при этом не будут выполнены.

required (требуемый): если один или несколько модулей стека вернут отрицательный ответ, все остальные модули будут выполнены, но запрос приложения будет отвергнут, sufficient (достаточный): если модуль помечен как достаточный и перед ним ни один из необходимых или достаточных модулей не возвратил отрицательного ответа, то все оставшиеся модули в стеке игнорируются, и возвращается положительный ответ, optional (дополнительный): если в стеке нет требуемых модулей, и если ни один из достаточных модулей не возвратил положительного ответа, то хотя бы один из дополнительных модулей приложения или службы должен вернуть положительный ответ

Модуль 23. Мандатное разграничение доступа

Механизм контроля мандатного разграничения доступа реализован, как и механизм дискреционного разграничения доступа, в ядре ОС (модуль parsec.ko). При этом принятие решения о запрете или разрешении доступа субъекта к объекту принимается на основе типа операции (чтение/запись/исполнение), мандатного контекста безопасности субъекта и мандатной метки объекта. При принятии решения учитывается наличие PARSEC-привилегий субъекта.

1) уровень ьо меньше уровня и (ьо<Ы), если численное значение L0 меньше численного значения L1;
2) уровень L0 равен уровню Ll (LO—L1), если численные значения L0 и Ы совпадают;
3) уровень целостности iLO меньше уровня iLl (iLOciLl), если численное значение 1L0 меньше численного значения 1Ы;
4) уровень целостности iLO равен уровню целостности iLl (iL0==iLl), если численные значения iLO и iLl совпадают;
5) категории CO меньше категорий C1 (C0<C1), если все биты набора CO являются подмножеством набора бит C1;
6) категории CO равны категориям C1 (C0«»C1), если значения CO и C1 совпадают;
7) операция записи разрешена, если L0=L1, iL0>*=iLl и co=ci;
8) операция чтения разрешена, если L0>=L1, C0>=C1, ViLO, ViLl;
9) операция исполнения разрешена, если L0>=L1 и C0>=C1, ViLO, ViLl.

В настоящий момент в системе могут использоваться два уровня целостности: высокий (hi) - значение 1; низкий (low) - значение 0.

При анализе критических маршрутов было установлено, что с каждым субъектом и объектом связаны мандатный контекст безопасности и мандатная метка, соответственно и механизм мандатного разграничения доступа затрагивает следующие подсистемы:

- механизмы IPC;
- стек TCP/IP;
- ФС Ext2/Ext3;
- сетевая ФС CIFS;
- ФС proc, tmpfs.

Модуль 31. Основы работы с СУБД PostgreSQL

PostgreSQL - свободная объектно-реляционная система управления базами данных (СУБД). PostgreSQL базируется на языке SQL и поддерживает многие из возможностей стандарта SQL:2003 (ISOAEC 9075).

Основные плюсы PostgreSQL:

- поддержка БД практически неограниченного размера;
- мощные и надёжные механизмы транзакций и репликации;
- расширяемая система встроенных языков программирования: в стандартной поставке поддерживаются PL/pgSQL, PL/Perl, PL/Python и PL/Tcl; дополнительно можно использовать PL/Java, PL/PHP, PL/Py, PL/R, PL/Ruby, PL/Scheme и PL/sh, а также имеется поддержка загрузки C-совместимых модулей;
- наследование;
- легкая расширяемость.

Входящий в состав СУБД PostgreSQL набор программных средств можно разделить на следующие классы:

— управление БД;

— выполнение запросов пользователя;

— оптимизация производительности;

— обеспечение средств копирования и восстановления.

Работа с СУБД требует установки соединения с сервером БД, что при использовании клиентских утилит командной строки обеспечивается заданием свойств соединения с помощью аргументов (ОПЦИИ) командной строки:

-h, —host=HOSTNAME - указывает имя сервера БД или каталог сокетов UNIX, если начинается с символа «/»

-p, —port=PORT - указывает порт сервера БД или расширение имени сокета UNIX, по которым сервер принимает соединения

-и, —username=usERNAME - указывает имя пользователя для установки соединения -w, **—no-password** - подавление запроса пароля пользователя. В случае, когда установка соединения с сервером требует ввода пароля, а пароль недоступен, например из файла .pgpass, попытка установки соединения завершается ошибкой. Опция полезна при выполнении пакетов заданий или скриптов

-W, —password - принудительный запрос пароля при установке соединения.

При отсутствии перечисленных аргументов используются переменные окружения (PGDATABASE, PGHOST, PGPORT, PGUSER), определяющие параметры соединения по умолчанию.

Управление базами данных: создание и удаление БД, управление пользователями и процедурными языками. Создание кластера БД состоит из создания каталогов для хранения данных БД, создания разделяемых таблиц системного каталога (таблиц, относящихся ко всему кластеру БД, а не к конкретной БД), и создания БД template 1 и postgres. При создании в дальнейшем новых БД в них копируется содержимое БД template 1. (все, что установлено в БД template 1, автоматически будет скопировано в каждую создаваемую в дальнейшем БД.) БД postgres является БД по умолчанию для использования пользователями, утилитами и сторонними приложениями. Создание кластера выполняется администратором на сервере с помощью утилиты **initdb.**

Создание и удаление баз данных. Для создания новой БД используется утилита **createdb,** для удаления — утилита **dropdb.** По умолчанию владельцем новой БД становится пользователь, выполняющий команду. В тоже время в качестве владельца новой БД может быть указан другой пользователь с помощью опции -o, если выполняющий команду пользователь обладает соответствующими привилегиями. При этом удаление может выполнить только суперпользователь или владелец БД.

Обобщенный способ вызова заключается в передаче опций и имени БД. При этом используются правила установки соединения, рассмотренные выше: **createdb [ОПЦИИ]... [БАЗА_ДАННЫХ] [ОПИСАНИЕ] dropdb [ОПЦИИ]... [БАЗА_ДАННЫХ]**

Управление пользователями. В СУБД PostgreSQL для управления правами на доступ к БД используется концепция ролей. Под ролью понимается пользователь или группа пользователей БД, в зависимости от параметров роли. Роли могут являться владельцами объектов БД (например, таблиц) и могут назначать привилегии на управление объектами для других ролей, имеющих доступ к данным объектам. Кроме того, существует возможность предоставления членства в роли для другой роли, что позволяет членам роли использовать привилегии, назначенные роли, членами которой они являются. Таким образом, концепция ролей объединяет концепции «пользователи» и «группы».

Корректная работа с СУБД предполагает использование механизма ЕПП, что подразумевает использование в качестве пользователей СУБД пользователей домена ЕПП.

Для создания нового пользователя или роли используется утилита **createuser,** для удаления — **dropuser.** Только суперпользователи и пользователи с привилегией

CREATEROLE могут создавать и удалять пользователей и роли. Удалять суперпользователя может только суперпользователь.

Синтаксис:

createuser **[ОПЦИИ]... [РОЛЬ]**

dropuser **[ОПЦИИ]... [РОЛЬ]**

Использование процедурных языков. СУБД PostgreSQL предоставляет пользователям возможность создавать хранимые процедуры (функции) и триггеры для обработки данных, хранящихся в БД. Для этого могут использоваться следующие процедурные языки: PL/Perl, PL/pgSQL, PL/Python и PL/Tcl.

Для возможности использования конкретного процедурного языка его необходимо установить в конкретную БД.

Для установки поддержки процедурного языка в БД используется утилита createlang, для удаления поддержки языка из БД — droplang.

Синтаксис:

createlang [ОПЦИИ] . . . ЯЗЫК [БАЗА_ДАННЫХ] droplang [ОПЦИИ]... ЯЗЫК [БАЗА_ДАННЫХ]

Несмотря на то, что поддержка процедурного языка может быть выполнена непосредственно некоторыми SQL-командами (например, DROP LANGUAGE), рекомендуется использовать данные утилиты, т. к. они осуществляют необходимые проверки.

Выполнение запросов. Взаимодействие пользователя с СУБД в основном осуществляется с помощью прикладного ПО, созданного для решения конкретных прикладных задач.

В то же время в состав СУБД входят средства интерактивного взаимодействия с пользователем. Для этого предлагается консольная утилита psql (интерактивный терминал) и утилита администрирования с визуальным пользовательским интерфейсом fly-admin- postgres.

Интерактивный терминал. Утилита psql является интерактивным клиентом PostgreSQL и позволяет интерактивно набирать запросы, отправлять их серверу и получать результаты. Так же ввод может осуществляться из файла. В дополнение утилита поддерживает метакоманды и некоторые возможности командной оболочки для облегчения создания скриптов и автоматизации широкого круга задач. Синтаксис: psql [ОПЦИИ]... [БАЗА_ДАННЫХ [ПОЛЬЗОВАТЕЛЬ]]

Утилита psql является клиентским приложением PostgreSQL. Для установки соединения требуется указание БД, имя и номер порта сервера и имя пользователя, под которым устанавливается соединение. Существует возможность указать эти параметры с помощью аргументов командной строки -d, -h, -p и -и, соответственно. Если аргумент не

соответствует ни одной из опций, он воспринимается как имя БД (или имя пользователя, если имя БД уже было получено).

Если значения по умолчанию не верны, существует возможность их переопределения установкой переменных окружения PGDATABASE, PGHOST, PGPORT и/или PGUSER в соответствующие значения. Так же удобно использовать файл ~/ .pgpass для устранения необходимости регулярного ввода пароля.

Альтернативным путем задания параметров соединения является строка соединения, используемая вместо имени БД. Этот механизм предоставляет широкие возможности по управлению установкой соединения. Например:

```
$ psql "service=myservice sslmode=require"
```

При невозможности установки соединения в силу тех или иных причин (например, недостаток прав доступа, сервер не запущен, и т.п.) утилита psql возвращает ошибку и завершает работу.

При нормальном функционировании psql выводит приглашение с именем БД, с которой в настоящее время установлено соединение, за которым следует =>. Например:

```
$ psql testdb
psql (x.x.O)
Наберите "help" для справки.
testcLb=>
```

После приглашения пользователь имеет возможность ввода SQL-команд. Обычно введенный запрос отсылается серверу после ввода завершающего символа «;». Перевод строки не завершает команду. Команда может быть записана в несколько строк для лучшего восприятия. Если команда была отослана серверу и выполнена без ошибок, на экран выводится результат ее выполнения.

В случае ввода строки, начинающейся с не заключенного в кавычки символа \, она воспринимается как метакоманда и обрабатывается непосредственно утилитой psql. Подобные команды делают утилиту более удобной для администрирования и создания скриптов.

В процессе запуска psql пытается прочитать и выполнить команды из общесистемного файла psqlrc и пользовательского файла ~/ .psqlrc (см. . . . /share/psqlrc. sample для примера общесистемного файла). Он может быть использован для настройки параметров клиента или сервера (используя команды set или SET) .

История команд сохраняется в файле ~/ .psql_history.

Примеры: **1**. Разбиение команды при вводе в несколько строк (следует обратить внимание на изменение приглашения при этом)

```
^escaJD=> CREATE TABLE my_table ( testdb(> first
integer not null default 0, testdb(> second
text) testdb-> ;
CREATE TABLE
```

2. Просмотр определения таблицы `testdb=> \d my_table`

```
                Таблица "my_table"
  Атрибут | Тип |             Модификатор
----------------+------------+----------------------------
 first       |  integer |  not null default 0
 second      |     text |
```

3. Просмотр содержимого таблицы

```
peterdlocalhost testdb=> SELECT * FROM my_table; first
 | second
        -----------+-----------
              1  | one
              2  | two
              3  | three
              4  | four
 (4 строк)
```

Утилита администрирования с визуальным пользовательским интерфейсом. Утилита `fly-admin-postgres` предназначена для администрирования БД СУБД PostgreSQL и **позволяет:**

- просматривать иерархическую структуру БД;
- удаленно редактировать конфигурационные файлы СУБД и загружать их на сервер;
- управлять пользователями и группами СУБД;
- управлять дискреционным и мандатным доступом к объектам;
- выполнять SQL-запросы;
- создавать, изменять и удалять различные объекты БД;
- просматривать и редактировать данные таблиц.

Системные операции. Существует необходимость осуществлять ряд системных операций как для оптимизации работы СУБД, так и в качестве регламентных работ по обеспечению отказоустойчивости и возможности восстановления после сбоев.

Оптимизация баз данных. С целью оптимизации работы СУБД для увеличения производительности используются как архитектурные способы при разработке конкретной

схемы БД, так и применение различных способов индексирования информации. При этом может возникать необходимость перестройки индексов в процессе изменения большого количества данных.

Для пересоздания индексов в БД PostgreSQL используется утилита reindexdb, а для кластеризации (оптимизации индексов) ранее кластеризованных таблиц в БД используется утилита clusterdb. Она находит таблицы, которые были ранее кластеризованы, и кластеризует их заново по тем же индексам, которые были указаны до этого. Таблицы, которые до этого не были кластеризованы, не затрагиваются.

Так же существует понятие сборки мусора, т. е. очистки таблиц от ранее удаленных записей.

Для сборки «мусора» и сбора статистики, необходимой для работы оптимизатора запросов,

БД PostgreSQL используется утилита vacuumdb.

reindexdb [ОПЦИИ]... [БАЗА_ДАННЫХ]

clusterdb [ОПЦИИ]... [БАЗА_ДАННЫХ]

vacuumdb [ОПЦИИ]... [БАЗА_ДАННЫХ]

Резервное копирование и восстановление. Для создания резервной копии БД в виде файла в текстовом или других форматах используется утилита pg_dump. Утилита создает согласованную копию, даже если БД используется, при этом доступ к ней других пользователей (как читающих, так и пишущих) не блокируется. Резервная копия может создаваться в виде скрипта или форматах упакованного файла. Скрипт резервной копии представляет собой текст, содержащий последовательность SQL-команд, необходимых для воссоздания БД до состояния, в котором она была сохранена. Для восстановления из скрипта он подается на вход утилиты psql.

Альтернативные форматы упакованного файла могут быть использованы утилитой pg_restore для пересоздания БД. Они позволяют выбирать, что именно восстанавливать, или даже менять порядок элементов перед восстановлением.

Утилита pg_dump предоставляет гибкий механизм архивирования и переноса при использовании одного из форматов упаковки файла и комбинирования с pg_restore. Например, может быть выполнено создание резервной копии всей БД, после чего может быть использована утилита pg_restore для просмотра и/или выбора частей резервной копии для восстановления. pg_dump [ОПЦИИ]... [БАЗА_ДАННЫХ]

Созданный утилитой pg_dump архивный файл не содержит информации о статистике, которую использует оптимизатор запросов, таким образом рекомендуется выполнять команду **ANALYZE** после восстановления из резервной копии для достижения лучшей

производительности. Архивный файл так же не содержит команд `ALTER DATABASE ...` SET, эти установки архивируются утилитой pg_dumpall вместе с информацией о пользователях и других глобальных параметров установки.

Примеры:

Создание резервной копии БД mydb в виде SQL-скрипта $ pg_dump mydb > db.sql Загрузка подобного скрипта в новую БД newdb $ psql -d newdb -f db.sql

Создание резервной копии всех схем, начинающихся с east или west и заканчивающихся на gsm, исключая все схемы, содержащие слово test

$ pg_dump -n 'east*gsm' -n 'west*gsm' -N '*test*' mydb > db.sql Утилита pg_dump создает за раз дамп только одной БД, при этом информация о ролях или табличных пространствах не сохраняется (эта информация относится ко всему кластеру, а не к каждой отдельной БД). Для обеспечения удобного сохранения дампа всего содержимого кластера предназначена утилита pg_dumpall. Она создает резервную копию каждой БД кластера, а также сохраняет информацию о кластере, такую как определения ролей и табличных пространств.

В качестве стартовой БД возможно указание любого имени, но при загрузке данных в пустой кластер, как правило требуется указание postgres. При восстановлении дампа, полученного с помощью pg_dumpall, необходимо обладать правами суперпользователя БД, поскольку они требуются для восстановления информации о ролях и табличных пространствах. При использовании табличных пространств следует убедиться, что пути табличных пространств из дампа подходили для новой конфигурации.

Утилита pg_dumpall сначала выполняет команды для воссоздания ролей, табличных пространств и пустых БД, и лишь затем запускает pg_dump для каждой БД. Это означает, что хотя каждая БД будет обладать внутренней целостностью, «снимки» различных БД могут не быть полностью синхронизированы.

Примеры:

. Создание резервной копии всех БД $ pg_dumpall > db.out Восстановление сохраненных БД $ psql -f db.out postgres

Не имеет значения, с какой БД было осуществлено соединение, т.к. созданный с помощью pg_dumpall скрипт содержит соответствующие команды для создания и соединения для указанных БД.

Для восстановления БД из архивов, созданных утилитой **pg_dump** в одном из нетекстовых форматов, предназначена утилита **pg_restore**. Она осуществляет команды, необходимые для воссоздания БД до состояния на момент времени создания резервной копии. Архивные файлы так же позволяют выбирать с помощью утилиты **pg_restore**, что именно восстанавливать, и даже менять порядок восстанавливаемых элементов.

Примеры:

1. Создание резервной копии БД mydb в формате «custom»

$ pg_dump -Fc mydb > db.dump

3. Удаление БД и воссоздание ее из резервной копии **$**

dropdb mydb

$ pg_restore -C -d postgres db.dump

БД, указанной в опции **-d,** может быть любая **БД** кластера. **pg_restore** используется только для выполнения команды CREATE DATABASE. С опцией -с данные всегда восстанавливаются в **БД,** указанную в резервной копии.

4. Загрузка резервной копии в новую БД newdb **$**

createdb -T templateO newdb

$ pg_restore -d newdb db.dump

Необходимо отметить, что опция -с не была использована, вместо этого осуществлялось подключение непосредственно к восстанавливаемой БД. Новая БД была создана из шаблона templateO, а не template 1, для обеспечения первоначальной чистоты базы.

Совместная работа сервера СУБД PostgreSQL с ALD

Для работы СУБД PostgreSQL с ALD необходимо выполнение следующих условий:

1) наличие в системах, на которых функционируют сервер и клиенты СУБД PostgreSQL, установленного пакета клиента ALD — **aid-client;**

2) разрешение имен должно быть настроено таким образом, чтобы имя системы разрешалось, в первую очередь, как полное имя (например, **myserver. example. ru);**

3) клиент ALD должен быть настроен на используемый ALD домен.

Для проведения операций по настройке ALD и администрированию Kerberos необходимо знание паролей администраторов ALD и Kerberos.

Для обеспечения совместной работы сервера СУБД PostgreSQL с ALD необходимо, чтобы сервер СУБД PostgreSQL функционировал как сервис Kerberos. Выполнение данного условия требует наличия в БД Kerberos принципала для сервера СУБД PostgreSQL, имя которого задается в формате: servicename/hostname@realm где имя сервиса servicename

соответствует имени учетной записи пользователя, от которой осуществляется функционирование сервера СУБД PostgreSQL (по умолчанию — postgres), и указывается в конфигурационном файле сервера PostgreSQL как значение параметра krb srvname. В качестве значения hostname указывается полное доменное имя системы, на которой функционирует сервер СУБД PostgreSQL, а в качестве значения realm — имя домена ALD.

Таким образом, для обеспечения совместной работы сервера СУБД PostgreSQL с ALD необходимо:

1) создать в БД ALD с помощью утилиты администрирования ALD принципала, соответствующего устанавливаемому серверу PostgreSQL. Принципал создается с автоматически сгенерированным случайным ключом:

```
#    ald-admin service-add postgres/server.my_domain.org
```

2) ввести созданного принципала в группу сервисов mac, используя следующую команду:

```
#    ald-admin sgroup-svc-add postgres/server .my_domain.org —sgroup=mac
```

3) создать файл ключа Kerberos для сервера СУБД PostgreSQL с помощью утилиты администрирования ALD aid-client, используя следующую команду (пример приведен для кластера БД по умолчанию):

```
#    aid-client update-svc-keytab postgres/server .my_domain.org \
                    —ktfile="/etc/postgresql/x.x/main/krb5.keytab"
```

Полученный файл должен быть доступен серверу СУБД PostgreSQL по пути, указанному в конфигурационном параметре krb_server_keyfile (в данном случае — /etc/postgresql/x.x/main/krb5. keytab). Права доступа к этому файлу должны позволять читать его пользователю, от имени которого работает сервер СУБД PostgreSQL (как правило, владельцем файла назначается пользователь postgres);

4) сменить владельца, полученного на предыдущем шаге, файла krb5. keytab на пользователя posgtres, выполнив следующую команду:

```
#    chown postgres /etc/postgresql/x.x/main/krb5.keytab
```

5) задать в конфигурационном файле сервера СУБД PostgreSQL /etc/postgreslq/x.x/main/postgresql. conf для приведенных ниже параметров соответствующие значения:

```
krb_server_keyf ile = ' /etc/postgresql/x. x/main/krb5. keytab' krb srvname = fpostgres'
```

6) указать для внешних соединений в конфигурационном файле сервера СУБД PostgreSQL /etc/postgreslq/x 1 x/main/pg_hba. conf метод аутентификации gss.

Пример

```
host all all 192.168.32.0/24 gss
```

Общие условия, при которых обеспечивается совместное функционирование клиентов СУБД PostgreSQL с ALD. Кроме того, сервер СУБД PostgreSQL должен быть также настроен соответствующим образом. Для настройки клиента СУБД PostgreSQL необходимо:

1) создать в БД ALD учетную запись пользователя, зарегистрированного в СУБД PostgreSQL (например, pgusemame). Дополнительная информация приведена в руководстве man на ALD;

2) задать в качестве значения параметра соединения `krbsrvname` имя сервиса servicename, используемое при создании принципала сервера СУБД PostgreSQL. Имя принципала сервера СУБД PostgreSQL задается в формате: `servicename/hostname@realm` где `servicename` — обычно имя учетной записи сервера СУБД PostgreSQL, используемое при создании принципала сервера СУБД PostgreSQL (по умолчанию — `postgres)`, а `hostname` — полное доменное имя системы, на которой функционирует сервер СУБД PostgreSQL.

Мандатное разграничение доступа СУБД PostgreSQL. В Astra Linux доступно расширение команд SQL при мандатном разграничении в СУБД. К примеру, при необходимости обеспечения сквозной аутентификации из скриптов, запускаемых на WEB-сервере, которые должны взаимодействовать с другими службами в режиме ЕПП, например, с защищенным сервером СУБД, в конфигурационном файле для виртуального хоста следует дополнительно прописать:

KrbSaveCredentials on

В настройках вашего браузера необходимо задать в качестве значений параметра `network.negotiate-auth.delegation-uris`, маски доменов которым можно передавать данные для сквозной аутентификации.