

-МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
Факультет математики, информационных и авиационных технологий
Кафедра «Телекоммуникационных технологий и сетей»

Чичев А.А., Чекал Е.Г.

**АРХИТЕКТУРА И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ
ИНФОКОММУНИКАЦИОННЫХ УСТРОЙСТВ**

Часть 2. Методические указания
к выполнению лабораторных работ

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
Факультет математики, информационных и авиационных технологий
Кафедра «Телекоммуникационных технологий и сетей»

Чичев А.А., Чекал Е.Г.

**АРХИТЕКТУРА И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ
ИНФОКОММУНИКАЦИОННЫХ УСТРОЙСТВ**

Часть 2. Методические указания
к выполнению лабораторных работ

Ульяновск
2015

УДК 621.39:004.7(0758)
ББК 32.883я73+32.971.35я73
Ч-72

*Печатается по решению Ученого совета
факультета математики, информационных и авиационных технологий
Ульяновского государственного университета
(протокол № от)*

Рецензенты:

Заведующий кафедрой телекоммуникационных технологий и сетей
ФГБОУ ВО «УлГУ», доктор технических наук, профессор А.А. Смагин

Заведующий кафедрой информатики ФГБОУ ВПО «УлГПУ им. И.Н. Ульянова» доктор
педагогических наук, кандидат технических наук, профессор В.Г. Шубович

Чичев А.А.

Ч-72

Архитектура и программное обеспечение инфокоммуникационных устройств. Часть 2. Методические указания к выполнению лабораторных работ.
/ Чичев А.А., Чекал Е.Г. – Ульяновск: УлГУ, 2015. – с.

Учебно-методическое пособие составлено в соответствии с программой дисциплины «Архитектура и программное обеспечение сетевых инфокоммуникационных устройств», и предусматривает подготовку бакалавров по направлению 11.03.02 «Инфокоммуникационные технологии и системы». Может использоваться студентами родственных специальностей и направлений.

Пособие состоит из двух частей. В первой части пособия описываются основы организации сетей, архитектура и программное обеспечение сетевых инфокоммуникационных устройств. Во второй части пособия приведены задания, методические указания и дополнительный справочный материал к лабораторным работам.

Пособие предназначено для практического руководства при проведении преподавателями лабораторных занятий и выполнении заданий студентами указанного направления всех форм обучения.

УДК 621.39:004.7(0758)
ББК 32.883я73+32.971.35я73

© Ульяновский государственный университет, 2015
© Чичев А.А., Чекал Е.Г., 2015

ОГЛАВЛЕНИЕ

ПРЕДИСЛОВИЕ КО ВТОРОЙ ЧАСТИ ПОСОБИЯ	5
1. ОБЩИЕ ТРЕБОВАНИЯ К СДАЧЕ И ОФОРМЛЕНИЮ ЛАБОРАТОРНЫХ РАБОТ	6
2. ЛАБОРАТОРНЫЕ РАБОТЫ: ВВЕДЕНИЕ В ОПЕРАЦИОННЫЕ СИСТЕМЫ	7
2.1. Лабораторная работа №1. Создание пользователя	7
2.2. Лабораторная работа №2. Терминал: файловый менеджер mc	12
2.3. Лабораторная работа №3. Терминал: команды работы с файлами	29
2.4. Лабораторная работа №4. Терминал: переменные окружения	35
2.5. Лабораторная работа №5. Терминал: редактор vi (vim)	45
2.6. Лабораторная работа №6. Терминал: атрибуты файлов	51
2.7. Лабораторная работа №7. Терминал: управление процессами	55
2.8. Лабораторная работа №8. Установка Linux на flash-носитель	64
2.9. Лабораторная работа №9. BASH-программирование	72
2.10. Лабораторная работа № 10. Технология виртуализации: Wine	80
2.11. Лабораторная работа № 11. Технология виртуализации: VirtualBox	91
2.12. Лабораторная работа №12. Установка 4-х ОС	100
2.13. Лабораторная работа №13. Программирование: работа с процессами	108
2.14. Лабораторная работа №14. Программирование: учет пользователей ОС	114
3. ЛАБОРАТОРНЫЕ РАБОТЫ: ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ИНФОКОММУНИКАЦИОННЫХ УСТРОЙСТВ	121
3.1. Лабораторная работа № 1. Настройка локальной вычислительной сети в условиях отсутствия DNS	121
3.2. Лабораторная работа № 2. Контроль активности в локальной вычислительной сети	128
3.3. Лабораторная работа № 3. Работа с СУБД MySQL	138
3.4. Лабораторная работа № 4. Сохранение и восстановление базы данных под управлением СУБД MySQL	144
3.5. Лабораторная работа № 5. Удаленный терминальный доступ к СУБД MySQL	150
3.6. Лабораторная работа № 6. Программирование: создание базы данных	153
3.7. Лабораторная работа № 7. Программирование: управление учетными записями	163
3.8. Лабораторная работа № 8. Анализ вычислительной сети организации	165
3.9. Лабораторная работа № 9. Установка и конфигурирование файлового сервера рабочей группы	169
3.10. Лабораторная работа № 10. Установка и запуск веб-сервера Apache	182
3.11. Лабораторная работа № 11. Удалённая работа с X сервером	192
3.12. Лабораторная работа № 12. Установка и удалённая работа с почтовым сервером	196
3.13. Лабораторная работа № 13. Трансляция с веб-камеры на сайт в Интернет	204
3.14. Лабораторная работа № 14. Установка DNS для Intranet фирмы	211
3.15. Лабораторная работа № 15. Корпоративная сеть фирмы и её защита	213
3.16. Лабораторная работа № 16. Проект структурированной кабельной системы организации	228
ИСПОЛЬЗОВАННАЯ И РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА	237

ПРЕДИСЛОВИЕ КО ВТОРОЙ ЧАСТИ ПОСОБИЯ

В части второй пособия помимо лабораторных работ, имеющих отношение к дисциплине, также включены работы по операционным системам. Это обусловлено тем, что для понимания работы и конфигурирования инфокоммуникационных устройств знание особенностей установки, настройки и работы с операционными системами совершенно необходимы, поскольку современные инфокоммуникационные устройства давно вышли из младенческого возраста (когда они реализовывались почти полностью аппаратно) и в настоящее время, как правило, представляют собой почти полнофункциональные ЭВМ с операционными системами. Причём, в качестве операционных систем в них чаще всего используются unix-подобные ОС, например, ios в устройствах фирмы Cisco или linux в устройствах фирмы D-Link.

В силу этой специфики необходимо уже в самом начале обучения оценить (и при необходимости - повысить) степень знакомства студентов с основами unix-овых операционных систем (прежде всего с их интерфейсом и работой с ОС).

В второй части пособия приводится количество лабораторных работ превышающее возможность их выполнения в рамках планируемого одного учебного семестра. Это позволяет делать выборки работ, варьируя их по годам. Все приведённые лабораторные работы прошли апробацию в ходе изучения учебных дисциплин со студентами факультета математики, информационных и авиационных технологий УлГУ.

В первой части пособия приводятся сведения по архитектуре и программному обеспечению сетевых инфокоммуникационных устройств, по основам организации сетей, в частности, рассматривается: архитектура концентратора, моста, коммутатора, маршрутизатора, модема, сетевой карты; состав программного обеспечения роутеров типа Cisco, D-Link, конфигурационные файлы и скрипты; базовая модель взаимодействия открытых систем, стеки сетевых протоколов, взаимодействие процессов и др.

1. ОБЩИЕ ТРЕБОВАНИЯ К СДАЧЕ И ОФОРМЛЕНИЮ ЛАБОРАТОРНЫХ РАБОТ

1.1. Лабораторные работы предназначены для получения практических навыков работы с программным обеспечением инфокоммуникационных устройств.

К выполняемым лабораторным работам предъявляются следующие требования:

1) Работа выполняется самостоятельно и индивидуально по выбранной теме в полном объёме.

2) Не разрешается выполнение одного и того же задания по одной теме более чем одному человеку.

3) По каждой лабораторной работе оформляется и сдаётся отчёт преподавателю.

4) Работа выполняется самостоятельно в произвольное время и сдаётся в строго оговоренные сроки только в лаборатории в часы занятий.

5) Выполнение лабораторной работы предполагает достаточно подробное изучение и правдоподобное отражение предметной области.

6) Для проверки полноты усвоения материала и самостоятельности выполнения работы преподаватель может задать дополнительные вопросы и предложить выполнить дополнительные задания.

7) Лабораторные работы выполняются в операционной среде, используемой в лаборатории «ОС и системное программирование» кафедры ИТ. Допускается использование других операционных сред, но в этом случае студентом самостоятельно должны решаться проблемы совместимости.

1.2. Отчёт должен содержать следующие разделы:

- титульный лист;

- введение: формулировка темы (то есть, формулировка своей работы);

- основная часть отчёта (содержание этой части поясняется отдельно для каждой лабораторной работы).

Параметры страниц А4, поля 30-10-20-20 мм, междустрочный интервал «точно»=0,60см, шрифт DejaVu Serif или Liberation Serif 12 пунктов.

1.3. Отчёт сдаётся в электронном и бумажном виде. По мере сдачи лабораторных работ все отчеты выкладываются на личном сайте в Интернет.

2. ЛАБОРАТОРНЫЕ РАБОТЫ: ВВЕДЕНИЕ В ОПЕРАЦИОННЫЕ СИСТЕМЫ

Лабораторная работа № 1

Тема: СОЗДАНИЕ ПОЛЬЗОВАТЕЛЯ

Цель: Научиться создавать учётные записи пользователей

Задание:

Для создания учётных записей пользователей выполните следующие действия.

1. Создать пользователя с использованием графической оболочки.
 - 1.1. Использовать команды меню «Центр управления системой» → «Пользователи» → «Локальные учётные записи».
 - 1.2. Прочесть справку.
 - 1.3. Логин определить следующим образом:

Первая буква (**малая латинская буква**):

№ п/п	Название группы студентов	Первая буква логина
1	МОАИС-11	z
2	ПМ-11	y
3	ПМ-12	x
4	ИС-21	w
5	ИТСС-21	v
6	КБ-21	k
7	ИТСС-31	u
8	МОАИС-31	t
9	КБ-31	s
10	ИТСС-41	r
11	ИТСС-61	q
12	ПРИ-11у	p
13	ПРИ-12у	o

14	ИС-21 (ИДО)	n
15		m
16		l

Вторая, третья и четвёртая буквы — фио (**малые латинские буквы**).

1.4. Ввести новый логин в поле «Новая учётная запись». Нажать клавишу «Создать». Логин появится в списке слева (в окне учётных записей).

1.5. Выделить Var'ом созданную учётную запись в списке (в окне слева). Поставить галочку «входит в группу администраторов». Пароль придумать самостоятельно, записать на промокашке и сохранить. Ввести пароль в поля ввода пароля (дважды). Нажать клавишу «Применить».

2. В дальнейшем работать только под своей учётной записью.

Порядок сдачи лабораторной работы:

1. Создать пользователя.
2. Продемонстрировать вход/выход с созданным логином.
3. Создать ссылку на программу «Терминал (Консоль)» на рабочем столе.
4. Запустить программу Терминал.
5. Продемонстрировать возможность выполнения команды su в Терминале.
6. Скопировать на рабочий стол каталог с заданиями на лабораторные работы.
7. Открыть файловым менеджером каталог с заданиями, найти задание на эту лабораторную.
8. Открыть задание на работу и громко прочесть пункт 2 задания.
9. Представить отчёт.

Общие требования к отчёту указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчёт должен содержать следующую информацию:

- а) задание на работу;
- б) описание процесса создания учётной записи пользователя в графическом режиме.

Дополнительная справочная информация

"Учётная карточка" (user account)

Linux, как Unix-подобная ОС, изначально создавалась как система многопользовательская. Это значит, что в операционной системе:

- разграничиваются права различных пользователей, то есть, какие файлы они могут читать/писать/изменять, какие программы запускать и т. п.; это требование реализуется посредством реализации в операционной системе дискреционного метода разграничения доступа;

- даётся возможность каждому пользователю создавать свою "среду обитания" (environment), то есть, иметь свои настройки для своих программ, свои папки-директории, свои настройки терминала и т.п.: это требование реализуется посредством предоставления пользователю «домашнего каталога», принадлежащего ему и все объекты в котором принадлежат ему, «хозяину» домашнего каталога.

Для этого в системе должна быть некоторая база данных (пусть даже очень примитивная), в которой хранятся основные сведения о каждом пользователе, допущенном к работе на данной машине.

Одна запись в этой БД, содержащая сведения о пользователе, называется user account. Для того, чтобы допустить нового пользователя в систему, необходимо создать для него этот account. Слово "account" обычно переводится как "банковский счёт" - не очень подходящий перевод. Поэтому, более подходящим по смыслу будет - "личная учётная карточка пользователя".

При создании учётной карточки (при регистрации пользователя, точнее, при регистрации ЛОГИНа пользователя) каждому пользователю присваивается уникальный номер (user ID, uid), именно этот цифровой номер и используется внутри операционной системы при проверке прав доступа, сборе и хранении статистики и т.д. Однако, для человека оперировать номерами пользователей неудобно. Поэтому, при регистрации пользователя в системе регистрируется алфавитно-цифровое имя пользователя — логин (login, username), которое тоже должно быть уникальным. Это символьное имя может быть даже осмысленным и потому человеку более понятно.

Если какой-нибудь программе требуется в качестве аргумента указать пользователя, обычно используется это "login name". И, наоборот, если какая-либо программа в своем выводе как-нибудь упоминает конкретных пользователей, она обычно называет их этим именем, а не просто печатает номер юзера.

То есть, между user ID и login всегда существует однозначное соответствие. Просто user ID используется во внутренних данных системы, а login при общении человек-компьютер.

Структура учётной карточки пользователя (основные атрибуты):

Name:Password:user ID:group ID:General information:Home directory:Shell

Назначение полей учётной карточки

Name или login. Уникальное имя пользователя. Его система спрашивает при входе пользователя и оно же используется в командах администрирования.

Password. Пароль. Для того, чтобы никто другой, кроме этого пользователя не мог войти в систему под логином этого пользователя, естественно, у каждого пользователя должен быть его собственный секретный пароль для входа. Вообще-то, пароль может и отсутствовать, но это очень не рекомендуется. Особенно, если машина доступна по сети. Пароли хранятся в БД в закодированном виде, поэтому, даже администратор (root) не может его узнать. (Однако, администратору он и не нужен, у него и так права не ограничены. А если юзер забудет свой пароль, то проще попросить администратора записать новый пароль, чем пытаться раскодировать старый).

User ID. Уникальный номер юзера, хранится в виде unsigned int, то есть, беззнакового целого в диапазоне от 0 до 65535. Он однозначно соответствует имени (login) и используется внутри системы.

Group ID. Номер группы, к которой принадлежит пользователь. В Unix (и в Linux) все пользователи объединяются в группы. Причем,

- каждый пользователь входит по крайней мере в одну группу, но
- может быть "членом" нескольких различных групп.

Каждая группа имеет своё имя (group name) и числовой номер (groupID), которые однозначно соответствуют друг другу. Группы используются при определении прав доступа пользователей к различным объектам в системе.

General information. Это некоторые "анкетные данные" того реального человека или иногда реальной организации, которые скрываются под Name. Очень часто это поле не заполняется вообще. Иногда там пишут реальное имя/фамилию юзера (например, Lev Tolstoi) в помощь администратору (должен же админ знать, что за пользователи работают в его системе?).

Полностью это поле может состоять из

Full Name - Имя Фамилия,

Location - адрес (имеется в виду рабочее место),

Office Phone - рабочий телефон,

Home Phone - домашний телефон.

А если пользователь — юридическое лицо, то название, телефон и, иногда, адрес фирмы.

Home dir. Домашний каталог пользователя. Именно в этом каталоге помещаются настроечные файлы для различных программ с настройками под конкретного пользователя (профиль пользователя). Здесь же пользователю даётся полная свобода создавать/удалять свои файлы. Все файлы в домашнем каталоге пользователя должны принадлежать этому пользователю, иначе они могут оказаться недоступными для этого пользователя.

Shell. Программа, которая запускается для пользователя, когда он входит в систему. Обычно это "исполнитель команд" (shell, программа-оболочка операционной системы), который принимает команды с терминала и запускает программы для исполнения этих команд.

В современных Unix'ах есть несколько таких оболочек (sh, csh, tcsh, bash ...), из которых можно выбрать наиболее подходящий. Однако, вместо shell'a здесь может быть указана любая другая программа, что часто используется для некоторых задач.

Место хранения учётных карточек

Доступная часть учётной информации хранится в файле /etc/passwd. Недоступная обычным пользователям часть учётной информации хранится в других местах, причём, в разных дистрибутивах Unix и Linux — в разных местах.

Подробнее обо всем этом можно прочитать в соответствующем справочном руководстве (man 5 passwd).

Лабораторная работа № 2

Тема: ФАЙЛОВЫЙ МЕНЕДЖЕР mc

Цель: Научиться работать с файловым менеджером mc

Задание:

Умные люди утверждают, что 90% всей информации человек получает с помощью зрения. Недаром говорят, что «лучше один раз увидеть, чем сто раз услышать». Следствием этого положения является то, что при работе в терминале желательно пользоваться некоторым средством, которое визуально показывало бы текущее состояние. Иначе пользователю системы придётся помнить состояние и запоминать, что он делал, что делает, а это излишнее напряжение.

Одним из таких несложных средств, которое представляет информацию визуально и которое позволяет несколько автоматизировать работу пользователя, является mc (Midnight Commander). Но для удобства и эффективности работы это средство необходимо правильно настроить.

Указания к выполнению работы:

Предполагается, что вы работаете в текстовом (терминальном) режиме. Если вы ещё в графическом (в KDE, Gnome или где-то ещё), то перейдите в текстовый режим. Для этого нажмите одновременно клавиши **Ctrl-Alt-F2**

1. В левом верхнем углу терминала видите приглашение вида

```
<имя_компа> login:
```

2. Входим в систему: вводим логин и пароль.

Появляется приглашение вида:

```
[login@имя_компа ~]$
```

Это означает, что мы в системе. Мы вошли под именем <login> на компьютер <имя_компа>. Значок ~ после двоеточия означает, что мы находимся в своём домашнем каталоге, который называется /home/<login>. Для нас система запустила оболочку bash. Значок \$ - приглашение оболочки вводить команды.

3. Вводим команду:

```
[login@имя_компа ~]$ mc
```

Нажимаем Enter. В результате выполнения команды должен запуститься файловый менеджер Midnight Commander. На рисунке 1 показано, как выглядит окно mc сразу после

установки пакета и при первом запуске.

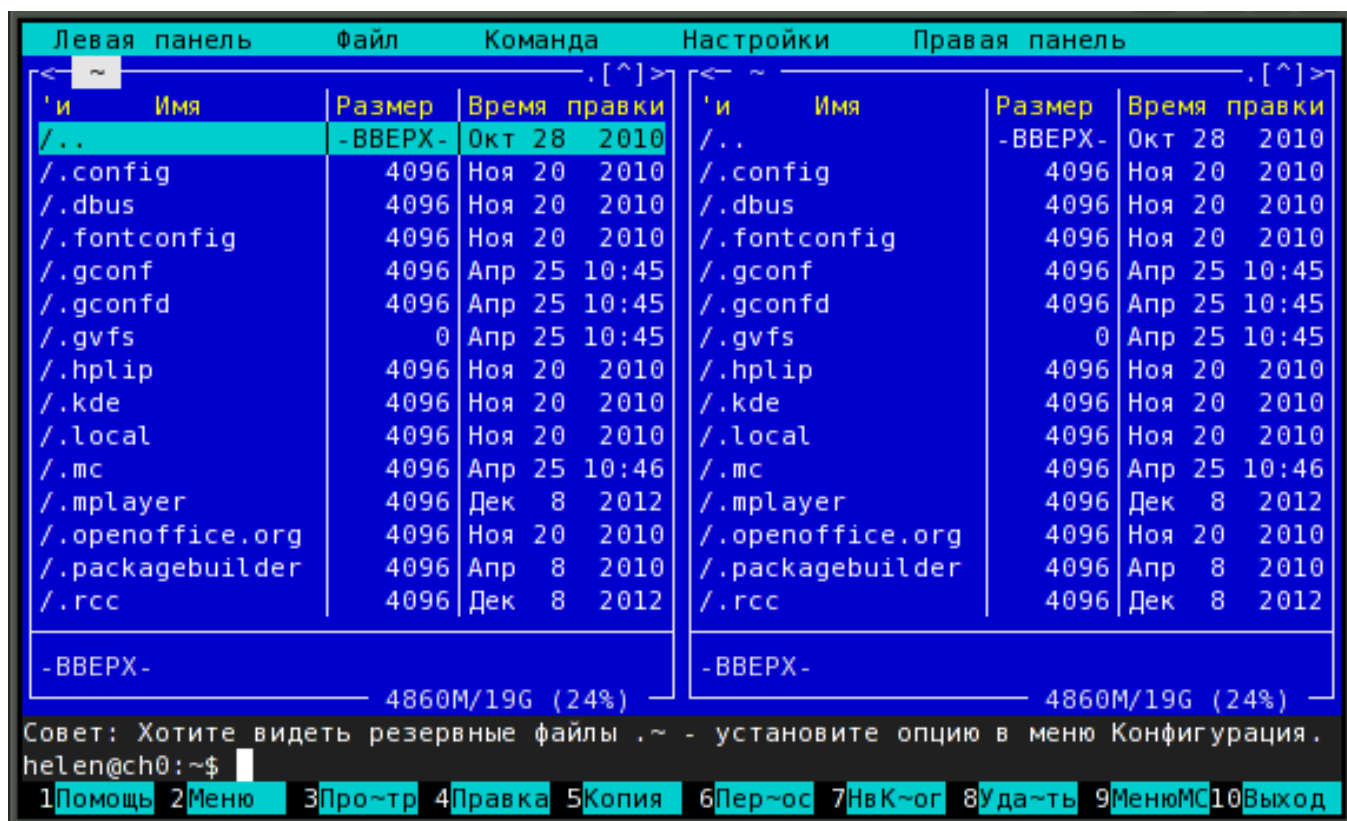


Рис. 1. Окно mc. Ненастроенное

Самая верхняя строка окна — строка меню. Вход в меню осуществляется посредством нажатия функциональной клавиши **F9**, а выход («без сохранения») - двойное нажатие клавиши **Esc**. Поскольку в меню всегда можно попасть с помощью клавиши F9, то строку меню можно удалить.

Среднюю часть окна занимают две панели — левая и правая. В них отображается содержание текущих каталогов. Отображаться может один и тот же каталог или разные. Видно, что в левой панели на каталоге «..» установлена полоса-курсор (Bar). Она может передвигаться по каталогам и файлам с помощью клавиш «стрелка_вверх» и «стрелка_вниз», а между панелями Bar переключается клавишей Tab. Панель, в которой находится Bar, называется **активной**.

Нижнюю часть панелей занимает статусная строка («мини-статус»), в которой с левой стороны выводится имя каталога/файла, на котором установлен Bar (на рисунке 1 Bar установлен на каталоге «..», который обозначает вышестоящий (родительский) каталог, поэтому вместо имени каталога в статусе видим слово «ВВЕРХ»), а с правой стороны выводится итоговая информация по файловой системе в форме: <сколько_свободно>/<общий_размер_файловой_системы> и в круглых скобках — процент свободного места.

Следующая строчка начинается со слова «Совет:». Это «строка подсказки», которую

обычно никто не читает. Поэтому её тоже можно удалить.

Следующая строчка — командная строка оболочки. В ней можно ввести команду оболочки (bash'a), нажать Enter и команда оболочки будет выполнена. Но результат выполнения команды (вывод команды) мы не увидим, потому что сразу после завершения команды будет восстановлено окно mc. Чтобы увидеть результат выполнения команды, нужно нажать клавиши Ctrl-o, чтобы вернуться в mc нужно нажать Ctrl-o ещё раз.

И самая нижняя строка окна — ещё одна подсказка: в ней перечислены используемые в mc функциональные клавиши и их назначение:

F1 — вызов большой справки (online help);

F2 — меню пользователя; используется для запоминания некоторых часто используемых действий, оформленных в виде скриптов;

F3 — просмотр содержимого файла во встроенном просмотрщике; выход из просмотра — F3;

F4 — встроенный текстовый редактор, простой и удобный;

F5 — копирование выделенного Ваг'ом файла из одной панели в другую;

F6 — перенос выделенного Ваг'ом файла из одной панели в другую;

F7 — создание нового каталога;

F8 — удаление выделенного Ваг'ом файла или каталога;

F9 — вход в меню mc;

F10 — выход из mc.

4. Рекомендуемые настройки mc.

4.1. Показывать экран терминала после выполнения команды. Для этого выполняем следующие действия: F9 → Настройки → Конфигурация. Открывается панель выбора «Параметров конфигурации» (см. рис. 2). На этой панели в разделе «Пауза после выполнения . . .» клавишами перемещения курсора (клавиши со стрелками на клавиатуре) выделить строку

() Всегда

и нажать клавишу пробел. Символ «*», который стандартно стоит в строке «На тупых терминалах», переместится в строку «Всегда». Нажать клавишу «Enter».

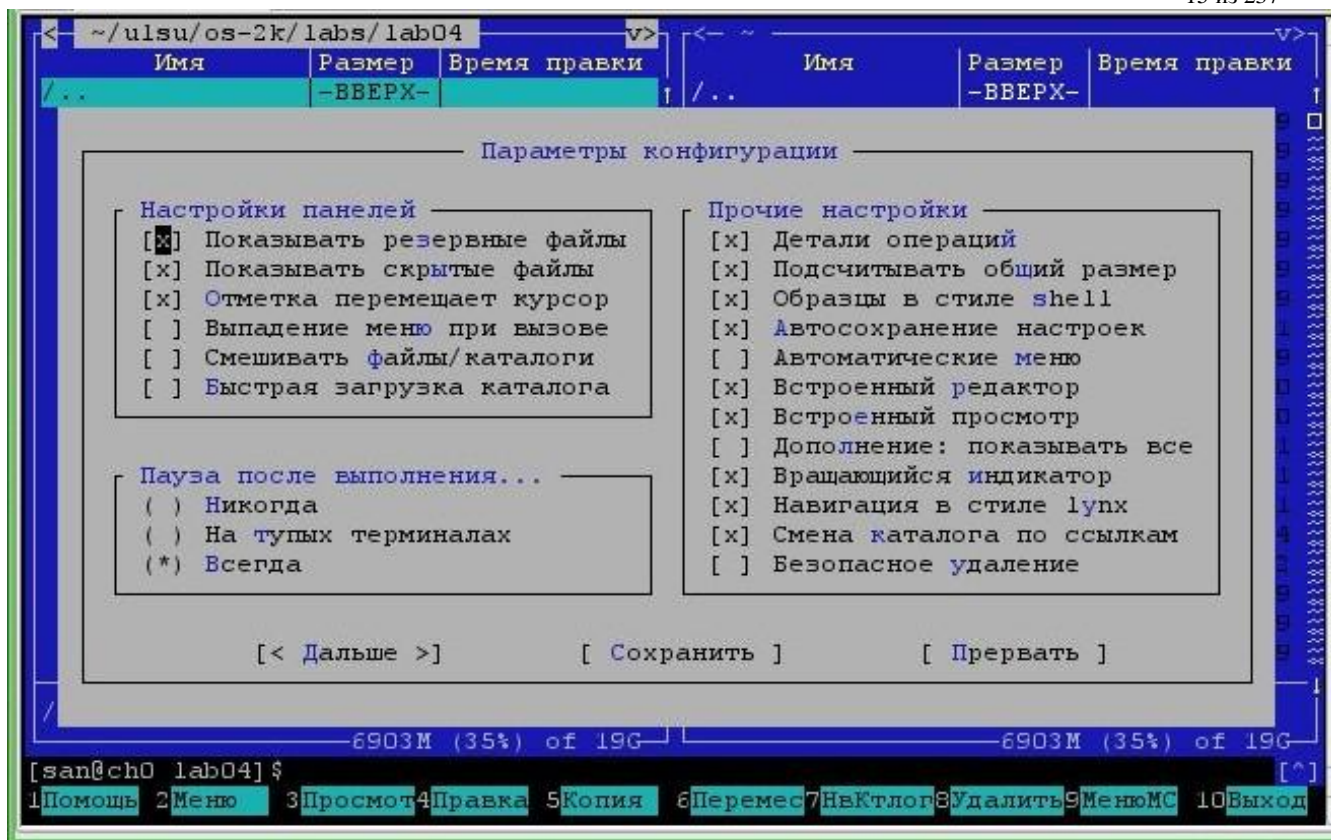


Рис. 2. Панель выбора параметров конфигурации —
установка паузы после выполнения

4.2. Увеличиваем «жизненное пространство» - убираем линейку меню и строку подсказки. Меню всегда можно вызвать функциональной клавишей F9, а подсказки всё равно никто не читает. Для этого выполняем следующие действия: F9 → Настройки → Внешний вид. Открывается панель настройки внешнего вида (см. рис. 3). На этой панели в разделе «Прочие настройки» клавишей пробел убираем крестики в строках «Линейка меню» и «Строка подсказки». После чего нажимаем клавишу «Enter».

4.3. Сохраняем сделанные настройки: F9 → Настройки → Сохранить настройки (см. рис. 4). Настройки сохраняются в персональном конфигурационном файле пользователя, который находится в каталоге `.mc` (обратите внимание на то, что перед именем каталога стоит «.»).

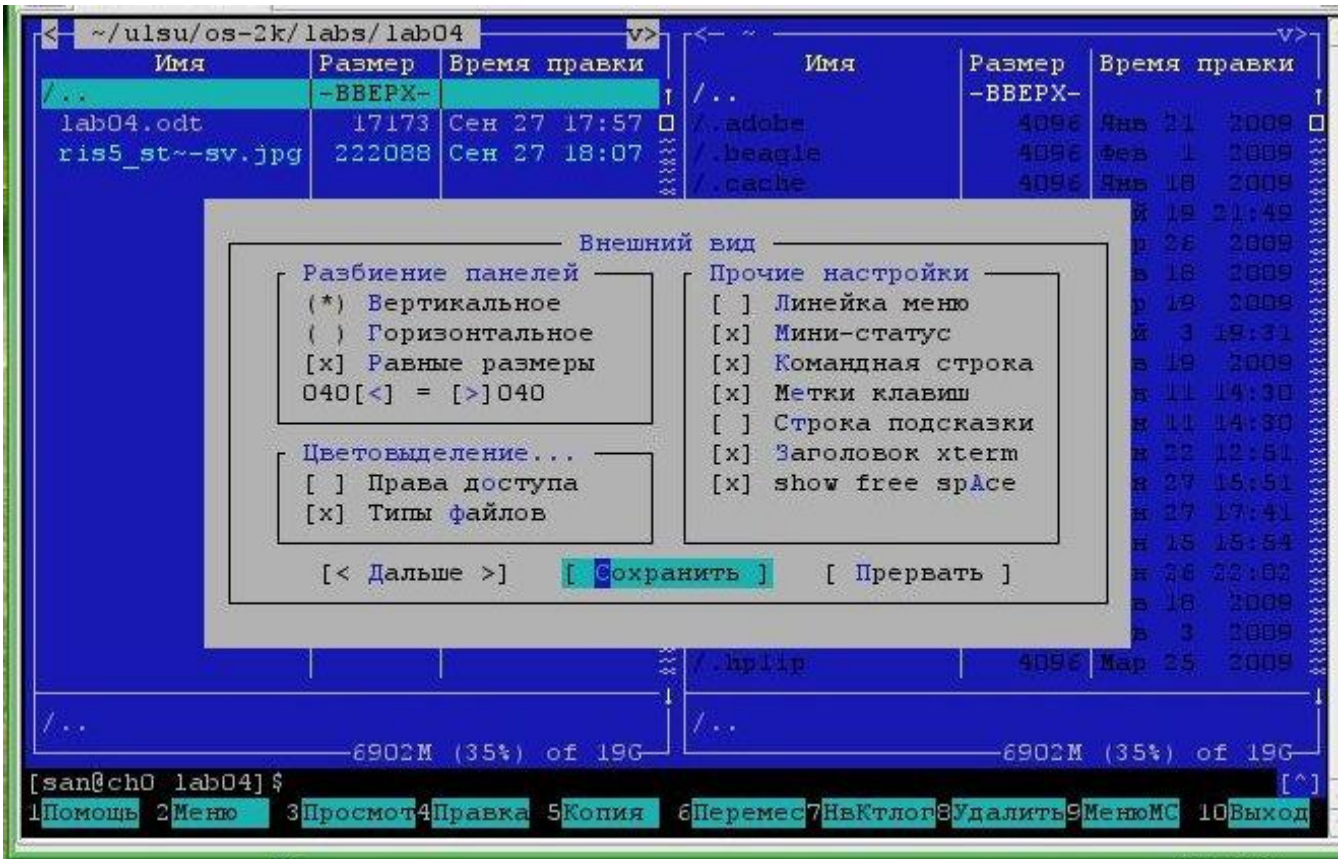


Рис. 3. Панель настройки внешнего вида

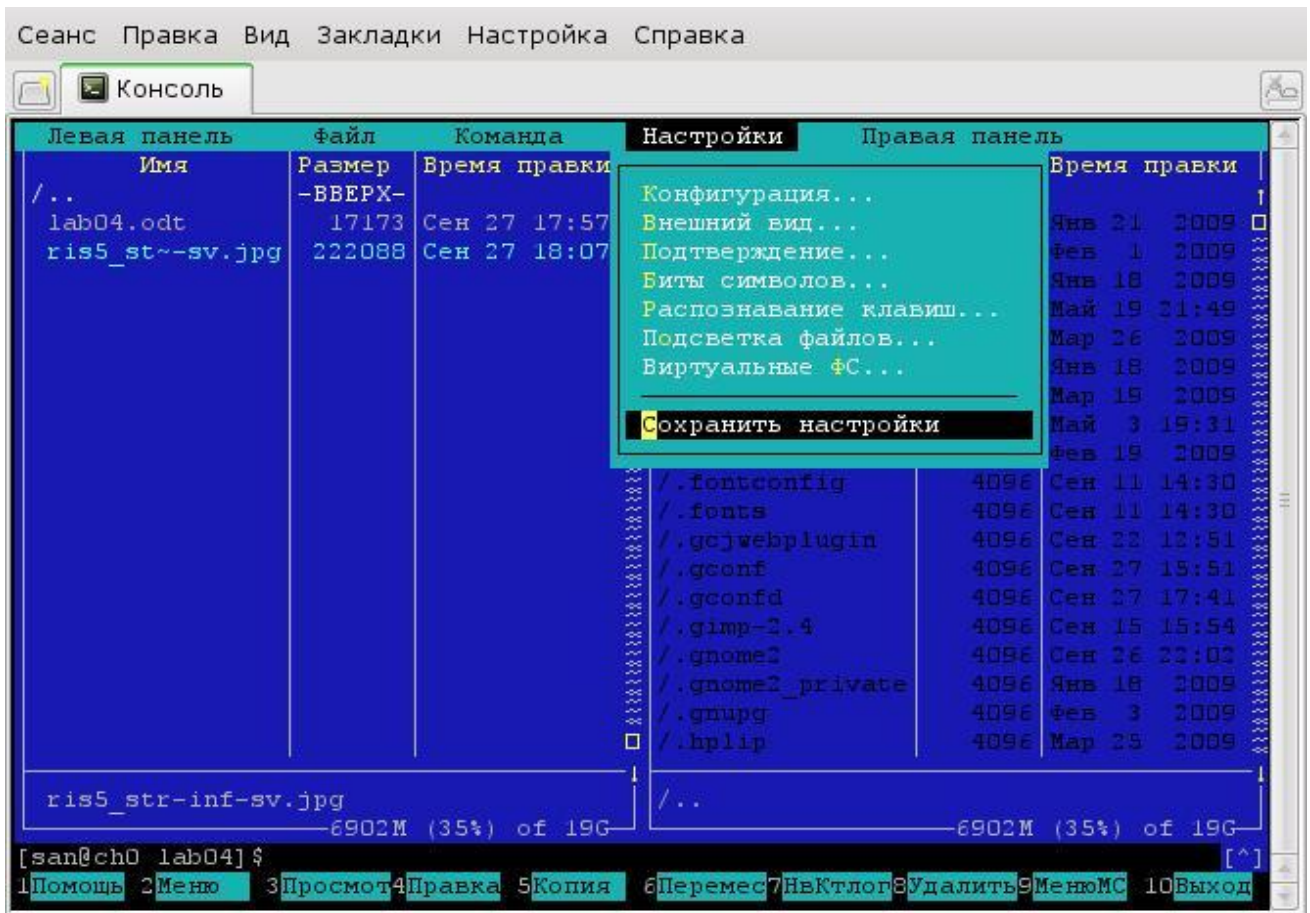


Рис. 4. Сохранение настроек

4.4. Теперь окно mc выглядит так как на рисунке 5, то есть, за счёт исключения лишних элементов «жизненное пространство» увеличилось на две строки. Кроме того, после выполнения команды оболочки, введённой в командной строке, окно mc восстанавливается не сразу, а только после нажатия any key, что позволяет увидеть вывод команды — результат её выполнения.

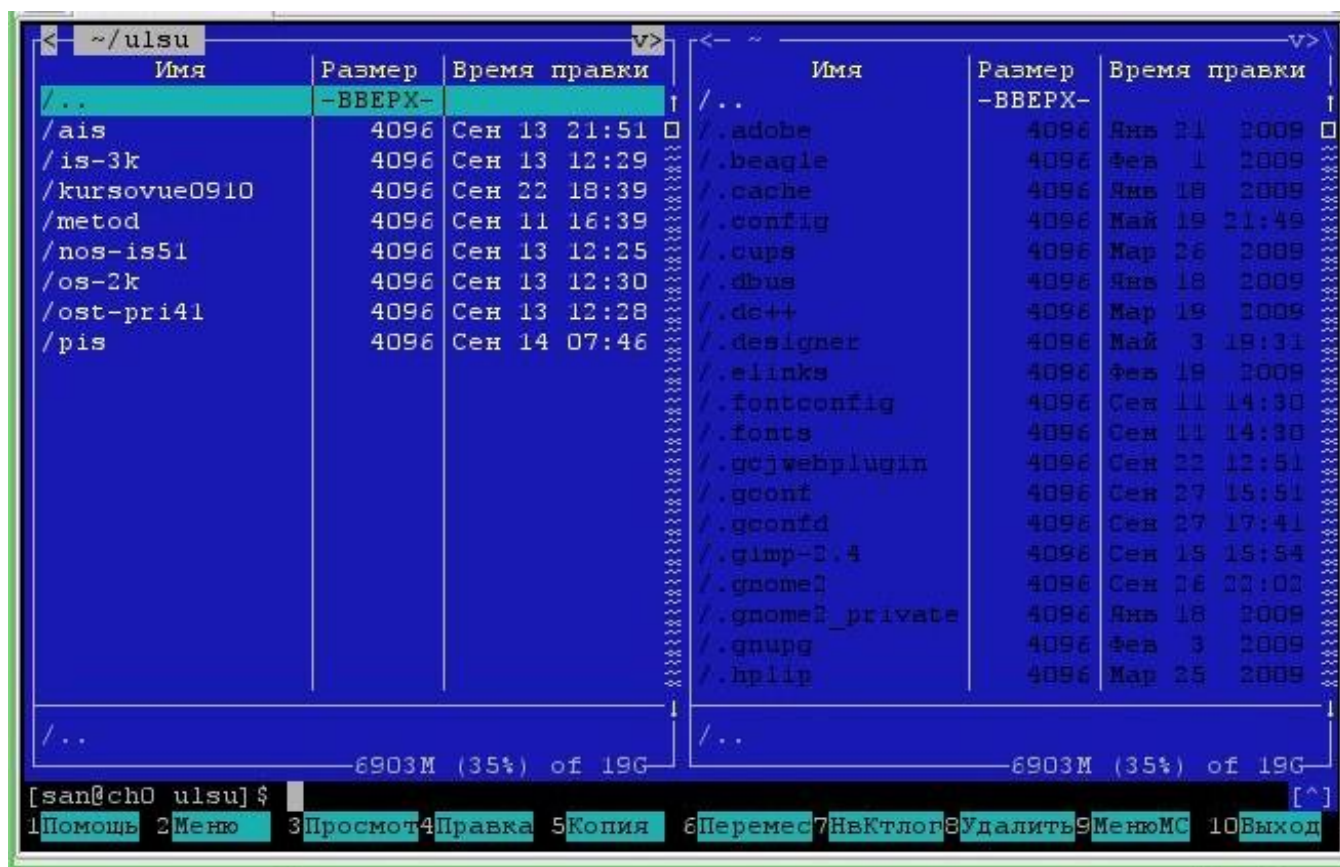


Рис. 5. Так окно mc должно выглядеть после произведённых настроек

5. Справочная система mc. Справка вызывается функциональной клавишей F1. Читать. В системе русский файл справки расположен обычно здесь: /usr/share/mc/mc.hlp.ru.

Примечание-требование. Важное. Везде далее для имён каталогов и файлов использовать только латинские буквы, цифры и спецсимволы «-» - тире, «_» - подчёркивание. Все остальные символы — запрещены.

6. Выполнить в mc следующие действия:

- 6.1. Сделать активной левую панель; переключение между панелями — клавиша Таб;
- 6.2. Перейти в ней в **свой домашний каталог**; перемещение между файлами и каталогами осуществляется с помощью клавиш «стрелка вниз» и «стрелка вверх»;
- 6.3. Создать каталог <фамилия_латинскими_буквами_>; в конце имени каталога должен стоять знак подчёркивания; создание каталога — функциональная клавиша F7;
- 6.4. Сделать активной правую панель;

- 6.5. Перейти в ней в **свой домашний каталог**;
- 6.6. Создать каталог <фамилия_латинскими_буквами_1>;
- 6.7. Зайти в каталог <фамилия_латинскими_буквами_1>, то есть сделать его текущим;
- 6.8. Нажать Shift-F4, откроется окно встроенного редактора mc; ввести текст «Я, ФИО, выполняю лабораторную работу № X»;
- 6.9. Сохранить файл с именем <фио_латинскими_буквами.txt>; сохранение файла во встроенном редакторе — функциональная клавиша F2;
- 6.10. Добавить в созданный файл текущую дату и время командой date;
- 6.11. Посмотреть содержимое созданного файла — функциональная клавиша F3; выход из режима просмотра — повторное нажатие клавиши F3;
- 6.12. Скопировать созданный файл в каталог <фамилия_латинскими_буквами_>; копирование файла — функциональная клавиша F5; **не забывайте: чтобы скопировать файл из одного каталога в другой, нужно открыть каталоги в обеих панелях, выделить нужный файл Var'ом, нажать клавишу F5, нажать клавишу Enter**;
- 6.13. Найти в файловой системе файл с именем hosts (начинайте поиск от корневой файловой системы, а не из домашнего каталога); как искать — смотреть в справочной системе mc;
- 6.14. Скопировать файл hosts в каталог <фамилия_латинскими_буквами_1>;
- 6.15. Сделать копию файла hosts в файл hosts1, а файл hosts удалить; удаление файла — функциональная клавиша F8;
- 6.16. Командой cat добавить содержимое файла hosts1 в файл <фио_латинскими_буквами.txt>;
- 6.17. Командой cat добавить содержимое файла /etc/fstab в файл <фио_латинскими_буквами.txt>;
- 6.18. Добавить в файл <фио_латинскими_буквами.txt> текущую дату и время командой date;
- 6.19. Посмотреть что получилось в этом файле — клавиша F3.

Порядок сдачи лабораторной работы:

1. По требованию преподавателя повторить работу в лаборатории и объяснить, что, собственно, делал.
2. Представить отчет.

Общие требования к отчету указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчет должен содержать следующую информацию:

- а) задание на работу;

- б) распечатку файла с именем <фамилия_латинскими_буквами.txt>;
- в) объяснение (комментарии) проделанной работы.

Дополнительная справочная информация

Краткие сведения о Midnight Commander'e

Здесь приведены краткие сведения о Midnight Commander'e, полный текст смотрите в файле /usr/share/mc/mc.hlp.ru.

Что такое Midnight Commander

Midnight Commander (mc) - это программа, предназначенная для просмотра содержимого каталогов и выполнения основных функций, управления файлами в Unix-подобных операционных системах. То есть, mc — это программа-оболочка, которая работает поверх shell и в некоторой степени автоматизирует работу пользователя.

Главное окно программы

Главное окно программы Midnight Commander состоит из трёх полей. Два поля, называемые "панелями", идентичны по структуре и обычно отображают перечни файлов и подкаталогов каких-то двух каталогов файловой структуры. Эти каталоги в общем случае различны, хотя, в частности, могут и совпасть. Каждая панель состоит из заголовка, списка файлов и информационной строки.

Третье поле экрана, расположенное в нижней части экрана, содержит командную строку текущей оболочки. В этом же поле (самая нижняя строка экрана) содержится подсказка по использованию функциональных клавиш F1 - F10.

Самая верхняя строка экрана содержит строку «горизонтального меню» (Menu Bar). Эта строка может не отображаться на экране; в этом случае доступ к ней можно получить, щелкнув мышью по верхней рамке или нажав клавишу F9.

Панели Midnight Commander обеспечивают просмотр одновременно двух каталогов. Одна из панелей является активной в том смысле, что пользователь может выполнять некоторые операции с отображаемыми в этой панели файлами и каталогами.

В активной панели подсвечено имя одного из каталогов или файлов, а также выделен цветом заголовок панели в верхней строке. Этот заголовок совпадает с именем отображаемого в данной панели каталога, который является текущим каталогом той оболочки, из которой запущена программа. Вторая панель - пассивна. Почти все операции выполняются в активной панели, то есть в соответствующем (текущем) каталоге. Некоторые операции (типа копирования или переноса файлов) по умолчанию используют каталог, отображаемый в пассивной панели, **как место назначения операции.**

Вы можете выполнить любую команду операционной системы или запустить на исполнение любую программу непосредственно из программы Midnight Commander, просто набрав имя этой команды (программы) в командной строке и нажав клавишу Enter.

Поддержка мыши

Программа Midnight Commander обеспечивает поддержку мыши. Это свойство обеспечивается независимо от того, откуда запущен терминал (xterm или console) (даже если терминал запущен на удаленном компьютере, используя соединение через telnet, ssh или rlogin) или если вы работаете за консолью Linux и запущена программа управления мышью gpm.

Если вы щелкаете мышью на имени файла в одной из панелей, файл выбирается (подсветка перемещается на это имя); если вы щёлкните правой кнопкой мыши, файл отмечается (или отметка с файла снимается, в зависимости от предыдущего состояния).

Двойной щелчок мыши на имени файла означает попытку запустить файл на исполнение (если это исполняемая программа); либо, если «файл расширений» (Extension File Edit) содержит программу, ассоциированную с данным расширением, запускается эта программа и ей передаётся на обработку выбранный файл.

Точно также можно выполнить команду, ассоциированную с любой функциональной клавишей, щёлкнув по соответствующей экранной кнопке в самой нижней строке экрана.

Если щёлкнуть мышью по верхней рамке панели, отображающей очень длинный список файлов, происходит перемещение списка на одну колонку назад. Щелчок по нижней рамке панели приводит, соответственно, к перемещению по списку на целую колонку вперёд. Этот метод перемещения работает также при просмотре «встроенной подсказки» (Help'a) и просмотре окна «Дерево каталогов» (Directory Tree).

По умолчанию скорость эмуляции повторных нажатий на клавишу в случае её удержания (auto repeat rate) составляет 400 миллисекунд. Это значение можно изменить путём изменения параметра «mouse_repeat_rate» в файле ~/.mc/ini.

Если Commander запущен с поддержкой мыши, вы можете обойти Commander и добиться того, что мышь будет вести себя так же, как она ведёт себя по умолчанию (обеспечивая вырезание и вставку текста), если будете удерживать клавишу Shift.

Некоторые команды

Если Midnight Commander скомпилирован с поддержкой подболочки (subshell), вы можете в процессе выполнения приложения из-под Midnight Commander в любой момент набрать C-o и вернуться к главному экрану Midnight Commander-a.

Для возврата к вашему приложению достаточно снова набрать C-o. Если вы застопорите выполнение приложения, использовав этот приём, вы не сможете запустить других программ из

Midnight Commander пока отложенное приложение не закончит работу, либо пока вы не прервете его выполнение.

Часто используемые команды:

- **Tab (или C-i)** Сменить текущую (активную) панель. Подсветка перемещается с панели, которая была активной ранее, в другую панель, которая становится активной.
- **Insert (или C-t)** Чтобы отметить файл, на который указывает в данный момент подсветка, используйте клавишу Insert (the kich1 terminfo sequence). Для снятия отметки с файла используются те же комбинации.
- **Alt-t** Циклически переключает режимы отображения списка файлов текущего каталога. С помощью этой комбинации клавиш можно быстро переключаться из режима стандартного вывода (long listing) к сокращенному или к режиму, определяемому пользователем.
- **+ (plus)** Эта клавиша используется для того, чтобы выбрать (отметить) группу файлов по регулярному выражению, задающему эту группу. Когда включена опция «Только файлы», то выделены будут только файлы. Если опция «Только файлы», отключена, то выделены будут как файлы, так и каталоги. Если включена опция «Образцы в стиле shell» (Shell Patterns), регулярные выражение строятся по тем же правилам, которые действуют в оболочке shell (* означает ноль или большее число любых символов, а ? заменяет один произвольный символ). Если опция «Образцы в стиле shell» (Shell Patterns) отключена, то пометка файлов производится по правилам обработки нормальных регулярных выражений (смотрите man ed). Если включена опция «С учётом регистра» то пометка файлов и каталогов будет производиться с учётом регистра символов имён. Если опция «С учётом регистра» отключена, то регистр символов учитываться не будет.
- **\ (backslash)** Клавиша "\" снимает отметку с группы файлов, то есть производит действие, обратное тому, которое вызывается по клавише "+".
- **Page-Up (или C-p)** Перемещает подсветку на предыдущую позицию в списке файлов панели.
- **Page-Down (или C-n)** Перемещает подсветку на следующую позицию в списке файлов панели.
- **home (или Alt-<)** Перемещает подсветку на первую позицию списка файлов.
- **end (или Alt->)** Перемещает подсветку на последнюю позицию списка файлов.

Командная строка оболочки

В этом разделе перечислены команды, которые позволяют сократить число нажатий на клавиши во время ввода и редактирования команд в командной строке:

- **Alt-Enter** Копирует подсвеченное имя файла или каталога в командную строку.
- **C-Enter** То же самое, что M-Enter, но работает только на консоли Linux.
- **C-q** Эта команда (the quote command) используется для того, чтобы вставить символы, которые каким-то образом интерпретируются самим Midnight Commander-ом (например, символы '+', '-', '*',

'!' и другие; поэтому нельзя использовать эти и подобные символы в именах файлов).

Клавиши управления перемещением

Встроенная программа просмотра файлов, программа просмотра подсказки и программа просмотра каталогов используют один и тот же программный код для управления перемещением.

Следовательно, для перемещения используются одни и те же комбинации клавиш. Но в каждой подпрограмме имеются и комбинации, применяющиеся только в ней.

Другие части Midnight Commander'a тоже используют некоторые из комбинаций клавиш управления перемещением, так что настоящая секция руководства может быть также полезна при изучении этих частей.

Клавиши управления перемещением:

- «Стрелка вверх» (или **C-p**) Перемещение на одну строку назад или вверх
- «Стрелка вниз» (или **C-n**) Перемещение на одну строку вперёд
- **Page Up** (или **M-v**) Перемещение на одну страницу назад
- **Page Down** (или **C-v**) Перемещение на одну страницу вперёд
- **Home** Перемещение к началу.
- **End** Перемещение к концу.

Редактирование строк ввода

При вводе команд в строку ввода используются некоторые управляющие комбинации клавиш. Они описаны в руководстве на `tc`.

Главное меню программы Midnight Commander

Строка главного меню появляется в верхней части экрана после нажатия клавиши F9 или щелчка мыши по верхней рамке экрана. Меню состоит из пяти пунктов: "Левая", "Файл", «Команды", "Настройки" и "Правая" (в английской версии соответственно "Left", "File", "Command", "Options" и "Right"). При выборе одного из этих пунктов появляется соответствующее выпадающее меню. Все они подробно описаны в руководстве на `tc`.

Встроенная программа просмотра файлов

Встроенная программа просмотра файлов имеет два режима просмотра: режим ASCII и шестнадцатеричный (hex). Для переключения режимов используется клавиша F4. Если у вас установлена программа `gzip` проекта GNU, она будет использована для автоматического просмотра сжатых файлов.

Встроенная программа просмотра всегда пытается использовать для отображения информации лучший из методов, предоставляемых вашей системой для данного типа файла.

Некоторые последовательности символов интерпретируются для задания таких атрибутов, как жирный шрифт и подчёркивание, обеспечивая более наглядное представление информации.

В шестнадцатеричном режиме функция поиска позволяет задать строку поиска как в обычном текстовом виде (заклѳченном в кавычки), так и в виде шестнадцатеричных констант. Можно даже одновременно использовать в шаблоне поиска как ту, так и другую форму представления, например:

```
"String" -1 0xBB 012 "more text"
```

Обратите внимание, что 012 является восьмеричным числом, -1 преобразовывается в 0xFF, а текст между кавычками и константами игнорируется.

Встроенный редактор

Встроенный редактор обеспечивает выполнение большинства функций редактирования, присущих полноэкранным редакторам текста. Он вызывается нажатием клавиши F4 при условии, что в инициализационном файле установлена в 1 опция use_internal_edit. Размер редактируемого файла не может превышать 16 Мегабайт. С помощью этого редактора можно редактировать двоичные файлы без потери данных.

Поддерживаются следующие возможности: копирование, перемещение, удаление, вырезание и вставка блоков текста; отмена предыдущих операций (key for key undo); выпадающие меню; вставка файлов; макроопределения; поиск и замена по регулярным выражениям; выделение текста по комбинации клавиш shift-стрелки в стиле MSW-MAC (только для Linux-консоли); переключение между режимами вставки-замены символа; а также операция обработки блоков текста командами оболочки (an option to pipe text blocks through shell commands like indent).

Редактор очень прост и практически не требует обучения. Для того, чтобы узнать, какие клавиши вызывают выполнение определённых действий, достаточно просмотреть выпадающие меню, которые вызываются нажатием клавиши F9 в окне редактора. Не перечисленные в меню комбинации клавиш:

- Shift-<клавиши стрелок> выделение блока текста.
- Ctrl-Ins копирует блок в файл cooledit.clip.
- Shift-Ins производит вставку последнего скопированного в cooledit.clip блока в позицию курсора.
- Shift-Del удаляет выделенный блок текста, запоминая его в файле cooledit.clip.
- По клавише Enter вставляются символы конца строки, причем на следующей строке автоматически устанавливается отступ.

Работает выделение текста с помощью мыши, причѳм если удерживать клавишу Shift, то управление мышью осуществляется терминальным драйвером мыши.

Для того, чтобы определить макрос, нажмите Ctrl-R, после чего введите строки команд,

которые должны быть выполнены. После завершения ввода команд снова нажмите Ctrl-R и свяжите макрос с какой-нибудь клавишей или комбинацией клавиш, нажав эту клавишу (комбинацию). Макрос будет вызываться нажатием Ctrl-A и назначенной для него клавиши. Макрос можно также вызвать нажатием любой из клавиш Meta (Alt), Ctrl, или Esc и назначенной макросу клавиши, при условии, что данная комбинация не используется для вызова какой-либо другой функции. Макрокоманды после определения записываются в файл `.mc/cedit/cooledit.macros` в вашем домашнем каталоге. Вы можете удалить макрос удалением соответствующей строки в этом файле.

По клавише F19 (ее нет на обычной клавиатуре IBM PC, так что придется пользоваться соответствующим пунктом меню, вызываемым по клавише F9, или переназначить клавишу) будет осуществляться форматирование выделенного блока кода на языке C, C++ или других. Форматирование управляется файлом `/usr/share/mc/edit.indent.rc`, который при первом вызове копируется в `.mc/cedit/edit.indent.rc` в вашем домашнем каталоге.

Встроенный редактор обрабатывает символы из второй половины кодовой таблицы (160+). Но когда **редактируете бинарные файлы**, лучше установить опцию "Биты символов" (Display bits) из меню "Настройки" в положение "7 бит", чтобы сохранить формат файла (to keep the spacing clean).

Описать все функции встроенного редактора в данной подсказке невозможно. Запомните только, что все основные операции можно выполнить через пункты меню, которое вызывается нажатием клавиши F9 в окне редактирования. Кроме того, можно прочитать man-страницу по команде `man mcedit` (или `info mcedit [Internal File Editor / options]`).

Виртуальные файловые системы

Программа Midnight Commander содержит подпрограммы, обеспечивающие доступ к различным файловым системам. Эти подпрограммы (их совокупность называется переключателем виртуальных файловых систем - virtual file system switch) позволяют Midnight Commander-у манипулировать файлами, расположенными на не-Unix-овых файловых системах.

В настоящее время Midnight Commander обеспечивает поддержку нескольких Виртуальных Файловых Систем - ВФС (VFS):

- локальной файловой системы, используемой для обычных файловых систем Unix;
- файловой системы `ftpfs`, используемой для манипулирования файлами на удаленных компьютерах по протоколу FTP;
- файловой системы `tarfs`, используемой для обработки tar- и сжатых tar-файлов;
- файловой системы `undelfs`, используемой для восстановления удаленных файлов в файловой системе `ext2` (файловая система, используемая в Linux по умолчанию);
- файловой системы `fish` (для манипулирования файлами при работе с оболочкой через

такие программы как rsh и ssh);

- и, наконец, сетевой файловой системы nfs.

МС может быть собран с поддержкой файловой системы smbfs, используемой для манипулирования файлами на удаленных компьютерах по протоколу SMB (CIFS).

Подпрограммы работы с виртуальными файловыми системами интерпретируют все встречающиеся имена путей и формируют корректные обращения к различным файловым системам. Форматы обращения к каждой из виртуальных файловых систем описаны в отдельных разделах по каждой ВФС.

Файловая система ftpfs (FTP File System)

Файловая система ftpfs позволяет работать с файлами на удаленных компьютерах. Для этого можно использовать команду "FTP-соединение" (доступную из меню левой и правой панелей) или же непосредственно сменить текущий каталог командой cd, задав путь к каталогу следующим образом:

```
ftp:[!][user[:pass]@]machine[:port][remote-dir]
```

Элементы user, port и remote-dir не обязательны. Если элемент user указан, то Midnight Commander будет пытаться зарегистрироваться на удалённом компьютере с этим именем, в противном случае будет использовано имя anonymous или имя из файла ~/.netrc. Необязательный элемент pass#(если указан) используется как пароль для входа. Однако явно задавать его не рекомендуется (также не записывайте его в ваши hotlist, если только вы не обеспечили соответствующую защиту этих файлов; но и тогда нельзя быть полностью уверенным в безопасности).

Примеры:

```
ftp:ftp.nuclecu.unam.mx/Linux/local
```

```
ftp:tsx-11.mit.edu/pub/Linux/packages
```

```
ftp:!behind.firewall.edu/pub
```

```
ftp:guest@remote-host.com:40/pub
```

```
ftp:miguel:xxx@server/pub
```

Для того, чтобы соединиться с сервером, который расположен за firewall, нужно использовать префикс ftp:! (то есть добавить восклицательный знак перед именем сервера), чтобы указать Midnight Commander на необходимость использовать прокси для осуществления передач по ftp. Вы можете задать имя прокси в диалоговом окне «Виртуальные ФС...» (Virtual FS) меню "Настройки".

Чтобы не задавать имя прокси-сервера каждый раз, можно поставить отметку в квадратных скобках возле опции «Всегда использовать FTP прокси» в диалоговом окне «Виртуальные ФС...»

(Virtual FS) меню "Настройки". В таком случае программа всегда будет использовать указанный прокси-сервер. При этом (если опция установлена) программа делает следующее: считывает из файла `/usr/share/mc/mc.no_proxu` имена локальных машин (если имя начинается с точки, оно считается именем домена), и, если заданное при установлении FTP-соединения имя машины совпадает с одним из имен, указанных в файле `mc.no_proxu` без точки, то производит прямое обращение к данной машине.

При подключении к ftp-серверу через фильтрующий пакеты маршрутизатор (If you are using the ftpfs code with a filtering packet router), который не позволяет использовать обычный режим открытия файлов, можно заставить программу работать в режиме пассивного открытия файла (the passive-open mode). Для этого установите в инициализационном файле опцию `ftpfs_use_passive_connections` в 1.

Midnight Commander сохраняет в течение заданного интервала времени список файлов удалённого каталога, прочитанный по FTP, в оперативной памяти. Величина этого интервала времени задаётся в диалоговом окне «Виртуальные ФС...» (Virtual FS) меню "Настройки". В силу этого возможен побочный эффект, заключающийся в том, что даже если вы сделали какие-то изменения в каталоге, они не будут отображаться в панели до тех пор, пока вы не обновите содержимое панели командой `C-g`. Это не является недоработкой (если вы думаете, что это ошибка, поразмыслите над тем, как происходит работа по FTP с файлами, находящимися на другой стороне Атлантического океана).

Файловая система FISH (File transfer over SHell)

Файловая система `fish` — это сетевая файловая система, которая позволяет работать с файлами на удалённом компьютере так, как если бы они были расположены на вашем диске. Для того, чтобы это было возможно, на удалённом компьютере должен быть запущен `fish`-сервер, или `bash`-совместимая оболочка `shell`.

Для соединения с удалённым компьютером нужно выполнить команду перехода в каталог (`chdir`), имя которого задаётся в следующем формате:

```
sh:[user@]machine[:options]/[remote-dir]
```

Элементы `user`, `options` и `remote-dir` не обязательны. Если задан элемент `user`, то Midnight Commander будет регистрироваться на удалённый компьютер под этим именем, в противном случае - под тем именем, с которым вы зарегистрированы в локальной системе.

В качестве `options` могут использоваться:

'C' - использовать сжатие;

'r' - использовать `rsh` вместо `ssh`;

`port` - использовать данный порт для подключения к удалённому компьютеру.

Если задан элемент `remote-dir`, то указанный каталог станет текущим после соединения с

удалённым компьютером.

Примеры:

```
sh:onlyrsh.mx:r/Linux/local
sh:joe@want.compression.edu:C/private
sh:joe@noncompressed.ssh.edu/private
sh:joe@somehost.ssh.edu:2222/private
```

Файловая система NFS (Network File System)

Файловая система `nc` - это ещё одна сетевая файловая система, которая позволяет работать с файлами на удалённом компьютере. Для того, чтобы можно было воспользоваться этой ФС, на удалённом компьютере должна быть запущена серверная программа `ncserv`.

Для соединения с удалённым компьютером нужно выполнить команду перехода в каталог, имя которого строится в соответствии со следующим форматом:

```
nc:[user@]machine[:port][remote-dir]
```

Элементы `user`, `port` и `remote-dir` не обязательны.

Если задан элемент `user`, то Midnight Commander будет регистрироваться на удалённый компьютер под этим именем, в противном случае — под тем именем, с которым вы зарегистрированы в локальной системе.

Элемент `port` используется в том случае, если удалённый компьютер использует специальный порт (чтобы узнать, что такое порт и как его использовать, смотрите страницу руководства `man ncserv`. Если задан элемент `remote-dir`, то указанный каталог станет текущим после соединения с удалённым компьютером.

Примеры:

```
nc:ftp.nuclecu.unam.mx/Linux/local
nc:joe@foo.edu:11321/private
```

Файловая система UFS (Undelete File System)

В ОС Linux можно сконфигурировать файловую систему `ext2fs`, используемую по умолчанию, таким образом, что появится возможность восстанавливать удалённые файлы (но только в файловой системе `ext2`). Файловая система UFS (Undelete File System) представляет собой интерфейс к библиотекам `ext2fs`, позволяющий восстановить имена всех удалённых файлов, выбрать некоторое количество таких файлов и восстановить их.

Для того, чтобы воспользоваться этой возможностью (этой файловой системой), нужно выполнить команду перехода (`chdir`) в специальный каталог, имя которого образуется из префикса "undel" и имени специального файла устройства, на котором находится реальная файловая система.

Например, чтобы восстановить удалённые файлы на втором разделе первого SCSI-диска, нужно использовать следующее имя:

```
undel:sda2
```

Загрузка списка удалённых файлов требует некоторого времени, так что наберитесь терпения. Имейте в виду, что имена файлов в полученном списке будут цифровыми, так что поиск нужного придется проводить либо по дате, либо последовательным просмотром содержимого.

Файловая система smbfs

Файловая система smbfs позволяет работать с файлами на удалённых компьютерах по протоколу SMB (CIFS) (Windows for Workgroups, Windows 9x/ME/XP, Windows NT, Windows 2000 и Samba). Для этого можно использовать пункт "SMB связь..." (доступный из меню левой и правой панелей) или же непосредственно сменить текущий каталог командой `cd`, задав путь к каталогу следующим образом:

```
smb:[username@]machine[/service][remote-dir]
```

Элементы `username`, `service` и `remote-dir` необязательны. `username`, `domain` и `password` могут быть указаны в окне диалога.

Примеры:

```
smb:machine/Share
```

```
smb:other_machine
```

```
smb:guest@machine/Public/Irlex
```

Лабораторная работа № 3

Тема: ТЕРМИНАЛ: КОМАНДЫ РАБОТЫ С ФАЙЛАМИ

Цель: Научиться работать в терминале с командами работы с файлами ОС Linux

Задание: Изучить команды работы с файлами

Указания к выполнению работы:

Предполагается, что вы работаете в текстовом (терминальном) режиме. Если вы ещё в графическом (в KDE, Gnome или где-то ещё), то перейдите в текстовый режим. Для этого нажмите одновременно клавиши **Ctrl-Alt-F2**

1. В левом верхнем углу терминала отображается приглашение вида

```
<имя_компа> login:
```

2. Входим в систему: вводим логин и пароль.

Появляется приглашение вида:

```
[login@имя_компа ~]$
```

Это означает, что мы в системе. Мы вошли под именем <login> на компьютер <имя_компа>. «Собачка» @ между login и именем компьютера является напоминанием о том, что вообще-то эта комбинация является локальным почтовым адресом пользователя в данной вычислительной системе. Значок ~ (тильда) означает, что мы находимся в своём домашнем каталоге, который называется /home/<login>.

Если мы находимся в каком-либо другом каталоге, то вместо тильды будет стоять название этого каталога. Для нас система запустила оболочку bash. Значок \$ - приглашение оболочки вводить команды. Курсор в виде мигающего белого прямоугольника установится после знака «\$». То есть, ваши команды будут вводиться после этого знака.

Знак «\$» - это приглашение к вводу для обычного пользователя (в отличие от root'a, для которого знак приглашения - «#»).

Таким образом, после знака «\$» вы вводите команду, Linux её выполняет, что-то выдаёт на экран (если у неё есть вам сказать пару слов, так она вам скажет, а если нет, то выполнит вашу команду молча) и, если команда выполнена, то Linux снова выдаёт знак «\$», приглашая вас вводить следующую команду.

Ввод команды всегда завершается нажатием клавиши Enter, только после Enter система начинает выполнять команду. Если вы Enter нажали, курсор перешёл на следующую строку

и ... всё,

знак \$ не появляется. Это означает, что Linux ждёт ваших дальнейших действий — команда ещё не завершена. Примеры подобных команд есть в лабораторных работах.

! Не забывайте, что «/» («слэш») - это символ-разделитель между именами объектов файловой системы - именами каталогов и файлов. Пример:

/usr/share/mc/mc.hint.ru — абсолютный путь (от корня файловой системы) к файлу mc.hint.ru.

! В этом пути не должно быть пробелов, в том числе, символ «/» не должен отделяться пробелами от имён.

Синтаксис команды:

`<имя_команды><пробел><ключи_команды><пробел><аргументы_команды>`

где

`<имя_команды>` - название встроенной команды программы-оболочки или название какой-либо программы в системе;

`<пробел>` - символ разделитель;

`<ключи_команды>` - один или несколько ключей команды, вводятся в виде «-буква» (минус — признак ключа, буква — название (идентификатор) ключа); если вводятся несколько ключей подряд, то они разделяются пробелом; некоторые ключи могут иметь аргументы;

`<аргументы_команды>` - некоторое алфавитно-цифровое имя, идентифицирующее какой-либо объект в системе.

Пример команды:

```
ls -i -l -a /home
```

Запуск программы из текущего каталога. Например, вы находитесь в каталоге /usr/bin и видите в нем программу xman. Но при попытке запустить её получаете в ответ

```
bash: xman команда не найдена.
```

Правильно так:

```
./xman
```

То есть, символами «./» вы указываете, что путь к программе должен исчисляться от текущего каталога, а имя «.» (точка) - это и есть текущий каталог.

Частые причины возникновения ошибочной ситуации:

- обычный пользователь пытается выполнить команду goot'a, а она не всегда доступна;
- в текущем каталоге нужной программы нет; необходимо указать путь к данной программе, например /sbin/fdisk.

Далее в задании будет опускаться содержимое квадратных скобок ([login@имя компа каталог]), а будет указываться только символ «\$».

Внимание: При сдаче работы возможно придётся отвечать на вопросы о назначении и смысле

команд. Справка по команде:

```
$ man <команда> <Enter>
```

В том числе можно получить справку и по самой системе man:

```
$ man man <Enter>
```

3. Ввести команду:

```
$ script
```

Скрипт запущен, файл - typescript

```
$
```

4. Прежде всего, убедитесь, что вы находитесь в своём домашнем каталоге. Это можно сделать командой:

```
$ pwd
```

5. Чтобы посмотреть, какие файлы уже находятся в вашем домашнем каталоге, нужно ввести команду:

```
$ ls
```

6. Ввести команду:

```
$ ls -i -l
```

7. Обратит внимание на первую колонку вывода этой команды. В отчёте объяснить, что означают эти числа.

8. Ввести команду:

```
$ ls -i -l -a
```

9. Обратит внимание на появившиеся каталоги и файлы с точкой. В отчёте объяснить, что это за файлы и каталоги.

10. Наиболее часто используемая форма команды ls:

```
$ ls -l
```

11. Создать файл с именем = фио (например, ivanov_i.txt) командой

```
$ touch <имя_файла>
```

12. Ввести в этот файл следующую информацию «Я, <фамилия имя отчество>, студент УлГУ, ФМИАТ, группа <группа>» командой:

```
$ cat > <имя_файла>
```

<информация, указанная выше>

<Ctrl-D>

Обратите внимание, что <Ctrl-D> вводится на новой строке; то есть, сначала вводим <информацию, указанную выше>, затем нажимаем Enter (переходим на новую строку), затем нажимаем <Ctrl-D>.

13. Проверить, что информация в файл введена, командой:

```
$ cat <имя_файла>
```

14. Дополнить созданный файл следующей информацией «Это результат выполнения лабораторной № 2» с помощью команды:

```
$ echo <информация> >> <имя_файла>
```

Внимание: Если <информация> содержит пробелы (пробел — это разделитель!), то <информацию> заключить в кавычки; тогда всё, что содержится в кавычках будет восприниматься командой echo как один аргумент.

15. Проверить, что информация в файл введена правильно, командой:

```
$ cat <имя_файла>
```

16. Дополнить созданный файл следующей информацией «Дисциплина «Архитектура и программное обеспечение инфокоммуникационных устройств», курс N-ый, семестр K-ый» с помощью команды:

```
$ tee >> <имя_файла>.
```

После ввода команды вводим указанную информацию, затем нажимаем Enter (переходим на новую строку), затем нажимаем <Ctrl-C>

! Обратите внимание, что <Ctrl-C> вводится на новой строке.

17. Проверить, что информация в файл введена правильно, командой:

```
$ cat <имя_файла>
```

18. И в конце файла поставить дату и время:

```
$ date >> <имя_файла>
```

19. Нажать на клавиатуре клавишу PrintScreen. В открывшемся окне программы Ksnapshot выбрать режим копирования «Окно под курсором», нажать клавишу «Новый снимок», указать мышкой окно терминала, после восстановления окна программы Ksnapshot клавишей <Сохранить как . . .> сохранить копию экрана в файл labaN.jpg в свой домашний каталог.

20. Завершение задания: ввести Ctrl-D. В терминале появится сообщение «Скрипт выполнен, файл - typescript»

21. Таким образом, в вашем домашнем каталоге образовался файл с именем typescript. Это протокол вашей работы. А также была создана копия экрана перед завершением работы. На копии **должны хорошо читаться** последние ваши команды в терминале.

Порядок сдачи лабораторной работы:

1. По требованию преподавателя повторить работу в лаборатории и объяснить, что, собственно, делал.
2. Продемонстрировать протокол работы.
3. Представить отчёт.

Общие требования к отчёту указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчёт должен содержать следующую информацию:

- а) задание на работу;
- б) распечатку файла typescript с именем <фамилия_латинскими_буквами.txt>;
- в) распечатку копии **окна терминала** с качеством, достаточным, чтобы можно было прочесть информацию в окне терминала.
- г) объяснение (комментарии) проделанной работы.

Дополнительная справочная информация

Краткое описание основных команд для работы с файлами

Здесь приведены не все команды, более полное описание смотрите в справочных руководствах (тап <команда>).

cd [каталог]

Меняет текущий каталог на указанный. Если параметр опущен, то текущим становится домашний каталог.

ls [-alFR] [файл ...]

Выводит список файлов в указанном (или текущем) каталоге. Ключ -a заставляет выводить все файлы (в том числе, скрытые), ключ -l служит для вывода подробной информации о файлах, ключ -F приводит к тому, что к именам каталогов добавляется символ '/', к именам ссылок '@', к именам выполняемых файлов '*'. При использовании ключа -R выводится список файлов не только указанного каталога, но и его подкаталогов.

touch файл ...

Меняет время доступа и изменения файла. **Если файл не существовал, то он будет создан.**

mkdir каталог

Создает каталог.

rmdir каталог

Удаляет каталог.

cp [-Rp] файл1 файл2

cp [-Rp] файл ... каталог

Копирует один файл в другой (затирая прежнее содержание этого другого файла) или копирует файлы в указанный каталог. Ключ -R предназначен для копирования каталогов, ключ -p позволяет сохранять владельцев файлов, режим доступа и время доступа и изменения.

rm [-r] файл ...

Удаляет файлы. Ключ -r позволяет удалять каталоги.

mv файл1 файл2

mv file ... directory

Переименовывает файл или перемещает файлы в заданный каталог.

cat [файл ...]

Объединяет содержимое указанных файлов и выводит на стандартный вывод.

cat [файл1+файл2+...+файлN файлM]

Объединяет содержимое указанных файлов и выводит их в один файл (объединяет).

Лабораторная работа № 4

Тема: ТЕРМИНАЛ: ПЕРЕМЕННЫЕ ОКРУЖЕНИЯ

Цель: Научиться работать в терминале с командами работы с профилем пользователя ОС Linux

Задание: Изучить команды работы с профилем пользователя ОС Linux.

Указания к выполнению работы:

Предполагается, что вы работаете в текстовом (терминальном) режиме. Если вы ещё в графическом (в KDE, Gnome или где-то ещё), то перейдите в текстовый режим. Для этого нажмите одновременно клавиши **Ctrl-Alt-F2**

1. В левом верхнем углу терминала видите приглашение вида

```
<имя_компа> login:
```

2. Входим в систему: вводим логин и пароль.

Появляется приглашение вида:

```
[login@имя_компа ~]$
```

Прочтите пункт 2 задания на лабораторную работу 2.

Внимание: При сдаче работы возможно придётся отвечать на вопросы о назначении и смысле команд. Справка по команде:

```
$ man <команда> <Enter>
```

В том числе можно получить справку и по самой системе man:

```
$ man man <Enter>
```

3. Прежде всего, убедитесь, что вы находитесь в своём домашнем каталоге. Это можно сделать командой:

```
$ pwd
```

Если вы не в домашнем каталоге, то перейти в домашний каталог.

4. Прочитать в manual'e описание команды *touch*. Создать файл с именем = фио (например, obama_b.txt) командой

```
$ touch <имя_файла>
```

5. Вывести в этот файл вывод команды *pwd*.

6. Добавить в этот файл следующую информацию «Я, <фамилия имя отчество>, группа <группа>, лабораторная №».

7. Прочитать в manual'e описание команды *date*. Добавить в этот файл дату командой «*date*».
8. Добавить в этот файл две пустых строки.
9. Добавить в этот файл вывод команды «*ls -la*».
10. Добавить в этот файл две пустых строки, а затем строку «ПРОФИЛЬ ПОЛЬЗОВАТЕЛЯ <login>:», где *login* — логин пользователя, под которым вы вошли в систему.
11. Прочитать в manual'e описание команды *set*. При помощи команды *set* добавить в этот файл профиль текущего пользователя (переменные среды).
12. Добавить в этот файл две пустых строки.
13. Прочитать в manual'e описание команды *uname*. Добавить в этот файл вывод команды «*uname -a*».
14. Добавить в этот файл две пустых строки.
15. Прочитать в manual'e описание команды *free*. Добавить в этот файл вывод команды «*free*».
16. Добавить в этот файл две пустых строки.
17. Прочитать в manual'e описание команды *df*. Добавить в этот файл вывод команды «*df*».

Порядок сдачи лабораторной работы:

1. По требованию преподавателя повторить работу в лаборатории и объяснить, что, собственно, делал.
2. Продемонстрировать протокол работы.
3. Представить отчёт.

Общие требования к отчёту указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчёт должен содержать следующую информацию:

- а) задание на работу;
- б) распечатку созданного файла с именем <фамилия_латинскими_буквами.txt>;
- в) объяснение (комментарии) проделанной работы.

Дополнительная справочная информация

Общие сведения

Unix — многопользовательская, многозадачная операционная система с разделением времени. В любой момент в системе выполняется множество процессов, каждый процесс

принадлежит некоторому пользователю. **Пользователь** - это объект обладающий определенными правами в системе. Каждый пользователь идентифицируется уникальным идентификатором пользователя (UID — user identifier). Пользователю присваиваются имя и пароль. Пользователь с UID 0 (root) обладает неограниченными правами. Кроме того каждый пользователь входит в одну или несколько групп. Принадлежность к группе добавляет пользователю определенные права в системе. Каждая группа идентифицируется уникальным идентификатором группы (GID — group identifier).

Информация о пользователях хранится в файле /etc/passwd. Каждая строка файла содержит информацию об одном пользователе: регистрационное имя (логин), зашифрованный пароль (в настоящее время это поле не используется, пароль хранится в другом месте), UID, GID, полное имя пользователя, домашний каталог, командная оболочка.

Командная оболочка (командный интерпретатор, shell) — средство интерактивного взаимодействия с системой.

Домашний каталог — каталог, в котором хранятся файлы пользователя. Имя домашнего каталога обычно совпадает с логином. При входе пользователя в систему этот каталог становится текущим для оболочки.

Файловая система

Файловая система — это набор структур данных на носителе + алгоритмы в ОС (драйверы файловой системы), с помощью которых ядро операционной системы организует и представляет пользователям ресурсы внешней памяти системы. Сюда относится память (внешняя, в отличие от «внутренней» - ОЗУ) 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 / (**слэш — корень файловой системы**) и разветвляется, охватывая произвольное число подкаталогов.

Структуры некоторых файловых систем рассмотрены в лекциях.

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 (абсолютными, каноническими) или относительными. Полное имя легко идентифицировать: оно всегда начинается с символа "/".

В любой момент каждый процесс (каждая запущенная программа) привязан к некоторому текущему каталогу — «рабочему каталогу» этой программы. Относительные имена интерпретируются с текущего каталога.

Пример полного имени:

/home/student/andreevas/aas.txt

Однако, если пользователь student находится в своём домашнем каталоге student

(/home/student), то для того, чтобы поиметь доступ к файлу aas.txt не обязательно указывать полное имя, достаточно ввести относительное имя файла aas.txt:

```
andreevas/aas.txt
```

Пример: В каталоге andreevas находятся файлы aas.txt и hosts1. Команда pwd показывает, где вы находитесь:

```
$ pwd
```

```
/home/student/andreevas/
```

Вам нужно добавить содержание файла hosts1 в файл aas.txt. Это делается командой:

```
cat hosts1 >> aas.txt
```

Относительное имя файла может быть очень коротким, например, просто f. Если в имени файла вообще нет знака "/" (слэш), то имя относится к файлу текущего каталога. Если слэш есть (но не в начале имени – как в вышеуказанном примере), то все, что находится слева от первого в имени слэша, расценивается как подкаталог текущего каталога.

Особенности:

а) символ ~ (тильда), как правило, обозначает домашний каталог пользователя; например, если пользователь находится где-то в файловой системе и ему нужно вернуться в свой домашний каталог, то это можно сделать командой

```
cd ~
```

б) каталог . (точка) — текущий каталог; каталог .. (две точки) — вышестоящий каталог.

в) предположим, пользователь student, **находясь в своём домашнем каталоге**, транслировал программу proga и она создалась в его домашнем каталоге (то есть, полный путь к ней /home/student/proga); тогда, чтобы запустить программу proga на выполнение нужно дать команду

```
./proga <Enter>
```

то есть, символы ./ указывают, что путь к программе должен исчисляться от текущего каталога (того каталога, где сейчас находится пользователь).

Иерархическая структура файловой системы может быть произвольного размера. Однако существуют определенные ограничения, зависящие от конкретной операционной системы. Как правило имена файлов и каталогов не должны содержать более 256 символов, а в определении одного (полного) пути не должно быть более 1023 символов. Имена файлов могут состоять из любых символов, за исключением слэша и символа с кодом ноль. Однако на самом деле **в именах файлов НАСТОЯТЕЛЬНО РЕКОМЕНДУЕТСЯ использовать только следующие символы и**

ничего кроме них: **a-z,A-Z,0-9** и спецсимволы «-», «_» и «.», в противном случае возникают проблемы.

Максимальная длина имени файла определяется конкретной системой. Для каждого файла определён владелец этого файла и группа владельца данного файла. Для каждого файла определяются права доступа владельца файла, права доступа группы (приближённых к владельцу), права доступа всех остальных. Есть три типа прав доступа: чтение, запись, выполнение/поиск. Изменить права доступа к файлу может только владелец и суперпользователь (root) командой **chmod**. Изменить владельца файла и группу владельца может только пользователь root командой **chown**.

В ОС Unix существует семь типов файлов:

1) Обычный файл (обозначаются **-**) — это просто последовательность байтов. Обычный файл может содержать выполняемую программу, главу книги, графическое изображение и т.п.

2) Каталоги (обозначаются **d**) могут содержать файлы любых типов в любых сочетаниях. Специальные имена **.** и **..** обозначают соответственно сам каталог и его родительский каталог.

3,4) Файлы устройств символьные и блочные (символьные обозначаются **c**, блочные обозначаются **b**) - позволяют программам взаимодействовать с аппаратными средствами и периферийными устройствами системы. **Файлы устройств находятся в каталоге /dev и нигде больше.** При конфигурировании ядра к нему добавляются те модули, которые знают, как взаимодействовать с каждым из устройств системы. За всю работу по управлению конкретным устройством отвечает специальный программный модуль, называемый 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 передаёт этот запрос соответствующему драйверу устройства.

Каждому типу устройств системы может соответствовать несколько файлов устройств. Поэтому файлы устройств характеризуются двумя номерами: старшим и младшим. Старший определяет драйвер (тип устройства), а младший конкретное устройство (экземпляр устройства этого типа).

Файл устройства является псевдофайлом, он не размещён на диске, о нём есть только запись (строка в индексной таблице), которая используется при доступе к устройству. Первое число, которое стоит в поле длины файла в выводе программы **ls** для файлов устройств – это **major** номер, а второе, после запятой – **minor** номер. Первый из них означает номер типа устройств и одновременно – позицию в ядре (номер строки в таблице драйверов), в которой следует искать адрес драйвера устройства. Второй – номер экземпляра устройства данного типа. Поэтому файлы однотипных устройств в выводе команды **ls** имеют одинаковые **major** номера. Устройство каждого типа имеет свой **major** номер. **Major** номера назначаются соответствующей международной организацией. **Minor** номер определяется ОС в процессе загрузки при обнаружении устройства.

5) Локальные сокеты (sockets) — это файлы специального типа, которые связаны с определёнными структурами в ядре, с помощью которых обеспечивается взаимодействие между процессами в пределах одной ОС. Обращение к ним осуществляется через объект файловой системы (файл типа socket), а не через сетевой порт.

6) Именованные каналы (именованные pipe), также как и сокеты обеспечивают взаимодействие двух процессов, выполняющихся на одной машине (точнее, в пределах одной ОС).

7) Символические ссылки — это запись в каталоге, ссылающаяся на файл с определённым именем. Символическая ссылка содержит путевое имя файла, на который она ссылается, обеспечивая возможность указывать вместо путевого имени файла имя ссылки. То есть, символическая ссылка — это отдельный файл типа "символическая ссылка", и индексный дескриптор этого файла содержит только путь к файлу или каталогу, на который указывает ссылка.

Можно создать символическую ссылку на любой каталог, а также на файл, находящийся в другом разделе Unix. При удалении символической ссылки с файлом или с каталогом, на который она ссылается, ничего не происходит. При удалении файла, на который ссылается символическая ссылка, она "повисает в воздухе", ссылаясь на пустоту. В этом случае при обращении к такой "пустой" ссылке возникнет ошибка file not found, несмотря на то, что сама ссылка будет видна и доступна в списке файлов.

Перенаправление ввода и вывода

Если некоторый процесс намерен производить ввод или вывод информации в файл, то он должен сначала открыть этот файл. При открытии файла процесс получает дескриптор файла — некоторое число (беззнаковое целое), которое используется, в дальнейшем для обращения к файлу. При запуске процесса ему операционной системой передаются дескрипторы трёх открытых файлов: 0 — стандартный ввод (stdin), 1 — стандартный вывод (stdout), 2 — стандартный вывод ошибок (stderr). Как правило все эти дескрипторы указывают на терминал — tty. Оболочка позволяет назначать другие файлы для ввода и вывода при помощи команд перенаправления:

программа < файл

Таким образом, при запуске команды дескриптор 0 будет связан с файлом, т.е. программа будет считывать данные не с клавиатуры, а из файла. Файл будет открыт для чтения.

программа > файл

Здесь при запуске команды дескриптор 1 будет связан с файлом, т.е. программа будет выводить результаты работы не на экран, а в заданный файл. Файл будет открыт для записи, (**!внимание**) если файл существовал, он будет очищен, если нет, то он будет создан.

программа >> файл

При запуске команды дескриптор 1 будет связан с указанным файлом, как и в предыдущем случае.

Однако в данном случае, если файл существовал, то он **не будет перезаписан**, данные будут добавляться в конец файла.

программа `n > файл`

При запуске команды дескриптор с номером `n` будет связан с указанным файлом. Например, если указать `2 > err.log`, то вывод сообщений об ошибках будет производиться в файл `err.log`. Аналогично, можно указывать дескриптор перед операторами перенаправления `>` и `>>`.

программа `n < > файл`

При запуске команды дескриптор с номером `n` будет связан с указанным файлом. Файл будет открыт для чтения и записи.

При перенаправлении можно вместо имени файла указывать дескриптор, для этого следует поставить перед дескриптором знак `&`. Например: `2 > &1` скопирует содержимое дескриптора `2` в дескриптор `1`. Копируемый дескриптор должен быть открыт для чтения или записи в зависимости от операции.

Операции перенаправления выполняются слева направо. В случаях, когда используется копирование дескрипторов, порядок выполнения операций может влиять на результат.

Конвейер

Конвейер — последовательность из одной или более команд, разделённых символом `|` (вертикальная черта - конвейер). Формат конвейера следующий:

[time [-p]] [!] command [| command2 ...]

Стандартный вывод `command` подключается к стандартному вводу команды `command2`. Это подключение производится до выполнения любых перенаправлений.

Если конвейеру предшествует зарезервированное слово **!** (восклицательный знак), то код завершения конвейера равен логическому отрицанию кода завершения последней команды. Иначе код завершения конвейера равен коду завершения последней команды. Интерпретатор ожидает завершения всех команд до того как вернет значение кода завершения.

Если конвейеру предшествует зарезервированное слово **time**, то после завершения выполнения конвейера будет выведена информация о времени выполнения конвейера и о затраченном времени процессора в режимах пользователя и системы.

Каждая команда в конвейере выполняется как отдельный процесс (т.е. в подоболочке).

Переменные окружения и команды

У каждого процесса имеется область памяти, называемая программным окружением (`program environment`) — это набор строк, заканчивающийся нулевым символом. Эти строки называются переменными окружения. Каждая строка имеет вид: имя переменной = значение. Имя

переменной может состоять из алфавитно-цифровых символов и знака подчёркивания. Цифра не может быть первым символом имени. Присвоение значения переменной в оболочке производится следующим образом:

Имя = Значение

Для того, чтобы значение переменной передавалось процессам, порождаемым оболочкой, следует использовать встроенную команду **export**. Следующие две команды помечают переменные VAR и TST как экспортируемые и присваивают переменной TST значение /usr/doc:

```
export VAR
```

```
export TST=/usr/doc
```

Для того, чтобы просмотреть значения переменных окружения можно использовать команду **set**, которая выводит значения всех переменных окружения.

Для того чтобы получить значение переменной, перед её именем указывается знак доллара. Такое выражение будет заменяться интерпретатором на значение переменной. Например, команда **echo** выводит в стандартный вывод свои аргументы, тогда следующее выражение:

```
echo TST=$TST
```

выведет на экран TST=/usr/doc (при условии, что значение переменной TST – /usr/doc).

touch - изменяет временные штампы файла, синтаксис команды:

```
touch [-acm] [-r ref_file|-t время] [--] файл...
```

в версии GNU:

```
touch [-acfm] [-r файл] [-t decimtime] [-d time] [--time={atime,access,use,mtime,modify}]
```

```
 [--date=время] [--reference=файл] [--no-create] [--help] [--version] [--] файл...
```

Команда изменяет время последнего доступа и/или время последней модификации каждого заданного файла. Эти временные штампы устанавливаются в текущее время; или, если задана опция -r, то эти штампы устанавливаются в те же, что имеет файл ref_file; или, если задана опция -t, то эти штампы устанавливаются на заданное время. Оба штампа изменяются, если не задана ни одна из опций -a и -m или если заданы они обе. Если задана только опция -a или только -m, то изменяться будет, соответственно, только время последнего доступа или время последней модификации. Если заданный файл еще не существует, то он создается (как пустой файл с правами доступа 0666, с учетом umask), если не задана опция -c.

Опции:

-a

Изменить время последнего доступа к файлу.

-c

Не создавать файл.

-m

Изменить время последней модификации файла.

-r ref_file

Использовать соответствующий временной штамп от файла ref_file в качестве нового значения для изменяемого временного штампа (или штампов).

-t время

Использовать заданное время в качестве нового значения для изменяемого временного штампа (или штампов). Аргумент является десятичным числом вида [[ВВ]ГГ]ММДДччмм[.СС] с обозначениями (ВВ - век, ГГ - год, ММ - месяц, ДД - день, чч - часы, мм - минуты, СС - секунды). Если ВВ не задан, то год ВВГГ берется из диапазона 1969-2068. Если СС не задано, то берется 0. Секунды могут быть заданы в диапазоне 0-61, чтобы можно было указать високосную секунду. Считается, что результирующее время соответствует часовому поясу, заданному в переменной окружения TZ. Если в результате получилось время до 1 января 1970 года, то будет выдана ошибка.

uname - сообщает информацию о данном компьютере и операционной системе, синтаксис:

uname [Опции]

Опции:

-a, --all

выводит подробную информацию в следующем порядке:

-s, --kernel-name

выводит название ядра операционной системы

-n, --nodename

выводит сетевое название узла (сетевое имя компьютера)

-r, --kernel-release

выводит выпуск ядра операционной системы

-v, --kernel-version

выводит версию ядра операционной системы, включая дату выпуска

-m, --machine

выводит сведения о типе компьютера

-p, --processor

выводит тип процессора для данного компьютера

-i, --hardware-platform

выводит сведения о платформе компьютера

-o, --operating-system

выводит тип операционной системы

df - отчет об использовании файловой системы, синтаксис:

```
df [ОПЦИЯ]... [ФАЙЛ]...
```

Команда показывает количество места на диске, доступное в файловой системе, в которой содержатся указанные ФАЙЛЫ. Если ни один ФАЙЛ не указан, показывается доступное место на всех смонтированных файловых системах. Размеры указаны в блоках по 1 кб по умолчанию, за исключением заданной переменной окружения POSIXLY_CORRECT (в этом случае используются 512-байтовые блоки).

Если ФАЙЛ указан как абсолютный путь к узлу, в который смонтирована файловая система, df показывается доступное место в этой файловой системе, а не содержимое файловой системы узла устройства (которая всегда показывается как корневая файловая система).

Опции:

-a, --all

включать виртуальные файловые системы

-B, --block-size=РАЗМЕР использовать блоки указанного РАЗМЕРА (в байтах)

-h, --human-readable

печатать размеры в удобочитаемом формате (напр., 1K 234M 2G)

-H, --si

то же, но использовать степени 1000, а не 1024

-i, --inodes

вывести информацию об индексных дескрипторах, а не об использовании блоков

-k аналог **--block-size=1K**

-l, --local

перечислить только локальные файловые системы

-P, --portability

выводить в формате POSIX

-t, --type=ТИП

перечислить только файловые системы указанного ТИПА

-T, --print-type

печатать тип файловой системы

-x, --exclude-type=ТИП

исключить файловые системы указанного ТИПА

Лабораторная работа № 5

Тема: ТЕРМИНАЛ: РЕДАКТОР Vi (Vim)

Цель: Научиться редактировать файлы с помощью редактора vim

Задание: Изучить редактор vim.

Указания к выполнению работы:

Предполагается, что вы работаете в текстовом (терминальном) режиме. Если вы ещё в графическом (в KDE, Gnome или где-то ещё), то перейдите в текстовый режим. Для этого нажмите одновременно клавиши Ctrl-Alt-F2

1. В левом верхнем углу терминала видите приглашение вида

```
<имя_компа> login:
```

2. Входим в систему, вводим логин и пароль.

Появляется приглашение вида:

```
<login>@<имя_компа>:~$
```

Прочтите пункт 2 задания на лабораторную работу № 2.

Внимание: При сдаче работы возможно придётся отвечать на вопросы о назначении и смысле команд.

Справка по команде:

```
$ man <команда> <Enter>
```

В том числе можно получить справку и по самой системе man:

```
$ man man <Enter>
```

3. Прежде всего, убедитесь, что вы находитесь в своём домашнем каталоге. Это можно сделать командой:

```
$ pwd
```

Если вы не в домашнем каталоге, то перейти в домашний каталог.

4. Создать файл с именем = фио (например, obama_b.txt) командой

```
$ vim <имя_файла>
```

5. Ввести следующую информацию «Я, <фамилия имя отчество>, группа <группа>, лабораторная № 4».

6. Добавить в этот файл две пустых строки.

7. Ввести следующую информацию (выделено курсивом):

= = =

Запуск:

***vim** имя_файла*

Перемещение по файлу:

клавиши hjkl,

или клавиши со стрелками вверх-вниз-вправо-влево, если они есть на вашей клавиатуре.

Редактирование:

i - начать ввод текста перед курсором,

a - начать ввод текста после курсора,

<esc> (<Ctrl><[>) - выход из режима редактирования.

Выход из редактора:

:q!<Enter> выйти без сохранения файла

:wq<Enter> сохранить файл и выйти из редактора

Выйти можно только когда редактор находится в командном режиме, то есть, прежде чем вводить «:», нужно сначала нажать клавишу <esc>.

Если вводился русский текст, то нужно переключить раскладку клавиатуры, прежде чем вводить команды.

= = =

8. Выйти из редактора с сохранением.

9. Добавить в этот файл дату командой «date».

Порядок сдачи лабораторной работы:

1. По требованию преподавателя повторить работу в лаборатории и объяснить, что, собственно, делал.
2. Продемонстрировать протокол работы.
3. Представить отчёт.

Общие требования к отчёту указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчёт должен содержать следующую информацию:

- а) задание на работу;
- б) распечатку созданного файла с именем <фамилия_латинскими_буквами.txt>;
- в) объяснение (комментарии) проделанной работы.

Дополнительная справочная информация

Знакомство с редактором Vim

Одним из самых старых текстовых редакторов является редактор vi. Этот редактор обладает несколько своеобразным интерфейсом и, поначалу, работа с ним вызывает у неопытного пользователя серьёзные затруднения, но тем не менее этот редактор очень популярен и многие тысячи людей используют именно его для редактирования текстов. Практически в любой Unix совместимой системе имеется какая-либо реализация vi или vim. **Vim отличается от vi тем, что понимает локализацию, то есть, умеет работать не только с латинскими буквами, но и с буквами других алфавитов.**

Для освоения редактора vim запустите команду vimtutor и выполните упражнения, содержащиеся в открывшемся файле. Если Вам не очень понятен английский язык, воспользуйтесь приведённой ниже краткой справкой по vi.

Редактор vi

Vi экранный текстовый редактор. Большая часть экрана используется для отображения редактируемого файла. Последняя строка экрана используется для ввода команд и вывода различной информации. Редактор может находиться либо в режиме редактирования, либо в режиме ввода команд. Для того, чтобы совершать какие либо действия Вы должны находиться в нужном режиме. После запуска редактор находится в командном режиме. Для перехода из режима редактирования в командный режим используется клавиша Esc. Для того, чтобы начать редактирование файла используется команда

vi имя_файла

Основные возможности в командном режиме:

• Перемещение по файлу:

h, left-arrow

переместить курсор влево на один символ

j, down-arrow

переместить курсор вниз на одну строку

k, up-arrow

переместить курсор вверх на одну строку

l, right-arrow

переместить курсор вправо на один символ

/text<cr>

найти строку text в файле и поместить курсор на ее первый символ. После этого можно использовать клавиши n и Shift-n для перемещения к следующему или предыдущему включению строки.

• **Переход в режим редактирования:**

i

начать ввод текста перед курсором

a

начать ввод текста после курсора

o

вставить строку после текущей и начать ввод текста в ней

O

вставить строку перед текущей и начать ввод текста в ней

Выход из режима редактирования — клавиша <esc> (<Ctrl><[>).

• **Копирование, вставка и удаление:**

yy u\$ uw

скопировать строку, строку от позиции курсора до конца, слово.

dd d\$ dw

удалить строку, строку от позиции курсора до конца, слово.

X

удалить символ

p

вставить содержимое буфера после курсора

P

вставить содержимое буфера перед курсором

u

Отменить последнюю операцию

• **Сохранение и чтение файлов, выход из редактора:**

:w<Enter>

сохранить файл

:w filename<Enter>

сохранить файл под указанным именем

:r filename<Enter>

вставить содержимое указанного файла

:q<Enter>

выйти из редактора

:wq<Enter>

сохранить файл и выйти из редактора

:q!<Enter>

выйти без сохранения файла

Почему важно уметь работать с редактором vi

Если вы всегда работаете только на полноценных терминалах (то есть, и дисплей большой цветной, и мышка есть, и клавиатура полная, как минимум 101 клавиша), то знать **vi** вам не обязательно. И Word'a с вас вполне достаточно. Или Writer'a.

Однако, если вам приходится работать

- с системами удалённо,

- или на всяком-разном оборудовании, которому мышка не предусмотрена, а клавиатура - урезанная всего лишь с тридцатью клавишами или даже меньше и функциональные клавиши и рядом с ней не лежали, то запустить Word не удастся. Как, впрочем, и Writer. А возможно не удастся запустить и более простые вещи, типа kate или nano.

А, вот, **vi** будет работать даже в этих условиях!

Самые минимальные знания vi, которые легко запомнить и которые могут вас выручить в кризис

Запуск:

vi имя_файла

Перемещение по файлу:

клавиши h,j,k,l,

или клавиши со стрелками вверх-вниз-вправо-влево, если они есть на вашей клавиатуре.

Редактирование:

i - начать ввод текста перед курсором,

a - начать ввод текста после курсора,

<esc> (<Ctrl><[>) - выход из режима редактирования.

Выход из редактора:

:q!<Enter> выйти без сохранения файла

:wq<Enter> сохранить файл и выйти из редактора

Выйти можно только тогда, когда редактор находится в командном режиме, то есть, прежде чем вводить «:», нужно сначала нажать клавишу <esc>.

Если вводился русский текст, то нужно переключить раскладку клавиатуры, прежде чем вводить команды.

Лабораторная работа № 6

Тема: ТЕРМИНАЛ: АТТРИБУТЫ ФАЙЛОВ

Цель: Научиться читать и изменять атрибуты файлов

Задание: Изучить возможности просмотра и изменения атрибутов файлов

Указания к выполнению работы:

Предполагается, что вы работаете в текстовом (терминальном) режиме. Если вы ещё в графическом (в KDE, Gnome или где-то ещё), то перейдите в текстовый режим. Для этого нажмите одновременно клавиши **Ctrl-Alt-F2**

1. В левом верхнем углу терминала видите приглашение вида

```
<имя_компа> login:
```

2. Входим в систему: вводим логин и пароль.

Появляется приглашение вида:

```
[login@имя_компа ~]$
```

Прочтите пункт 2 в задании на лабораторную работу №2.

Далее в задании будет опускаться содержимое квадратных скобок ([<login>@<имя компа>]), а будет указываться только символ «\$».

Внимание: При сдаче работы, возможно, придётся отвечать на вопросы о назначении и смысле команд. Справка по команде:

```
$ man <команда> <Enter>
```

В том числе можно получить справку и по самой системе man:

```
$ man man <Enter>
```

3. Прежде всего, убедитесь, что вы находитесь в своём домашнем каталоге. Это можно сделать командой:

```
$ pwd
```

Если вы не в домашнем каталоге, то перейти в домашний каталог.

4. Перейти в режим root.

5. Выполнить команды

```
cp /var/log/syslog/messages messages
```

```
cp /var/log/dmesg dmesg
```

```
cp /etc/passwd passwd
```

6. Прочитать в manual'е описание команды *chown*. Сменить хозяина скопированных файлов на себя (на свой логин) командой *chown*.

7. Выйти из *root*.

8. Создать файл с именем = фио (например, *barak_o.txt*) командой

```
$ touch <имя_файла>
```

9. Ввести следующую информацию «Я, <фамилия имя отчество>, группа <группа>, лабораторная №5».

10. Добавить в этот файл две пустых строки.

11. Прочитать в manual'е описание команды *tail*. Добавить в этот файл вывод следующих команд

```
tail messages
```

```
tail dmesg
```

12. С помощью команды **cut** выделить из файла *passwd* первое и третье поле, вывод этой команды отсортировать по алфавиту и добавить в созданный ранее файл фио.

11. Добавить в файл фио количество строк, слов и байт, содержащихся в файлах *messages*, *dmesg* и *passwd*.

12. Добавить в этот файл дату командой «*date*».

Порядок сдачи лабораторной работы:

1. По требованию преподавателя повторить работу в лаборатории и объяснить, что, собственно, делал.

2. Продемонстрировать протокол работы.

3. Представить отчёт.

Общие требования к отчёту указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчёт должен содержать следующую информацию:

а) задание на работу;

б) распечатку созданного файла с именем <фамилия_латинскими_буквами.txt>;

в) объяснение (комментарии) проделанной работы.

Дополнительная справочная информация

Команды обработки текстовых файлов

Здесь приведены краткие описания некоторых команд (более подробно: `man <команда>`)

head [-n count] [file...]

Выводит первые count строк файла (по умолчанию 10).

tail [-f] [-n count] [file...]

Выводит последние count строк файла (по умолчанию 10). Если указан ключ `-f`, то ожидает добавления данных в конец файла и выводит их. Например, с помощью команды

```
tail -f /var/log/syslog/messages
```

вы можете контролировать, что пишется в общий протокол системы.

comm [-123] file1 file2

Считывает файлы `file1` и `file2`, которые должны быть предварительно отсортированы, и выводит три колонки текста. В первой колонке строки имеющиеся только в `file1`, во второй имеющиеся только в `file2`, в третьей имеющиеся в обоих файлах. Параметры `-1`, `-2`, `-3` позволяют подавить вывод соответствующей колонки.

cut {-c list|-f list [-d delim]} [file...]

Вырезает из каждой строки указанные символы и выводит их. Аргумент `list` — список чисел и диапазонов чисел разделенных запятыми. Для `-c` числа указывают номера символов в строке, подлежащих выводу, для `-f` номера полей в таблице. Поля таблицы разделены символом `delim` (по умолчанию символ табуляции). Пример: выделить и вывести на экран 3, 6 и 7 поля из файла `passwd`

```
cut -d : -f 3,6,7 passwd
```

sort [-c|-m] [-o output] [-urnb] [file...]

Производит сортировку строк файлов, их объединение или проверяет, отсортирован файл или нет. Значения параметров:

- c только проверить правильность сортировки
- m объединить предварительно отсортированные файлы
- u удалять повторяющиеся элементы
- r сортировка в обратном порядке

- n сортировка чисел
- b игнорировать лидирующие пробелы
- o file производить вывод в файл file

wc [-c|-m][-lw][file...]

Читает один или более входных файлов и, по умолчанию, выводит число символов новой строки, слов и байт содержащихся в каждом файле на стандартный вывод. Значения параметров:

- c вывести число байт в каждом входном файле
- l вывести число символов новой строки в каждом входном файле
- m вывести число символов в каждом входном файле
- w вывести число слов в каждом входном файле

iconv -f codepage1 -t codepage2 [file...]

Конвертирует файлы из кодировки codepage1 в кодировку codepage2 и выводит результат на стандартный вывод. Например, команда

```
iconv -f windows-1251 -t koi8-r file
```

перекодирует файл из кодировки CP1251 в кодировку KOI8-R.

Лабораторная работа № 7

Тема: УПРАВЛЕНИЕ ПРОЦЕССАМИ

Цель: Научиться работать с процессами из терминала

Задание: Изучить команды работы с вычислительными процессами

Указания к выполнению работы:

Предполагается, что вы работаете в текстовом (терминальном) режиме. Если вы ещё в графическом (в KDE, Gnome или где-то ещё), то перейдите в текстовый режим. Для этого нажмите одновременно клавиши **Ctrl-Alt-F2**

1. В левом верхнем углу терминала видите приглашение вида

```
<имя_компа> login:
```

2. Входим в систему: вводим логин и пароль.

Появляется приглашение вида:

```
[login@имя_компа ~]$
```

Прочтите пункт 2 задания на лабораторную работу 2.

Далее в задании будет опускаться содержимое квадратных скобок ([login@имя компа ~]), а будет указываться только символ «\$».

Внимание: При сдаче работы, возможно, придётся отвечать на вопросы о назначении и смысле команд.

Мануал по команде:

```
$ man <команда> <Enter>
```

В том числе можно получить справку и по самой системе man:

```
$ man man <Enter>
```

3. Прежде всего, убедитесь, что вы находитесь в своём домашнем каталоге. Это можно сделать командой:

```
$ pwd
```

Если вы не в домашнем каталоге, то перейти в домашний каталог.

4. Создать файл с именем = фио (например, ivanov_i.txt) командой

```
$ touch <имя_файла>
```

5. Ввести в этот файл следующую информацию «Я, <фамилия имя отчество>, группа <группа>, лабораторная №8».
6. Добавить в этот файл две пустых строки.
7. Добавить в этот файл вывод команды **ps** так, чтобы были видны **id** пользователя, запустившего процесс, **id** процесса, **id** родительского процесса, приоритет процесса, использование памяти процессом, использование CPU процессом, терминал процесса, команда запуска процесса.
8. Добавить в файл отчёта две пустых строки.
9. Добавить в файл отчёта информацию о процессах запущенных пользователем root. Вывод должен быть отсортирован по номеру процесса. Как это сделать, смотри man ps.
10. Добавить в файл отчёта информацию о процессах запущенных пользователем student так, чтобы были видны **id** пользователя, запустившего процесс, **id** процесса, **id** родительского процесса, приоритет процесса, использование памяти процессом, использование CPU процессом, терминал процесса, команда запуска процесса.
Вывод должен быть отсортирован по номеру процесса. Выборку процессов, принадлежащих пользователю student, делать с помощью команды grep. Сортировку вывода можно сделать с помощью программы sort.
11. Перейти на другой terminal (командой Alt-F3). Ввести команду top так, чтобы контролировать только процессы пользователя student (см. man top).
12. Вернуться во второй терминал. В этом терминале продемонстрировать работу команды kill — найти процесс top, запущенный в третьем терминале и убить его. Вывести результат в отчёт.
13. Вывести в отчёт результат выполнения команд


```
tty
w
uname -a
uptime
```
14. Добавить в этот файл дату командой «date».

Порядок сдачи лабораторной работы:

1. По требованию преподавателя повторить работу в лаборатории и объяснить, что, собственно, делал.
2. Продемонстрировать созданный файл, который, по сути, является протоколом работы.
3. Представить отчёт.

Общие требования к отчёту указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчёт должен содержать следующую информацию:

- а) задание на работу;
- б) распечатку созданного файла <фамилия_латинскими_буквами.txt>;
- в) объяснение (комментарии) проделанной работы.

Дополнительная справочная информация

Управление процессами

1. Процессы

Процесс (process) — совокупность области адресного пространства, в котором выполняется запущенная программа, и PCB — блока управления процессом. Любой процесс может запускать другие процессы. То есть, есть процессы родительские и есть процессы — потомки. Таким образом, процессы в среде Unix образуют иерархическую структуру. На вершине этой структуры находится процесс `init`, являющийся предком всех остальных процессов.

1.1. Атрибуты 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 имён.

Текущий корневой каталог - это каталог от которого производится разрешение абсолютных имён; процесс не имеет доступа к файлам находящимся выше корневого каталога; Это свойство используется при создании «песочниц» для процессов.

Идентификаторы пользователя и группы - с каждым процессом связаны действительные идентификаторы пользователя (real user ID) 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 своё процессорное время другим процессам; чем больше значение nice, тем ниже приоритет процесса.

2. Основные сведения о работе с процессами

2.1. Создание процессов — вызовы fork и exec

Основным средством для создания процессов является системный вызов **fork**. При выполнении данного вызова ядро создаёт новый процесс, который является копией процесса, вызвавшего fork. Созданный процесс называется дочерним, а процесс осуществивший вызов fork — родительским. В дочернем процессе вызов возвращает значение ноль, а в родительском он возвращает идентификатор дочернего процесса. Дочерний процесс наследует дескрипторы открытых файлов и значения переменных окружения родительского процесса.

Другой системный вызов для работы с процессами — **exec**. Он позволяет сменить выполняемую программу. Вызову exec передаются в качестве аргументов имя программы, которую надо выполнить и список её аргументов. При выполнении вызова в пространство памяти вызывающего процесса загружается новая программа, которая запускается сначала. При выполнении вызова exec дескрипторы открытых файлов сохраняют своё значение.

2.2. Завершение процессов

Для завершения процесса используется системный вызов **exit**. Вызов имеет целочисленный аргумент, называемый кодом завершения процесса. Как правило, при успешном завершении процесса код завершения равен нулю, а в случае возникновения ошибки отличен от нуля. Родительский процесс может получить статус завершения дочернего процесса, выполнив системный вызов **wait** или **waitpid**. Если родительский процесс завершается раньше дочернего (не сделав вызов wait или waitpid), то дочерний процесс по завершению переходит в состояние **зомби**, когда он выполняться уже не может, а ресурсы занимает.

3. Механизмы межпроцессного взаимодействия

Unix имеет большое число механизмов межпроцессного взаимодействия. Наиболее популярными средствами являются сигналы, программные каналы (pipes), именованные каналы (FIFO), shared memory, сообщения и сокеты. Особенно большое распространение получили последние. Большинство современного распределённого программного обеспечения используют для взаимодействия сокеты, информация о которых приводится в третьей части пособия.

Средства взаимодействия могут иметь локальный характер (то есть, могут передавать информацию только между процессами, запущенными в одной операционной системе) и глобальный (то есть, могут передавать информацию между процессами, запущенными в разных операционных системах).

3.1. Сигналы

Сигналы обеспечивают простой метод прерывания работы процессов. Сигналы используются в основном для обработки исключительных ситуаций. Процесс может определять действия выполняемые при поступлении сигнала, блокировать сигналы, посылать сигналы другим процессам. Существует более двадцати различных сигналов. Основные:

SIGCHLD - сигнал о завершении дочернего процесса.

SIGHUP - сигнал освобождения линии. Посылается всем процессам, подключенным к управляющему терминалу при отключении терминала. Многие демоны при получении данного сигнала заново просматривают файлы конфигурации и перезапускаются.

SIGINT - сигнал посылается всем процессам сеанса, связанного с терминалом, при нажатии пользователем клавиши прерывания (CTRL-C).

SIGTERM - сигнал приводит к немедленному прекращению работы получившего сигнал процесса.

SIGKILL - сигнал приводит к немедленному прекращению работы получившего сигнал процесса. В отличие от SIGTERM процесс не может блокировать и перехватывать данный сигнал.

SIGSEGV - сигнал посылается процессу, если тот пытается обратиться к неверному адресу памяти.

SIGSTOP - сигнал приводящий к остановке процесса. Для отправки сигнала SIGSTOP активному процессу текущего терминала можно воспользоваться комбинацией клавиш (CTRL-Z).

SIGCONT - сигнал возобновляющий работу остановленного процесса.

SIGUSR1, SIGUSR2 - сигналы определяемые пользователем.

Для того, чтобы отправить процессу сигнал можно использовать команду **kill**. Для того, чтобы процесс мог отправить сигнал другому процессу необходимо чтобы эффективные идентификаторы пользователя у посылающего процесса и у процесса получателя совпадали. Процессы с эффективным идентификатором пользователя равным нулю могут посылать сигналы любым процессам.

Пример: Предположим, что мы в терминале запустили программу `top`. В другом терминале в выдаче программы `ps` мы видим, что `pid` процесса `top` равен 8628. Тогда, чтобы прибить процесс `top`, нужно дать команду `kill 8628`.

3.2. Каналы

Часто возникает ситуация когда два процесса последовательно обрабатывают одни и те же данные. Для обеспечения передачи данных от одного процесса к другому в подобных ситуациях используются программные каналы. Программный канал (**pipe**) служит для установления связи, соединяющей один процесс с другим. Запись данных в канал и чтение из него осуществляются при помощи системных вызовов `write` и `read`, т.е. работа с каналами аналогична работе с файлами. Для создания программного канала используется системный вызов **pipe**. Вызов возвращает два дескриптора файлов, первый из которых открыт для чтения из канала, а второй для записи в канал.

Каналы используются, например, при организации конвейера. При выполнении команды:

```
find /usr/bin -name a* | sort
```

создается канал, команда `find` выводит в него результаты своей работы, а команда `sort` считывает из этого канала данные для сортировки.

Главным недостатком программных каналов является то, что они могут использоваться только для связи процессов имеющих общее происхождение (например, родительский процесс и его потомок). Другой недостаток - ограниченное время существования канала (программные каналы уничтожаются после завершения обращающегося к ним процесса).

Именованные каналы идентичны программным в отношении записи и чтения данных, но они являются объектами файловой системы. Именованный канал имеет имя, владельца и права доступа. Открытие и закрытие именованного канала осуществляется как открытие и закрытие любого файла, но при чтении и записи он ведёт себя аналогично каналу.

Для создания именованного канала используется команда **mkfifo**. Если некоторый процесс открывает именованный канал для записи, то этот процесс блокируется до тех пор, пока другой процесс не откроет этот канал для чтения, и наоборот.

4. 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, подключённых к терминалу. Значения параметров следующие:

- a вывести информацию о процессах всех пользователей;
- x вывести информацию о процессах, не подключённых 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, -u 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, разделённых 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.

Для различных систем параметры и ключевые слова могут сильно различаться. Подробности об использовании ps на конкретной системе можно получить при помощи команды man ps.

kill [-s signal | -signal] pid

Посылает сигнал указанному процессу. Если значение сигнала опущено, предполагается SIGTERM. signal — символическое имя сигнала без префикса SIG, либо номер сигнала. Пример:

kill -HUP 172 — послать сигнал SIGHUP процессу с идентификатором 172.

nice [-nice] команда [аргументы]

Выполняет команду с меньшим (уменьшенным, пониженным) приоритетом. Если nice не задан, то предполагается 10. Значение nice может быть от -20 (наивысший приоритет) до 20 (наименьший приоритет). Отрицательные числа задаются как -nice. Увеличение приоритета может выполнить только суперпользователь.

Пример:

nice -10 john users — запустить программу john с пониженным приоритетом.

mkfifo [-m режим_доступа] имя

Создаёт именованный канал с указанным именем и режимом доступа.

tty

Выводит имя текущего терминала.

who [am i]

Выводит список пользователей работающих в системе.

uname [-amnrsv]

Выводит информацию о системе.

uptime

Выводит время работы системы и её среднюю загрузку за последние 5, 10 и 15 минут.

5. Средства оболочки для работы с процессами

Список — последовательность из одного или более конвейеров, разделённых операторами `;`, `&`, `&&` или `||`. Более высокий приоритет у операторов `&&` и `||`. Если команда завершается оператором `&`, то оболочка выполняет ее в фоновом режиме. Если между двумя командами стоит оператор `&&`, то вторая команда будет выполнена только в том случае, если первая завершится успешно. Если между двумя командами стоит `||`, то вторая команда будет выполнена только в том случае, если код завершения первой команды отличен от нуля. Если команды разделены точкой с запятой, то вторая команда будет выполнена после завершения первой, независимо от результата выполнения первой команды.

Оболочка содержит несколько встроенных команд для работы с процессами:

wait [pid]

Ожидает завершения процесса с указанным идентификатором. Если идентификатор опущен, то ожидает завершения всех процессов запущенных оболочкой.

exec команда [аргумент]...

Указанная команда заменяет оболочку и получает в качестве параметров заданные аргументы.

exit [n]

Приводит к завершению оболочки с кодом завершения n. Если аргумент опущен, то код завершения ноль.

trap [действие условие...]

Устанавливает обработчик события. Условие либо EXIT, либо имя сигнала без префикса SIG. EXIT соответствует завершению работы оболочки. Если действие равно “-”, то обработчик сбрасывается

в значение по умолчанию. Например, после выполнения команды:

```
trap "echo PRESSED" INT
```

оболочка будет выводить слово PRESSED после каждого нажатия клавиш CTRL-C. (Нажатие клавиш CTRL-C приводит к отправке сигнала SIGINT процессам, подключенным к терминалу).

Более подробную информацию по всем выше упомянутым командам и функциям можно посмотреть в `man'e`.

Лабораторная работа № 8

Тема: УСТАНОВКА LINUX НА FLASH-НОСИТЕЛЬ

Цель: Научиться устанавливать ОС Linux на flash-носитель

Задание: Установить ОС Linux на flash-носитель.

Указания к выполнению работы

1. Наличие загрузочной флешки — необходимый атрибут системного/сетевого администратора. Но и для обычного продвинутого пользователя она также необходима, поскольку достаточно часто возникают задачи, требующие загрузки системы с другого носителя. Flash-диск в силу миниатюрности и достаточно большого объёма памяти вполне подходит для создания внешнего загружаемого устройства.

Для подобной установки есть специально созданные дистрибутивы, как правило, небольшого объёма: puppy — 130 Mb, frenzy — 200-250 Mb, DSL ~ 50-70Mb, feather ~ 120Mb и другие. Некоторые из них специально созданы как средства системного администратора.

2. Рекомендуемый дистрибутив — puppy (сайт www.puppyrus.org). Почему: при загрузке автоматически распознаются все виндовые и Linux'овые разделы (нескрытые, которые реально можно смонтировать), наличествующие на винчестере ПЭВМ и их иконки выводятся на Desktop.

Рекомендуемая версия — **puppyrus-rga**. Это последняя (самая свежая) русская версия; на ftp-сервере можно видеть много разных версий puppy, которые суть либо адаптации английских версий, в работе менее удобных, либо несколько устарели.

Puppy устанавливается на файловую систему FAT32, то есть, после установки флешка остаётся в прежнем состоянии, только на неё добавляется ещё четыре файла, общим объёмом ~ 140 Мбайт.

3. Руководство по установке имеется на сайте <http://docs.puppyrus.org/setups/start> в разделе USB Flash.

Основная идея установки. Puppy рекомендуется ставить методом, так называемой, frugal-установки. Смысл её в том, что на флешку копируются три файла:

vmlinuz — ядро операционной системы, 2-3 Мб;

initrd.gz — образ RAM-диска, сжатый архив, который при запуске распаковывается в область оперативной памяти, организуемой как раздел диска; в архиве — нужное для работы ядра системное ПО; объём — несколько десятков Мб;

pup_xxx.sfs — это тоже архив, содержащий прикладное ПО — те самые программы, которые будут видны из главного меню.

Процесс копирования этих трёх файлов как раз и организует «Универсальный инсталлятор Puppy» при установке puppy на флешку. Это вполне можно сделать вручную, выделив эти файлы из iso-образа puppy. На флешке должна быть файловая система FAT32 (как правило, так и есть) или файловая система EХТ-2 («ну, это вряд ли»). После копирования этих файлов, останется...

4. Установить загрузчик. Рекомендуется grub-2 и рекомендуется это сделать из установленного puppy. В главном меню выбираем пункт «Конфигурирование загрузчика Grub4dos», в открывшемся окне указываем флешку (не ошибитесь!) и ждём примерно минуту. Иногда система спрашивает, не хотите ли что-нибудь подправить в конфигурационном файле. Если плохо соображаете, то не стоит ничего подправлять («а пусть она . . .»).

Но можно поставить и другой загрузчик, например, стандартный CD-шный sysLinux. Разные варианты описаны на сайте <http://docs.puppyrus.org/setups/start>

5. Но можно поставить на Flash и обычный дистрибутив. Например, AltLinux (Центр управления системой → Система → Создание загружаемого usb устройства). Однако это потребует флешки большого объёма и придётся создавать под Линукс отдельный раздел на флешке. Не рекомендуется. И вообще, большие дистрибутивы с флэшки работают медленно!

Порядок сдачи лабораторной работы:

1. Продемонстрировать работу с установленной на флэшке системой:
 - загрузиться с флешки,
 - выполнить на системе действия, по указанию преподавателя (как минимум, настройка Интернета и работа с hdd программой fdisk).
2. Представить отчёт.

Общие требования к отчёту указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчёт должен содержать следующую информацию:

- а) задание на работу;
- б) описание процесса установки системы на flash-диск;
- в) краткое описание возможностей установленного на флешку дистрибутива.

Дополнительная справочная информация

Виды установки: FULL vs FRUGAL

Установка **FULL** — полная установка. Всех сбивает с толку название «полная», но в случае с Pupru это уступка для особо слабых машин, особенно с малым объёмом оперативной памяти, когда из-за свопирования машина начинает заметно тормозить. Реально скорость **FULL** выше всего лишь приблизительно на 20%, зато к ней, и именно к ней, справедливы упреки в небезопасности постоянной работы под root-ом в графическом режиме. Кроме того, при **FULL**-установке отсутствует `/initrd`, который является точкой монтирования для sfs-модулей, что приводит к необходимости их ручной распаковки и установки, а удаление установленных таким образом программ — очень сложный процесс.

Установка **FRUGAL** — ошибочно переводится как «формальная», более точно будет «упрощённая», «лёгкая». Это относится к лёгкости процедуры установки Pupru на жёсткий диск таким методом, которая сводится к копированию трёх-пяти файлов (зависит от версии), а не к работе установленной таким методом системы.

Фактически при такой установке происходит эмулирование загрузки с LiveCD, что для Pupru является основным режимом работы. **FRUGAL**-установка обеспечивает:

1. Работу с sfs-модулями, как постоянно подключенными, так и «на одну сессию», так называемое «горячее подключение».
2. Обеспечивает повышенную безопасность, так как sfs-файлы, будучи архивами, подключаются к системе «только для чтения» (ro), что исключает повреждение их содержимого случайными действиями пользователя. Такой файл можно только намеренно переименовать или удалить, но и восстановить не составит труда. Просто копируем на место удалённого sfs его «эталон» с CD.
3. Обеспечивает лёгкий бэкап, так как все изменения в системе хранятся в `pup_save.2fs`, то его можно просто скопировать в другое место или под другим именем, и в случае серьёзного сбоя заменить «испорченный» save на «дубликат».

Для этого существует опция загрузки `rfix=ram`, которая добавляется в строку `kernel` файла конфигурации `menu.lst` загрузчика `grub`, при загрузке с LiveCD — в нижнюю строку загрузочного меню `boot`: (здесь пишется `pupru rfix=ram`). После этого происходит загрузка «с чистого листа» и можно проводить «восстановительные работы».

Для экономии места можно копировать только содержимое save-файла

```
cp -r /initrd/pup_rw /mnt/hdaN/savedir
```

Правда такой метод усложняет восстановление, так как вместо простой замены файла

нужно заменить его содержимое, а для этого «неисправный» save надо примонтировать, очистить и скопировать сохранённое из savedir. Этот метод оправдан только при малом объёме жёсткого диска.

Существует ещё метод обеспечения безопасности — создание собственного sfs. Для этого достаточно скопировать содержимое /initrd/pup_ro2 в отдельно созданную директорию (например root-dir), «наложить» сверху содержимое /initrd/pup_rw и создать свой sfs командой

```
mksquashfs root-dir pup_301 -mydisk.sfs
```

После создания sfs (процесс не быстрый) заменить им «штатный» sfs. Тогда необходимость в pup_save.2fs и zdrv-301.sfs отпадает.

Два замечания. Копирование лучше производить в графическом режиме (мышкой), почему-то меньше ошибок. И новый sfs будет пытаться стартовать в консоли, при первом запуске точно потребуются команда xwin, но это решаемо.

Также при **FRUGAL**-установке можно сделать минимального размера save-файл с самыми необходимыми настройками. Его легко вернуть «на родину» после краха и увеличить размер при необходимости. А потом просто кликнуть на старом save-файле (с другим именем), примонтировав таким образом, и скопировать оттуда необходимые настройки, которые обычно находятся в /root/имя_программы, в рабочую /root. Настройки из других директорий так же легко копируются.

frugal (экономная или безопасная) установка — козырь pupru Linux. Такой простой установки нет ни в одном дистрибутиве. Тем более, что сейчас можно использовать до 25 sfs (в pupru 4.1) Как показывает опыт, такая установка очень устойчива к сбою электричества или случайному выключению.

В случае pupru 3 (и puprugus) надо периодически удалять **wh.файлы**, чтобы не было проблем со «слоями» (это немного отдельная тема)

O Puppy Linux

Введение. На данный момент дистрибутивы Linux выпускают с основательно проработанным интерфейсом и со всякими удобствами. В сравнении с другими дистрибутивами Puppy Linux выглядит устарелым и менее привлекательным. Puppy Linux может и не выиграет конкурс по красоте, однако, тут важно, что внутри, а не снаружи. Если вы взгляните на дистрибутив, не обращая внимание на внешний вид, то обнаружите жемчужину дистрибутива Linux.

Puppy Linux написан австралийским профессором Барри Каулером (Barry Kauler).

Дистрибутив создан, чтобы быть малым, эффективным и дружелюбным к пользователю. К этой категории относятся хорошо знакомые дистрибутивы, такие как Damn Small Linux, SLAX и SAM Linux, но у Puppy Linux есть серьёзные преимущества:

- Собран практически с нуля. Puppy очень мал и не требователен к железу.
- При загрузке с CD весь дистрибутив загружается в оперативную память и стартует без нужды доступа на CD, что делает Puppy очень быстрым.
- Puppy дает возможность сохранять данные сессии в отдельный файл, даже если вы запускаете дистрибутив с CD-RW.
- Puppy Linux устанавливается на любые носители включая USB флеш, жёсткий диск или на карту памяти.
- Системную конфигурацию можно легко изменить при помощи удобного инструмента настройки.
- Puppy Linux включает в себя быстрые и удобные приложения для интернета, офиса, графики, видео, звука и даже несколько игр для развлечения.
- Puppy включает в себя собственный файловый менеджер, делая установку дополнительных приложений простой.

В результате делает Puppy Linux идеальным дистрибутивом для использования на старых компьютерах.

Так же как и с любым другим Linux дистрибутивом для начала вам нужно скачать ISO образ Puppy последней версии и записать его на CD. Убедитесь, что в BIOSе первичное загрузочное устройство назначен CD привод.

Подобно другим Live CD дистрам, Puppy поддерживает загрузочные параметры. Например, puppy rfix = ram параметр заставляет Puppy Linux загружаться в RAM без загрузки сохраненной сессии, в тоже время puppy rfix = purge делает глобальную зачистку файлов, которая может быть очень полезна для восстановления системы. Полный лист загрузочных параметров и их дескрипторы можно посмотреть на странице WIKI Puppy Linux.

В процессе загрузки вы должны выбрать графический сервер X, состоящий из двух опций Xorg и Xvesa. Xorg поддерживает множество продвинутых настроек для современного железа, но может не запуститься на старых компьютерах. Xvesa имеет ограниченное количество настроек, но запускается практически на любой конфигурации компьютера. Обычно пользователи выбирают сначала Xorg, если экран после этого ничего не показывает, то можно выбрать Xvesa. Как только Puppy загрузился нужно выбрать оптимальное разрешение экрана.

У Puppy есть отличная система управлением разрешений. Все что вам нужно это выбрать нужное разрешение и нажать кнопку TEST. Если все отображается на экране корректно, но можно

продолжить работу, нажав **Ok**. Также можно определить разрешение вручную. Как только Puppy окончательно загрузился, взгляните на картинку на рабочем столе, которая содержит в себе несколько подсказок, включая информацию о доступной RAM памяти, состоянием подключения к интернету и сохранение ваших настроек и данных.

Установка Puppy Linux. Хотя Puppy Linux отлично запускается с CD, вы также можете установить его на любой носитель. Puppy включает в себя собственный установщик. Запустите его, выбрав в меню **Menu** → **Setup** → **Puppy universal installer**. Установщик включает в себя подробную информацию о процессе установки, и мы рекомендуем, чтобы вы прочли все внимательно, выбирая нужные опции.

Например, для загрузки puppy с USB флеш, которая использует файловую систему FAT32, вам нужно установить файлы в загрузочный сектор. Для установки файлов в загрузочный сектор выберете опцию **mbr.bin** когда появится диалог со списком доступных загрузчиков. Если вы устанавливаете Puppy в новую USB флешку, то скорее всего она не отформатирована как загрузочное устройство. В этом случае вы должны запустить **GParted**.

Запустите **GParted**, нажмите на разделе флешки правой кнопкой мыши и выберете **Manage Flags**. Далее выбираем **Boot** и жмем **OK** для закрытия окна и подтверждаем наши изменения, нажав кнопку **Apply**. Затем закрываем **GParted** и установщик доделает все остальное самостоятельно.

Установить Puppy на жёсткий диск также легко. Вам нужно выбрать между минимальной (**frugal**) и полной (**full**) установкой. При минимальной установке Puppy просто копирует несколько файлов (**vmlinuz**, **initrd.gz**, **pup_301.sfs** и **zdrv_301.sfs**) с CD на выбранный логический диск, что позволяет вам запускать Puppy Linux как Live CD дистрибутив, только с жёсткого диска и сохраняя сессию и данные на жёстком диске. Так же вам необходимо настроить загрузчик **GRUB** вручную. Полная установка позволяет установить весь дистрибутив на жёсткий диск в выбранный вами логический диск.

Запуск Puppy Linux с QEMU. Puppy, установленный на USB флешку, делает дистрибутив очень компактным. Вместо того чтобы таскать с собой ноутбук, вы можете с помощью флешки запустить Puppy на любом компьютере. Однако в некоторых случаях вам не разрешат перезагрузить Windows и зайти в Puppy Linux. **QEMU Manager** - это эмулятор, позволяющий запустить Puppy на платформе Windows. Также немаловажно, что **QEMU Manager** - это компактная программа и поэтому ее можете установить на USB флеш с Puppy Linux. Чтобы создать виртуальную машину, основанную на QEMU с Puppy Linux, нужно скачать программу **QEMU** и образ последней версии Puppy Linux. Распаковать **QEMU Manager** и скопировать папку на USB флеш. Копируем ISO образ в каталог с **QEMU Manager** и запускаем **QemuManager.exe**. Нажимаем **Create New Virtual Machine**, далее появится помощник, который поможет настроить

новую виртуальную машину. Все опции помощника вполне понятные и вы без проблем сможете установить новую виртуальную машину (VM).

Как только все шаги в создании VM пройдены, проверьте, что отмечен пункт View Advanced Configuration Options After Saving Box. Далее жмем кнопку Save Virtual Machine, которая сохраняет новую VM и открывает окно настроек. Далее переходим на вкладку Disk Configuration. В секции CD-ROM жмём кнопку Browse и выбираем ISO образ с Puppy Linux. Выбираем опцию Boot From CD-ROM. Сохраняем настройки, нажав на кнопку Save и теперь, вы можете закрыть окно. После этого можно запустить Puppy на VM нажав на кнопку Launch.

Настройка Puppy Linux. Puppy Linux имеет панель управления, которая позволит вам без проблем настроить ОС. Чтобы вызвать панель управления, выбираем Menu → Setup → Wizard Wizard. Эта панель управления поможет вам настроить любой аспект Puppy, включая локальные настройки, звук, X видео, соединение с интернетом и файрвол. Если Puppy не настроил, как следует драйвера на WIFI, вы можете установить их вручную. Для этого нажмите Load Module, выберете нужный модуль из списка драйверов, и жмём Load. Если драйвера для вашей WIFI карты нет в списке, у вас есть возможность установить драйвер для Windows при помощи NDISwrapper. Перейдите в секцию More, выбираем NDISwrapper, указываем нужный драйвер и жмём ОК.

Как только файл загружен, вам нужно создать новый профиль (New profile). Указываем нужное устройство, жмём кнопку appropriate, выбираем Wireless, Create new profile и заполняем требуемые поля. Помощник поддерживает мультипрофили. С помощью него можно переключаться между различными беспроводными сетями. Для возвращения настроек по умолчанию можно воспользоваться утилитой Menu → Desktop → Puppybackground image. Также можете удалить иконку с рабочего стола. Жмём правой кнопкой мыши на нужной иконке и выбираем Remove. Если вы выбрали минимальную установку или вы запускаете Puppy с USB флешки или другого съёмного устройства все ваши настройки и данные сохранятся в отдельном файле pup_save.2fs. При следующей загрузке Puppy автоматически загрузит созданный pup_save.2fs файл.

Установка приложения. Puppy Linux имеет свой собственный менеджер пакетов, который можно использовать для установки дополнительных пакетов с официального репозитория. Puppy использует свой собственный формат называемый PET, поэтому список приложений доступных в PET пакетах не большой, но он содержит основные приложения, такие как Mozilla FireFox, OpenOffice.org, GIMP и другие. Для установки приложения при помощи Puppy package manager просто, нужно всего лишь выделить нужное приложение и нажать Okay. Затем Manager скачивает выбранный пакет, проверяет его целостность и устанавливает. Кроме того можно управлять и .deb пакетами, которые позволяют вам пользоваться debain'ановскими пакетами. Для доступа к этой функции вам нужно установить 2 пакета при помощи Puppy Package

Manager: Веб браузер Dillo и pb_debianinstaller.

Далее можно скачивать .deb пакеты с debian'ановкого репозитария. Запускаем Терминал Menu → Utility → RXVT Terminal Emulator и вводим команду pb-debianinstaller. Эта команда запустит установщик и браузер Dillo. Наживаем кнопку Choose и выбираем скачанный .deb пакет, жмем Check dependencies и устанавливаем требуемые пакеты, если таковы требуются. После нажатия кнопки Install now и Finish, все готово.

После этого можете запустить установленную программу из терминала. Для удаления установленной программы вы можете использовать Puppy package manager. После установки пакетов Debian имейте ввиду что pb_debianinstaller все ещё экспериментальная версия, и может сделать вашу систему нестабильной. Используйте эту программу с осторожностью и не забывайте делать резервную копию системы.

Сборка дистрибутива Puppy Linux. После того, как вы настроили систему и установили нужные приложения, вы можете собрать свой собственный дистрибутив Puppy Linux. Нужная программа включена в Puppy (Menu → Setup → Remaster Puppy Live-CD) позволяет вам пересобрать его всего в с помощью нескольких кликов. Программа всего-навсего создает файл pup_301.sfs (Где 301 - номер версии Puppy), создаёт ISO образ и записывает его на CD-DVD. Все что вам нужно это выбрать логический или диск, из которого программа сделает ISO образ.

Заключение. Со всем вниманием и рекламой, которые окружают основные дистрибутивы, такие как Ubuntu, Mandriva и openSUSE, легко пропустить маленькое чудо как Puppy Linux. Этот дистрибутив быстро загружается, даже на старом железе, его можно установить на любой носитель. Вдобавок к этому, Puppy использует свой собственный файловый менеджер, имеет возможность устанавливать debian-приложения, а также возможность пересобрать дистрибутив. И у вас будет удивительный Linux дистрибутив.

Лабораторная работа № 9**Тема: BASH-ПРОГРАММИРОВАНИЕ**

Цель: Научиться создавать простые скрипты

Задание:

Разработать скрипт, выполняющий указанные действия согласно вариантам. Скрипт должен выдавать ФИО студента, комментарии по поводу предстоящих действий, и делать задержку после выдачи результатов на экран, иметь меню. Скрипт должен содержать функции, выполняющие указанные действия.

Вар	Действия	Вар	Действия	Вар	Действия	Вар	Действия
1	23,32,19.5,	8	7,19.3,30,	15	4.5,14.3,36	22	1,9,25.2,
2	24.1,30,19.6,	9	8.1,19.4,31,	16	4.6,15,25.1,	23	2,10,25.3,
3	24.2,29,26.1,	10	8.2,19.5,32,	17	4.7, 16,26.1,	24	3,11,26.3,
4	24.3,28,26.2,	11	9,19.6,19.1,	18	5,17,26.2,	25	4.1,12,34.1,
5	24.4,27,26.3,	12	19.7,20,19.2,	19	6.1,18,27,	26	4.2,13,34.2,
6	24.5,26.1,35.1	13	19.8,21,19.3,	20	6.2,19.1,28,	27	4.3,14.1,35.1,
7	25.1,26.2,35.2	14	22,33,19.4,	21	6.3,19.2,29,	28	4.4,14.2,35.2,

Список действий:

1. Выдать краткое описание указанной команды.
2. Отобразить тип аппаратной платформы.
3. Вычислить контрольную сумму указанного файла.
4. Вывести текущую дату в формате:
 - 1) дня недели,
 - 2) названия месяца,
 - 3) номера месяца,
 - 4) года,
 - 5) в национальном формате,
 - 6) номера дня недели,
 - 7) номера недели.
5. Вывести указанное сообщение.

6. Вывести информацию об указанном пользователе в формате:

- 1) полном,
- 2) только группы,
- 3) только номера.

7. Вывести информацию на заданную тему:

- 1) из указанного каталога,
- 2) краткой справки по указанной команде,
- 3) номера версии этой утилиты.

8. Вывести страницы руководства в формате:

- 1) все страницы,
- 2) краткого описания команды.

9. Отобразить список процессов.

10. Сохранять выводимые на экран символы в указанном файле путём дописывания в конец.

11. Приостановить выполнение команд на 10 сек.

12. Найти текстовую строку в заданном объектном файле.

13. Отобразить график загрузки системы.

14. Отобразить информацию:

- 1) о периоде времени после последней перезагрузки,
- 2) о числе пользователей, подключённых к системе,
- 3) о средней загрузке системы.

15. Выдать информацию о подключённых пользователях.

16. Вывести информацию о системе.

17. Сменить текущий каталог.

18. Отобразить режим доступа к указанному файлу.

19. Изменить режим доступа к личному файлу:

- 1) группе пользователей,
- 2) прочим пользователям,
- 3) установить разрешение на чтение,
- 4) установить разрешение на запись,
- 5) установить разрешение на исполнение,
- 6) установить запрет на чтение,
- 7) установить запрет на запись,
- 8) установить запрет на исполнение,

20. Отобразить владельца файла.

21. Сменить владельца файла.

22. Выполнить копирование указанного файла с сохранением атрибутов.

23. Выполнить копирование указанного каталога.
24. Вывести список файлов и подкаталогов указанного каталога путём:
 - 1) вывода списка по строкам,
 - 2) вывода без сортировки,
 - 3) вывода с сортировкой по расширениям,
 - 4) вывода с сортировкой по дате,
 - 5) вывода с пометками для файлов, каталогов и символических списков.
25. Найти во всей файловой системе:
 - 1) указанный файл,
 - 2) файлы по указанному пути,
 - 30 файл, изменяемый указанное количество дней назад,
26. Отобразить список файлов указанного каталога путём:
 - 1) вывода по столбцам,
 - 2) в полном формате,
 - 3) отсортированный по дате изменения.
27. Создать указанный каталог.
28. Переместить указанный файл.
29. Вывести имя текущего каталога.
30. Удалить указанный каталог.
31. Вывести размер указанной программы.
32. Упаковать указанный файл или каталог.
33. Распаковать указанный файл или каталог.
34. Вывести содержимое указанного файла:
 - 1) на экран с нумерацией строк,
 - 2) в конец другого файла.
35. Сравнить содержимое двух указанных файлов:
 - 1) с выводом позиций всех отличий,
 - 2) с выводом отличающихся символов.
36. Отформатировать текст в указанном файле, разбив его на несколько столбцов.

Порядок сдачи лабораторной работы:

1. Продемонстрировать выполнение скрипта.
2. Представить отчёт.

Общие требования к отчёту указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчёт должен содержать следующую информацию:

- а) задание на работу;
- б) распечатку созданного скрипта с именем <фамилия_латинскими_буквами.txt>;
- в) объяснение (комментарии) проделанной работы.

Дополнительная справочная информация

Краткая справка по языку Bash

Bourne-again shell (GNU Bash) — это реализация Unix shell, написанная на C в 1987 году Брайаном Фоксом (Brian Fox) для GNU Project.

Синтаксис языка Bash является надмножеством синтаксиса языка Bourne shell. Подавляющее большинство скриптов для Bourne shell могут быть исполнены интерпретатором Bash без изменений, за исключением скриптов, использующих специальные переменные или встроенные команды Bourne shell.

Также синтаксис языка Bash включает идеи, заимствованные из Korn shell (ksh) и C shell (csh): редактирование командной строки, история команд, стек директорий, переменные \$RANDOM и \$PPID, синтаксис POSIX для подстановки команд: \$(...).

Комментарии. Строки, начинающиеся с #, трактуются как комментарии.

Подстановка результатов выполнения команд. Выражения можно заключать в обратные кавычки (`). Такие выражения вычисляются в месте использования. Они могут быть, например, частью строк. Пример. Пусть параметром макрокоманды является имя файла с расширением .for. Требуется удалить одноименный файл с расширением .err.

```
name=`ena -n $1`
rm -f ${name}.err
```

Значение, полученное в результате выполнения команды

```
ena -n $1
```

присваивается переменной name. Фигурные скобки использованы для выделения аргумента операции перехода от имени к значению. Без них .err приклеилась бы к имени.

Переменные и подстановка их значений. Все переменные в языке - текстовые. Их имена должны начинаться с буквы и состоять из латинских букв, цифр и знака подчёркивания (_). Чтобы воспользоваться значением переменной, надо перед ней поставить символ \$. Использование

значения переменной называется подстановкой.

Различается два класса переменных: позиционные и с именем. Позиционные переменные - это аргументы командных файлов, их именами служат цифры: \$0 - имя команды, \$1 - первый аргумент и т.д. Значения позиционным переменным могут быть присвоены и командой set .

Пример. После вызова программы, хранящейся в файле ficofl:

```
ficofl -d / \*.for
```

значением \$0 будет ficofl, \$1 - -d, \$2 - /, \$3 - *.for, значения остальных позиционных переменных будут пустыми строками. Заметим, что если бы символ * при вызове ficofl не был экранирован, в качестве аргументов передались бы имена всех fortan'ных файлов текущей директории.

Ещё две переменные хранят командную строку за исключением имени команды: @\$ эквивалентно \$1 \$2 ..., а \$* - "\$1 \$2 ...".

Начальные значения переменным с именем могут быть установлены следующим образом:

```
<имя>=<значение> [ <имя>=<значение> ] ...
```

Не может быть одновременно функции и переменной с одинаковыми именами.

Следующие переменные автоматически устанавливаются:

#	количество позиционных параметров (десятичное)
-	флаги, указанные при запуске shella или командой set
?	десятичное значение, возвращённое предыдущей синхронно выполненной командой
\$	номер текущего процесса
!	номер последнего асинхронного процесса
@	эквивалентно \$1 \$2 \$3 ...
*	эквивалентно "\$1 \$2 \$3 ..."

Напомним: чтобы получить значения этих переменных, перед ними нужно поставить знак \$.

Пример: выдать номер текущего процесса:

```
echo $$
```

Управляющие конструкции

Простая команда - это последовательность слов, разделённая пробелами. Первое слово является именем команды, которая будет выполняться, а остальные будут переданы ей как аргументы. Имя команды передаётся ей как аргумент номер 0 (т.е. имя команды является значением \$0). Значение, возвращаемое простой командой - это ее статус завершения, если она завершилась нормально, или (восьмеричное) 200+статус, если она завершилась аварийно.

Список - это последовательность одного или нескольких конвейеров, разделённых символами

;, &, && или || и быть может заканчивающаяся символом ; или &. Из четырёх указанных операций ; и & имеют равные приоритеты, меньшие, чем у && и ||. Приоритеты последних также равны между собой. Символ ; означает, что конвейеры будут выполняться последовательно, а & - параллельно. Операция && (||) означает, что список, следующий за ней будет выполняться лишь в том случае, если код завершения предыдущего конвейера нулевой (ненулевой).

Команда - это либо простая команда, либо одна из управляющих конструкций. Кодом завершения команды является код завершения её последней простой команды.

Цикл ДЛЯ

```
for <переменная> [ in <набор> ]
do
<список>
done
```

Если часть in <набор> опущена, то это означает in "\$@" (то есть in \$1 \$2 ... \$n). Пример. Вывести на экран все сpp-файлы текущего проекта:

```
for f in *.cpp
do
  cat $f
done
```

Оператор выбора

```
case $<переменная> in
  <шаблон> | <шаблон>... ) <список> ;;
  ...
esac
```

Оператор выбора выполняет <список>, соответствующий первому <шаблону>, которому удовлетворяет <переменная>. Форма шаблона та же, что и используемая для генерации имен файлов. Часть | шаблон... может отсутствовать.

Пример. Определить флаги и откомпилировать все указанные файлы.

```
          # инициализировать флаг
flag= ;;
          # повторять для каждого аргумента
for a
do
  case $a in
    # объединить флаги, разделив их пробелами
    -[ocSO]) flag=$flag' '$a ;;
    -*) echo 'unknown flag $a' ;;
    # компилировать каждый исходный файл и сбросить флаги
    *.c) cc $flag $a; flag= ;;
```

```

*.s) as $flag $a; flag= ;;
*.f) f77 $flag $a; flag= ;;
    # неверный аргумент
*) echo 'unexpected argument $a' ;;
esac
done

```

Условный оператор

```

if <список1>
then
  <список2>
[ elif <список3>
then
  <список4> ]
...
[ else
  <список5> ]
fi

```

Выполняется <список1> и, если код его завершения 0, то выполняется <список2>, иначе - <список3> и, если и его код завершения 0, то выполняется <список4>. Если же это не так, то выполняется <список5>. Части elif и else могут отсутствовать.

Цикл ПОКА

```

while <список1>
do
  <список2>
done

```

До тех пор, пока код завершения последней команды <списка1> есть 0, выполняются команды <списка2>.

При замене служебного слова while на until условие выхода из цикла меняется на противоположное.

В качестве одной из команд <списка1> может быть команда true (false). По этой команде не выполняется никаких действий, а код завершения устанавливается 0 (-1). Эти команды применяются для организации бесконечных циклов. Выход из такого цикла можно осуществить лишь по команде break.

Функции

Синтаксис объявления функции:

```

function <имя> ()
{
  <список>;
}

```

Определяется функция с именем <имя>. Тело функции - <список>, заключённый между скобками { и }.

Примеры вычисления факториала.

Используется рекурсивное определение факториала.

```
function factorial
{
    typeset -i n=$1
    if [ $n = 0 ]; then
        echo 1
        return
    fi
    echo $(( n * $(factorial $(( n - 1 )) ) ))
}
for i in {0..16}
do
    echo "$i! = $(factorial $i)"
done
```

Используется итеративное определение факториала.

```
f=1
for (( n=1; $n<=17; $(n++) ));
do
    echo "$((n-1))! = $f"
    f=$((f*n))
done
```

Лабораторная работа № 10

Тема: ТЕХНОЛОГИЯ ВИРТУАЛИЗАЦИИ: Wine

Цель: Научиться использовать эмулятор Wine

Задание:

1. Изучить особенности функционирования эмулятора Wine.
2. Провести установку программного пакета, разработанного для ОС Windows, в эмуляторе Wine. Но! Устанавливаемые программы должны быть разные! То есть, если студент Сидоров описал установку в Wine «Словаря Даля», то студент Петров может описать установку «Словаря Мюллера», но никак не «Словаря Даля». Игры не устанавливать.
3. Составить руководство для системного администратора по установке данного пакета в Wine.

Порядок сдачи лабораторной работы:

1. В Wine установить некоторый программный пакет: ГИС «Ульяновск-регион», или «Словарь Даля», или что-либо иное по собственному выбору.
2. Продемонстрировать функционирующий установленный в Wine программный пакет.
3. Быть готовым ответить на дополнительные вопросы преподавателя

Общие требования к отчёту указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчёт должен содержать следующую информацию:

- задание на работу;
- копию экрана окна Wine с функционирующей программой;
- руководство для системного администратора по установке данного пакета в Wine.

Дополнительная справочная информация

Здесь приведена краткая информация о WINE, более полное описание смотрите в справочных руководствах на сайтах etersoft.ru/products/wine, altlinux.org, docs.altlinux.org.

WINE@Etersoft

Продукт WINE@Etersoft разработан специально для того, чтобы предоставить возможность организациям и отдельным пользователям полностью перейти на платформу Unix/Linux, сохранив унаследованные Win-приложения.

Особое внимание уделяется поддержке российских коммерческих приложений, которым затруднительно найти аналоги среди свободного ПО для Linux: в первую очередь это приложения, необходимые для ведения дел на предприятии (бухгалтерия, учёт, налоговая отчётность, правовые базы данных и др.).

Программное средство обеспечивает:

- поддержку наиболее популярных приложений;
- совместную работу с данными в среде WINE по сети и в терминальном режиме;
- поддержку ключей защиты (Guardant, HASP, Sentinel, Eutron);
- полную и корректную поддержку русского языка в работе приложений.

Списки поддерживаемых приложений и систем постоянно расширяются, самую актуальную информацию можно найти на сайте Etersoft www.etersoft.ru. Там же можно найти более подробную информацию о вариантах поставки WINE@Etersoft, новых версиях и обновлениях, а также о дополнительных услугах, предоставляемых компанией Etersoft.

Поддерживаемые приложения

Каждая версия продукта WINE@Etersoft сопровождается списком поддерживаемых приложений. Вошедшие в этот список Win-приложения тестируются на совместимость с WINE@Etersoft и гарантированно устанавливаются, запускаются и стабильно работают. Приложения, не вошедшие в этот список, также могут успешно работать в WINE@Etersoft, однако компания Этерсофт не может этого гарантировать и не обязуется отвечать на вопросы по их эксплуатации.

Подробная информация о совместимости WINE@Etersoft с программами доступна на сайте AppDB (<http://appdb.winehq.org.ru/>). С планами по поддерживаемым программам в новых версиях WINE@Etersoft можно ознакомиться (<http://etersoft.ru/wine/roadmap>).

Ниже приводятся списки программ полностью или частично поддерживаемых WINE@Etersoft. Актуализируемый список поддерживаемых приложений представлен на сайте www.etersoft.ru.

Программы, имеющие полную поддержку:

Бухгалтерские и торговые программы:

- 1С: Предприятие 7.7 (локальная, сетевая, SQL)
 - компоненты Бухгалтерский учёт, Оперативный учёт, Расчёт, УРБД

- 1С: Предприятие 8.0 (локальная, сетевая (файловая))
- 1С: Предприятие 8.1 (локальная, сетевая, SQL)
- 1С: Предприятие 8.2 (локальная, сетевая, SQL)
- Borland Database Engine (BDE)
- 1С: Предприятие - Работа с файлами
- Программы подготовки обязательной отчётности
 - Налогоплательщик ЮЛ 4.15
 - ПЕРС 2.4
- Банк-клиенты
 - Балтийского банка (версия 3.10.070)
 - Восточно-Европейской Финансовой Корпорации (ВЕФК), Петро-Аэро-Банка, ИнкасБанка, Таврического банка
 - Inter-Pro v 5

Правовые системы и базы данных:

- Гарант F1, Гарант F1 Турбо (локальная, файл-серверная, клиент-серверная версии)
- Консультант+ (локальная, сетевая версии)

Поддерживаемые ключи защиты:

Следующие ключи защиты поддерживаются для поддерживаемых программ:

- HASP 3/HASP 4/HASP HL от Aladdin (HASP SRM пока не поддерживается)
- Smartkey 3 от Eutron
- SafeNet и UltraPro/SuperPro от Sentinel
- SenseLock (USB)
- Катран (USB и LPT)
- Guardant Stealth/Net II USB, Stealth/Net III USB, Stealth III Sign/Time USB HID от компании Актив

Подробнее см. в разделе документации аппаратные ключи защиты, а также в полном списке поддерживаемых ключей.

Программы, имеющие поддержку производителя:

- Система управления предприятием AVARDA.ERP компании AnSoft
- Система лояльности “VT:Дисконт” компании “Версия-Т”
- Программа составления строительных смет «Гектор:Сметчик-строитель» компании «Гектор»

Программы, имеющие базовую поддержку:

Бухгалтерские и торговые программы:

- Конфигурации и расширения для 1С: Предприятия 7.7
 - конфигурации 1С: Парус (ключи Eutron)
 - расширение 1С++
 - расширение [openconf](#)
 - различные внешние компоненты (ВК)
- Конфигурации и расширения для 1С: Предприятие 8
 - конфигурации 1С: Парус АльфаАвто
 - конфигурация Континент-Страхование (ключи HASP)
- Программа “Офис-Склад-Магазин”
- ШТРИХ-М: Рабочее место кассира
- БЭСТ 4+
- ФолиоWinСклад версии 4.0
- Инфо-Бухгалтер 8.6 (локальный ключ Sentinel)
- 1С: Бухгалтерия 6.0 (сетевая) для Windows 95 **
- БизнесПак 6.35
- САМО-Тур
- Налогоплательщик 2009
- Налогоплательщик ЮЛ (4.17)
- CheckXML (4.03.2010)

Банк-клиенты:

- Межбанковский процессинговый центр Faktura.ru (через браузер IE *)
- ВТБ24 (шифрование через Inter-Pro v4), браузер konqueror или firefox****, браузер IE *
- ИБанк (Юниаструм Банк) (через браузер IE *)
- РосЕвроБанк (шифрование через Inter-Pro v4)

Программы для сдачи отчетности в электронном виде:

- СБиС++2.4

Программы для школьной администрации:

- 1С: Хронограф Школа 2.5

Правовые системы и базы данных:

- Кодекс 5.2 (локальная, сетевая версии, однако есть штатные версии для Linux)

- Референт Дельта
- Оболочка базы данных CronosPlus
- Triasoft Express- АРМ Нотариуса

Прочие программы:

- OpenOffice.org 2.4
- OpenOffice.org 3.x
- Microsoft Word 2000, Microsoft Excel 2000, Microsoft Access 2000
- Microsoft Word XP, Microsoft Excel XP, Microsoft Access XP
- Microsoft Word 2003, Microsoft Excel 2003
- Microsoft Word 2007, Microsoft Excel 2007
- Microsoft Word Viewer 97, Microsoft Word Viewer 2003
- Microsoft Excel Viewer 97, Microsoft Excel Viewer 2003
- TeamViewer
- RAdmin
- ABBYY Fine Reader 7.0
- ABBYY Fine Reader 8.0/9.0
- Lotus Notes 6.0/7.0
- Программы для интернет-трейдинга: QUIK, MetaTrader 4
- Православный ежедневник
- Total Commander 6.56
- Java Runtime Environment Version 5
- Mono
- .NET 1.1 *
- .NET 2.0 *
- Программное обеспечение “Анкета ПВДНП” (на загранпаспорт)

Графические программы:

- ДубльГИС (карты городов) 3.0.x
- OziExplorer

Программы, имеющие полную поддержку в WINE@Etersoft CAD 1.1:

- КОМПАС-3D 10

Программы, имеющие базовую поддержку в WINE@Etersoft CAD 1.1:

- КОМПАС-3D 10

- КОМПАС График 10

Программы, имеющие базовую поддержку в WINE@Etersoft Games 1.2:

- Старкрафт

Программы, которые готовятся:

- Гранд-смета
- БЭСТ 5.3х
- Инфо-Бухгалтер 10
- Парус 7
- Парус 8
- Галактика

Программы, которые могут работать, но не поддерживаются:

- ПРОМТ
- VentaFAX

Программы, поддерживаемые производителем в свободном Wine:

- Турбо 9 Бухгалтерия (ключи SenseLock и СУБД MySQL)
- Турбо 9 Торговля (ключи SenseLock и СУБД MySQL)

Программы, которые не работают:

- Microsoft Navision
- Microsoft Ахарта
- MS SQL Server
- Программы, защищённые StarForce (эта защита устанавливает свой драйвер cd-привода) (например, диски ИТС от 1С)
- Локальные ключи защиты, использующие VXD-драйвера
- HASP SRM от Aladdin (пока нет поддержки)
- КОМПАС-3D V11 (защищён HASP SRM, который пока не поддерживается)
- Клиент-банк “Уралсиб”, использующий USB-токен SafeNet iKey1000

Для использования программ, отмеченных символом *, потребуется лицензия на MS Windows.

Для программ, отмеченных символом **, в некоторых конфигурациях возможны проблемы с ключами защиты.

Для программ, отмеченных символом****- потребуется использование расширений Firefox FoxyProxy и UserAgent Switcher

WINE@Etersoft Local

Версия WINE@Etersoft Local ориентирована на индивидуальных пользователей, которым требуется работать с win-приложениями на платформе GNU/Linux или другой Unix-подобной системе. Поддерживается большое количество платформ: разнообразные дистрибутивы Linux, FreeBSD, OpenSolaris.

В версии WINE@Etersoft Local не поддерживается одновременная работа нескольких пользователей с одними и теми же файлами (например, общей базой данных). Соответственно, программы, которые предусматривают режим совместной работы (например, 1С: Предприятие), можно использовать только в монопольном режиме.

WINE@Etersoft Network

WINE@Etersoft Network ориентирована на корпоративных клиентов, которым требуется возможность совместной работы с win-приложениями на платформе GNU/Linux или другой Unix-подобной системе. В этом продукте реализована возможность одновременного доступа к файлам, позволяющая, например, использовать 1С: Предприятие 7.7 в многопользовательском режиме. С помощью WINE@Etersoft Network можно организовать работу win-приложений с клиентских станций как с разделяемым каталогом по протоколам CIFS и NFS, так и в режиме «тонких клиентов» с использованием терминального сервера. Совместная работа по сети windows- и linux-клиентов возможна только при использовании CIFS.

WINE@Etersoft SQL

WINE@Etersoft SQL ориентирована на организации, которым требуется возможность совместной работы с win-приложениями на платформах GNU/Linux или FreeBSD. В этом продукте наравне с возможностью одновременной работы с файлами реализована работа приложений с SQL-серверами (например, СУБД MS SQL Server 2000 или PostgreSQL). Рекомендуется организациям, работающих в системе «1С: Предприятие 7.7 SQL» или «1С: Предприятие 8.1 PostgreSQL».

Установка WINE

Пакеты, составляющие WINE@Etersoft, устанавливаются обычным образом, Вы можете воспользоваться привычной программой управления пакетами, например, для AltLinux такой привычной программой является synaptic.

Первый запуск WINE

При первом запуске WINE необходимо создать win-окружение для каждого пользователя.

Для этого нужно выполнить команду (в терминале):

```
$ wine
```

и дождаться её завершения. Процесс создания окружения отображается на графической заставке.

Вывод на экран сообщений может различаться при разных системах и конфигурациях.

Пример 1. Создание win-окружения

```
First running... Using WINEPREFIX=/net/wine/.wine
Creating default file tree...
Copying prepared tree from '/usr/share/wine/skel' ...
Initialize registry and environments...
Building local environment...
Flash Player 9 NPAPI installing...
Device 'e:' created as link for '/media' target.
Device 'e:.' created as link for '/dev/cdrom' target.
Device 'd:' created as link for '/net/wine/Documents' target.
Communication dlls installing...
MSI installing...
Windows Scripting installing...
Successfully registered DLL msxml3.dll
Successfully registered DLL msxml4.dll
Successfully registered DLL mfc40.dll
Successfully registered DLL mfc42.dll
Successfully registered DLL msscript.ocx
Successfully registered DLL mfc42u.dll
WINE@Etersoft 1.0 SQL 1.0.10-eter16/11

Licensed for OOO "Этерсофт" with registration number ****-****
Contact person: Системный администратор
Use /net/wine/wine_c as WINE C:\ disk.
Copy your program into and install it.
```

Пример 2. Проверка правильности установки (в терминале): Содержимое каталога wine.

```
$ ls -la ~/.wine
drwxrwxr-x  3 test test  4096 Окт 30 11:31 .
drwx----- 28 test test  4096 Окт 30 11:45 ..
drwxrwxr-x  4 test test  4096 Окт 30 11:31 dosdevices
-rw-rw-r--  1 test test   177 Окт 30 11:31 .etersoft-release
-rw-rw-r--  1 test test    72 Окт 30 11:31 install.log
-rw-rw-r--  1 test test 573550 Окт 30 11:31 system.reg
-rw-rw-r--  1 test test  3801 Окт 30 11:31 userdef.reg
```

```
-rw-rw-r-- 1 test test 26467 Окт 30 11:31 user.reg
```

Не забудьте скопировать файл лицензии WINE-ETERSOFT.LIC (ссылка на него присылается в письме при заказе продукта) в один из каталогов: ~/.wine, C:\WINDOWS\INF или в /etc/wine. Если файл лицензии не будет найден, это будет сообщено при выводе сведений и версии wine.

Пример 3. Вывод сведений о версии wine

```
$ wine --version
```

```
WINE@Etersoft 1.0 SQL 1.0.12-eter2/3
```

```
Licensed for OOO "Этерсофт" with registration number ****-****
```

```
Contact person: Системный администратор
```

```
License expired at 28/04/2010
```

Установка приложений в WINE

Можете приступить к установке win-приложений. В роли диска C: выступает каталог ~/.wine_c (находится в домашнем каталоге).

Скопируйте туда дистрибутив программы и выполните команду

```
$ wine имя_программы.exe
```

Использование WINE@Etersoft

Запуск win-приложений. Общее правило для запуска всех win-приложений в WINE — запускаемые файлы должны находиться в области видимости WINE, то есть на одном из логических дисков WINE или в его подкаталогах. Если программа поставляется на компакт-диске, то не забудьте должным образом смонтировать диск **1**, прежде чем обращаться к нему из WINE. Обратите внимание, что в этом случае у вас должен быть разрешён запуск приложений с компакт-диска. Если приложение распространяется не на диске — не забудьте сначала скопировать его в область видимости WINE.

Запуск win-приложений производится двойным щелчком мыши на значке в любом файловом менеджере.

Также приложение может быть запущено с помощью команды в командной строке. Не забудьте сначала перейти в каталог с программой.

Для запуска exe-файлов в терминале нужно выполнить команду:

```
$ wine программа.exe
```

Для запуска консольных приложений, например файлового менеджера Far, используется

команда:

```
$ wineconsole Far.exe
```

Для получения командной строки запустите

```
$ wineconsole cmd
```

или выберите в меню программ пункт “Командная строка WINE”.

При запуске программы в WINE на самом деле запускается не только сама программа, но и несколько вспомогательных, в частности, программа `wineserver`, реализующая функции ядра Windows, и предназначенная для синхронизации различных win-программ, запущенных пользователем.

Запуск сервисов

Некоторые win-приложения должны быть запущены как сервисы. От обычных программ сервисы отличаются тем, что не ведут диалога с пользователем, и могут выполняться незаметно.

При запуске сервисов следует иметь в виду, что они завершаются вместе с завершением `wineserver`, поэтому следует предварительно запустить `wineserver` с ключом `-p`, отменяющим автоматическое завершение.

Пример 4. Пример запуска сервиса

```
$ wineserver -p
```

```
$ wine pssvc.exe &
```

В указанном примере программа `pssvc` будет запущена как сервис, причём в фоновом режиме.

Установка и удаление win-приложений

Как и в Windows, перед использованием большую часть win-приложений сначала потребуется установить. Установка производится обычным для Windows способом — с помощью поставляемой вместе с win-приложением программы установки. Разница в том, что в случае WINE, программа будет установлена в локальном win-окружении пользователя.

Для установки win-приложения следует любым удобным способом запустить программу установки (чаще всего `setup.exe`). Далее можно действовать по инструкции, предлагаемой поставщиком win-приложения.

Многие win-приложения запрашивают перезагрузку для завершения установки. Перезагружать host-систему при этом не следует. В локальном win-окружении процедуре загрузки Windows соответствует команда `wineboot` — её можно вызвать из любой командной строки. Если в этот момент в WINE выполняются другие приложения, то рекомендуется их завершать до перезагрузки.

Для удаления win-приложения, установленного в win-окружении, следует воспользоваться

программой `uninstaller`. Запустить её можно через меню или командой `wine uninstaller`. Эта утилита выводит список установленных в win-окружении приложений (если они зарегистрированы в реестре). Чтобы удалить приложение, выберите его из списка и нажмите кнопку «Uninstall». Если в списке нет приложения, которое вы хотите удалить, то достаточно просто удалить каталог с приложением (можно воспользоваться для этого программой `winefile`, а можно — стандартными средствами host-системы).

Иногда приложение требует дополнительные компоненты, отсутствующие в стандартной поставке WINE. В этом случае можно обратиться к программе `winetricks`, запустив её из командной строки. Она позволяет установить различные компоненты, при этом все необходимые вспомогательные действия берёт на себя. Используйте с осторожностью, установка некоторых компонент или их сочетание может сломать работающее win-окружение.

Лабораторная работа № 11

Тема: ТЕХНОЛОГИЯ ВИРТУАЛИЗАЦИИ: Virtualbox

Цель: Научиться использовать систему виртуализации Virtualbox

Задание: Изучить особенности функционирования систему виртуализации Virtualbox. Установить и настроить систему виртуализации. В VirtualBox установить другую операционную систему. Подготовить отчет.

Порядок сдачи лабораторной работы:

1. Проверить, что VirtualBox установлен в системе. Если нет – установить.
2. Настроить VirtualBox.
3. В VirtualBox установить другую операционную систему. Рекомендуется puppet.
4. Составить руководство для системного администратора по установке ОС в VirtualBox.

Общие требования к отчету указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчет должен содержать следующую информацию:

- задание на работу;
- копию экрана окна VirtualBox с запущенной операционной системой;
- руководство для системного администратора по установке операционных систем в VirtualBox.

Дополнительная справочная информация

*Прежде всего, надо иметь в виду, что при установке ОС **ALTLinux Kdesktop** автоматически ставится пакет *Virtualbox*. А теперь читайте весь процесс его установки и настройки!*

VirtualBox

Virtualbox — программный продукт виртуализации для операционных систем DOS, GNU/Linux, Mac OS X и SUN Solaris/OpenSolaris, и даже Microsoft Windows. Программа была создана компанией Innotek с использованием исходного кода Qemu. Первая публично доступная версия VirtualBox появилась 15 января 2007 года. Существует две версии — свободная (OSE, англ.

Open Source Edition), выпущенная под GNU GPL, и проприетарная (PUEL), различающиеся по функциональности. Полнофункциональная проприетарная версия для личного использования распространяется бесплатно. В феврале 2008 Innotek был приобретён компанией Sun Microsystems, модель распространения VirtualBox при этом не изменилась.

Пакеты свободной версии VirtualBox есть в составе дистрибутивов ALT Linux, начиная с Desktop 4.1. При установке системы можно выбрать группу "Виртуализация" для установки Virtualbox по умолчанию.

1. Установка свободной версии VirtualBox из пакетов. Чтобы воспользоваться Virtualbox, необходимо установить следующий набор пакетов (рекомендуется пользоваться для этого synaptic).

- virtualbox-версия программы
- virtualbox-common-версия программы
- virtualbox-doc-версия программы
- kernel-modules-virtualbox-std-def-версия-программы-"номер_сборки_ядра" (нужен для хостовой ОС)
- kernel-modules-virtualbox-addition-std-def-версия-программы-"номер_сборки_ядра" (не нужен для хостовой ОС)
- virtualbox-guest-additions-версия программы (не нужен для хостовой ОС)

Пример.

Пусть установлен VirtualBox версии:

- virtualbox-4.3.22-alt0.M70P.1

и ядро

- kernel-image-std-def-3.14.35-alt1

Анализируем модули virtualbox:

- kernel-modules-virtualbox-addition-std-def-4.3.22-alt1.200227.1 (можно удалить - вы не внутри Virtualbox)
- kernel-modules-virtualbox-std-def-4.3.22-alt1.200227.1 (актуальное ядро, от актуальной версии VirtualBox)
- kernel-modules-virtualbox-std-def-4.3.22-alt1.200225.1 (для старой версии ядра)
- kernel-modules-virtualbox-std-def-4.3.14-alt1.200225.1 (от старого Virtualbox и старого ядра)
- virtualbox-common-4.3.22-alt0.M70P.1 - Ок
- virtualbox-doc-4.3.22-alt0.M70P.1 - Ок
- virtualbox-guest-additions-4.3.22-alt0.M70P.1 - (не нужный пакет, если вы не внутри VirtualBox)/

При этом они должны быть одной версии (4.3.22-alt0.M70P.1 в данном случае) а модули ядра - от текущего ядра. Кроме того, модуль ядра должен соответствовать сборочной ветки ядра (std-def в данном случае). Чтобы это обеспечить, ядро и пакеты необходимо устанавливать из одного и того же репозитория.

Если VirtualBox был недавно обновлён в репозитории, а ядро давно не обновляли, вероятно, что VirtualBox не будет работать со старым ядром. Придётся обновить ядро и/или модули. Это всё можно сделать утилитой update-kernel.

Узнать версию загруженного ядра можно командой

```
$ uname -r ,
```

которая выдаст, например,

```
3.14.35-std-def-alt1.
```

Если у вас в виртуальной машине гостевая система тоже ALTLinux, то можно установить в ней "дополнения гостевой системы" командой:

```
kernel-modules-virtualbox-addition-std-def-версия
```

и программы-"номер_сборки_ядра"

```
virtualbox-guest-additions-версия программы-"номер_сборки_ядра" .
```

Используемое устройство. Узнать, какое устройство Virtualbox использует для работы можно командой:

```
$ ls -l /dev/vboxdrv,
```

которая, например, выдаст:

```
crw-rw---- 1 root vboxusers 10, 58 Май 5 08:46 /dev/vboxdrv
```

Если такого файла нет, то следует проверить наличие соответствующего загруженному ядру (по типу и версии) пакета

```
kernel-modules-virtualbox-*
```

и дать команды (от root'a, конечно):

```
chkconfig virtualbox on
```

```
service virtualbox start
```

Проверяем список установленных пакетов командой:

```
$ rpm -qa | grep virtual-*
```

Анализируем ответ:

```
kernel-modules-virtualbox-addition-std-def-4.3.22-alt1.200227.1
```

```
virtualbox-doc-4.3.22-alt0.M70P.1
```

kernel-modules-virtualbox-4.3.22-alt1.200227.1 - не нужен в гостевой машине, можно

удалить.

virtualbox-4.3.22-alt0.M70P.1 - этот пакет можно смело удалять, если вы не хотите устраивать "матрёшки" — вложенные друг в друга одинаковые системы.

```
virtualbox-guest-additions-4.3.22-alt0.M70P.1
```

```
virtualbox-common-4.3.22-alt0.M70P.1
```

Далее включаем своего пользователя (или себя — свой логин) в группу vboxusers командой:

```
gpasswd -a user vboxusers
```

а если унас коммерческая сборка, то и в группу:

```
gpasswd -a user vboxadd
```

Usb-устройства. Для работы с usb устройствами необходимо настроить fstab (однако, в версии p7 и старше не надо настраивать — это уже сделано!).

В конец /etc/fstab добавляем

```
none /proc/bus/usb/ usbfs devgid=500,devmode=666 0 0
```

где

devgid=500 - идентификатор пользователя; посмотрите, какой у вас gid командой `cat /etc/passwd`

Для работы usb в режиме usb2, надо скачать с сайта загрузки VirtualBox (<https://www.virtualbox.org/wiki/Downloads>) и установить "Oracle VM VirtualBox Extension Pack". См. ниже пункт 4.

Перезагружаем систему.

После перезагрузки, можно запускать Virtualbox, например, в KDE из меню:

К — Система — Виртуализатор Virtualbox.

2. Настройка сети в гостевой системе Virtualbox. Чтобы гостевая система получила доступ к подсети, в которой находится хост-система, можно поступить следующим образом.

Настроить в хост-системе туннель и мост:

- сначала необходимо запустить интерфейс туннеля tap0,
- затем объединить его в мост с интерфейсом вашей физической сетевой карты (например, eth0; на компах лаборатории этот интерфейс называется enp3s0:). Подробности смотрите на форуме altlinux.org.

После того, как вы убедитесь, что сеть на хост-системе после этой настройки работает как и раньше, вы можете использовать туннель в Virtualbox: зайдите в свойства виртуальной машины,

раздел "сеть" и выберите "Подсоединён к" — "Хост-интерфейс", а ниже, в поле "Хост-интерфейсы" выберите tap0. После этого, при загрузке, гостевая система получит доступ к той же подсети, что и хост-система, и будет выступать равноправным компьютером в сети со своим IP адресом. IP адрес и другие настройки интерфейсу следует назначать средствами гостевой системы.

3. Решение проблем.

1) При запуске дистрибутивов на Седьмой платформе, как гостевых ОС в VirtualBox, после загрузки ядра показывается чёрный экран и в журнале пишется

```
Guest Log: BIOS: KBD: unsupported int 16h function 03
```

Причина и решение: для современных ядер, на которых основаны дистрибутивы Седьмой платформы, нужно указывать в настройках виртуальной машины Система ⇒ Материнская плата ⇒ Чипсет: ICH9 (заодно там же включите Ю APIC). По умолчанию в альтовом VirtualBox стоит PIIX3, что и является причиной указанного поведения.

2) При запуске гостевой ОС на Седьмой платформе VirtualBox показывает сообщение о не достаточных правах доступа для /dev/vboxdrv и просит запустить "/etc/init.d/vboxdrv setup", которого нет.

Решение: необходимо выполнить:

```
chkconfig virtualbox on  
service virtualbox start
```

3) В гостевой ОС после выбора подсказки Установка или LiveCD отображается чёрный экран и ничего не происходит.

Решение: выберите PIIX3, и при запуске установки (или загрузки с LiveCD) выберите клавишей F5 загрузку "Без локального APIC" (см. рис. 6).

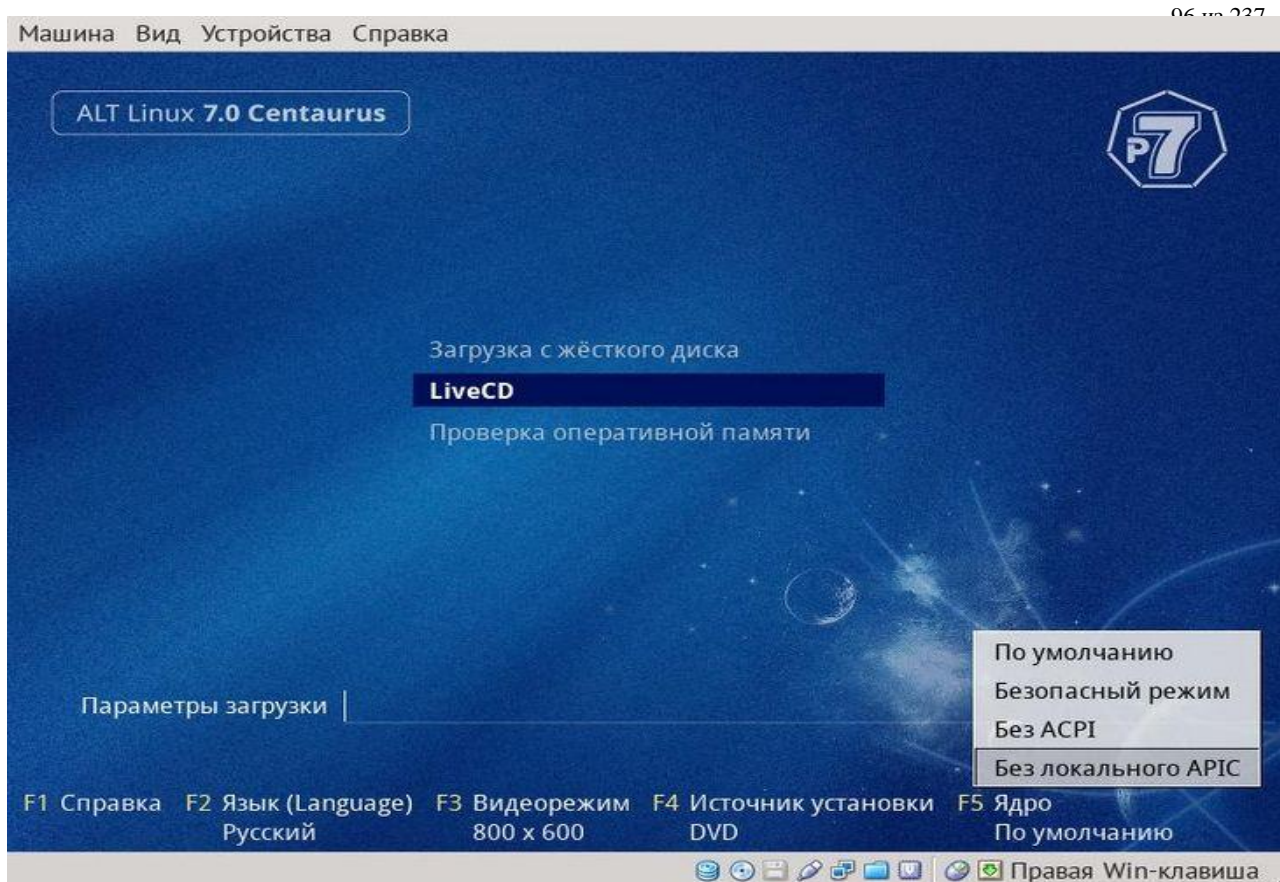


Рис. 6. Вид загрузки системы

4) В гостевой системе не работает звук.

Решение: откройте регулятор громкости PulseAudio (pavucontrol) и на вкладке «Проигрывание» включите звук для виртуальной машины.

Примечание: VirtualBox через PulseAudio позволяет настраивать вывод звука для каждой виртуальной машины. При этом включение для первой машины включает звук и для других запускаемых виртуальных машин

5) Плохо работает микрофон в гостевой системе (заикается).

Решение: всё дело в PulseAudio. Для того, чтобы микрофон работал без заиканий, для виртуальной машины выставьте ALSA: Аудио ⇒ Аудиодрайвер: ALSA и выключите на время работы PulseAudio: в терминале введите команду:

```
pulseaudio -k
```

6) Если virtualbox запускается, но модули не загружаются, отключите автозапуск виртуализатора virtualbox и создайте файл /etc/modules-load.d/virtualbox.conf такого содержания:

```
vboxdrv  
vboxpci
```


vboxnetflt

vboxnetadp

7) Происходит зависание гостевой ОС на словах Local IPI :

Причина: Это [altbug #29344](#), исправлено в virtualbox-4.2.16-alt2 (но вовсе не факт, что в других дистрибутивах порядок).

4. Установка Oracle Extension Pack. Установленная версия VirtualBox OSE позволяет использовать только USB 1.0, однако иногда необходимо использовать USB 2.0. Для корректной работы USB 2.0 в VirtualBox (на момент написания данного материала - актуальной версией в репозитории Sisyphus является VirtualBox 4.0.12) требуется Oracle VM VirtualBox Extension Pack. Его можно скачать с сайта Oracle (<http://www.oracle.com/technetwork/server-storage/virtualbox/downloads/index.html#extpack>). Установку дополнения можно запустить из графического интерфейса или из командной строки:

```
# VirtualBox extpack install /path/to/Oracle_VM_VirtualBox_Extension_Pack-...vbox-extpack
```

Для установки дополнения необходимо запустить VirtualBox с правами администратора, иначе при установке будет выдана ошибка:

The installer failed with exit code 127: Error creating textual authentication agent:

Error opening current controlling terminal for the process (^/dev/tty'): No such device or address.

Примечание. Однако новые версии Virtualbox умеют устанавливать это расширение от обычного пользователя

Если в системе установлена программа gksu или kdesudo, то VirtualBox попытается с их помощью поднять уровень привилегий для выполнения установки.

На данный момент Oracle VM VirtualBox Extension Pack является единственным дополнением. Оно обеспечивает следующую функциональность:

- USB 2.0 EHCI;
- VirtualBox Remote Desktop Protocol, VRDP;
- Intel PXE boot ROM с поддержкой контроллера E1000;
- экспериментальная поддержка «проброса» PCI с хост-системы.

Если установлен и используется плагин Oracle Extension Pack , то обязательно обновить его до последней версии (скачать с virtualbox.org), иначе виртуальная машина начинает грузиться и «схлопывается» (подробности см. <http://lists.altlinux.org/pipermail/community/2015-February/683600.html>)

5. Установка проприетарной сборки VirtualBox

Также можно использовать закрытую бинарную сборку с сайта разработчиков программы.

Настоятельно не рекомендуется: не нужно пользоваться тем, на чём можно попасть на деньги.

Для очень любопытных — см. как это сделать на сайте altlinux.org — есть подробное описание, но процесс далеко не простой.

6. Установка «Дополнений гостевой ОС» для гостевого ALT Linux

1. Установите модуль ядра `kernel-modules-virtualbox-addition-<тип ядра>` и вспомогательные пакеты для интеграции

```
apt-get install kernel-modules-virtualbox-addition-$(uname -r|cut -f2,3 -d -) virtualbox-guest-additions xorg-drv-virtualbox
```

Перед любыми операциями с установкой модуля ядра, надо обновить ядро с помощью команд:

```
apt-get update
```

```
update-kernel
```

И ни в коем случае не устанавливайте модуль от другой ядерной сборки.

2. Выберите в Центре управления системой ⇒ Дисплей драйвер `vboxvideo`.

Примечание: Драйвер можно выбрать и в командной строке (под правами `root`):

```
alterator-cmdline /x11 action write _objects driver driver vboxvideo
```

Примечание: Если пункта «Дисплей» нет в Центре управления системой, установите пакет `alterator-x11`

3. Перезагрузите гостевую систему или только сеанс

4. Включите в меню виртуальной машины

- Устройства ⇒ Общий буфер обмена ⇒ Двухнаправленный
- Устройства ⇒ Drag'n'Drop ⇒ Двухнаправленный.

6.1. Общие папки для взаимодействия хостовой и виртуальной систем.

1. Установите в гостевой операционной системе дополнения и утилиты:

```
apt-get install kernel-modules-virtualbox-addition-$(uname -r | cut -d "-" -f2,3) virtualbox-guest-utils
```

Этот пункт не нужен, если уже установлен пакет `virtualbox-guest-additions`, как описано выше.

2. Добавьте в свойствах виртуальной машины (раздел «Общие папки») папку. Для этого выберите путь и укажите имя папки (по умолчанию используется имя последней папки в указанном пути). Если хотите чтобы папка осталась настроенной и после перезагрузки гостевой операционной системы, установите флажок «Создать постоянную папку».

3. Для монтирования общих папок VirtualBox должен быть загружен модуль vboxsf:

```
modprobe vboxsf
```

4. В гостевой операционной системе выполните

```
mount -t vboxsf <имя папки> /mnt
```

Внимание! Использование параметра `-t` в команде `mount` разрешено только пользователю `root`.

5. Общая папка становится доступна в `/mnt`.

6.2. Краткая инструкция для установки дополнений вручную

Внимание! Здесь предлагается идеологически неправильное решение!!!

Идеологически правильно дождаться, пока мейнтейнер соберёт новую версию драйверов в дистрибутив.

- Запускаем гостевой Линукс, и удаляем пакеты:

```
xorg-drv-vboxvideo
```

- Ставим пакет `kernel-headers-modules` для *своего* ядра.
- В окне VirtualBox, в меню "Устройства", выбираем "Установить дополнения гостевой ОС".
- В гостевом Линуксе заходим на CD-ROM, находим там файл `VBoxLinuxAdditions-x86.run` (или `-amd64.run`, если у вас 64-разрядная ОС), копируем его в `/tmp`.
- В гостевом Линуксе открываем терминал, переходим в нём в рута (командой `su -`) и выполняем команду:


```
chmod a+x /tmp/VboxLinuxAdditions-x86.run /tmp/VBoxLinuxAdditions-x86.run
```
- Далее исполняется скрипт, который сам все распакует, скомпилирует и установит.
- Убеждаемся, что в гостевом Линуксе в `xorg.conf` прописан драйвер `vboxvideo` для видео.
- Перезагружаем виртуальную машину.
- Наслаждаемся автоизменением размеров окна, и "незалипающей" мышкой.

Лабораторная работа № 12

Тема: УСТАНОВКА 4-х ОС НА ПЭВМ

Цель: Научиться устанавливать различные операционные системы: Win-XP + 3 Linux: Alt, Mops, Puppy на ПЭВМ.

Задание:

1. Произвести разбиение винчестера ПЭВМ на разделы с помощью программы fdisk следующим образом:

sda1 — 30 Gb — ntfs для Windows

sda2 — 40Gb - a5

sda3 — 2Gb - swap

sda4 — extended

sda5 — 30 Gb - ALTLinux

sda6 — 30 Gb - MOPS

sda7 — всё, что остаётся, для Puppy.

2. Установить четыре операционные системы: Windows + 3 Linux: Alt, Mops, Puppy – на ПЭВМ лаборатории.

Указания к выполнению работы

Свободно распространяемые версии дистрибутивов скопировать с сайтов разработчиков.

Учебники и пособия по Linux рекомендуется смотреть на сайтах:

<http://www.altlinux.org> , <http://docs.puppyrus.org>, <http://uco.puppyrus.org/stati>

Наиболее сложная часть работы — это настройка загрузчика grub с обеспечением загрузки всех 4-ёх систем.

Порядок сдачи лабораторной работы

1. Продемонстрировать работу с установленными системами.
2. Представить отчет.

Общие требования к отчету указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчёт должен содержать следующую информацию:

- а) задание на работу;
- б) описание подготовки винчестера (разделов),
- в) описание установки дистрибутивов,
- г) описание установки и настройки загрузчика.

Дополнительная справочная информация

Основные шаги по подготовке HDD к использованию

Подключение. HDD, CD-Rom'ы, DVD-Rom'ы и другие аналогичные устройства хранения информации подключаются к ПЭВМ (не сильно новых) с помощью интерфейса IDE (параллельный ATA - PATA). Интерфейс IDE реализуется на системных платах в виде спаренного (двухканального) контроллера IDE — разъёмы ide0 и ide1 (иногда на системных платах они помечены как ide1 и ide2). К каждому каналу (разъёму) с помощью специального интерфейсного кабеля может быть подключено два устройства, из которых один должен быть (обязательно) master'ом, а другой (обязательно) slave. Более точно используются следующие интерфейсы:

- либо интерфейс UDMA-33 (40-pin'овый кабель) — в этом случае необходимо вручную с помощью перемычек на HDD установить статус (режим работы HDD): master или slave;

- либо интерфейс UDMA-66/100/133 (80-pin'овый кабель) — в этом случае кто есть кто определяется кабелем (цветной разъём — на разъём системной платы, серый — slave, чёрный — мастер), а перемычки на HDD устанавливаются в положение cs — cabel select.

На более новых ПЭВМ вместе с параллельным ATA используется также последовательный ATA (serial ATA — SATA). В этой разновидности интерфейса на один разъём на системной плате предусмотрено подключение только одного устройства и потому перемычки на HDD не предусмотрены. То есть, с помощью каждого кабеля SATA подключается только один HDD и он всегда master.

Разбиение HDD на разделы. Разбиение HDD на разделы осуществляется с помощью программы **fdisk**. Назначение программы **fdisk** — создание и/или редактирование таблиц разделов (Partition Table — PT), основной и дополнительных.

Разделы являются контейнерами всего своего содержимого. Этим содержимым является, как правило, файловая система. Под файловой системой с точки зрения диска понимается «система разметки секторов и блоков диска для хранения файлов». После того, как на разделе создана файловая система и в ней размещены файлы операционной системы, раздел может стать загружаемым. Загружаемый раздел имеет в своих первых секторах небольшую программу, которая производит загрузку операционной системы — вторичный загрузчик. То есть, для загрузки

операционной системы нужно явно запустить ее загрузчик из первых секторов раздела.

!!!Разметка диска на разделы еще не означает создания файловых систем. То есть, разбитый на разделы диск ещё не готов для использования.

В самом начале HDD находится «системная область» размером 63 сектора. Почему 63? Это «тяжёлая наследственность», оставшаяся от тех стародавних времён, когда дорожки HDD разбивались на 63 сектора и вся крайняя (наружная, самая первая) дорожка была системной.

Самый первый сектор системной области (именно первый, с порядковым номером один; вообще-то, счёт, как обычно, идёт с нуля, но нулевой сектор на дорожке служит меткой начала дорожки) содержит Главную Загрузочную Запись (Master Boot Record — MBR) размером 446 байт — первичный загрузчик. Но! Первичный загрузчик находится в самом первом секторе системной области только в том случае, если на этом HDD установлена операционная система и при её установке было предписано установить загрузчик в MBR (то есть, в системную область диска). Windows никогда об этом не спрашивает и всегда MBR перезаписывает при установке. Unix'овые системы практически всегда спрашивают, куда ставить загрузчик. Это значит, что даже если на HDD есть ОС, то это ещё не означает, что первый сектор HDD содержит MBR, поскольку пользователь/админ может «захотеть» и, не понимая что делает, поставить первичный загрузчик в другое место, например, в какой-нибудь раздел, или даже на флешку. А, возможно, именно так и надо.

Внимание! Не путайте MBR (и первичный загрузчик) с вторичным загрузчиком в начале какого-либо раздела!

В следующих 64 байтах этого сектора содержится Главная Таблица Разделов (Partition Table — PT). Это таблица из 4-х строк, каждая строка которой содержит следующую структуру:

```
struct pt_struct
{
    u8 bootable;      // флаг активности раздела      1 байт
    u8 start_part[3]; // координаты начала раздела 3 байта
    u8 type_part;     // системный идентификатор 1 байт
    u8 end_part[3];  // координаты конца раздела  3 байта
    u32 sect_before; // число секторов перед разделом 4 байта
    u32 sect_total;  // число секторов в разделе   4 байта
};
```

А последние два байта сектора MBR должны содержать число 0xAA55. По наличию этой сигнатуры BIOS проверяет, что первый блок был загружен успешно и что он действительно содержит MBR + PT.

Каждая строка таблицы PT описывает один раздел диска. Эти разделы (описанные в Главной PT) называются первичными или основными — primary. И поскольку строк четыре, то, следовательно, на HDD могут быть только четыре первичных раздела: в Linux'овом именовании это, например, hda1, hda2, hda3 и hda4.

Один из этих первичных разделов (любой, кроме первого) может быть расширенным разделом — extended. С номера 5 начинают считаться логические разделы, которые создаются в **расширенном разделе, как в контейнере**. В начале каждого логического раздела - в первый сектор раздела, пишется структура, похожая на сектор MBR с одним отличием: самого MBR в нём нет, то есть первые 446 байт заполнены нулями. Но PT и сигнатура 0xAA55 в нём присутствуют.

Важно! Не путайте диск (винчестер, всё устройство в целом) с разделом на диске. Раздел — это часть диска. Но Windows этого не знает.

Программа **fdisk** умеет работать с Главной PT - создавать первичные разделы, определять один из первичных разделов расширенным, создавать в расширенном разделе логические разделы. При создании логического раздела fdisk автоматически создаёт PT (вторичную) в первом секторе логического раздела.

При создании раздела пользователь должен ввести информацию о расположении и размере раздела. Эта информация используется для заполнения строки PT, описывающей этот раздел.

Таким образом, в результате работы программы **fdisk** на HDD пишутся:

- первый сектор диска — сектор MBR с заполненной главной PT;
- и, если на HDD создаются логические разделы, то в начале каждого раздела по адресу сектора, с которого начинается раздел (самый первый сектор раздела) пишется сектор с вторичной PT, в первой строке которой описан этот логический раздел.

Внимание! На самом деле, реальность несколько по-другому выглядит, но детали опущены. Если хотите точно знать, как всё происходит — читайте документацию и исходники ОС linux.

Создание файловой системы на разделе. Создание файловой системы на разделе— это операция форматирования раздела с помощью команды mkfs (см. лекции или man mkfs).

Монтирование раздела. Монтирование раздела для использования осуществляется командой mount (см. лекции или man mount).

Установка 4-х ОС на hdd

Варианты установки. Возможны два варианта установки:

- а) у вас новый диск или «старый», но информация на нём вам не нужна;
- б) у вас уже стоит на hdd некоторая система (например, Windows), тогда вам нужно освободить часть диска под новые ОС; освободить место всегда нужно с конца hdd, иначе нумерация разделов на hdd может оказаться не соответствующей реальному порядку

расположения разделов на hdd.

Разбиение диска (формат PC BIOS). Далее загружаетесь с какого-нибудь live-CD, например, puppy. Здесь важно то, что разбиение диска настоятельно рекомендуется делать в Linux, ибо в Windows вы далеко не всегда сможете переразбить диск так как надо.

В варианте а) вы весь диск программой fdisk разбиваете (переразбиваете) на разделы.

В варианте б) вы тоже самое проделываете с освобождённым на диске местом.

Этот пункт требует умения работать с программой fdisk и вообще понимать, что такое винчестер, раздел, таблицы разделов, файловые системы и как со всем этим добром работать.

Умение работать с fdisk — обязательно!

Разбиение диска для 40-гигабайтных hdd (в лаборатории, см. рис. 7):

sda1—win—7Gb,
sda2—swap—1Gb,
sda3—alt—12Gb,
sda4—extended,
sda5—MOPS—12Gb,
sda6—puppy—что_останется.

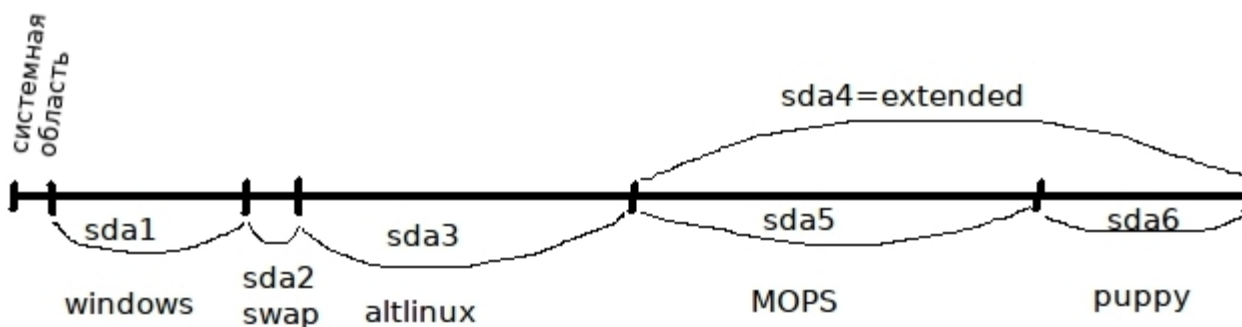


Рис. 7. Разбиение диска для 40-гигабайтных hdd

Разбиение диска для 160- или 500-гигабайтных hdd (в лаборатории, см. рис. 8) должно быть следующим:

sda1-win-30Gb,
sda2-bsd-40Gb,
sda3-swap-2Gb,
sda4-extended,
sda5-alt-30Gb,
sda6-MOPS-30Gb,
sda7-puppy-что_останется.

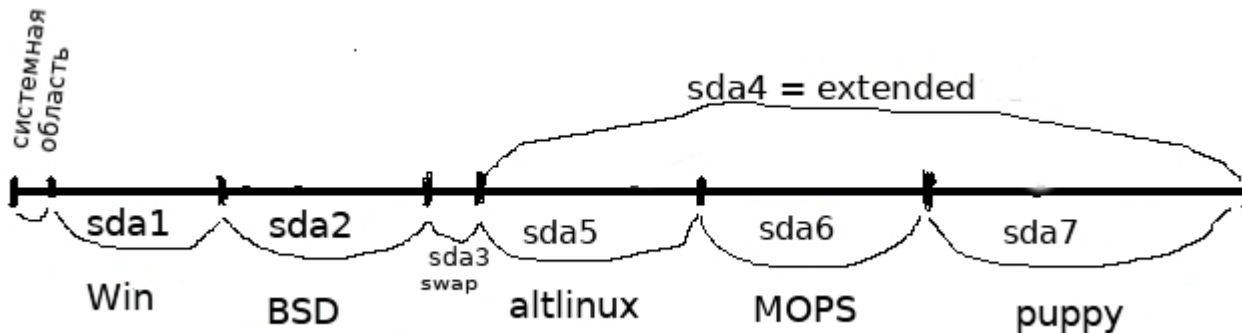


Рис. 8. Разбиение диска для 160(500)-гигабайтных hdd

В домашних условиях у вас разбиение hdd может быть другим, хотя для варианта а) вы вполне можете использовать аналогичные разбиения.

Установка операционных систем. Здесь важно соблюсти порядок установки: первой ставится Windows (поскольку она несовместима ни с чем, даже сама с собой), а затем ставятся Linux, в порядке, вам удобном.

Замечание о загрузчиках. В итоге при сдаче лабораторной вы должны продемонстрировать загрузку 4-х ОС с помощью загрузчика grub-2, именно grub 2-ой версии, а не lilo, grub4dos или ещё чего. Среди операционных систем, которые должны быть установлены на ПЭВМ (Windows, altLinux, MOPS, puppy), загрузчик grub-2 используется в altLinux и MOPS.

Пример конфигурационного файла grub-2 приведён ниже.

```
# GRUB2
set timeout=10
set default=0
set root=(hd0,3)
insmod video
insmod vbe
insmod font
loadfont /boot/grub/unifont.pf2
insmod gfxterm
set gfxmode="640x480x24;640x480"
terminal_output gfxterm
insmod png
#background_image /boot/grub/grub640.png
# End GRUB global section
# Linux bootable partition config begins
```

```
menuentry "ALTLinux 7.03 KDesktop on /dev/sda3"
{
    insmod ext2
    set root=(hd0,3)
    set gfxpayload="1024x768x24;1024x768"
    Linux (hd0,3)/boot/vmlinuz root=/dev/sda3 ro acpi=force quiet splash
    initrd (hd0,3)/boot/initrd.img
}
#
menuentry "MOPSLinux 7.0 on /dev/sda5"
{
    set root=(hd0,5)
    set gfxpayload="1024x768x24;1024x768"
    Linux /boot/vmlinuz ROOTDEV=6142e232-da21-431c-96ee-2e74aea986b5 ro acpi=force quiet
splash
    initrd /boot/initrd.gz
}
#
menuentry "MOPSLinux 7.0 (режим восстановления) на /dev/sda5"
{
    set gfxpayload="1024x768x24;1024x768"
    Linux (hd0,5)/boot/vmlinuz root=/dev/sda5 ro acpi=force quiet splash single
}
#
menuentry "PuppyLinux 2.03 KDesktop on /dev/sda6"
{
#    insmod gzio
#    insmod part_msdos
    insmod ext2
    set root=(hd0,6)
    set gfxpayload="1024x768x24;1024x768"
    Linux (hd0,6)/boot/vmlinuz root=/dev/sda6 ro acpi=force quiet splash
    initrd (hd0,6)/boot/initrd.gz
}
#
```

```
# Other bootable partition config begins
```

```
menuentry "Windows XP on /dev/sda1"
```

```
{
```

```
    set root=(hd0,1)
```

```
    chainloader +1
```

```
}
```

```
#
```

```
menuentry "Memtest86+-4.20"
```

```
{
```

```
    insmod part_msdos
```

```
    insmod ext2
```

```
    set root='hd0,3'
```

```
    search --no-floppy --fs-uuid --set=root 0ed46199-81a7-4f32-8657-caab1f45115c
```

```
    Linux16 /boot/memtest-4.20.bin
```

```
}
```

Лабораторная работа № 13

Тема: ПРОГРАММИРОВАНИЕ: РАБОТА С ПРОЦЕССАМИ

Цель: Научиться разрабатывать консольные программы работы с процессами

Задание: Разработать консольную программу на языке C согласно приведенным вариантам.

Варианты:

Задание 1. Написать программу, которая:

- создаёт подпроцесс,
- процессы идентифицируют себя, печатая сообщения и свой PID.

Задание 2. Написать программу, которая:

- печатает на дисплее собственный текст.

Задание 3. Написать программу, которая:

- создаёт подпроцесс,
- процессы идентифицируют себя, печатая сообщения и свой PID,
- процесс-потомок печатает на дисплее собственный текст.

Задание 4. Написать программу, которая:

- создаёт файл с именем <имя программы>.txt и пишет в него 10 строк: «N строки: этот файл создан программой <имя программы>-автор <ФИО>».

Задание 5. Написать программу, которая:

- создаёт файл с именем <имя программы>.txt и пишет в него исходный текст программы; в конце, с новой строки - «Автор — ФИО».

Задание 6. Написать программу, которая:

- открывает файл с исходным текстом программы и выводит его на экран.

Задание 7. Написать программу, которая:

- создаёт новый поток исполнения,
- основной процесс и поток идентифицируют себя, печатая сообщения (своё имя) и свой PID.

Задание 8. Написать программу, которая:

- создаёт новый поток исполнения,
- основной процесс и поток идентифицируют себя, печатая сообщения (своё имя) и свой PID,
- в основном процессе производится вычисление факториала 10 и печатается результат,
- в потоке производится вычисление факториала 12 и печатается результат.

Задание 9. Написать программу, которая:

- создаёт новый поток исполнения,
- основной процесс и поток идентифицируют себя, печатая сообщения (своё имя) и свой PID,
- в потоке производится вычисление факториала 12,
- в основном процессе печатается результат вычисления, сделанного в потоке.

Задание 10. Написать программу, которая:

- может обработать аргументы (ключи) строки запуска -a, -b, -c, -d и -e;
- реакция на ключи: печать сообщения «Имя программы: задан аргумент «-ключ» - обработано»,
- если ни один ключ не задан, то выдать сообщение: «Usage: <имя программы> -a, -b, -c, -d, -e».

Задание 11. Написать программу, которая:

- может обработать аргументы (ключи) строки запуска -a, -b, -c, -d, -e;
- реакция на ключи: печать сообщения «Имя программы: задан аргумент «-ключ» - обработано»,
- если задан ключ -b, то программа выводит на экран собственный текст.,
- если ни один ключ не задан, то выдать сообщение: «Usage: <имя программы> -a, -b, -c, -d, -e».

Задание 12. Написать программу, которая:

- может обработать аргументы (ключи) строки запуска -a, -b, -c, -d, -e;
- реакция на ключи: печать сообщения «Имя программы: задан аргумент «-ключ» - обработано»,
- если задан ключ -a, то программа создаёт файл с именем <имя программы>.txt и пишет в него 10 строк: «N строки: этот файл создан программой <имя программы>-автор <ФИО>»,
- если ни один ключ не задан, то выдать сообщение: «Usage: <имя программы> -a, -b, -c, -d, -e».

Задание 13. Написать программу, которая:

- может обработать аргументы (ключи) строки запуска -a, -b, -c, -d, -e;
- реакция на ключи: печать сообщения «Имя программы: задан аргумент «-ключ» - обработано»,
- если задан ключ -c, то программа создаёт файл с именем <имя программы>.txt и пишет в него исходный текст программы; в конце, с новой строки - «Автор — ФИО»,
- если ни один ключ не задан, то выдать сообщение: «Usage: <имя программы> -a, -b, -c, -d, -e».

Задание 14. Написать программу, которая:

- может обработать аргументы (ключи) строки запуска -a, -b, -c, -d, -e;
- реакция на ключи: печать сообщения «Имя программы: задан аргумент «-ключ» - обработано»,
- если задан ключ -c, то программа создаёт файл с именем <имя программы>.txt и пишет в него исходный текст программы; в конце, с новой строки - «Автор — ФИО»,
- если ни один ключ не задан, то выдать сообщение: «Usage: <имя программы> -a, -b, -c, -d, -e».

Задание 15. Написать программу, которая может обработать аргументы (ключи) строки запуска -a, -b, -c, -d, -e. Исполняемый файл программы должен называться progа. Создать жёсткую ссылку на этот файл progа1. Работа программы:

- если программа запущена из исполняемого файла с именем progа с каким-либо ключом (ключами), то печатается сообщение «Имя программы: задан аргумент «-ключ» - обработано»,
- если программа запущена из исполняемого файла с именем progа1, то выдаётся сообщение: «Error: Неверный вызов программы.
Usage: progа -a, -b, -c, -d, -e».

Задание 16. Написать программу, которая может обработать аргументы (ключи) строки запуска -a, -b, -c, -d, -e. Исполняемый файл программы должен называться progа. Создать жёсткую ссылку на этот файл progа1. Работа программы:

- если программа запущена из исполняемого файла с именем progа с каким-либо ключом (ключами), то печатается сообщение «Имя программы: задан аргумент «-ключ» - обработано»,
- если программа запущена из исполняемого файла с именем progа1, то программа создаёт файл с именем progа.txt и пишет в него исходный текст программы; в конце, с новой строки - «Автор — ФИО»,

Порядок сдачи лабораторной работы:

1. Показать функционирования программы в лаборатории. Провести доработку или исправление по указанию преподавателя.
2. Представить отчет.

Общие требования к отчету указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчет должен содержать следующую информацию:

- а) задание на работу;
- б) распечатку файла программы с именем = <фамилия_латинскими_буквами.txt>;
- в) описание программы;
- г) вместе с отчетом в электронном виде сдаётся исполняемый файл и исходные тексты программы.

Дополнительная справочная информация

Создание неименованного канала

Основные аспекты создания неименованного канала, родственного процесса и организации взаимодействия между родственными процессами через неименованный канал показано в нижеследующей программе.

```
#include <stdio.h>
#include <utmp.h>
#include <fcntl.h>
#include <unistd.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/stat.h>
int main()
{
    struct utmp record;           // структура для записи данных из utmp
    struct utmp record2;        // она же но для записи из межпроцессного канала
    int utmpfd;                  //fd для utmp
    int reclen=sizeof(struct utmp); //значение len для read()
    int fd[2];
    int filefd;
    pid_t childpid;
    if ((utmpfd = open (UTMP_FILE,O_RDONLY)) == -1) //открываем файл utmp
    {
        perror (UTMP_FILE);
        exit (1);
    }
}
```

```

}
pipe (fd); //создание межпроцессного канала
while (read(utmpfd,&record,reclen) == reclen)
{
if ((childpid = fork()) == -1) //проверка на ошибки
{
perror("fork");
exit(1);
}
if (childpid == 0) //процесс потомок
{
close (fd[0]); //закрытие межпроцессного канала на чтение
write (fd[1], &record, sizeof(struct utmp));
printf ("\nrec1 - %-8.8s ",record.ut_name); //проверяю считывается ли
//при каждой итерации
// в record информация из utmp
printf ("%8.8s \n",record.ut_line); //проверяю считывается ли при
//каждой итерации
// в record информация из utmp
exit (0);
}
else //процесс родитель
{
close (fd[1]); //закрытие межпроцессного канала на запись
read (fd[0], &record2, sizeof(struct utmp));
printf ("\nrec2 - %-8.8s ",record2.ut_name);
printf ("%8.8s ",record2.ut_line);
}
}
printf ("\n");
close (utmpfd); //закрываем utmp
return 0;
}

```

Аргументы функции main()

В языке C заданы два встроенных аргумента функции main: argc и argv.

Выглядит это так:

```
int main (int argc, char *argv[]) { ... }
```

Аргумент `argc` типа `integer` содержит в себе количество аргументов командной строки.

Аргумент `argv` типа `char` - указатель на массив строк. Каждый элемент массива указывает на аргументы командной строки (ключи программы). Один параметр отделяется от другого пробелами. Здесь:

- `argv[0]` - полное имя запущенной программы;
- `argv[1]` - первая строка записанная после имени программы (первый ключ);
- `argv[2]` - вторая строка записанная после имени программы (второй ключ);
- `argv[argc-1]` - последняя строка записанная после имени программы;
- `argv[argc]` - `NULL`.

Существует еще и третий аргумент `env`, который, так же как и `argv` является указателем на массив строк, но содержит установки среды:

```
int main (int argc, char *argv[], char *env[]) { ... }
```

Если необходимо в качестве параметра иметь строку, содержащую пробелы, то ее надо заключить в двойные кавычки. Если аргументом является число, то оно рассматривается как строка. Для работы с ним, как с числом необходимо его преобразовать, используя соответствующую функцию.

Пример:

```
#include "stdio.h"
#include "stdlib.h"
int main(int argc, char *argv[], char *env[]) {
    int i;
    printf("Количество аргументов командной строки %d \n", argc);
    printf("Аргументы командной строки:\n");
    for (i = 0; i < argc; i++)
        printf("%s\n", argv[i]);
    printf("\nАргументы состояния среды:\n");
    for (i = 0; env[i] != NULL; i++)
        printf("%s\n", env[i]);
    return 0;
}
```

В листинг целесообразно добавить `getchar()`; перед `return 0`; чтобы окно при выполнении программы сразу не закрывалось. Но лучше не `getchar()`, а `getch()` из библиотеки `conio.h`

Лабораторная работа № 14

Тема: ПРОГРАММИРОВАНИЕ: УЧЕТ ПОЛЬЗОВАТЕЛЕЙ ОС

Цель: Научиться разрабатывать системные программы учета пользователей

Задание: Разработать программу, удовлетворяющую ниже приведенным требованиям.

1. Интерфейс может быть:

- терминальный режим;
- псевдографика Perl или Tcl/Tk — тоже терминальный режим;
- графический режим.

Соответственно определяются четыре варианта реализации:

- терминальный (на языке C);
- терминальный с псевдографикой (на языках Perl или Tcl/tk);
- графический-1 (на языке C++);
- графический-2 (на языке Java).

2. Языки программирования: Perl, Tcl/Tk, C, C++, Java. Использование других языков предварительно согласовать.

3. Описания структур данных pwd и utmp смотреть в справке (в man'e) по утилитам utmp, getutent, и в заголовочных файлах /usr/include/pwd.h и /usr/include/utmp.h.

4. Интерфейс системы должен обеспечивать

- получение статистики о пользователях в табличной форме на экране;
- печать отчётов (таблиц, отображаемых на экране);
- работу с программой: в терминальном режиме и режиме псевдографики — клавиатура, в графическом режиме — мышь и клавиатура.

Указания к выполнению работы

Для реализации графического интерфейса рекомендуется использовать библиотеку xlib (см. [dfe/petrus.ru/posob/X/index.html](http://dfe.petrus.ru/posob/X/index.html)) либо библиотеку QT.

Порядок сдачи лабораторной работы:

1. Показать функционирования программы в лаборатории. Провести доработку или исправление по указанию преподавателя.
2. Представить отчет.

Общие требования к отчету указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчет должен содержать следующую информацию:

- а) задание на работу;
- б) распечатку файла программы с именем = <фамилия_латинскими_буквами.txt>;
- в) документ «Описание программы» и документ «Руководство оператора»;
- г) вместе с отчётом в электронном виде сдаётся исполняемый файл и исходные тексты программы.

Дополнительная справочная информация

Структура utmp:

```
struct utmp
{
    short int ut_type;           /* тип входа */
    pid_t ut_pid;               /* идентификатор процесса входа */
    char ut_line[UT_LINESIZE]; /* 32 символа */
    char ut_id[4];              /* идентификатор inittab */
    char ut_user[UT_NAMESIZE]; /* 32 символа */
    char ut_host[UT_HOSTSIZE]; /* 256 символов */
    struct timeval ut_tv;
    struct exit_status ut_exit; /* состояние бездействующего процесса */
    long ut_session;
    int32_t ut_addr_v6[4];
};
```

Пример. В программе демонстрируются некоторые методы чтения данных из utmp и wtmp.

```
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <time.h>
```

```

#include <sys/time.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <utmp.h>
#include <popt.h>
//-----

void print_utmp_entry(struct utmp * u)
{
    struct tm *tp;
    char * type;
    char addrtext[INET6_ADDRSTRLEN];
    switch (u->ut_type)
    {
        case EMPTY: type = "EMPTY"; break;
        case RUN_LVL: type = "RUN_LVL"; break;
        case BOOT_TIME: type = "BOOT_TIME"; break;
        case NEW_TIME: type = "NEW_TIME"; break;
        case OLD_TIME: type = "OLD_TIME"; break;
        case INIT_PROCESS: type = "INIT_PROCESS"; break;
        case LOGIN_PROCESS: type = "LOGIN_PROCESS"; break;
        case USER_PROCESS: type = "USER_PROCESS"; break;
        case DEAD_PROCESS: type = "DEAD_PROCESS"; break;
        case ACCOUNTING: type = "ACCOUNTING "; break;
    }
    printf("%-13s:", type);
    switch (u->ut_type)
    {
        case LOGIN_PROCESS: ;
        case USER_PROCESS: ;
        case DEAD_PROCESS: printf(" line: %s", u->ut_line);
                                                                    /* fall through */
        case INIT_PROCESS: printf("\n pid: %6d id: %4.4s", u->ut_pid, u->ut_id);
    }
}

```

```

}
printf ("n");
tp = gmtime(&u->ut_tv.tv_sec);
printf("time: %24.24s.%lun", asctime(tp), u->ut_tv.tv_usec);
switch (u->ut_type)
{
  case USER_PROCESS: ;
  case LOGIN_PROCESS: ;
  case RUN_LVL: ;
  case BOOT_TIME: printf("пользователь: %sn", u->ut_user);
}
if (u->ut_type == USER_PROCESS)
{
  if (u->ut_session)
    printf(" сеанс: %lun", u->ut_session);
  if (u->ut_host)
    printf (" хост: %sn", u->ut_host);
  if (u->ut_addr_v6[0])
  {
    if (!(u->ut_addr_v6[1] | u->ut_addr_v6[2] | 60:   u->ut_addr_v6[3]))
    {
      /* заполнение только первой группы означает адрес IPV4 */
      inet_ntop(AF_INET, u->ut_addr_v6,
        addrtext, sizeof(addrtext));
      printf(" IPV4: %sn", addrtext);
    }
    else
    {
      inet_ntop(AF_INET_6, u->ut_addr_v6,
        addrtext, sizeof(addrtext));
      printf (" IPV6: %sn", addrtext);
    }
  }
} // if
} //if

```

```

if (u->ut_type == DEAD_PROCESS)

{
    printf(" завершение : %u: %un", u->ut_exit.e_termination, u->ut_exit.e_exit);
}
printf("n");
} // end print_utmp_entry
//-----

struct utmp * get_next_line (char * id, char * line)

{
    struct utmp request;
    if (!id && !line)
        return gettutent();
    memset(&request, 0, sizeof(request));
    if (line)
    {
        strncpy(&request.ut_line[0], line, UT_LINESIZE);
        return getutline(&request);
    }
    request.ut_type = INIT_PROCESS;
    strncpy(&request.ut_id[0], id, 4);
    return getutid(&request);
} // end get_next_line
//-----

void print_file(char * name, char * id, char * line)

{
    struct utmp *u;
    if (utmpname(name))
    {
        fprintf(stderr, "сбой при открытии базы данных utmp %sn", name);
        return;
    }
    setutent();
    printf("%s:n=====n", name);
    while ((u = get_next_line(id, line)))

```

```

{
    print_utmp_entry(u);
        // POSIX требует очистки статических данных перед
        //     повторным вызовом getutline или getutid
    memset(u, 0, sizeof(struct utmp));
}
endutent();
} // end print_file
//=====
int main(int argc, const char **argv)
{
    char * id = NULL, *line = NULL;
    int show_utmp = 1, show_wtmp = 0;
    int c;
    poptContext optCon;
    struct poptOption optionsTable[] =
        {
            {"utmp", 'u', POPT_ARG_NONE|POPT_ARGFLAG_XOR, &show_utmp,
0, "переключить просмотр содержимого файла utmp", NULL},
            {"wtmp", 'w', POPT_ARG_NONE | POPT_ARGFLAG_XOR, show_wtmp,
0, "переключить просмотр содержимого файла wtmp", NULL},
            {"id", 'i', POPT_ARG_STRING, &id, 0,
"показать записи процесса для заданного идентификатора inittab",
"<inittab id>" },
            {"line", 'l', POPT_ARG_STRING, &line, 0,
"показать записи процесса для заданной строки устройства",
"<line>" },
            POPT_AUTOHELP,
            POPT_TABLEEND
        };
    optCon = poptGetContext("utmp", argc, argv, optionsTable, 0);
    if ((c = poptGetNextOpt(optCon)) < -1)
    {
        fprintf(stderr, "%s:%sn",
            poptBadOption(optCon, POPT_BADOPTION_NOALIAS),

```

```
poptStrerror(c));
return 1;
}
poptFreeContext(optCon);
if (id && line)
    fprintf(stderr, "Невозможно выбирать сразу по идентификатору и строке,"
               "выбор по строкам");
if (show_utmp)
    print_file(_PATH_UTMP, id, line);
if (show_utmp && show_wtmp)
    printf("nnn");
if (show_wtmp)
    print_file(_PATH_WTMP, id, line);
return 0;
} //end main
```


3. ЛАБОРАТОРНЫЕ РАБОТЫ: ИНФОКОММУНИКАЦИОННЫЕ УСТРОЙСТВА

Лабораторная работа № 1

Тема: НАСТРОЙКА ЛОКАЛЬНОЙ ВЫЧИСЛИТЕЛЬНОЙ СЕТИ В УСЛОВИЯХ ОТСУТСТВИЯ DNS

Цель: Научиться настраивать локальную сеть в условиях отсутствия DNS посредством настройки resolver'a.

Задание: Необходимо получить стандартно работающую локальную сеть, в которой обращение к другому компьютеру локальной сети обеспечивается:

- а) при указании ip-адреса компьютера,
- б) по полному имени компьютера,
- в) по краткому имени компьютера.

Для проверки правильности настройки сети на всех компьютерах установить сервис telnet (допустимо использовать ssh).

Для выполнения лабораторной работы используется следующее аппаратно-программное обеспечение лаборатории:

- дистрибутивы AltLinux, MOPS, Puppy;
- 10 компьютеров, коммутатор, кабельная система.

Предполагается, что DNS и DHCP не установлены, сервис telnet может быть заменён на сервис ssh. В этой ситуации настройка сети осуществляется посредством настройки резолвера на каждом компьютере сети.

Порядок сдачи лабораторной работы:

1. Выполнить необходимые работы по настройке сети на компьютерах лаборатории.
2. Продемонстрировать доступность компьютеров сети с помощью команд:

```
$ telnet 127.0.0.1
```

```
$ telnet localhost.localdomain
```

```
$ telnet localhost
```

```
$ telnet <ip-адрес своего компа>
```

```
$ telnet <полное имя своего компа>
```

```
$ telnet <hostname своего компа>  
$ telnet <ip-адрес другого компа>  
$ telnet <полное имя другого компа>  
$ telnet <имя host'a другого компа>
```

3. Представить отчёт, содержащий описание процесса настройки локальной сети в условиях отсутствия DNS.

Примечание: Если не можете установить telnet, используйте ssh.

Общие требования к отчету указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчет должен содержать следующую информацию:

- а) задание на работу;
- б) описание выполненных работ по настройке компьютеров для работы в сети;
- в) содержание изменённых в процессе настройки конфигурационных файлов;
- г) копии экрана, демонстрирующие выполнение указанных в п. 2 команд;
- д) методику «Порядок правильной настройки локальной сети».

Дополнительная справочная информация

Общие сведения

Определение локальной вычислительной сети (ЛВС). В стеке протоколов TCP/IP сеть называется локальной, если

- все компьютеры этой сети доступны непосредственно по физическому адресу (по MAC-адресу),
- все компьютеры сети имеют общее (одно и то же) доменное имя,
- все компьютеры имеют ip-адреса из одной сетки класса А, В или С (или той сетки, которую определил администратор с помощью маски).

Примечание. Это правильное определение локальной сети в стеке протоколов TCP/IP. В Internet вы можете найти массу других определений понятия «локальная сеть», но почти всегда это будут определения для бухгалтеров/филологов/экологов и прочих юристов, то есть, для непрофессионалов, для людей, не знающих (и часто не желающих знать) что такое сеть и как она работает.

Резолвер

Резолвер — это совершенно необходимая часть стека TCP/IP, выглядит как библиотека, в составе которой с десятков функций, очень часто эти функции включаются в стандартную библиотеку языка C — `libc`.

В стеке TCP/IP каждому компу (сетевому узлу) назначается логическое имя. Это имя компьютера имеет две формы:

- символьную (доменное имя), предназначено для человека;
- цифровую (так называемый IP-адрес), предназначено для программ.

И поскольку человек всегда работает с символьными именами (кроме тех случаев, когда администратор непутёвый и как настроить сеть не знает, а также некоторых сугубо специальных случаев), а программы всегда в сеть посылают/принимают пакеты данных с цифровыми именами, то программам очень часто приходится преобразовывать («разрешать») имя компьютера из одной формы в другую и обратно. Эту функцию преобразования имени («разрешения») выполняет резолвер.

Краткое руководство по настройке ЛВС

Необходимость «правильной» настройки локальной сети обуславливается тем, что некоторые сетевые сервисы не смогут работать полностью или частично, если разрешение имён работает неверно или не работает вовсе.

Настройка сети выполняется в три шага (на «раз-два-три»):

Шаг 1. Определение домена и политики именования host'ов.

Здесь необходимо выбрать домен для локальной сети.

Если данная ЛВС является частью корпоративной сети фирмы и домен фирмы уже существует, то поддомен для данной ЛВС определяется просто как поддомен существующего домена (даже если создаваемая сеть находится в удалённом филиале).

Если сеть вновь создаваемая для (пока ещё) небольшой фирмы, то необходимо определить новое доменное имя второго уровня (как правило). Причём при подборе доменного имени нужно исходить из предположения, что фирма будет бурно расти в ближайшие годы и домен вероятно:

- а) будет делиться на поддомены в соответствии с ростом организационной структуры фирмы,
- б) его придётся регистрировать официально как доменное имя данной фирмы у некоторого регистратора (например, в РосНИИРОС, www.nic.ru).

Поэтому рекомендуется сразу, подобрав доменное имя, проверить его на уникальность с

помощью сервиса whois (например, на www.nic.ru).

При подборе имени следует руководствоваться двумя правилами: имя должно быть максимально коротким и имя должно соответствовать миссии и цели фирмы (бренду), то есть, быть осмысленным.

При определении доменной политики необходимо руководствоваться правилом: **каждому подразделению — свой поддомен.**

В результате будем иметь некоторый домен второго уровня, например, `firma.ru`. В нём для каждого подразделения создаём поддомены, например:

`acca.firma.ru` - бухгалтерия;
`market.firma.ru` - отдел маркетинга;
`sale.firma.ru` - отдел продаж;
`factory.firma.ru` - производственный отдел;
`store.firma.ru` - склад;
`logistika.firma.ru` - отдел снабжения.

Соответственно, компьютеры (`host'ы`) в подразделениях будут теперь именоваться, например, так:

`general.acca.firma.ru` - комп главбуха;
`pay.acca.firma.ru` - комп бухгалтера по расчёте зарплаты;
`keeper.store.firma.ru` - комп кладовщика;
`disp.factory.firma.ru` - комп диспетчера в производственном отделе;
`agent.logistika.firma.ru` - комп агента в отделе снабжения;

. . . и так далее.

В общем виде

`comp1.department1.firma.ru`,
 . . . ,
`compN.departmentL.firma.ru`.

Здесь:

`department.firma.ru` — доменная часть имени `host'a`,
`compN` — `hostname`,
`compN.department.firma.ru` — полное (каноническое) имя компьютера (оно также называется FQDN — полностью квалифицированное доменное имя).

Примечание. (Очень важно!) При реальном выполнении данного шага в некоторой фирме настоятельно рекомендуется утвердить руководством (собственником) фирмы и сам домен фирмы и порядок присвоения имён компьютерам, написав служебную записку, проект решения «О домене и именовании компьютеров» и сами правила именования, чтобы потом вас не обвиняли в самоуправстве и чтобы оградить себя от вольностей пользователей в именовании своих машин.

Шаг 2. Определение политики присвоения IP-адресов.

Здесь также нужно исходить из предположения, что фирма будет бурно расти в ближайшие годы, и ВС будет делиться на подсети в соответствии с ростом организационной структурой фирмы.

Также следует иметь в виду, что с ростом структуры фирмы появятся требования по разграничению доступа и защите, а функции разграничения доступа и защиты чаще всего базируются на проверке IP-адресов по маскам.

В настоящее время в основном используется протокол IP версии 4 (IPV4) и имеет место быть дефицит «белых» (настоящих интернетовских) IP-адресов. Очень часто фирмы имеют один-два «белых» адреса для обеспечения связи с Интернет, а для адресации компьютеров внутри фирмы используют «приватные» адреса из сетей 10.0.0.0, 172.16-32.0.0, 192.168.0.0.

Прежде всего, при проектировании структуры сети следует исходить из правила: каждому подразделению — отдельная сетка адресов. На первом шаге вы определили для подразделений фирмы поддомены, теперь для подразделений фирмы определяем отдельные сетки. То есть, получаем соответствие: подразделение — поддомен — сетка. Это важно, поскольку на этом соответствии в дальнейшем будет базироваться разграничение доступа сотрудников фирмы к информационным ресурсам.

Рекомендуется принять самое простое и надёжное решение — выбрать сетки из диапазона 192.168.0.0. Именно этот диапазон сетей рекомендуется использовать в небольших и средних фирмах по следующим причинам:

а) количество доступных для использования адресов в одной сетке класса С — 254, а редко в каком подразделении малого и среднего бизнеса требуется количество компьютеров больше этого числа;

б) всего в диапазоне 192.168.0.0 может быть использовано 254 сетки (сетки 0 и 255 использовать не рекомендуется), а редко в какой фирме малого и среднего бизнеса имеется более 254 подразделений;

в) защита базируется на проверке ip-адресов по маскам и,

- если вы придерживаетесь правила: каждому подразделению — отдельную сетку класса С, то авария/сбой/ошибка может привести к «слёту» маски, и **по умолчанию** она восстановится снова в нужном виде, то есть, 255.255.255.0, и, следовательно, защита не будет нарушена;

- если же вы используете, например, сетку класса А (10.0.0.0) и делите её искусственно на подсетки с помощью масок, то в аналогичной ситуации все маски по умолчанию восстановятся к виду 255.0.0.0 и защита нарушится;

- кроме того, сам по себе расчёт и проверка масок совсем нетривиальная задача, а защиту проверять нужно постоянно.

Таким образом, при выборе сетки адресов для локальной сети необходимо выбирать сетку из диапазона 192.168.0.0.

Шаг 3. Правка конфигурационных файлов.

На этом шаге необходимо исправить конфигурационные файлы сетевого интерфейса eth0 (сетевой платы) и конфигурационные файлы резолвера. То есть, сделать «привязку» имён к интерфейсу.

Полшага 3.1. Конфигурационные файлы интерфейса.

В дистрибутиве ALTLinux наиболее просто их поправить с помощью «Центра управления системой» (раздел «Сеть»). На страничке «Сетевые интерфейсы» необходимо ввести полное доменное имя компьютера, как определено на шаге 1, переключить порядок присвоения ip-адреса из «автоматического» в «ручное» и ввести ip-адрес сетевой карты в соответствии с шагом 2.

Можно их исправить и вручную: для ALTLinux версии 4 конфигурационные файлы сетевого интерфейса находятся в каталоге /etc/net/ifaces/, а hostname определяется в файле /etc/sysconfig/network.

Если в локальной сети есть компьютеры с ОС Windows, то необходимо тоже исправить имя компьютера, определить доменное имя, переключить порядок определения адреса на ручной и ввести ip-адрес на страничке определения сетевого интерфейса «Панели управления».

Замечание 1. Выполнение работ в полшаге 3.1 в дистрибутивах Linux осложняется тем, что велико многообразие дистрибутивов и большинство из них имеют несколько специфические конфигурационные файлы и их расположение. Более того, и конфигурационные файлы, и их расположение иногда меняются от версии к версии одного и того же дистрибутива. К тому же иногда используются разные системы конфигурирования. Поэтому, при ручном конфигурировании некоторого дистрибутива Linux, необходимо, прежде всего, выяснить, где расположены конфигурационные файлы и которые из них реально используются.

Полшага 3.2. Конфигурационные файлы резолвера:

- файл /etc/host.conf — определяет порядок разрешения имен и адресов, он должен содержать строчку

```
order hosts,bind или просто hosts bind
```

Это означает, что резолвер сначала будет смотреть файл hosts, а если в нём соответствия имя-адрес не найдено — обращаться дальше, к bind (к DNS).

- файл /etc/hosts — локальная база резолвера, файл должен содержать, например, следующее:

127.0.0.1	localhost.localdomain	localhost
192.168.199.111	pay.acca.firma.ru	pay

192.168.199.112	keeper.acca.firma.ru	keeper
192.168.199.113	general.acca.firma.ru	general
192.168.199.114	accer1.acca.firma.ru	accer1
192.168.199.115	accer2.acca.firma.ru	accer2
192.168.199.116	i.tak.dalee.ru	i

Первая колонка таблицы — ip-адреса хостов, вторая колонка — полные имена хостов, третья колонка — краткие имена хостов.

Первая строчка таблицы — определение адреса и имени для интерфейса локальной петли (lo0). **Эта строчка должна быть всегда именно такой и никакой другой.**

Последующие строчки - определение адресов и имён для интерфейсов сетевых плат всех компьютеров ЛВС (интерфейсов eth0) в предположении, что для локальной сети выбрана сетка 192.168.199.0.

Замечание 1. Файл должен содержать определение интерфейсов для **всех** компьютеров локальной сети, если какой-либо компьютер не будет описан в этом файле или в строчке описания будет ошибка, то этот компьютер не будет виден в сети и, следовательно, будет недоступен.

Замечание 2. Этот файл должен быть **одинаковым** на всех компьютерах ЛВС, в том числе, на тех, на которых установлена ОС Windows. В ОС Windows XP этот файл находится по пути C:\windows\system32\drivers\etc\ и по умолчанию называется hosts.SAM. Его необходимо исправить, как указано выше, и сохранить под именем hosts. В других версиях Windows — см. документацию на Windows.

В результате выполнения данных трёх шагов имеем правильно настроенную локальную сеть, в которой пользователи будут ходить на другие компьютеры по имени, как и положено в «приличных семьях».

Лабораторная работа № 2

Тема: КОНТРОЛЬ АКТИВНОСТИ В ЛОКАЛЬНОЙ СЕТИ

Цель: Научиться осуществлять мониторинг и контроль активности в локальной сети

Задание: Используя утилиты **icmpinfo**, **iptraf**, **mtr**, **netwatch**, **tcpdump**, проанализировать активность локальной сети.

Указания к выполнению работы:

Для выполнения лабораторной работы необходимо, прежде всего, проверить, установлены ли в системе данные программы и если не установлены, то доустановить необходимые пакеты и изучить их особенности функционирования, предварительно посмотрев справочные руководства по этим утилитам, например, man tcpdump.

Установку программ (при необходимости) проще всего выполнить при помощи synaptic.

Порядок сдачи лабораторной работы:

Продемонстрировать в лаборатории результаты работы утилит:

- a) **icmpinfo -vvv**
- b) **icmpinfo -l**
- c) **iptraf:** мониторинг трафика;
мониторинг сетевых узлов, выявление «болтливых» MAC-адресов;
статистика активности по протоколам;
статистика активности по портам.
- d) **mtr:** оценку качества сети и маршрутов;
- e) **netwatch -t:** мониторинг активности в сети по IP-адресам: кто «грузит» сеть;
- f) **tcpdump:** мониторинг активности в сети, анализ пакетов.

Общие требования к отчету указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчёт должен содержать следующую информацию:

- задание на работу;
- описание результатов применения указанных утилит с копиями экрана.

Дополнительная справочная информация

Подробную информацию об утилитах также читайте в справочных руководствах, например, в man'ах.

icmpinfo - интерпретатор сообщений ICMP.

Синтаксис:

```
icmpinfo [-v [v [v]]] [-n] [-p] [-s] [-l] [-k]
```

Это инструмент для рассмотрения сообщений ICMP, полученных на хосте. Может использоваться для того, чтобы обнаружить и зафиксировать пакеты ICMP, а также решить различные сетевые проблемы.

Ключи:

- v Даёт информацию обо всех icmp пакетах, за исключением ping
- vv Даёт информацию обо всех icmp пакетах, включая ping
- vvv Включает также содержание каждого пакета
- n Не пытается разрешать ip-адреса в символическую форму
- p Не пытается разрешать номера портов в символическую форму
- s Показывает интерфейс, который принял пакет. Полезно, только если на хосте есть несколько сетевых интерфейсов.
- l Создаёт родственный подпроцесс (fork()) и использует syslog(3), чтобы делать записи событий в протокол работы (рекомендуемое использование — доступно только root).
- k Убивает фоновый процесс, начатый с ключом -l.

iptraf — программа мониторинга сетевой активности компьютера. Устанавливается с помощью synaptic. Запуск программы осуществляется путем ввода iptraf в командной строке. Рабочее окно утилиты показано на рис.9.

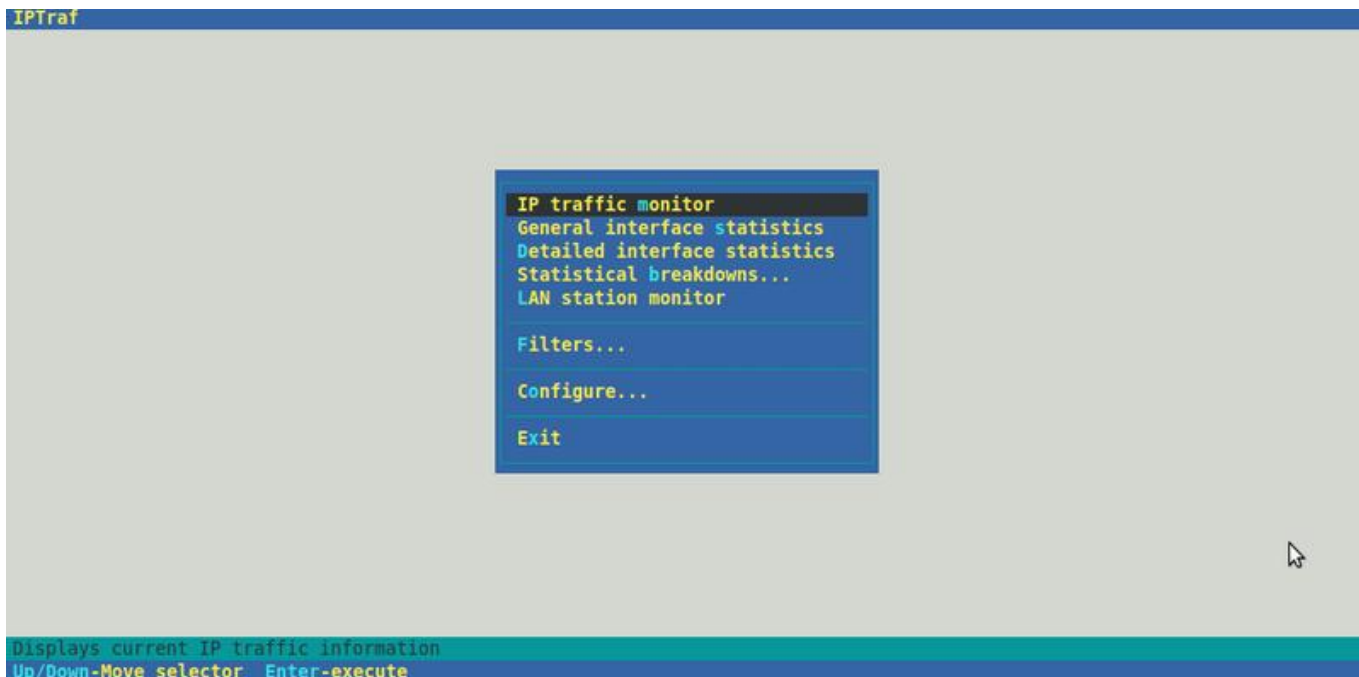


Рис. 9. Запуск iptraf

По умолчанию результаты мониторинга выдаются на экран. Для их сохранения в протоколе, нужно включить опцию логирования. Делается это в разделе «Configure» (рис. 10).

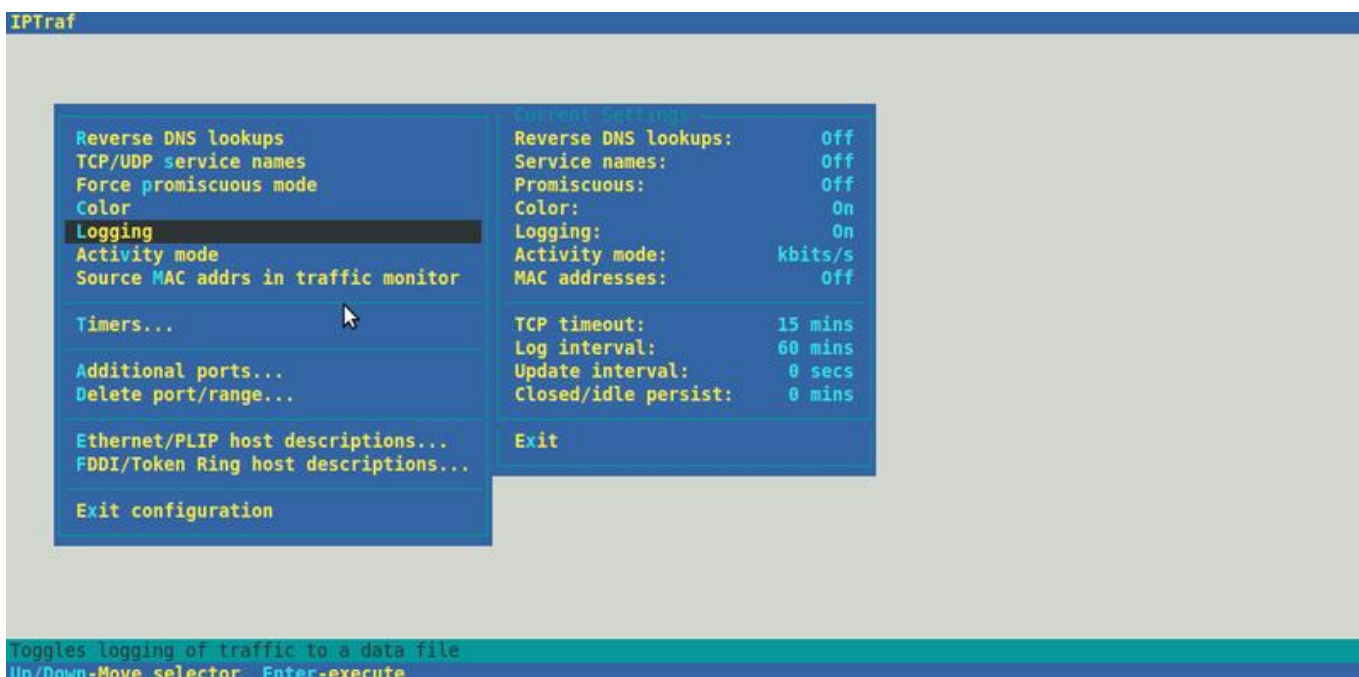


Рис.10. Включение логирования

Для инициирования мониторинга в главном меню выбирается опция «IP traffic monitor» и указывается путь к файлу, куда будем записываться сетевая активность (рис. 11).

```

IPTraf
TCP Connections (Source Host:Port) ----- Packets ----- Bytes ----- Flags ----- Iface -----
TCP: 0 entries ----- Active

Logging Enabled
Enter the name of the file to which to write the log.

If you don't specify a path, the log file will
be placed in /var/log/iptraf.

/var/log/iptraf/ip_traffic-1.log
Enter-accept Ctrl+X-cancel (turns logging off)

PKts captured (all interfaces): 0
Up/Dn/PgUp/PgDn-scroll M-more TCP info W-chg actv win S-sort TCP X-exit

```

Рис. 11. Указываем файл протокола работы

На рис. 12 показана информация о сетевой активности

```

IPTraf
TCP Connections (Source Host:Port) ----- Packets ----- Bytes ----- Flags ----- Iface -----
192.168.1.101:47046 > 3 1419 --A- eth1
74.125.43.139:80 > 2 434 -PA- eth1
192.168.1.101:48727 = 12 2533 --A- eth1
62.213.71.224:80 = 10 10222 -PA- eth1
192.168.1.101:48728 = 2 112 --A- eth1
62.213.71.224:80 = 1 60 S-A- eth1
192.168.1.101:48729 = 2 112 --A- eth1
62.213.71.224:80 = 1 60 S-A- eth1
192.168.1.101:48730 = 2 112 --A- eth1
62.213.71.224:80 = 1 60 S-A- eth1
192.168.1.101:48731 = 2 112 --A- eth1
62.213.71.224:80 = 1 60 S-A- eth1
192.168.1.101:48732 = 2 112 --A- eth1
62.213.71.224:80 = 1 60 S-A- eth1
192.168.1.101:59377 = 2 112 --A- eth1
TCP: 23 entries ----- Active

UDP (66 bytes) from 192.168.1.101:58467 to 192.168.248.21:53 on eth1
UDP (60 bytes) from 192.168.1.101:41426 to 192.168.248.21:53 on eth1
UDP (158 bytes) from 192.168.248.21:53 to 192.168.1.101:43825 on eth1
UDP (176 bytes) from 192.168.248.21:53 to 192.168.1.101:58467 on eth1
UDP (170 bytes) from 192.168.248.21:53 to 192.168.1.101:41426 on eth1
UDP (172 bytes) from 192.168.1.1:4825 to 192.168.1.255:162 on eth1
UDP (146 bytes) from 192.168.1.1:4826 to 192.168.1.255:162 on eth1
0:00:00 0:00:00 0:00:00
Pkts captured (all interfaces): 248 | TCP flow rate: 0,00 kbits/s
Up/Dn/PgUp/PgDn-scroll M-more TCP info W-chg actv win S-sort TCP X-exit

```

Рис. 12. Работа iptraf.

mtr – бесплатное приложение, которое сочетает в себе функциональность трассировки и пинг в качестве инструмента для диагностики единой сети. При этом, можно проверить потери пакетов и выяснить причины низкой скорости соединения, чтобы помочь в устранении неполадок программного обеспечения. Устанавливается с помощью `synaptic`.

Примеры использования.

```
$ mtr linux-notes.org
```

```

Macbook (0.0.0.0)                               My traceroute [v0.86]                               Sun Feb 1 02:03:40 2015
Keys: Help  Display mode  Restart statistics  Order of fields  quit

```

Host	Packets			Pings			
	Loss%	Snt	Last	Avg	Best	Wrst	StDev
1. OpenWrt.lan	0.0%	18	1.1	13.0	1.0	125.0	30.8
2. 254.212.86.109.triolan.net	0.0%	18	1.4	11.5	1.4	75.3	19.7
3. ae3-201.RT.BH.HRK.UA.retn.net	0.0%	18	1.4	12.3	1.4	76.1	23.4
4. ae2-9.RT.IRX.VIE.AT.retn.net	0.0%	17	31.4	52.5	30.9	199.4	47.3
5. vix.edge01.vie01.as13335.com	0.0%	17	49.9	44.6	31.5	128.5	27.5
6. 104.28.21.6	0.0%	17	31.4	49.1	31.3	146.4	35.7

Рис. 13. Результат работы mtr

В результате в консоли (см. рис. 13) отобразится подробная информация по всем узлам до сервера: процент потерянных пакетов, количество отправленных пакетов, последний пинг, средний пинг, лучший пинг, худший пинг.

Иногда в результате выполнения данной команды может получиться так, что процент потерь пакетов будет составлять достаточно большой. Дело в том, что некоторые маршрутизаторы не успевают обрабатывать запросы, идущие на них каждую секунду. В таком случае утилите mtr можно указать интервал, с которым она будет отправлять данные указанном серверу. Для этого используется ключ `-i` с указанием интервала в секундах. Для интервала в 2 секунды команда будет выглядеть так:

```
mtr -i 2 google.com
```

Максимальное количество отправляемых пакетов устанавливается через ключ `--report-cycles`. Например, для тестирования маршрута до сервера google.com с максимальным количеством отправленных пакетов равном 50, команда будет выглядеть следующим образом:

```
mtr --report-cycles 50 google.com
```

Во всех приведенных выше примерах результаты выводятся в терминал и отображаются только до тех пор, пока работает утилита mtr. Для того, чтобы по завершению работы утилиты результаты остались на экране, используется ключ `--report`. Выглядит команда так:

```
mtr --report google.com
```

По умолчанию для составления отчета отправляется 10 пакетов. Если для составления отчета необходимо отправить больше 10 пакетов (например 60 пакетов) с интервалом в две секунды, то необходимо воспользоваться приведенными выше ключами `--report-cycles` и `-i`. В таком случае в консоли необходимо будет ввести команду вида:

```
mtr -i 2 --report --report-cycles 60 google.com
```

А для вывода отчета в текстовый файл в конце строки необходимо добавить два знака больше (`>>`) и указать имя файла или полный путь к файлу, в который этот самый отчет будет сохраняться. Выглядеть такая команда будет так:

```
mtr -i 2 --report --report-cycles 50 google.com >> results.txt
```

В данном случае текстовый файл с отчетом необходимо искать в той папке, из которой был запущен терминал (обычно по умолчанию — это папка /home/user_name).

Еще одно важное замечание: после нескольких запусков подряд любой описанной выше команды оператор `>>` будет дописывать отчеты в конец текстового файла, а оператор `>` (одиночный знак «больше») будет каждый раз перезаписывать текстовый файл.

netwatch - отслеживает состояние хостов в сети. Осуществляется это посредством отправки ICMP-запросов (пингов) по заданным в списке IP-адресам. Для каждой записи в таблице netwatch вы можете указать IP-адрес, интервал между пингами и командные сценарии. Основным преимуществом netwatch является возможность запускать произвольные консольные команды при изменениях состояния хоста.

Предупреждение: netwatch запускает сценарии от имени системной учётной записи (пользователь *sys), поэтому любая заданная в сценарии NetWatch глобальная переменная будет недоступна для чтения планировщику или другим пользователям

Опции:

- e — указывает имя интерфейса для прослушки ;
- c — указывает путь к конфигурационному файлу;
- n — не резолвить ИП адреса;
- t — запуск netwatch в ТОП режиме (30 сек. задержка запуска);

Навигация в интерактивном режиме:

- tab — позволяет переключаться между двумя экранами;
- left/right — изменение параметров отображения (перемещение влево/вправо по опциям);
- up/down — переход к предыдущей/следующей странице пребывания на текущем списке;
- h — вызов справки;
- t — переход в ТОП режим (30 сек. задержка запуска);
- c — очистка всех сетчиков всех хостов;
- n — очистить таблицы удаленных и локальных хостов;
- N — очистите таблицы удаленных или локальных хостов;
- d — переключение в режим отображения записей домена (Name Server Запросы);
- w — войти в режим просмотра статистики маршрутизации и пакетов хоста;
- q — выход из программы.

Например , результат команды

```
# netwatch -e wlp5s2 -n
```

показан на рис. 14.

LOCAL NETWORK			REMOTE NETWORK		
HOST	IP	SERVICE	HOST	IP	SERVICE
192.168.0.15		UDP: 1900	>239.255.255.250		UDP: 0
sia		TCP: 38152	217.69.133.148		TCP: 80
			213.199.179.159		UDP: 40001
			213.199.179.149		UDP: 40001
			194.247.175.22		TCP: 80
			178.54.190.19		TCP: 30613
			176.8.57.182		TCP: 0
			173.194.113.223		TCP: 443
			173.194.113.199		TCP: 443
			173.194.113.192		TCP: 443
			173.194.32.229		TCP: 443
			173.192.82.194		TCP: 80
			157.56.193.11		TCP: 443
			157.56.52.15		TCP: 40027
			157.55.235.158		UDP: 40005
			157.55.130.169		UDP: 40015
			157.55.130.156		UDP: 40008
			157.55.130.153		UDP: 40019
			157.55.130.143		UDP: 40015
			157.55.56.161		UDP: 40018
			137.116.232.37		TCP: 443
			111.221.74.34		UDP: 40030
			111.221.74.14		UDP: 40012
			109.173.117.172		UDP: 0
			91.225.144.233		TCP: 48647
			85.198.155.112		UDP: 22579
			65.55.223.39		UDP: 40023
			64.233.164.188		TCP: 5228
			64.4.23.162		UDP: 40001
			64.4.23.157		UDP: 40007

ROUTER 163 kbits/sec Eth: 1345

Рис. 14. Результат работы netwatch

tcpdump - программа мониторинга трафика сетевого интерфейса для дальнейшего разбора. Без tcpdump сложно разобрать ошибки, которые возникают при работе с сетью и сервисами. Устанавливается с помощью synaptic.

tcpdump, по умолчанию, принимает только первые 68-96 байт данных из пакета. Для более подробного просмотра пакетов следует добавить опцию `-s <число>`, где число обозначает количество байтов, которые нужно захватить. Если установить `-s 0` (ноль) - это означает захватывать все пакеты.

Краткий список ключей, которые используются чаще всего:

- `-i any` – прослушивать трафик со всех имеющихся интерфейсов;
- `-n` – отображать IP адреса вместо имени хостов;
- `-nn` – отображать IP адреса и номера портов вместо имени хостов и названия протоколов;

- -X – показывать пакет в hex и ASCII формате;
- -XX – показывать пакет в hex и ASCII формате и выводить заголовок ethernet;
- -v, -vv, -vvv – уровни отображаемой информации о пакете;
- -c – получение определенного N количества пакетов, далее запись останавливается;
- -s – количество байт в пакете, которые обрабатывает tcpdump;
- -S – позволяет не преобразовывать абсолютные порядковые номера в относительные;
- -e – получение ethernet заголовка;
- -q – показывает минимальное количество информации о пакете;
- -E – расшифровать трафик IPSec, предоставляя ключ шифрования;
- -r – позволяет приложению tcpdump прочитать данные из файла;
- -t – не отображать метку времени в каждой строке;
- -tt – отображать неформатированную метку времени в каждой строке;
- -ttt – показывать дату и время;
- -w – записывать данные в файл.

Примеры использования приложения tcpdump

Различные выражения позволяют вам легко найти необходимый трафик. Tcpdump позволяет гибко настроить фильтры. В приложении tcpdump есть три типа параметров type, dir и proto, которые принимают следующие значения:

```
type - host, net и port,
dir - src, dst, host,
proto - tcp, udp и icmp.
```

Команда

```
tcpdump host 192.168.3.114
```

позволяет просматривать трафик на основе IP адреса или имени хоста, если не используется ключ -n.

Команды

```
tcpdump src 192.168.4.115
```

```
tcpdump dst 192.168.5.116
```

ищет трафик от определенного источника или по определенному назначению (выбирает трафик только с одной стороны).

Команда

```
tcpdump net 192.168.3.0/24
```

захватывает весь трафик в сети, но необходимо установить маску подсети.

Команда

```
tcpdump icmp
```

работает с протоколами tcp, udp и icmp.

Команда

```
tcpdump port 5060
```

захватывает трафик по определенному порту

Команды

```
tcpdump src port 5060
```

```
tcpdump dst port 5060
```

захватывают трафик по определенному порту от источника или к IP адресу назначения.

Команды

```
tcpdump src port 5060 and tcp
```

```
tcpdump udp and src port 5060
```

сочетание всех трех выражений

Команда

```
tcpdump portrange 5060-5062
```

фильтрации по диапазону портов (portrange – захват трафика по диапазону портов).

Команды

```
tcpdump less 32
```

```
tcpdump greater 128
```

захват трафика определенного размера в байтах. Так же в этом случае можно использовать символы (>, <, <=, >=):

```
tcpdump > 32
```

```
tcpdump <= 128
```


Команда

```
tcpdump -s1514 port 5060 -w voipnotes.cap
```

запись всего трафика по порту 5060 в файл. При сохранении файла желательно использовать расширение файла *.cap или *.pcap. В дальнейшем, полученный файл, можно будет открыть в WireShark.

tcpdump позволяет сохранять захваченный трафик в файл. Для этого необходимо использовать ключ -w. Для чтения трафика из файла необходимо использовать ключ -r.

Команда

```
tcpdump -r voipnotes.cap
```

загрузки файла в tcpdump.

Лабораторная работа № 3

Тема: РАБОТА С СУБД MySQL

Цель: Научиться работе с СУБД MySQL в терминальном режиме

Задание: Установить пакеты СУБД MySQL. Обеспечить запуск сервера при старте ПЭВМ. Создать базу данных и учетную запись пользователя.

Указания к выполнению работы:

Установка и настройка

1. Установить СУБД.

Сначала проверить установлены ли пакеты `mysql-server` и `mysql-client`.

Если не установлены, то необходимые пакеты установить (рекомендуется устанавливать с помощью `synaptic`), иначе - перейти к п.2.

В качестве репозитория рекомендуется использовать штатный диск дистрибутива, с которого ставилась система. Если на штатном диске нет нужных пакетов — скопировать из Интернета.

2. Обеспечить запуск СУБД при старте ПЭВМ.

3. Проверить наличие пароля *rootsqladm* у администратора СУБД командой

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

В случае наличия указанного пароля произойдет подключение к служебной базе данных, иначе установить пароль *rootsqladm* администратору `root`.

4. Установить соединение с СУБД MySQL под учетной записью администратора (`root'a`) так:

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

здесь: `-u root` — говорим, что зайдём пользователем `root`,

`-p` — говорим, что будем вводить пароль,

`mysql` — говорим, что подключаемся к БД `mysql`;

Создать базу с именем `<FIO>` (то есть, первые буквы своего ФИО, большие, латинские).

Создать обычного пользователя с логином `<io>` (то есть, первые буквы имени-отчества,

маленькие, латинские) с правами на созданную базу, и, возможно, с паролем.

Выйти из соединения с СУБД.

Примечание. Не путайте пользователя root в Linux с пользователем root в MySQL — это разные пользователи!

5. Подключиться к СУБД созданным пользователем:

```
mysql -u <io> -p <FIO> <Enter>
```

где io — логин созданного пользователя(см. пункт 4),
-p — говорим, что будем вводить пароль,
FIO — имя базы, созданной для данного пользователя.

Работа с СУБД

6. Далее необходимо создать в базе <FIO> таблицу **ITSS21**. Заполнить таблицу данными о всех студентах учебной группы. Структура таблицы:

«Фамилия, имя, отчество, год рождения, рост, вес, пол»

7. Создать в базе таблицу **mobila** — данные о мобильных телефонах (ввести не менее 10 марок!).

Структура таблицы:

«фирма, марка, ёмкость аккумулятора, время до перезарядки в режиме разговора, наличие диктофона, формат записи диктофона (MP3, WAV, OGG, иной — указать какой при заполнении), наличие фотоаппарата, разрешение фотоаппарата»;

8. Создать в базе таблицу **provider** — данные о провайдерах Internet'а Ульяновска, которые могут быть использованы студентами группы. Структура таблицы:

«название провайдера, сайт провайдера».

9. Создать в базе таблицу **tarifs** — тарифные планы провайдеров Internet'а, которые могут быть использованы студентами группы. Структура таблицы:

«название тарифного плана, провайдер, входящая_скорость, лимитный/безлимитный».

10. Выполнить запросы:

– «студенты женского пола, имеющие мобильник с фотоаппаратом»;

- «студенты мужского пола, у которых провайдер Ростелеком, тариф лимитный»;
- «студенты мужского пола, у которых мобильник Nokia, а провайдер Ростелеком, тариф безлимитный»;
- «студенты женского пола, имеющие мобильник Samsung, а провайдер Билайн (Корбина)»;

Требования к отображению информации: таблица в виде
«Фамилия, Имя, Отчество, <остальная информация запроса в текстовом виде>»

Порядок сдачи лабораторной работы:

1. По требованию преподавателя повторить работу в лаборатории и объяснить, что, собственно, делал.
2. Продемонстрировать протокол работы.
3. Представить отчет.

Общие требования к отчету указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчет должен содержать следующую информацию:

- задание на работу;
- описание порядка запуска СУБД;
- копию экрана окна xterm с выполненной командой
`ps -ax,`
показывающей, что СУБД запущена;
- копию экрана окна xterm с выполненной командой
`mysql -u <созданный пользователь> <созданная база>,`
подключением к базе и выполненным оператором
`show tables;`
- описание запросов пункта 9 к СУБД MySQL и распечатку выполненных запросов (можно читабельную копию экрана).
- описание своих действий по работе с СУБД MySQL.

Дополнительная справочная информация

Подробную информацию смотреть в «Руководстве администратора AltLinux» и в «Руководстве Администратора MySQL» (www.altlinux.org, docs.altlinux.org, heap.altlinux.org/issues/modules/init_d/index.html).

Общие сведения

Команда

```
ps ax | grep mysqld
```

показывает список активных процессов содержащих в имени строку "mysqld"

Команда

```
mysqladmin -u root password rootsqladm
```

создает пароль администратору root СУБД. Если пароль администратора был ранее определен, то ввести пароль администратора будет невозможно.

Команда

```
#!/etc/rc.d/init.d/mysqld start
```

позволяет запустить сервер СУБД, но выполняется с правами root.

Команда

```
mysql -u root -p (далее ввести пароль root СУБД)
```

позволяет подключиться root к серверу СУБД с помощью консольной утилиты mysql. При удачном соединении получите приглашение к работе в виде знака ">". Для выхода из утилиты вводится команда quit.

Создание учетной записи

Для создания пользователя нужно выполнить следующие действия.

- 1) В терминале с правами пользователя ОС вызвать утилиту

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

и ввести пароль администратора СУБД.

- 2) Получив приглашение, ввести команду создания суперпользователя СУБД со всеми правами ALL:

```
GRANT ALL ON имя_базы.* TO логин@localhost  
IDENTIFIED BY 'пароль';
```

3) Выйти из утилиты, и зайти вновь как пользователь для проверки существования введенного пользователями

```
mysql -u ЛОГИН -p (указать пароль пользователя)
```

Можно поработать в среде утилиты, вводя следующие команды отдельно каждую и записывая смысл этих команд в конспекте

```
SELECT NOW();
```

```
SELECT USER();
```

```
SELECT VERSION();
```

```
SHOW DATABASES;
```

Команды записываются после приглашения программы

```
mysql>
```

и заканчиваются обязательным знаком "точка с запятой"(;).

Создание базы данных

Для создания базы данных нужно выполнить следующие действия.

- 1) Ввести команду создания базы данных

```
CREATE DATABASE имя_базы;
```

- 2) Сделать созданную базу текущей

```
USE имя_базы;
```

- 3) Создать таблицу, например, так:

```
CREATE TABLE vuz
```

```
(in_vuz bigint not null auto_increment primary key,
```

```
name_vuz varchar(30) not null,
```

```
adres_vuz varchar(100) not null);
```

- 4) Проверить наличие созданной таблицы

```
SHOW TABLES;
```

- 5) Проверить структуру таблицы

```
DESCRIBE vuz;
```

- 6) Заполнить таблицу значениями

```
INSERT INTO vuz (name_vuz, adres_vuz)
```

```
VALUES ("УлГУ", "ул. Набережная р.Свияги,1");
```

- 7) Проверить заполнение таблицы

```
SELECT * FROM vuz;
```

При наличии в таблице составного первичного ключа его следует описывать следующим образом

```
(атрибут1 bigint not null,  
 атрибут2 bigint not null,  
 primary key(атрибут1, атрибут2))
```

Параметр `auto_increment` не может быть добавлен ключу, если этот ключ уже имел этот параметр в другой связанной таблице.

Сохранение базы данных

Для сохранения базы данных на личном внешнем носителе нужно сделать дамп базы данных с помощью команды

```
mysqldump -u root -p --databases имя_базы --add-drop-table >  
 путь_к_файлу.sql
```

Лабораторная работа № 4

Тема: СОХРАНЕНИЕ И ВОССТАНОВЛЕНИЕ БАЗЫ ДАННЫХ ПОД УПРАВЛЕНИЕМ СУБД MySQL

Цель: Научиться сохранять и восстанавливать базу данных

Задание: Изучить технологию сохранения содержания (таблиц) базы данных и последующего их восстановления.

Указания к выполнению работы:

1. В качестве файла данных для загрузки в Базу Данных взять (скопировать из Интернета) файл, например, словари Ожегова или Зализняка в текстовом виде (.txt подготовленном для загрузки в БД). Словарь в текстовом виде представлен в виде таблицы со строками переменной длины, колонки (графы) в которой разделены символом-разделителем.

2. Далее:

- установить и включить сервер MySQL;
- открыть файл словаря текстовым редактором (не Writer'ом, а именно текстовым редактором, например, vim, или редактором mc, или редакторами pico, gedit и т. д.) и определить структуру словаря: символ-разделитель и количество колонок (граф) таблицы словаря;
- создать базу для словаря;
- создать таблицу для словаря; структура таблицы должна соответствовать структуре файла .txt;
- загрузить словарь в СУБД MySQL командой утилиты mysql, например, так:

```
LOAD DATA LOCAL INFILE "oshegov.txt" INTO TABLE oshegov
```

Здесь предполагается, что вы находитесь в своём домашнем каталоге и файл oshegov.txt находится в нём же.

3. Командой backup выгрузить созданную базу со словарём в файл.

4. Командой restore проверить, что база восстанавливается правильно.

Порядок сдачи лабораторной работы:

1. По требованию преподавателя повторить работу в лаборатории и объяснить, что, собственно, делал.
2. Продемонстрировать протокол работы.
3. Представить отчёт.

Общие требования к отчёту указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчёт должен содержать следующую информацию:

- а) задание на работу;
- б) описание выполненной работы (включая копии экранов).

Дополнительная справочная информация:

Загрузка данных в таблицу

Создав таблицу, нужно позаботиться об ее заполнении. Для этого предназначены команды `LOAD DATA` и `INSERT`.

Предположим, ваши записи соответствуют приведенным в этой таблице (обратите внимание: MySQL принимает даты в формате ГГГГ-ММ-ДД; возможно, к такой записи вы не привыкли).

name	owner	species	sex	birth	death
Fluffy	Harold	cat	f	1993-02-04	
Claws	Gwen	cat	m	1994-03-17	
Buffy	Harold	dog	f	1989-05-13	
Fang	Benny	dog	m	1990-08-27	
Bowser	Diane	dog	m	1998-08-31	1995-07-29
Chirpy	Gwen	bird	f	1998-09-11	
Whistler	Gwen	bird		1997-12-09	
Slim	Benny	snake	m	1996-04-29	

Так как вы начинаете работу с пустой таблицей, заполнить ее будет проще всего, если создать текстовый файл, содержащий по строке на каждое из животных, а затем загрузить его содержимое в таблицу одной командой.

Создайте текстовый файл с именем ``pet.txt'`, содержащий по одной записи в каждой строке (значения столбцов должны быть разделены символами табуляции и даны в том порядке, который был определен командой `CREATE TABLE`). Незаполненным полям (например, неизвестный пол

или даты смерти живых на сегодняшний день животных), можно присвоить значение NULL. В текстовом файле это значение представляется символами \N. Например, запись для птицы Whistler должна выглядеть примерно так (между значениями должны располагаться одиночные символы табуляции):

name	owner	species	sex	birth	death
Whistler	Gwen	bird	\N	1997-12-09	\N

Загрузить файл `pet.txt` в таблицу можно с помощью следующей команды:

```
mysql> LOAD DATA LOCAL INFILE "pet.txt" INTO TABLE pet;
```

Маркер конца строки и символ, разделяющий значения столбцов, можно специально задать в команде LOAD DATA, но по умолчанию используются символы табуляции и перевода строки. Воспринимая их, команда сможет корректно прочитать файл `pet.txt`. Если в качестве символа-разделителя используется не табуляция, а некий другой символ, то его нужно указать в команде load.

При добавлении одиночных записей используется команда INSERT. В самом простом варианте ее применения необходимо задать значения каждого столбца, в том порядке, в каком они были перечислены в команде CREATE TABLE. Предположим, Диана (Diane) купила хомячка по имени Puffball. Соответствующую запись в таблицу можно внести с помощью команды INSERT примерно так:

```
mysql> INSERT INTO pet
-> VALUES ('Puffball','Diane','hamster','f','1999-03-30',NULL);
```

Обратите внимание на то, что здесь строковые выражения и даты представлены в виде ограниченных кавычками строк. Кроме того, в команде INSERT отсутствующие данные можно прямо заменять на NULL. Пользоваться эвфемизмом \N, как в команде LOAD DATA, нужды нет.

Этот пример наглядно показывает, что если бы с самого начала все данные вносились в базу при помощи нескольких команд INSERT, а не одной команды LOAD DATA, то набирать пришлось бы гораздо больше текста.

Предотвращение катастроф и восстановление

Резервное копирование баз данных. Поскольку таблицы MySQL хранятся в виде файлов, то резервное копирование выполняется легко. Чтобы резервная копия была согласованной, выполните на выбранных таблицах **LOCK TABLES**, а затем **FLUSH TABLES** для этих таблиц

(см. разделы **6.7.2 Синтаксис команд LOCK TABLES/UNLOCK TABLES** и **смотри 4.5.3 Синтаксис команды FLUSH**). При этом требуется блокировка только на чтение; поэтому другие потоки смогут продолжать запросы на таблицах в то время, пока будут создаваться копии файлов из каталога базы данных. Команда **FLUSH TABLE** обеспечивает гарантию того, что все активные индексные страницы будут записаны на диск прежде, чем начнется резервное копирование.

Если есть необходимость провести резервное копирование на уровне SQL, то можно воспользоваться **SELECT INTO OUTFILE** или **BACKUP TABLE**.

Существует еще один способ создать резервную копию базы данных - использовать программу **mysqldump** или сценарий **mysqlhotcopy**. Для этого нужно выполнить следующие действия:

1) Сделать полное резервное копирование баз данных:

```
shell> mysqldump --tab=/path/to/some/dir --opt --full
```

или

```
shell> mysqlhotcopy database /path/to/some/dir
```

Можно также просто скопировать табличные файлы (файлы `*.frm`, `*.MYD` и `*.MYI`) в тот момент, когда сервер не проводит никаких обновлений. Этот метод используется в сценарии **mysqlhotcopy**.

2) Если **mysqld** выполняется, остановить его, и затем запустить с опцией `--log-update[=file_name]`. В файлах журнала обновлений находится информация, необходимая для того, чтобы повторить в базе данных последовательность изменений, внесенных с момента выполнения **mysqldump**.

Можно проводить избирательное резервное копирование посредством

```
SELECT * INTO OUTFILE 'file_name' FROM tbl_name,
```

а восстановление - при помощи

```
LOAD DATA INFILE 'file_name' REPLACE ...
```

Чтобы избежать повторения записей, в таблице должен быть первичный или уникальный ключ. Ключевое слово **REPLACE** задает замену старых записей новыми в случае, когда новая запись в значении уникального ключа повторяет старую.

Если в системе, где выполняется резервное копирование, возникают проблемы с производительностью, то решить их можно, установив репликацию и выполняя резервное копирование на подчиненном сервере вместо головного.

Синтаксис BACKUP TABLE

Синтаксис команды:

BACKUP TABLE tbl_name[,tbl_name...] TO '/path/to/backup/directory'

Команда копирует в каталог резервного копирования тот минимум табличных файлов, который достаточен для восстановления таблицы. На данный момент работает только для таблиц MyISAM. Для таблиц MyISAM копирует файлы '.frm' (определений) и '.MYD' (данных). Индексные файлы могут быть реконструированы по этим двум.

В процессе резервного копирования будет установлена блокировка чтения отдельно для каждой таблицы на время ее копирования. Если необходимо сделать резервное копирование в виде мгновенного образа нескольких таблиц, необходимо сначала запросить LOCK TABLES установки блокировки чтения для каждой таблицы в группе.

Команда возвращает таблицу со следующими столбцами:

Столбец	Значение
Table	Имя таблицы
Op	Всегда ``backup''
Msg_type	Одно из значений status, error, info или warning.
Msg_text	Само сообщение.

Заметим, что BACKUP TABLE доступна только в версии MySQL 3.23.25 и выше.

Синтаксис RESTORE TABLE

Синтаксис команды:

RESTORE TABLE tbl_name[,tbl_name...] FROM '/path/to/backup/directory'

Восстанавливает таблицу(ы) из резервной копии, созданной с помощью BACKUP TABLE. Существующие таблицы не перезаписываются: при попытке восстановления поверх существующей таблицы будет выдана ошибка. Восстановление занимает больше времени, нежели BACKUP - из-за необходимости повторного построения индекса. Чем больше в таблице будет ключей, тем больше времени заберет реконструкция. Эта команда, так же как и BACKUP TABLE, в настоящее время работает только для таблиц MyISAM.

Команда возвращает таблицу со следующими столбцами:

Столбец	Значение
---------	----------

Table	Имя таблицы
Op	Всегда ``restore``
Msg_type	Одно из значений status, error, info или warning.
Msg_text	Само сообщение.

Лабораторная работа № 5

Тема: УДАЛЕННЫЙ ТЕРМИНАЛЬНЫЙ ДОСТУП К СУБД MySQL

Цель: Научиться работе с СУБД MySQL в терминальном режиме

Задание: Установить пакеты СУБД MySQL. Обеспечить запуск сервера при старте ПЭВМ. Обеспечить возможность удалённой работы с СУБД с других ПЭВМ локальной сети. Удаленно создать базу данных и учетную запись пользователя.

Указания к выполнению работы:

Установка и настройка

1. Установить СУБД.

Сначала проверить установлены ли пакеты `mysql-server` и `mysql-client`.

Если не установлены, то необходимые пакеты установить (рекомендуется устанавливать с помощью `synaptic`), иначе - перейти к п.2.

В качестве репозитория рекомендуется использовать штатный диск дистрибутива, с которого ставилась система. Если на штатном диске нет нужных пакетов — подключить репозитории дистрибутива.

2. Обеспечить запуск СУБД при старте ПЭВМ.

3. Проверить наличие пароля *rootsqladm* у администратора СУБД командой

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

В случае наличия указанного пароля произойдет подключение к служебной базе данных, иначе установить пароль *rootsqladm* администратору `root`.

3. Обеспечить возможность удалённой работы с СУБД с других ПЭВМ локальной сети, путем настройки локальной вычислительной сети (см. соответствующую лабораторную).

4. Установить соединение с СУБД MySQL под учетной записью администратора (`root'a`) так:

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

здесь: `-u root` — говорим, что зайдём пользователем `root`,

`-p` — говорим, что будем вводить пароль,

`mysql` — говорим, что подключаемся к БД `mysql`;

Создать *только* базу с именем `<FIO>` (то есть, первые буквы своего ФИО, большие, латинские).

Создать обычного пользователя с логином `<io>` (то есть, первые буквы имени-отчества, маленькие,

латинские) с правами на созданную базу, и с *возможностью доступа с другой ПЭВМ*,
 Выйти из соединения с СУБД.

Примечание. Не путайте пользователя root в Linux с пользователем root в MySQL — это разные пользователи!

5. С другой ПЭВМ подключиться к СУБД созданным пользователем:

```
mysql -h <hostname_server'a> -u <io> -p <FIO> <Enter>
```

где io — логин созданного пользователя(см. пункт 4),
 -p — говорим, что будем вводить пароль,
 FIO — база, созданная для данного пользователя

Работа с СУБД

6. Далее необходимо создать в базе <FIO> таблицу **ITSS21**. Заполнить таблицу данными о всех студентах учебной группы. Структура таблицы:

«Фамилия, имя, отчество, год рождения, рост, вес, пол»

7. Создать в базе таблицу **mobila** — данные о мобильных телефонах (ввести не менее 10 марок!).

Структура таблицы:

«фирма, марка, ёмкость аккумулятора, время до перезарядки в режиме разговора, наличие диктофона, формат записи диктофона (MP3, WAV, OGG, иной — указать какой при заполнении), наличие фотоаппарата, разрешение фотоаппарата»;

8. Создать в базе таблицу **provider** — данные о провайдерах Internet'а Ульяновска, которые могут быть использованы студентами группы. Структура таблицы:

«название провайдера, сайт провайдера».

9. Создать в базе таблицу **tarifs** — тарифные планы провайдеров Internet'а, которые могут быть использованы студентами группы. Структура таблицы:

«название тарифного плана, провайдер, входящая_скорость, лимитный/безлимитный».

10. Выполнить запросы:

- «студенты женского пола, имеющие мобильник с фотоаппаратом»;

- «студенты мужского пола, у которых провайдер Ростелеком, тариф лимитный»;
- «студенты мужского пола, у которых мобильник Nokia, а провайдер Ростелеком, тариф безлимитный»;
- «студенты женского пола, имеющие мобильник Samsung, а провайдер Билайн (Корбина)»;

Требования к отображению информации: таблица в виде

«Фамилия, Имя, Отчество, <остальная информация запроса в текстовом виде>»

Порядок сдачи лабораторной работы:

1. По требованию преподавателя повторить работу в лаборатории и объяснить, что, собственно, делал.
2. Продемонстрировать протокол работы.
3. Представить отчет.

Общие требования к отчету указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчет должен содержать следующую информацию:

- задание на работу;
- описание порядка запуска СУБД;
- копию экрана окна xterm с выполненной командой
`ps -ax,`
показывающей, что СУБД запущена;
- копию экрана окна xterm с выполненной командой
`mysql -u <созданный пользователь> <созданная база>,`
подключением к базе и выполненным оператором
`show tables;`
- описание запросов пункта 9 к СУБД MySQL и распечатку выполненных запросов (можно читабельную копию экрана).
- описание своих действий по работе с СУБД MySQL.

Дополнительная справочная информация:

«Руководство Администратора MySQL», главы 3 и 4.

Лабораторная работа № 6

Тема: ПРОГРАММИРОВАНИЕ: СОЗДАНИЕ БАЗЫ ДАННЫХ

Цель: Научиться программировать консольное приложение, создающее базу данных, с помощью библиотеки Qt в интегрированной среде IDE Qt Creator

Задание: Согласно варианту должны быть разработаны:

- структура базы данных с описанием атрибутов (описание, идентификатор, тип, особенности);
- программа создания базы данных, соответствующая варианту, с показом структуры созданных таблиц.

Программа разрабатывается на языке C++ как консольное приложение в среде Qt Creator.

Варианты:

1. ИС учёта мобильных устройств студенческой группы.

Примерное содержание Базы Данных:

Фамилия, имя, отчество, год рождения, рост, вес, пол;

- фамилия, имя отчество, год рождения, вес, рост, пол;
- фирма, марка, ёмкость аккумулятора, время до перезарядки в режиме разговора, наличие диктофона, формат записи диктофона (MP3, WAV, OGG, иной — указать какой при заполнении), наличие фотоаппарата, разрешение фотоаппарата;
- название провайдера, сайт провайдера.

2. ИС учёта переговоров/звонков абонентов АТС. Должны учитываться обычные и междугородние звонки.

Примерное содержание Базы Данных:

- ФИО, паспортные данные, адрес регистрации, адрес места жительства;
- номер телефона, спаренный / нет;
- есть ли льготы по оплате, какие;
- стоимость абонентской платы (для спаренных = 60 % от стоимости обычных);
- тариф: обычный, совмещённый, повременка;
- тариф междугородних переговоров (различается по направлениям);
- количество звонков обычных/междугородних, длительность;

- *прочее.*

3. ИС учёта абонентов сотовой связи (некоторого провайдера)

Примерное содержание Базы Данных:

- информация об абоненте, паспортные данные и др. информация;
- номер телефона, параметры номера, сим-карты...;
- параметры оборудования клиента (марка телефона, ...);
- есть ли договор на поддержку, ...;
- *прочее.*

4. ИС учета договоров, заключаемых со сторонними организациями.

Примерное содержание Базы Данных:

- номер, код договора, даты начала/завершения, статус договора;
- наименование договора, текст договора;
- наименование и атрибуты взаимодействующей организации;
- взаиморасчёты по договору;
- *прочее.*

5. ИС учёта оборудования абонентов АТС

Примерное содержание Базы Данных:

- ФИО, паспортные данные, адрес, место жительства, ...;
- номер телефона, спаренный/нет, ...;
- линия – характеристики, параметры оборудования клиента, ...;
- результаты тестирования линии, ...;
- реквизиты договора на установку, ...;
- *прочее.*

6. ИС Предприятие – система поддержки расчёта зарплаты работников.

Примерное содержание Базы Данных:

- ФИО, должность, разряд тарифной сетки, вид оплаты (тариф, сдельная, повременка, ...), паспортные данные, состав семьи, наличие льгот, стаж работы (дата приёма), общий стаж работы, ...;
- приказы (реквизиты) о премиях, надбавках, выплатах, ...;
- удержания: налоги, перечисления, взносы куда-либо, алименты, штрафы, ...;
- перечисления в соцстрах, пенсионный фонд, ...;
- вид выплаты: наличными или перечисление на счёт, ...;
- *прочее.*

Порядок сдачи лабораторной работы:

1. Показать функционирования программы в лаборатории. Провести доработку или исправление по указанию преподавателя.
2. Представить отчет.

Общие требования к отчету указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчет должен содержать следующую информацию:

- а) задание на работу;
- б) распечатку исходных текстов программы;
- в) вместе с отчётом в электронном виде сдаётся исполняемый файл и исходные тексты программы.

Указания к выполнению работы:

Используйте информацию о операторах SQL из предыдущих лабораторных работ и на сайте <http://qt-doc.ru/qt-database.html>.

Дополнительная справочная информация:***Программирование работы с СУБД***

Для работы с базой данных нужно сначала активизировать драйвер, а затем установить связь с базой данных. После этого посредством запросов SQL можно получать, вставлять, изменять и удалять данные.

Qt представляет модуль поддержки баз данных, классы которого разделены на три уровня: уровень драйверов, программный и пользовательский. Прежде чем начать работу с базой данных, необходимо соединиться с ней, активизировав драйвер. Запросы оформляются в виде строки. Для высылки запросов используется класс `QSqlQuery`.

При помощи концепции Интервью можно легко отображать данные SQL-моделей в представлениях. Далее подробнее.

Для использования баз данных, Qt предоставляет отдельный модуль `QtSql`. Для его использования необходимо сообщить об этом — просто добавьте в проектный файл следующую

строку:

```
QT += sql
```

А для того чтобы в состоянии работать с классами этого модуля, необходимо включить заголовочный метафайл QSql.

```
#include <QtSql>
```

Классы этого модуля разделяются на три уровня:

- уровень драйверов;
- программный уровень;
- уровень пользовательского интерфейса.

К первому уровню относятся классы для получения данных на физическом уровне. Это такие классы, как: QSqlDriver, QSqlDriverCreator<T*>, QSqlDriverCreatorBase, QSqlDriverPlugin и QSqlResult.

Классы второго уровня предоставляют программный интерфейс для обращения к базе данных. К классам этого уровня относятся следующие классы: QSqlDatabase, QSqlQuery, QSqlError, QSqlField, QSqlIndex и QSqlRecord.

Третий уровень предоставляет модели для отображения результатов запросов в представлениях интервью. К этим классам относятся: QSqlQueryModel, QSqlTableModel и QSqlRelationalTableModel.

Классы первого уровня вам не придется использовать, если вы не собираетесь писать свой собственный драйвер для менеджера базы данных. В большинстве случаев все ограничивается использованием конкретной СУБД (система управления базами данных), поддерживаемой Qt.

Для соединения с базой данных, прежде всего, нужно активизировать драйвер. Для этого вызывается статический метод QSqlDatabase::addDatabase(). В него нужно передать строку, обозначающую идентификатор драйвера СУБД.

Для того чтобы подключиться к базе данных, потребуется четыре следующих параметра:

- имя базы данных — передается в метод QSqlDatabase:: setDatabaseName();
- имя пользователя, желающего к ней подключиться, — передается в метод QSqlDatabase::setUserName();
- имя компьютера, на котором размещена база данных, — передается в метод QSqlDatabase::setHostName();
- пароль — передается в метод QSqlDatabase::setPassword().

Например,

```
static bool createConnection()
{
    QSqlDatabase db = QSqlDatabase::addDatabase("QSQLITE");
    db.setDatabaseName("addressbook");

    db.setUserName("elton");
    db.setHostName("epica");
    db.setPassword("password");
    if (!db.open()) {
        qDebug() << "Cannot open database:" << db.lastError();
        return false;
    }
    return true;
}
```

Методы должны вызываться из объекта, созданного с помощью статического метода `QSqlDatabase::addDatabase()`.

Само соединение осуществляется методом `QSqlDatabase::open()`. Значение, возвращаемое им, рекомендуется проверять. В случае возникновения ошибки, информацию о ней можно получить с помощью метода `QSqlDatabase::lastError()`, который возвращает объект класса `QSqlError`.

Его содержимое можно вывести на экран с помощью `qDebug()`. Если у вас возникла необходимость получить строку с ошибкой, то нужно вызвать из объекта класса `QSqlError` метод `text()`.

Для исполнения команд SQL, после установления соединения, можно использовать класс `QSqlQuery`. Запросы (команды) оформляются в виде обычной строки, которая передается в конструктор или в метод `QSqlQuery::exec()`. В случае конструктора, запуск команды будет производиться автоматически, при создании объекта.

Класс `QSqlQuery` предоставляет возможность навигации. Например, после выполнения запроса `SELECT` можно перемещаться по собранным данным при помощи методов `next()`, `previous()`, `first()`, `last()` и `seek()`. С помощью метода `next()` мы перемещаемся на следующую строку данных, а вызов метода `previous()` перемещает нас на предыдущую строку данных. При помощи методов `first()` и `last()` можно установить первую и последнюю строку данных соответственно. Метод `seek()` устанавливает строку данных по указанному целочисленному индексу в его параметре. Количество строк данных можно получить вызовом метода `size()`.

Дополнительные сложности возникают с запросом `INSERT`. Дело в том, что в запрос нужно внедрять данные. Для достижения этого можно воспользоваться двумя методами: `prepare()` и `bindValue()`. В методе `prepare()` мы задаем шаблон, данные в который подставляются методами

bindValue(). Например:

```
query.prepare("INSERT INTO addressbook (number, name, phone, email)
VALUES(:number, :name, :phone, :email);");
query.bindValue (" :number", "1");
query.bindValue(":name", "Piggy");
query.bindValue(":phone", " + 49 631322187");
query.bindValue(":email", "piggy@mega.de");
```

Также можно воспользоваться, известным из ODBC, вариантом использования безымянных параметров:

```
query.prepare("INSERT INTO addressbook (number, name, phone, email)
VALUES(?, ?, ?, ?);");
query.bindValue("1");
query.bindValue("Piggy");
query.bindValue("+ 49 631322187");
query.bindValue("piggy@mega.de");
```

В качестве третьего варианта — можно воспользоваться классом QString, в частности методом QString::arg(), с помощью которого можно произвести подстановку значений данных.

```
int main(int argc, char** argv)
{
    QApplication app(argc, argv);

    if (!createConnection())
    {
        return -1;
    }

    //Creating of the data base
    QSqlQuery query;
    QString str = "CREATE TABLE addressbook ("
        "number INTEGER PRIMARY KEY NOT NULL, "
        "name VARCHAR(15), "
        "phone VARCHAR(12), "
        "email VARCHAR(15) "
        ");";
```

```

if (!query.exec(str))
{
    qDebug() << "Unable to create a table";
}

//Adding some information
QString strF =
    "INSERT INTO addressbook (number, name, phone, email) "
    "VALUES(%1, '%2', '%3', '%4')";

str = strF.arg("1")
        .arg("Piggy")
        .arg("+49 631322187")
        .arg("piggy@mega.de");
if (!query.exec(str))
{
    qDebug() << "Unable to do insert opeation";
}

if (!query.exec("SELECT * FROM addressbook;"))
{
    qDebug() << "Unable to execute query - exiting";
    return 1;
}

//Reading of the data
 QSqlRecord rec    = query.record();
int    nNumber = 0;
QString  strName;
QString  strPhone;
QString  strEmail;

while (query.next()) {
    nNumber = query.value(rec.indexOf("number")).toInt();
    strName = query.value(rec.indexOf("name")).toString();
    strPhone = query.value(rec.indexOf("phone")).toString();
    strEmail = query.value(rec.indexOf("email")).toString();

    qDebug() << nNumber << " " << strName << ";\t"
        << strPhone << ";\t" << strEmail;
}

return 0;
}

```

Программа, приведенная выше, демонстрирует исполнение команд SQL. Производится

создание базы, запись данных и их опрос. В результате на консоль будут выведены следующие данные:

```
1 "Piggy" ; "+49 631322187" ; "piggy@mega.de"
```

В случае удачного соединения с базой данных с помощью `createConnection()` создается строка, содержащая команду SQL для создания таблицы. Эта строка передается в метод `exec()` объекта класса `QSqlQuery`. Если создать таблицу не удастся, то на консоль будет выведено предупреждающее сообщение. Ввиду того, что в таблицу будет внесена не одна строка, в строковой переменной `strF` при помощи символов спецификации определяется шаблон для команды `INSERT`. Вызовы методов `arg()` класса `QString` подставляют нужные значения используя шаблон.

Затем, когда база данных создана, и все данные были внесены в таблицу, выполняется запрос `SELECT`, помещающий строки и столбцы таблицы в объект `query`. Вывод значений таблицы на консоль производится в цикле. При первом вызове метода `next()` этот объект будет указывать на самую первую строку таблицы. Последующие вызовы приведут к перемещению указателя на следующие строки. В том случае, если записей больше нет, метод `next()` вернет `false`, что приведет к выходу из цикла. Для получения результата запроса следует вызвать метод `QSqlQuery::value()`, в котором необходимо передать номер столбца. Для этого мы воспользуемся методом `record()`. Этот метод возвращает объект класса `QSqlRecord`, который содержит информацию, относящуюся к запросу `SELECT`. С его помощью, вызовом метода `QSqlRecord::indexOf()`, мы получаем индекс столбца.

Метод `value()` возвращает значения типа `QVariant`. `QVariant` — это специальный класс, объекты которого могут содержать в себе значения разных типов. Поэтому, в нашем примере, полученное значение нужно преобразовать к требуемому типу, воспользовавшись методами `QVariant::toInt()` и `QVariant::toString()`.

Модуль `QtSql` поддерживает концепцию Интервью, предоставляя целый ряд моделей для использования их в представлениях. Класс `QSqlTableModel` позволяет, например, отображать данные в табличной и иерархической форме.

Использование Интервью — это самый простой способ отобразить данные таблицы. Здесь не потребуется цикла для прохождения по строкам таблицы.

Ниже программа демонстрирует такую возможность.

```
int main(int argc, char** argv)
{
```



```

QApplication app(argc, argv);

if (!createConnection())
{
    return -1;
}

QTableView view;
QSqlTableModel model;

model.setTable("addressbook");
model.select();
model.setEditStrategy(QSqlTableModel::OnFieldChange);

view.setModel(&model);
view.show();

return app.exec();
}

```

После соединения с базой данных, проводимого с помощью функции `createConnection()`, создается объект табличного представления `QTableView` и объект табличной модели `QSqlTableModel`. Вызовом метода `setTable()` мы устанавливаем актуальную базу в модели. Вызов метода `select()` производит заполнение данными.

Теперь настало время немного рассказать о возможностях редактирования и записи данных. Класс `QSqlTableModel` предоставляет для этого три следующие стратегии редактирования, которые устанавливаются с помощью метода `setEditStrategy()`:

- `onRowChange` — производит запись данных, как только пользователь перейдет к другой строке таблицы;
- `onFieldChange` — производит запись данных после того, как пользователь перейдет к другой ячейке таблицы;
- `OnManualSubmit` — записывает данные по вызову слота `submitAll()`. Если вызывается слот `revertAll()`, то данные возвращаются в исходное состояние.

В примере мы используем, вызовом метода `setEditStrategy()`, стратегию `QSqlTableModel::OnFieldChange`. Теперь данные нашей модели можно изменять после двойного щелчка на ячейке. В завершение мы устанавливаем модель в представлении вызовом метода `setModel()`.

Если вам понадобится произвести отображение данных какого-либо конкретного опроса `SELECT`, то для этого целесообразнее будет воспользоваться другим классом SQL-моделей —

классом QSqlQueryModel. Листинг ниже иллюстрирует отображение только электронных адресов и телефонных номеров всех контактов с именем Piggy. В нашем случае он будет всего лишь один.

```
int main(int argc, char** argv)
{
    QApplication app(argc, argv);

    if (!createConnection()) {
        return -1;
    }

    QTableView view;
    QSqlQueryModel model;
    model.setQuery("SELECT phone, email "
                  "FROM addressbook "
                  "WHERE name = 'Piggy';"
                  );

    if (model.lastError().isValid()) {
        qDebug() << model.lastError();
    }
    view.setModel(&model);
    view.show();
    return app.exec();
}
```

Здесь мы создаем табличное представление QTableView и модель опроса QSqlQueryModel. Строка запроса передается в метод setQuery(), после чего результат выполнения запроса проверяется в операторе if на наличие проблем исполнения, с помощью метода lastError(). Неверный объект класса QSqlError, возвращаемый этим методом, означает, что ошибок нет и никаких проблем не произошло. Это проверяется методом isValid(). Возникновение проблем повлечет их отображение в qDebug(). В завершение, модель устанавливается в представлении вызовом метода setModel().

Лабораторная работа № 7

Тема: ПРОГРАММИРОВАНИЕ: УПРАВЛЕНИЕ УЧЕТНЫМИ ЗАПИСЯМИ

Цель: Научиться программировать консольное приложение, базу данных, с помощью библиотеки Qt в интегрированной среде IDE Qt Creator

Задание: Согласно варианту должна быть разработана программа, управляющая учетными записями. Программа должна обеспечивать:

- создание, редактирование, просмотр учетных записей;
- в зависимости от варианта назначение тех или иных прав пользователю на те или иные таблицы базы данных.

Программа разрабатывается на языке C++ как консольное приложение в среде Qt Creator.

Варианты:

При создании программы предусмотреть по вариантам:

1. Создание суперпользователя (обязательно для всех вариантов).
2. Создание пользователя с правами (по вариантам):
 - 1) на запись и просмотр данных только в одну таблицы,
 - 2) на запись и просмотр данных во все таблицы,
 - 3) на просмотр данных только из двух таблицы;
 - 4) на просмотр данных из всех таблиц;
 - 5) на просмотр и изменение данных только в одной таблице;
 - 6) на просмотр и изменение данных во всех таблицах;
 - 7) на полное отсутствие прав.

Порядок сдачи лабораторной работы:

1. Показать функционирование программы в лаборатории. Провести доработку или исправление по указанию преподавателя.
2. Представить отчет.

Общие требования к отчету указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчёт должен содержать следующую информацию:

- а) задание на работу;
- б) распечатку исходных текстов программы;
- в) вместе с отчётом в электронном виде сдаётся исполняемый файл и исходные тексты программы.

Указания к выполнению работы:

Используйте информацию с предыдущих лабораторных работ.

Дополнительная справочная информация:

Документация по IDE Qt Creator, в том числе, на сайте

<http://qt-doc.ru/qt-database.html>

Лабораторная работа № 8

Тема: АНАЛИЗ ВЫЧИСЛИТЕЛЬНОЙ СЕТИ ОРГАНИЗАЦИИ

Цель: Научиться анализировать вычислительные сети организации

Задание: Проанализировать структуру вычислительной сети УлГУ.

Указания к выполнению работы

- 1) Осуществить запуск средств анализа вычислительной сети УлГУ.
- 2) Продемонстрировать доступные сетевые ресурсы и сетевые сервисы.
- 3) Построить карту вычислительной сети УлГУ.
- 4) Провести анализ вычислительной сети УлГУ.
- 5) **Рекомендуемое средство для выполнения работы — OpenNMS.**

Порядок сдачи лабораторной работы

1. По требованию преподавателя повторить работу в лаборатории и объяснить, что, собственно, делал.
2. Продемонстрировать протокол работы.
3. Представить отчет.

Общие требования к отчету указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчет должен содержать следующую информацию:

- а) задание на работу;
- б) общее описание сети УлГУ;
- в) перечень доступных ресурсов/сетевых сервисов (адреса) и комментарии к ним;
- г) граф сети (рисунок, можно в виде копии экрана);
- д) описание технологии анализа сети, то есть, какими средствами анализировалась сеть, краткое описание работы с данным средством и конкретное описание процесса анализа для данной работы (с копиями экрана).

Дополнительная справочная информация:

Описание средств анализа вычислительной сети представлено на странице <http://rus-linux.net/po.php?name=po/po-02.koi>

Описание документации представлено на странице <http://rus-linux.net/main.php?name=ivs-admin.koi#scan>

Рекомендуется установить пакет OpenNMS или аналогичный по возможностям построения карты сети.

OpenNMS на AltLinux

OpenNMS— система мониторинга сетевой инфраструктуры уровня предприятия. Сайт проекта <http://opennms.org>.

Предназначение пакетов системы:

- `opennms` — метапакет для быстрой установки;
- `opennms-common` — предоставляет структуру директорий остальных пакетов;
- `opennms-core` — серверная часть программы. Тут находится демон, который занимается мониторингом;
- `opennms-webapp-jetty` — веб-интерфейс для просмотра статистики интегрированный в `opennms-сервер`;
- `opennms-doc` — документация;
- `opennms-contrib` — некоторые дополнительные скрипты которые не вошли в `core`;
- `opennms-webapp-standalone` — веб-интерфейс для просмотра статистики без `opennms-сервера`;
- `opennms-remote-poller` — это часть программы для распределённого мониторинга.

Установка OpenNMS

Предполагается что все устанавливается на один компьютер. Поэтому используется команда:

```
apt-get install opennms opennms-webapp-jetty
```

Для установки потребуется сервер базы данных PostgreSQL (Использовать можно любой версии ≥ 7.4), `java` и др. нужные для работы пакеты. После этой команды мы будет установлен демон и веб-интерфейс.

Нужно запустить сервер PostgreSQL если он еще не запущен. И затем требуется создать базу данных и загрузить данные. Для этого выполняется команда:

```
/usr/share/opennms/bin/install -ids
```

Если требуется указать логин и пароль администратора для подключения к базе данных, то

дополнительно нужно указать их с помощью соответствующих ключей

```
/usr/share/opennms/bin/install --help
```

После чего можно запустить сервис:

```
service opennms start
```

Результат можно проверить командой:

```
# opennms -v status
```

Отобразится список установленных пакетов:

```
OpenNMS.Eventd      : running
OpenNMS.Trapd       : running
OpenNMS.Queued      : running
OpenNMS.Dhcpd       : running
OpenNMS.Actiond     : running
OpenNMS.Capsd       : running
OpenNMS.Notifd      : running
OpenNMS.Scriptd     : running
OpenNMS.Rtcd        : running
OpenNMS.Pollerd     : running
OpenNMS.PollerBackEnd : running
OpenNMS.Ticketer    : running
OpenNMS.Collectd     : running
OpenNMS.Threshd     : running
OpenNMS.Discovery   : running
OpenNMS.Vacuumd     : running
OpenNMS.EventTranslator: running
OpenNMS.PassiveStatusd : running
OpenNMS.Statsd      : running
OpenNMS.Importer    : running
OpenNMS.JettyServer : running
opennms is running
```

Советы:

- так как это java, то она очень требовательна к ресурсам. Выделяйте достаточное количество памяти, если это в VPS. Также возможно нужно увеличить переменную `JAVA_HEAP_SIZE` до 300—400 в файле `/etc/sysconfig/opennms`;
- обязательно настраивайте SNMP, если хотите получать максимальную информативность;

- рекомендуется включать linkd (для этого следует раскомментировать соответствующую секцию в service-configuration.xml).

Подключение к веб-интерфейсу

Подключение к веб-интерфейсу осуществляется по адресу: <http://IP:8980/opennms> с параметрами

- Логин: admin
- Пароль: admin

Конфигурация

Дальнейшая конфигурация хорошо описана в статьях:

- http://www.opennet.ru/base/net/opennms_monitor.txt.html
- http://www.opennet.ru/base/net/opennms_monitor2.txt.html
- а также на сайте <http://www.opennms.org> .

Лабораторная работа № 9

Тема: УСТАНОВКА И КОНФИГУРИРОВАНИЕ ФАЙЛОВОГО СЕРВЕРА РАБОЧЕЙ ГРУППЫ

Цель: Научиться устанавливать и настраивать файловый сервер подразделения (ftp+nfs+samba+video+web).

Задание:

- 1) Установить и настроить файловый сервер в лаборатории. Доступ к ftp, nfs и samba реализовать только с разрешённых хостов, к video и web — со всех.
- 2) Продемонстрировать доступ к ftp, nfs и samba с разрешённых компьютеров.
- 3) Продемонстрировать отсутствие доступа к ftp, nfs и samba с других компьютеров, где доступ не разрешен.
- 4) Продемонстрировать доступ к видео и web с любых компьютеров.

Указания к выполнению работы:

1. Для выполнения лабораторной работы необходимо:

- подготовить компьютеры для сервера и для клиентов, то есть, установить ОС и настроить;

- правильно настроить сеть.

2. Подключение к удалённому компьютеру осуществлять **только по hostname или по полному имени.**

3. Проверить, возможно, нужные пакеты уже в системе установлены.

4. Для выполнения лабораторной работы доустановить нужные пакеты (с помощью synaptic), если это необходимо и настроить соответствующие сервисы.

5. Необходимые пакеты рекомендуется устанавливать с помощью synaptic или apt-get в AltLinux или mpkg в Slackware. В rpm использовать подготовленные пакеты типа .pet. В качестве репозитория использовать штатный диск дистрибутива, с которого ставилась система **или официальные репозитории — правильной так.** При этом, перед установкой пакета предварительно сделать **обновление** системы, т.е. сначала в synaptic'e назначаем правильные репозитории, затем в терминале с правами root'a выполнить команды:

```
#apt-get update,
```

затем

```
#apt-get upgrade.
```

5. Обеспечить запуск сервисов при старте ПЭВМ.
6. Доступ к расшаренным ресурсам разрешать только для определённых host'ов — указывать точно, кроме видео и web — этими сервисами могут пользоваться все. Определить в конфигурациях — кто имеет право пользоваться файловым сервером.

Порядок сдачи лабораторной работы:

1. По требованию преподавателя повторить работу в лаборатории и объяснить, что, собственно, делал.
2. Продемонстрировать протокол работы.
3. Представить отчет.

Общие требования к отчету указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчет должен содержать следующую информацию:

- а) задание на работу;
- б) описание настройки сети;
- в) описание процесса конфигурирования файлового сервера (ftp+nfs+samba);
- г) описание выполненной работы с копиями экрана.

Дополнительная справочная информация

Простая настройка vsftp

Рассмотрим простую настройку vsftp сервера. Разрешим локальным пользователям добавлять файлы. Чтобы пользователи не выходили за пределы домашнего каталога, включается chroot-окружение. А так же разрешается анонимным пользователям копирование файлы из папки /var/ftp.

Устанавливается пакет:

```
apt-get install vsftpd
```

Редактируется файл /etc/vsftpd.conf:

```
listen=YES  
anonymous_enable=YES  
local_enable=YES
```

```

write_enable=YES
local_umask=022
dirmessage_enable=YES
use_localtime=YES
xferlog_enable=YES
connect_from_port_20=YES
chroot_local_user=YES
secure_chroot_dir=/var/run/vsftpd/empty
pam_service_name=vsftpd
rsa_cert_file=/etc/ssl/private/vsftpd.pem
anon_root=/var/ftp

```

Чтобы предоставить доступ пользователям без shell-доступа, редактируется /etc/shells, путем добавления в список оболочки:

```
/bin/false
```

Добавляется пользователь со своим домашним каталогом и дается пароль

```
useradd username -d /home/username -m -U -s /bin/false
```

```
passwd username
```

Перезапускается демон:

```
service vsftpd restart
```

Если при попытке подключения возникает ошибка:

1 Ответ: 500 OOPS: vsftpd: refusing to run with writable root inside chroot(),

нужно изменить права на домашний каталог локального пользователя

```
#chmod a-w /home/username
```

Теперь можно подключаться любым ftp-клиентом, например, filezilla.

На этом экспресс настройка закончена (источник: <http://www.zonepc.ru/prostaya-nastrojka-vsftpd/>).

Общая настройка сервера NFSv4 на AltLinux

Задача — экспортировать часть ФС (/ftp/pub) с iso`шками и репозиториями (часть репозиторияев — подмонтированные iso).

Можно применить решение, описанное в

<http://www.citi.umich.edu/projects/nfsv4/linux/using-nfsv4.html#exports>

выполнив команды:

```
mkdir -p /export/pub
```

```
mount --rbind /ftp/pub /export/pub
```

Теперь в `/export/pub` доступно всё содержимое `/ftp/pub`, в том числе — и смонтированные туда `/ftp/pub/foo*` (содержимое соответствующего `/ftp/pub/ISO/foo*.iso`). Несмотря на это, в `/etc/exports` придётся упомянуть каждую из неявно (за счёт `mount --rbind`) подмонтированных в `/export/pub/*` ФС: иначе клиенты их видеть не будут.

В `/etc/exports` требуется примерно следующие (сильно упрощённо):

```
/export (ro,nohide,fsid=0)
/export/pub (ro,nohide,fsid=1)
/export/pub/foo1 (ro,nohide,fsid=2)
/export/pub/foo2 (ro,nohide,fsid=3)
/export/pub/foo<N> (ro,nohide,fsid=<N>)
```

При этом числа в параметре `fsid` должны различаться. При совпадении — видна только одна из ФС с совпадающими `fsid`. Корень для всех экспортируемых систем должен быть указан с `fsid=0`.

Поправка от `ns@` (<http://lists.altlinux.org/pipermail/sysadmins/2006-July/001723.html>):

`fsid` надо указывать только у `/export`. То есть только там, где значение должно равняться "0". Эта точка станет корнем для NFSv4. То есть код ниже, тоже работает:

```
/export (ro,nohide,fsid=0)
/export/pub (ro,nohide)
/export/pub/foo1 (ro,nohide)
/export/pub/foo2 (ro,nohide)
/export/pub/foo<N> (ro,nohide)
```

Примечание: При экспортировании чего либо нерасположенного в `/export` (например `/ftp/pub` непосредственно) — экспортированное доступно только по NFSv3. Если не вдаваться в подробности (настройка `firewall` — ниже), то с сервером всё.

Монтирование клиентом

На клиенте всё монтируется в одной точке, но пути к ресурсу для NFSv4 и NFSv3 различаются:

```
mount -t nfs4 <сервер>:/ <точка монтирования>
mount -t nfs <сервер>:/export <точка монтирования>
```

При этом как `mount -t nfs4` смонтировать удаётся только содержимое `/export`: при экспорте какого-нибудь `/ftp/pub` напрямую — он доступен только по `mount -t nfs...`

Если при выполнении монтирования команда `mount` зависает — значит, вероятно, что включена опция использования `gssd`. Чтобы не мучиться с его настройками, пропишите в файл `/etc/sysconfig/nfs` параметр `SECURE_NFS=no`.

NFSv4 и NFSv3 через firewall

Если критична одновременная поддержка клиентов обеих версий NFS (v4 и v3), то фиксацию и открытие портов можно выполнить по рецептам для NFSv3, как для более беспроблемного варианта. Используйте <http://ipesin.linux.kiev.ua/translations/rhm/tipstricks10.htm> и <http://nfs.sourceforge.net/nfs-howto/ar01s06.html>

Фиксировать и открывать на серверном firewall нужно следующие:

- nfs — работает по 2049 tcp/udp по умолчанию. Если требуется сдвинуть специально — см. [altbug:9769](#)
- mountd — параметром MOUNTD_PORT (в /etc/sysconfig/nfs)
- nlockmgr — параметрами nlm_tcpport и nlm_udpport модуля lockd (строка вида options lockd nlm_tcpport=N nlm_udpport=M в /etc/modules.conf)
- portmapper — стандартные 111 tcp/udp

Простой Samba-сервер

Изложение основ настройки сервера Samba в windows-окружении основано на книге Джона Терпстры Samba-3 в примерах (John Terpstra. Samba-3 by Example), которую вы можете найти на сайте команды разработчиков Samba www.samba.org.

В этом примере речь пойдет о том, как организовать простой файловый сервер с безпарольным доступом к общим ресурсам. То есть можно будет "всем все", помните об этом, когда вдруг захотите скопировать через такую общую папку файл с паролями от вашей кредитки.

Требования и удаленное подключение к будущему файл-серверу. На будущем файл-сервере установлен ALD 5.0 KDE, имя компьютера altsf, ip/маска 192.168.0.1/24 (для такого файл сервера адрес лучше, а возможно даже необходимо, сделать статический). Делаем все от root. Если ваш файл-сервер нормально видит сеть (если не видит, возможно, вам сюда http://www.altlinux.org/Настройка_сетевой_карты,_краткое_пособие_для_начинающих), можете в общем-то всю работу выполнять не сидя локально, а сразу подключиться удаленно к файл-серверу (sergo555 это рабочий компьютер, sergo пользователь на удаленном компьютере altsf (который заведен при установке системы)):

из линукс

```
[sergo@sergo555 ~]$ ssh -l sergo 192.168.0.1
```

```
sergo@192.168.0.1's password:
```

```
Вводим пароль_пользователя_sergo.
```

```
Last login: Wed May 26 13:55:31 2010 from sergo555.local
```

```
[sergo@altsf ~]$
```

Всё, находимся в терминале на удаленном компьютере altsf, теперь включаем root'a:

```
[sergo@altsf ~]$ su -l
```

```
Password:
```

Вводим пароль пользователя root удаленного компьютера altsf и получаем права root.

```
[root@altsf ~]#
```

Подробнее смотрим тут

<http://forum.altlinux.org/index.php/topic,6364.msg89508.html#msg89508>

из windows:

воспользуйтесь утилитой `putty.exe` <http://www.putty.org/>. В разделе Session в строке Host Name (or IP adress) введите имя вашего файл-сервера или его IP-адрес. В разделе Window-Translation в кодировках поставьте UTF-8, чтобы не было проблем с отображением кириллицы. Жмем OPEN. На надпись в терминале login as: говорим sergo, вводим пароль. После входа набираем `su -l`, вводим пароль root файл-сервера. Можно работать.

Подготовка раздела для общих ресурсов. Допустим, раздел с будущими общими ресурсами представляет из себя

```
# mount -l
```

```
/dev/sda8 on /mnt/disk type ext3 (rw,noexec,nosuid,nodev)
```

То есть раздел, который определился в системе как `sda8`, примонтировался системой в папку `/mnt/disk`

Далее идут необязательные действия, но так удобнее, (если хотите, хороший тон), переименуем папку `/mnt/disk` во что-то более информативное, особенно это актуально, если у вас несколько жестких дисков, и они смонтированы соответственно в папки, например, `/mnt/sda2`, `/mnt/sdb3` и тому подобное. К тому же, если вы недавно в linux, вы узнаете кое-что новое о простых операциях и работе с файлами.

Отмонтируем раздел:

```
# umount /mnt/disk
```

Переименуем `/mnt/disk` в `/mnt/allfiles`:

```
# mv /mnt/disk /mnt/allfiles
```

Теперь нужно подправить файл `/etc/fstab` (файл, отвечающий за то, как и куда будут смонтированы разделы в вашей системе), чтобы раздел `/dev/sda8` монтировался при загрузке в новое место (папку `/mnt/allfiles`), открываем файл в редакторе

```
# mcedit /etc/fstab
```

и исправляем строку

```
UUID=fa40a5d4-ca38-4867-bace-5a63ebd29639 /mnt/disk ext3 nosuid,nodev,noexec 1 0
```

на

```
UUID=fa40a5d4-ca38-4867-bace-5a63ebd29639 /mnt/allfiles ext3 nosuid,nodev,noexec 1 0
```

Примечание: разумеется, ваш UUID будет другим. Жмем F2 (сохранить) и F10 (выход из mc). Монтируем все обратно:

```
# mount -a
```

Планирование общих ресурсов. Теперь определимся с общими ресурсами, для примера у нас будет два общих ресурса, music (тут вся музыка предприятия, например), и обмен файлами, где ваши сотрудники могут централизованно обмениваться какой-либо информацией. Последнее название выбрано специально, имя общего ресурса может вызвать затруднения при "расшаривании" и монтировании, ниже будет показано, что на самом деле сложностей никаких нет.

Данное решение использует параметр force user. Использование параметра force user гарантирует вам, что все файлы будут принадлежать одному и тому же идентификатору пользователя (user identifier UID) и здесь никогда не будет проблем с доступом к файлам.

Пояснение: это значит, что мы заведем на файл-сервере одного пользователя, который автоматически будет становиться владельцем всех файлов и папок в ваших общих ресурсах.

Создаем владельца общих ресурсов и сами общие папки. Создадим такого пользователя в системе, имя пользователя **sambauser**, его пароль **samba1USER**, при создании зададим каталог пользователя (ключ -m) и зададим пароль (ключ -p):

```
# useradd -m sambauser -p samba1USER
```

Убедитесь, что пользователь добавился в систему:

```
cat /etc/passwd
```

```
sambauser:x:501:501::/home/sambauser:/bin/bash
```

Также убедитесь, что создан домашний каталог пользователя sambauser:

```
# ls -l /home | grep sambauser
```

```
drwx----- 11 sambauser sambauser 4096 May 26 16:02 sambauser
```

Теперь создадим папку **sharefolder**, в которой будут находиться наши будущие общие ресурсы, можно и без нее, но удобнее, если все общие папки будут лежать в отдельном каталоге, а не просто в корне точки монтирования:

```
# mkdir /mnt/allfiles/sharefolder
```

Теперь создадим будущие общие папки:

```
# mkdir /mnt/allfiles/sharefolder/{music,"обмен файлами"}
```

И назначим нового владельца, нашего товарища sambauser, а также несколько изменим разрешения:

```
# chown -R sambauser:users /mnt/allfiles/sharefolder
```

```
# chmod -R ugo+rwX /mnt/allfiles/sharefolder
```

В итоге разрешения на наши общие ресурсы выглядят вот так:

```
# ls -l /mnt/allfiles/sharefolder
```

```
drwxrwxrwx 2 sambauser users 4096 May 26 16:27 music
```

```
drwxrwxrwx 2 sambauser users 4096 May 26 16:27 обмен файлами
```

Конфигурирование сервера Samba. Теперь займемся сервером Samba.

Конфигурационный файл для сервера Samba-3 находится в папке /etc/samba и называется smb.conf. После установки в нем написано много чего интересного, но мы не будем его править, а создадим свой. Сохраняем файл /etc/samba/smb.conf на память (и на всякий случай, там много чего есть интересного почитать, и вообще заведите себе привычку перед любым изменением любого конфигурационного файла делать его резервную копию):

```
# mv /etc/samba/smb.conf /etc/samba/smb.conf.default
```

Примечание: разработчики Samba настоятельно рекомендуют иметь файл smb.conf как можно более меньшего размера, а также не иметь внутри файла комментариев. Также они рекомендуют, как можно более тщательно комментировать все изменения, какие вы делаете. Данные противоречивые рекомендации исполняются следующим образом: мы сделаем файл smb.conf.comment, в котором опишем все, что мы делаем, а потом из него получим рабочий файл smb.conf:

```
# touch /etc/samba/smb.conf.comment
```

Теперь заполним файл smb.conf.comment примерно таким содержимым:

```
#Глобальные параметры
```

```
[global]
```

#название рабочей группы, в windows по умолчанию WORKGROUP, если вы имеете домен, укажите имя вашего домена, тогда

```
#в сетевом окружении windows ваш файл-сервер будет в общей массе компьютеров
```

```
workgroup = WORKGROUP
```

#NetBIOS-имя компьютера в вашей сети, имя, под которым файл-сервер будет отображаться в сетевом окружении

```
netbios name = ALTSERVER
```

```
#режим безопасности
```

```
security = SHARE
```

```
#то, как будет описываться в сетевом окружении
```

```
server string = files and music
```



```
#описание наших общих папок
#секция music
[music]
#комментарий, то, как будет подписан наш общий ресурс в сетевом окружении
comment=Наша музыка
#путь к общей папке
path=/mnt/allfiles/sharefolder/music
#данные параметры необязательны, но для чего они потребовались тут, скажу ниже
create mask = 0777
directory mask = 0777
#принуждаем быть владельцем общего ресурса пользователя sambauser
force user=sambauser
#принуждаем быть владельцем общего ресурса группу users
force group=users
#указываем, что можно не только читать, но и записывать
read only=No
#открываем гостевой доступ, по сути всем
guest ok=Yes
```

```
[обмен файлами]
comment=файлообмен
path="/mnt/allfiles/sharefolder/обмен файлами"
create mask = 0777
directory mask = 0777
force user=sambauser
force group=users
read only=No
guest ok=Yes
```

Получим рабочий файл smb.conf, а заодно проверим, нигде ли мы не ошиблись при помощи команды testparm:

```
# testparm -s smb.conf.comment>smb.conf
Load smb config files from smb.conf.comment
Processing section "[music]"
Processing section "[обмен файлами]"
Loaded services file OK.
WARNING: You have some share names that are longer than 12 characters.
```

These may not be accessible to some older clients.

(Eg. Windows9x, WindowsMe, and smbclient prior to Samba 3.0.)

Server role: ROLE_STANDALONE

Теперь у вас появился файл smb.conf, причем в нем нет ни одной строки с комментариями. Добейтесь, чтобы не было никаких ошибок, опечаток и т.п. Утилита testparm проверяет в основном опечатки, правильность заданных параметров она почти не контролирует. Например, если вы опечатались, сообщение об ошибке может выглядеть так:

```
# testparm -s smb.conf.comment>smb.conf
Load smb config files from smb.conf.comment
Unknown parameter encountered: "metbios name"
Ignoring unknown parameter "metbios name"
Processing section "[music]"
ERROR: Badly formed boolean in configuration file: "Yes/t".
lp_bool(Yes/t): value is not boolean!
Processing section "[обмен файлами]"
Loaded services file OK.
WARNING: You have some share names that are longer than 12 characters.
These may not be accessible to some older clients.
(Eg. Windows9x, WindowsMe, and smbclient prior to Samba 3.0.)
Server role: ROLE_STANDALONE
```

В данном случае опечатка **metbios name** вместо **netbios name**, а также ошибочное значение параметра в секции **[music]**

Примечание: параметр `workgroup=WORKGROUP` применяется по умолчанию, поэтому если у вас такая рабочая группа, то его можно не писать, а если какая то другая, то смело пишете, например, `workgroup=MYPNETWORK`

Если вы игнорируете рекомендации разработчиков и нещадно редактируете непосредственно файл smb.conf, то для проверки запускать testparm без параметров.

Итоговый файл smb.conf получился таким:

```
[global]
    netbios name = ALTSERVER
    server string = files and music
    security = SHARE

[music]
    comment = Наша музыка
    path = /mnt/allfiles/sharefolder/music
```

```
force user = sambauser
force group = users
read only = No
create mask = 0777
directory mask = 0777
guest ok = Yes
```

[обмен файлами]

```
comment = файлообмен
path = "/mnt/allfiles/sharefolder/обмен файлами"
force user = sambauser
force group = users
read only = No
create mask = 0777
directory mask = 0777
guest ok = Yes
```

Запуск сервера Samba и проверка работы. Теперь говорим Samba быть всегда включенной при загрузке/перезагрузке и запускаем ее:

```
# chkconfig smb --level 35 on
[root@altsf samba]# service smb start
```

```
Starting CIFS services: NetBIOS over TCP/IP server, Samba server
```

DONE]

Проверяем, что все нормально, подключимся анонимно сами к себе:

```
# smbclient -L altserver -U%
```

```
Domain=[WORKGROUP] OS=[Unix] Server=[Samba 3.0.37]
```

```
Sharename  Type  Comment
-----  ----  -
music      Disk  Наша музыка
обмен файлами Disk  файлообмен
IPC$       IPC   IPC Service (файл-сервер)
```

```
Domain=[WORKGROUP] OS=[Unix] Server=[Samba 3.0.37]
```

```
Server      Comment
-----  -
```

ALTSERVER

```
Workgroup      Master
-----
```

WORKGROUP

Так же вы можете подключиться командой

```
# smbclient -L altserver -Usambauser
```

```
password:
```

И получить аналогичный вывод

Монтирование общих ресурсов в windows- и linux-клиентах. Все, через 1-2 минуты идем, если в windows, в сетевое окружение, если linux, то Dolphin (или что там у вас) Сеть —> Samba Shares —> Workgroup —> Altserver. Там видим наши папки music и обмен файлами, убеждаемся, что везде есть доступ, как на чтение, так и на запись.

Временно смонтировать можно такой командой, например нашу музыку в папку /mnt/music:

```
# mkdir /mnt/music
```

```
# mount -t cifs //192.168.0.1/music /mnt/music -o users,username=guest,password=""
```

Но так ходить долго, особенно если пользуешься часто, поэтому организуем автоматическое монтирование на клиенте linux. Вот как раз для этого случая потребовались параметры:

```
create mask = 0777
```

```
directory mask = 0777
```

Создадим папки, куда будем монтировать (все делаем уже не на файл-сервере, а на рабочей станции с altlinux):

```
# mkdir /mnt/music
```

```
# mkdir /mnt/obmen
```

Допишем в файл /etc/fstab такие строки (лучше в конец файла):

```
//192.168.0.1/music /mnt/music cifs users,username=guest,password="",utf8 0 0
```

```
//192.168.0.1/обмен\040файлами /mnt/obmen cifs users,username=guest,password="",utf8 0 0
```

Ну, и сделаем ссылки на рабочем столе (их можно сделать уже от обычного пользователя):

```
ln -s /mnt/music /home/sergo/Desktop/музыка
```

```
ln -s /mnt/obmen /home/sergo/Desktop/"обмен файлами"
```

Затем в Dolphin слева на панели "Точки входа" можно щелкнуть правой клавишей, выбрать "Добавить", указать описание Музыка и в пути прописать /mnt/music, ну и тоже самое для обмена файлами. Так вы упростите навигацию к вашим общим ресурсам в файловом менеджере.

В windows мы щелкаем правой клавишей на "Мой компьютер", выбираем "Подключить сетевой диск", назначаем букву диска и указываем путь, либо делаем Пуск—>Выполнить—>cmd,

и в командной строке набираем (диск m: будет музыка, o: обмен файлами):

```
c:>net use m: \\altserver\music /persistent:Yes
```

```
c:>net use o: "\\altserver\обмен файлами" /persistent:Yes
```

Длинно получилось, на самом деле все за 15 минут делается!

Лабораторная работа № 10

Тема: УСТАНОВКА И ЗАПУСК WEB-СЕРВЕРА APACHE

Цель: Научиться устанавливать и настраивать web-сервер подразделения.

Задание: Разработать собственный сайт с описанием лабораторных работ и разместить его на созданном сервере. Добавить на сайт документацию по темам лабораторных работ и по учебным дисциплинам. По своему усмотрению на сайт можно выложить ещё что-нибудь.

Указания к выполнению работы:

1. Необходимые пакеты рекомендуется устанавливать с помощью `synaptic`. В качестве репозитория использовать штатный диск дистрибутива, с которого ставилась система.
2. Обеспечить запуск web-сервера при старте ПЭВМ. Как это сделать - смотреть в «Руководстве администратора AltLinux».
3. Web-сервер должен быть виден в локальной сети лаборатории.
4. Создать контент:

- **через меню главной странички обеспечить доступность отчётов по всем ранее выполненным лабораторным работам в формате html.**

5. Можете добавить на сайт что-то ещё.
6. Найти «лёгкий» хостинг и создать сайт на нём. Хостинг «лёгкий», если пустая страница (пустой шаблон) «весит» не более 20-30 килобайт. **Антипример:** пустая страница на `ucoz.ru` весит полмегабайта (!), поэтому категорически запрещается этим хостингом пользоваться. И другими аналогичными - трафик надо экономить.

Порядок сдачи лабораторной работы:

1. По требованию преподавателя повторить установку сервера в лаборатории и объяснить, что, собственно, делал.
2. Продемонстрировать доступность web-сервера в лаборатории:
 - локальной версии, установленной на компьютере лаборатории,
 - версии в Интернете с отчётами по выполненным лабораторным работам.
3. Представить отчет.

Общие требования к отчету указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчет должен содержать следующую информацию:

- а) задание на работу;
- б) screen окна xterm с выполненной командой ps -ax, показывающей, что apache запущен;
- в) screen окна firefox, показывающий главную страничку своего контента (то есть, файл <family_io>/index.html);
- г) описание порядка запуска и настройки web-сервера (apache);

В электронном виде также сдать отчет и сайт в виде архива.

Дополнительная справочная информация:

Правила и требования оформления сайта

Правильное оформление блока HEAD

```
<head>
```

```
<meta http-equiv="content-type" content="text/html;charset=windows-1251">
```

```
<title>Имя страницы – это то, что отобразится в заголовке браузера</title>
```

```
<meta name="Author" content="Автор – не забывайте себя порекламировать">
```

```
<meta name="description" content="Краткое описание страницы - аннотация">
```

```
<meta name="keywords" content="Ключевые слова – слова, чаще всего встречающиеся на
```

странице, их может быть много; теги keywords, description и title проверяются на согласованность поисковыми машинами, их содержание должно соответствовать содержанию самой страницы.

Лучшее решение: содержание title входит в состав description, а слова из description входят в

keywords + в keywords нужно добавить наиболее часто встречающиеся слова из текста страницы и которые наиболее соответствуют теме страницы">

```
</head>
```

Простейший вариант HTML-файла приведен ниже:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
```

```
<HTML>
```

```
<HEAD>
```

```
<TITLE>Моя первая WEB-страничка</TITLE>
```

```
</HEAD>
```

```
<BODY>
```

Добро пожаловать на мою первую WEB-страницу !

```
</BODY>
```

```
</HTML>
```

Первая строчка – необязательная. Обычно её добавляют Web-редакторы автоматически. Она объявляет о том, какая версия HTML использована. Здесь, как видите, версия 3.2.

Теги могут писаться и на нижнем регистре, это несущественно.

Как видно из приведенного текста, любой HTML-файл начинается тегом <HTML> и кончается </HTML> и включает блок <BODY>...</BODY> (символ наклонной черты перед именем тега ограничивает его область действия), теги могут иметь атрибуты.

После сохранения этой информации в файл index.html (используется также расширение имени файла htm) и двойного щелчка мышкой по имени, открывается браузер и вы видите то, что создали.

Имена файлов. Очень часто web-серверы работают на unix'овых машинах (бесплатные хостинги — все на unix-овых машинах), а в unix файлы index.html, Index.html, iNdex и inDEx.html (например) – это разные файлы, в то время, как в Windows – это один и тот же файл. Кроме того, unix может не понять русские буквы, кодировки могут оказаться разными. На какой машине ваш web-сервер? А завтра на какой будет? **Вывод: всегда пишите имена файлов только маленькими латинскими буквами.** Можно ещё использовать цифры, тире и знак подчёркивания. Пробелы и другие спецсимволы использовать нельзя.

Правильно оформляйте ссылки. Важным свойством языка HTML является возможность описания *гиперпереходов* между страницами (и/или отдельными точками на одной и той же странице), именно это определяет огромные возможности HTML по представлению информации со сложными логическими связями.

Ниже приведен пример строки с возможностью перехода по гиперсвязи (при интерпретации браузером выделенная тегами <A...> и часть строки отображается в отличном от других частей строки цвете, при наведении курсора 'мыши' на эту выделенную цветом часть строки и щелчке левой кнопкой 'мыши' браузер перейдет к странице (фактически загрузит новый файл) с именем about_me.htm (имя указано в параметрах тега <A>):

Добро пожаловать на [мою](about_me.html) вторую WEB-страницу !

Другой пример - переход к метке ('якорю') my_born на странице about_me.htm:

Добро пожаловать на [мою](about_me.html#my_born) третью WEB-страницу !

В HTML определены теги гиперпереходов по части текста, изображения, элементам мультимедиа, существует большой набор тегов описания стилей текста, расположения и масштабирования изображений и др. Ниже приведен пример использования URL-адресации в HTML-тексте

```

```

В данном случае браузер отобразит файл CRIM_ANI.GIF, находящийся в подкаталоге gif_89 WEB-сайта pilger.mgapi.edu (естественно, если этот сайт доступен по Сети, то есть это ссылка на чужой ресурс! Не злоупотребляйте этим, вы тем самым создаёте рекламу кому-то!); частным случаем (и более важным для вас) является ссылка на файл, расположенный на локальной машине (так часто и разрабатываются WEB-сайты).

Ссылки на WEB-сервер (или сайт) и на адрес электронной почты соответственно оформляются в виде

```
<a href="http://pilger.mgapi.edu ">Ссылка на WEB-сайт</a>
```

```
<a href="mailto:vep@oktava.msk.su">Ссылка на E-Mail</a>
```

Возможность разрешения глобальных гиперссылок еще больше расширяет мощь Интернет, фактически появляется возможность создания *распределенной системы* хранения (и обработки) данных.

Средством обеспечения диалога в Сети служат *формы* - специализированные конструкции языка HTML, позволяющие пользователю вводить данные в отображаемые браузером поля и пересылать их серверу.

Если вы ссылаетесь на локальные файлы, то имена файлов должны быть относительными. То есть, **не нужно указывать C:\<какой-то каталог>\... \<имя файла>**, а **нужно только <имя файла>**. Ещё раз: пусть у вас каталог C:\Personal\my_site, в нём вы создали файл index.html, в этом же каталоге вы создали файл about_me.html, тогда в файле index.html ссылка на файл about_me.html должна быть такой:

Добро пожаловать на [мою](about_me.html) вторую WEB-страницу!

Если же вы укажете в ссылке C:\Personal\my_site\about_me.html, то наверняка web-server вам выдаст: «Not found!».

Средства разработки. Очень хорошее и простое средство - pvu (читается «энваью»), без излишеств и легко осваивается. Можно также пользоваться и другими средствами – что вам проще освоить, решайте сами. В составе AltLinux имеется quanta+ (Quanta plus), документация на русском.

Не рекомендуется пользоваться Flash – далеко не у всех установлены новейшие браузеры, которые реализуют эту возможность, или не у всех загружены соответствующие плагины. Соответственно, они не смогут увидеть ваши гениальные разработки.

Запрещается использовать фреймы! Есть мнение, что в следующих версиях языка HTML они будут исключены.

Кодировка. Не забывайте правильно оформлять тег

```
<meta http-equiv="content-type" content="text/html; charset=windows-1251">
```

или

```
<meta http-equiv="content-type" content="text/html; charset=koi8-r">
```

Он очень важен для правильного отображения текста страницы браузером, поскольку определяет кодировку страницы. Естественно, и страница должна быть в означенной кодировке, иначе будет абракадабра.

Оформление. Не злоупотребляйте графикой: очень многие (до половины всего российского Internet'a) работают пока ещё по модемному каналу со скоростью 1.5 – 20.0 килобайт в секунду и, следовательно, фотка объёмом в 100 кб будет грузиться в десятки секунд, и всё это время потенциальный читатель вашего шедевра будет смотреть в экран и ждать появления всего лишь одной фотки на экране. А если вы их положили на страницу не одну? Вывод: а) изображения должны быть разумного размера; б) можно вставлять на страницу маленького размера изображения (6-10 кб) и делать их ссылками на большие, например, так:

```
<p>
```

```
<a href="pic2_big.jpg">
```

```
</a>
```

```
</p>
```


Кстати, обратите внимание на параметр alt="". . . “ и не забывайте его оформлять правильно. Текст из alt – один из важнейших параметров, анализируемых поисковиками при вычислении рейтинга страницы. Текст из alt должен соответствовать теме/содержанию страницы, и желательно, чтобы в

keywords слова из alt присутствовали.

E-mail. Регистрация на yandex.ru: Заходите браузером на Яндекс. Кнопочка "Регистрация". Заполняете анкету. Самое сложное здесь - подобрать логин: он должен быть как можно короче и быть осмысленным. Вводите пароль. **ЗАПИСЫВАЕТЕ НА ПРОМОКАШКУ!** Иначе забудете! Возможно операцию подбора логина придётся повторить несколько раз. Потому как не вы один жаждете зарегистрироваться на Яндексе. Кто-то раньше вас успел. По завершении регистрации вы станете обладателем электронного адреса <ваш_логин>@yandex.ru. И пишете письма!

Создание сайта. Ищете бесплатный хостинг для сайтов. Как правило такой хостинг предлагает некоторые инструменты для работы с сайтами, типо доступ по ftp, online-редактор и др. Регистрируетесь. «Заливаете» на хостинг свои файлы, смотрите свой шедевр.

Пример главной страницы сайта. На рисунке 15 приведена главная страница сайта, а ниже исходный текст.

	<p>Здесь вы можете ввести некоторый текст, например, про себя, или дать свои координаты, или поместить рекламу кого-либо (если вам за это заплатят!), или поместить рисунки (фотки) или ещё что-либо. А можете эту таблицу вообще убрать, если вам такой дизайн сайта не нравится.</p>
<p>ВАШ ЛОГИН.narod.ru</p>	<p>This is мой сайт: ВАШ_ЛОГИН.narod.ru</p>
<p>Содержание сайта Первый пункт меню - первый подпункт меню Второй пункт меню - первый подпункт 2-го пункта - второй подпункт 2-го пункта Третий пункт меню - первый подпункт 3-го пункта - второй подпункт 3-го пункта - третий подпункт 3-го пункта И так далее</p>	<p>Эта ячейка таблицы содержит основной текст и изображения страницы. Именно из этого текста прежде всего набираются ключевые слова для keywords и именно про этот текст составляется description</p>
<p>Здесь подвал страницы, здесь можно тоже что-то поместить</p>	

[Напишите мне письмо!](#)



Рис. 15. главная страница сайта

А ниже приведён исходный текст файла index.html, который даёт эту страницу.

```

<meta http-equiv="content-type" content="text/html; charset=windows-1251">
<title>Имя страницы &ndash; это то, что отобразится в
заголовке браузера</title>
<meta name="Author" content="Автор &ndash; не забывайте себя порекламирровать">
<meta name="description" content="Краткое описание страницы - аннотация">
<meta name="keywords" content="Ключевые слова &ndash; слова, чаще всего встречающиеся на
странице, их может быть много; теги keywords, description и title проверяются на согласованность
поисковыми машинами, их содержание должно соответствовать содержанию самой страницы, то
есть содержанию блока body">

</head>
<body>
<!-- это комментарий, так он оформляется -->
<div align="center">
<!-- начало первой таблицы, которая типа шапка -->
<table style="width: 771px; height: 117px; background-color: rgb(0, 102, 0);" border="0"
cellpadding="1" cellspacing="1">
<tbody>
<tr bgcolor="#ffffff">
<td align="center" width="13%"> <a href="mozilla.png"> </a></td>
<td style="text-align: justify; line-height: 19px; font-size: 14px;" width="85%"> <b>&nbsp;<span
style="color: rgb(0, 102, 0);">Здесь вы можете ввести
некоторый текст, например, про себя, или дать свои координаты, или
поместить рекламу кого-либо (если вам за это заплатят!), или поместить
рисунки (фотки) или ещё что-либо. А можете эту таблицу вообще убрать,
если вам такой дизайн сайта не нравится.</span></b></td>
</tr>
</tbody>
</table>
<!-- конец первой таблицы -->
</div>
<br style="text-align: justify; line-height: 4px;">

```

```

<div align="center">
<!-- начало второй таблицы -->
<table style="font-size: x-small; font-family: verdana; background-color: rgb(0, 102, 0); width: 773px;
height: 350px;" border="0" cellpadding="0" cellspacing="1">
  <tbody>
    <tr>
<!-- первая строка таблицы -->
      <td style="background-color: rgb(209, 255, 174);">
        <div style="background-color: rgb(233, 255, 233);font-size: 12px;" class="nnrus"
align="center"><small><span style="text-decoration:
underline;">ВАШ_ЛОГИН.narod.ru</span></small></div>
        </td>
        <td class="name_res" style="width: 560px; font-size: 12px; text-align: center; background-color:
rgb(187, 204, 186);">This
is мой сайт: ВАШ_ЛОГИН.narod.ru </td>
      </tr>
<!-- конец первой строки таблицы --> <tr>
<!-- вторая строка таблицы, в кот. меню в левой графе шириной 210px
и основной текст в правой графе -->
      <td class="leftcol" style="width: 210px; vertical-align: top; background-color: rgb(232, 255, 221);">
        <table bgcolor="#ffffff" border="0" cellspacing="2" width="99%">
<!-- в этой маленькой таблице заголовков меню -->
          <tbody>
            <tr class="sprav">
              <td class="sprav"><span style="text-decoration: underline;">Содержание сайта</span></td>
            </tr>
          </tbody>
        </table>
        <b style="font-size: 12px;">Первый пункт меню</b><br>
        &nbsp;<small><a href="ssylka_na_pervii_file.html">-
первый подпункт меню</a></small><br>-----<br>
        <b style="font-size: 12px;">Второй пункт меню</b><br>
        &nbsp;<small><a href="ssylka_na_2_file.htm">
- первый подпункт 2-го пункта</a></small><br>
        &nbsp;<small><a href="ssylka_na_3_file.htm">
- второй подпункт 2-го пункта</a></small><br>-----<br>

```

```

    <b style="font-size: 12px;">Третий пункт меню</b><br>
    &nbsp;<small><a href="ssylka_na_4_file.htm">
- первый подпункт 3-го пункта</a></small><br>
    &nbsp;<small><a href="ssylka_na_5_file.htm">
- второй подпункт 3-го пункта</a></small><br>
    &nbsp;<small><a href="ssylka_na_6_file.htm">
- третий подпункт 3-го пункта</a></small><br>-----<br>
    <b style="font-size: 12px;">И так далее</b><br>
    </td>

    <td class="info" style="width: 565px; vertical-align: top; font-size: 14px; background-color: rgb(250,
250, 255);"><br>
Эта ячейка таблицы содержит основной текст и изображения страницы.
Именно из этого текста прежде всего набираются ключевые слова для keywords
и именно про этот текст составляется description
    </td>
</tr>
<tr>
<!-- это уже пошёл подвал страницы -->
<td style="background-color: rgb(237, 236, 249);font-size: 12px;" class="info" colspan="2">Здесь
подвал страницы, здесь можно тоже что-то
поместить
..... </td>
    </tr>
</tbody>
</table>
<!-- Конец второй таблицы -->
</div>
<center>
<!-- Это линия после таблицы. Она вам нужна? -->
<hr noshade="noshade" size="1" width="775">
<div align="center">
<table border="0" cellpadding="3" cellspacing="0" height="70">
    <tbody>
        <tr>
            <td valign="bottom"> <a href="mailto:vash_login@yandex.ru">Напишите
мне письмо!</a>

```

```
<a href="mailto:vash_login@yandex.ru"></a></td>
</tr>
</tbody>
</table>
</div>
</center>
<!-- Не забудьте заменить ВАШ_ЛОГИН и vash_login на свой реальный логин -->
</body>
</html>
```

Лабораторная работа № 11

Тема: УДАЛЕННАЯ РАБОТА С X-СЕРВЕРОМ

Цель: Научиться удалённо работать с X-сервером

Задание: Настроить локальную вычислительную сеть, удалённый доступ, проанализировать трафик X-протокола.

Указания к выполнению работы:

1. Для выполнения лабораторной работы необходимо правильно настроить сеть. Подключение к удалённому компьютеру осуществлять по hostname или по полному имени.
2. Для выполнения лабораторной работы настроить доступ по ssh: сгенерить ключи, поправить конфигурационные файлы X-сервера (разрешить удалённый доступ).
3. Доступ разрешать только для определённых host'ов — указывать точно.
4. Выполнить действия:
 - подключиться к другому host'у, запустив на нём браузер;
 - браузер должен быть запущен на «другом» компьютере, а его окно должно отображаться на первом компьютере (то есть, «вы смотрите, а за трафик платит сосед»);
 - на обоих компьютерах запустить терминал и в окне терминала показать, что браузер реально выполняется на другом компьютере;
 - запустить ещё один терминал и в нём с помощью программы tcpdump показать трафик X-протокола (обмен между компьютерами);
 - копии экрана сохранить для отчёта.

Порядок сдачи лабораторной работы:

1. По требованию преподавателя повторить работу в лаборатории и объяснить, что, собственно, делал.
2. Продемонстрировать в лаборатории удалённую работу с X сервером.
3. Представить отчёт.

Общие требования к отчёту указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчет должен содержать следующую информацию:

- а) задание на работу;
- б) описание настройки сети;
- в) описание конфигурирования доступа к X-серверу;
- г) описание выполненной работы с копиями экрана.

Дополнительная справочная информация:

*Дополнительную информацию смотреть, например, на сайте
www.ashep.org/category/rabochiy-stol/
или искать в Интернет «запуск удалённых приложений в linux».*

Протокол X-Window

Система X-window была разработана в Массачусетском Технологическом институте (сотрудники этого института внесли существенный вклад и в разработку всего комплекса TCP/IP-протоколов) в качестве многооконного программного интерфейса для ЭВМ с побитовым отображением графической информации. Система предполагала отображение результатов работы нескольких программ одновременно. Сегодня это одна из наиболее популярных систем.

Для каждой программы выделялась отдельная область на экране - "окно". С самого начала система предназначалась для работы в различных сетях (TCP, IPX/SPX и т.д.). Система может управлять окнами как на локальной, так и удаленной ЭВМ. Для управления окнами используются специальные сообщения. Для обмена этими сообщениями разработан x-windows протокол. Система X-window состоит из двух частей Xlib и X-сервера (RFC-1198) (см. рис.16).

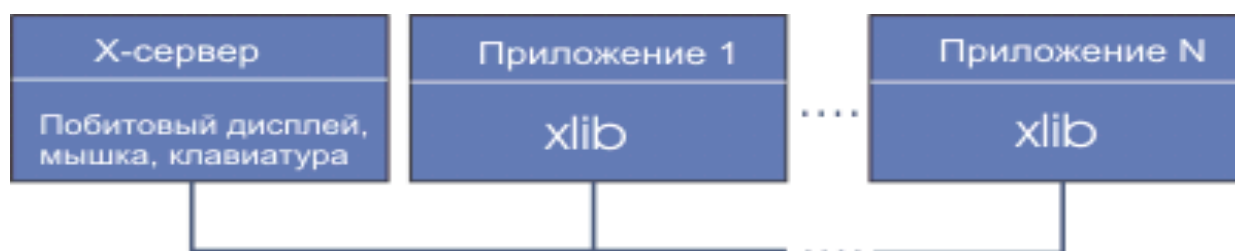


Рис. 16. Схема взаимодействия различных частей X-window

Программа **xlib** является интерфейсом для любого прикладного процесса и обычно представляет собой программу, написанную на С. Xlib отвечает за обмен информацией между сервером и терминалом пользователя (X-клиент). Под приложениями здесь подразумеваются независимые процессы. Для каждого терминала устанавливается отдельный X-сервер. Один X-

сервер может обслуживать несколько клиентов. X-сервер осуществляет отображение на экране всех окон, в то время как функция клиентов - управление окнами. Для управления окнами используются структуры типа стеков.

Прикладная программа-клиент и сервер взаимодействуют друг с другом через системный протокол X-window (RFC-1013 и RFC-1198). При этом используется четыре вида сообщений:

1. **Запрос:** инструкция, направляемая серверу рабочей станции, например, нарисовать окружность.
2. **Отклик:** направляется от сервера в ответ на запрос.
3. **Событие:** используется сервером, чтобы сообщить прикладной программе об изменениях, которые могут повлиять на ее работу (например, нажатие клавиши на терминале или мышке, запрос из сети и т.д.).
4. **Ошибка:** посылается сервером прикладной программе, если что не так (переполнение памяти, неправильно заданные параметры делают выполнение задания невозможным и пр.).

Форматы таких сообщений представлены на рис. 17.



Рис. 17. Форматы сообщений об ошибках

Некоторые X-запросы не нуждаются в откликах (например, связанные с перемещением манипулятора мышью), такие сообщения могут группироваться и посылаться единым потоком (batch stream). Такой подход позволяет пользователю выдать Xlib-запрос и перейти к выполнению других операций, в то время как схема запрос-отклик требует ожидания.

Сообщение типа событие посылается прикладной программе только в случае, если она запрашивала такого рода информацию.

X-Window приложения должны установить канал связи между собой, прежде чем они смогут обмениваться сообщениями. Приложение для того, чтобы установить связь с рабочей станцией использует запрос XOpenDisplay из библиотеки Xlib. Xlib формирует структуру дисплея, которая содержит необходимую информацию, определяющую режим обмена прикладная программа <-> рабочая станция. После того как связь установлена прикладная программа и

рабочая станция готовы к работе. Если была нажата клавиша мышки (событие - ButtonPress), а не известно, в каком из окон находится ее указатель, прикладная программа выдает в Xlib запрос XQueryPointer. Положение указателя будет прислано в отклике на этот запрос.

Лабораторная работа № 12

Тема: УСТАНОВКА И УДАЛЕННАЯ РАБОТА С ПОЧТОВЫМ СЕРВЕРОМ

Цель: Установка почтового сервера фирмы. Удалённая работа telnet'ом с почтовым сервером. Изучить протоколы telnet, smtp, pop.

Задание:

1. Установить почтовый сервер (postfix+pora3d).
2. Установить клиент telnet.
3. Настроить почтовый сервер для обслуживания локальной сети.

Указания к выполнению работы:

1. Настроить локальную сеть виртуальной фирмы: предполагается, что фирма небольшая и деления на подсети/поддомены нет.
2. Для настройки использовать домен firma.ru и сетку 192.168.80.0.
3. Доступ к другим компьютерам данной сети должен осуществляться только по именам компьютеров (hostname и полному имени).
4. На одной из машин сети настроить внутренний почтовый сервер фирмы с именами mail.firma.ru, smtp.firma.ru, pop.firma.ru и адресом 192.168.80.99. Для создания почтового сервера использовать postfix и pora3d.
5. На почтовом сервере создать аккаунты пользователей, настроить их как «почтовых» пользователей. В качестве пользователей (сотрудников фирмы) использовать штатный состав группы.
6. Для выполнения лабораторной работы установить клиент telnet.
7. С рабочего места сотрудника фирмы подключиться к почтовому серверу smtp.firma.ru программой telnet и выполнить следующее:
 - написать письмо себе (на свой почтовый ящик) от имени почтового пользователя postmaster@firma.ru;
 - скрин сохранить для отчёта;
 - написать письмо другу — другому сотруднику фирмы от своего имени;
 - скрин сохранить для отчёта.
8. С рабочего места сотрудника фирмы подключиться к почтовому серверу pop.firma.ru

программой telnet и выполнить следующее:

- открыть свой почтовый ящик и открыть письма, в нём находящиеся;
- скрин сохранить для отчёта.

Порядок сдачи лабораторной работы:

1. По требованию преподавателя повторить работу в лаборатории и объяснить, что, собственно, делал.
2. Продемонстрировать в лаборатории удалённую работу с почтовым сервером.
3. Представить отчет.

Общие требования к отчету указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчет должен содержать следующую информацию:

- а) задание на работу;
- б) описание настройки сети;
- в) описание конфигурирования почтового сервера;
- г) описание выполненной работы с копиями экрана.

Дополнительная справочная информация:

Настройка почтового сервера Postfix

Возможно, вас удивит то, что сервер передачи электронной почты Postfix рекомендуется к установке в любой конфигурации *ALT Linux*. Это объясняется тем, что в Unix-подобных системах способность отправлять почту с помощью простого вызова команды из командной оболочки практически обязательна. Некоторые программы (например, сервис cron) пользуются этим для отправки сообщений пользователям.

Пересылкой всей электронной почты, проходящей через машину, занимается MTA (Mail Transport Agent), в нашем случае это Postfix. Хотя многие почтовые клиенты способны отправлять сообщения на удалённый SMTP-сервер, имеет смысл поручить и эту задачу системному процессу, чтобы достигнуть эффекта “отправил и забыл”.

Существуют и другие популярные MTA (например qmail, exim), но они по разным причинам не вошли в данную версию дистрибутива. Sendmail, ветеран Интернета, проигрывает Postfix по ряду параметров, в том числе безопасности, к тому же он неоправданно сложен в настройке.

В данном руководстве мы ограничимся рекомендациями по настройке Postfix для нескольких типичных конфигураций. Более полные сведения можно получить из превосходной документации на английском языке, которая входит в состав пакета postfix.

Пакеты Postfix

Базовый RPM-пакет для установки сервера Postfix в *ALT Linux* носит, как нетрудно догадаться, имя postfix. Есть также несколько дополнительных пакетов, предоставляющих сервисы по приёму и доставке сообщений по сети с различной степенью защищённости. Один из пакетов SMTP-серверов, postfix-smtpd либо postfix-smtpd-sasl, нужен Postfix для того, чтобы принимать сообщения по протоколу SMTP (или ESMTP) как извне, так и локально. Второй из этих пакетов реализует расширения SASL. Есть также пакет postfix-sasl, который расширяет возможности доставки сообщений на случай, если какие-либо принимающие серверы, с которыми взаимодействует данный сервер, пользуются авторизацией по методу SASL.

Конфигурационные файлы. Файлы настройки Postfix располагаются в каталоге `/etc/postfix`. Основные параметры определяются в файле `main.cf`; в частности, параметры, о которых говорится далее в этой главе, устанавливаются в этом файле, если другой не указан специально.

В изначальном виде этот файл содержит конфигурацию, позволяющую серверу работать в пределах машины, а также развёрнутые комментарии с примерами. После редактирования конфигурации при работающем Postfix её нужно активизировать командой `service postfix reload` или просто `postfix reload`.

Доменная информация. Имя хоста и домена, которые считаются локальными при обработке email-адресов, необходимы для функционирования почтового сервера. Если эти имена для Postfix должны быть отличны от того, что выдаёт команда `hostname`, установите их с помощью параметров `myhostname` и `mydomain`.

Postfix на dialup-машине

Существует несколько проблем, возникающих при попытке отправки исходящей почты с машин, которые не являются полноценными узлами интернет, например, в системах с модемным и другими непостоянными соединениями не всегда возможно немедленно отправить сообщения удалённым адресатам по SMTP и их приходится держать в очереди до тех пор, пока соединение не будет установлено. Для этого используется параметр `defer_transports`, например:

```
defer_transports = smtp
```

Доставка активизируется командой `/usr/sbin/sendmail -q`, которая в *ALT Linux* выполняется

автоматически при установке PPP-соединения.

Будучи полноценным МТА, Postfix способен находить серверы, обслуживающие получателей сообщений, при помощи DNS. Тем не менее, для dialup-машин непосредственная доставка сообщений нежелательна, поскольку время соединения ограничено. К тому же это излюбленная тактика распространителей спама, поэтому многие серверы сверяют IP-адрес отправителя с базой известных адресов провайдерских пулов, после чего сообщения с таких адресов отвергаются. Поэтому целесообразно доверить доставку исходящей почты SMTP-серверу провайдера. Этим управляет параметр *relayhost*, например:

```
relayhost = [smtp.provider.net]
```

Postfix на клиентской машине локальной сети

Рабочие станции локальной сети или машины в провайдерской сети, отделённой от Интернета с помощью межсетевого экрана/NAT, должны переправлять исходящую почту на почтовый сервер, обслуживающий данную сеть. Для этого также используется параметр *relayhost*, описанный выше. Если сервер задан IP-адресом, можно отключить использование DNS для ускорения работы:

```
disable_dns_lookups = yes
```

Для того, чтобы в доменной части адреса отправителя фигурировал домен сети, а не имя конкретной машины, установите параметр *myorigin* в имя домена:

```
myorigin = $mydomain
```

Если почтовые ящики пользователей монтируются с сервера по NFS, Postfix на клиентских машинах служит лишь для отправки почты. В такой конфигурации следует отключить агенты *local* и *smtp* в файле `/etc/postfix/master.cf`.

Почтовый сервер для небольших доменов и сетей

Домены, для которых сервер получает почту, отличные от значения *mydomain* и не сконфигурированные как виртуальные домены Postfix, нужно перечислить с помощью параметра *mydestination* либо в дополнительном файле, на который ссылается этот параметр. Аналогичным образом параметр *mynetworks* описывает блоки IP-адресов, которые считаются внутренними и с которых разрешён приём исходящих сообщений. Не следует записывать в *mynetworks* блоки адресов, не принадлежащих сети, которую обслуживает сервер, поскольку этим могут воспользоваться распространители спама.

Для SMTP-аутентификации внешних пользователей, желающих отправлять сообщения через данный сервер, можно использовать поддержку авторизации SASL. Пакет `postfix-smtpd-sasl`

предоставляет альтернативу postfix-smtpd со включенной поддержкой SASL; возможный недостаток этого расширения— включение кода, в меньшей степени проверенного в плане безопасности. Настройка аутентификации SASL описана в файле SASL_README в документации Postfix.

Преобразование глобальных адресов в локальные адреса назначения устанавливается с помощью таблиц типа virtual (см. virtual(5)):

```
virtual_maps = hash:/etc/postfix/virtual
```

Пример содержимого /etc/postfix/virtual:

```
domain1.ru # Домен в стиле Postfix (текст здесь игнорируется)
name1@domain1.ru user1
name2@domain2.ru user2@otherbox
@domain2.ru user3
```

После редактирования оттранслируйте таблицу в рабочий образ командой `postmap /etc/postfix/virtual`.

Если каким-либо пользователям сети почта должна доставляться по SMTP на их рабочие станции (это предполагает, что на их машинах работают МТА), подставляйте в доменной части их адресов имена машин в таблицах virtual либо aliases.

Алиасы и преобразования адресов

Имена локальных адресатов либо совпадают с именами пользователей системы, либо подставляются из таблицы aliases (см. aliases(5)):

```
alias_maps = hash:/etc/postfix/aliases
alias_database = hash:/etc/postfix/aliases
```

При установке Postfix “с нуля” в этой таблице создаётся алиас на имя root для доставки всей корреспонденции, предназначенной администратору и поступающей на другие системные адреса, на имя реального пользователя, который осуществляет функции администратора. Изначально им становится первый зарегистрированный в системе реальный пользователь. Таблица алиасов отличается от остальных таблиц, используемых Postfix; имена слева, которые являются ключами для поиска, отделяются от значений справа двоеточиями. Адресаты справа перечисляются через запятую и могут быть адресами, командами (обозначаются символом | в начале правой части; сообщение подаётся на стандартный поток ввода команды) и именами файлов:

```
John.Smith: john
chief: chief@bosscomputer
trio: stock, hausen, walkman
```


robot: | /usr/bin/robot --process-mail

filebox: /dir/file

Рабочий образ таблицы строится с помощью команд `postalias /etc/postfix/aliases` или `newaliases`.

При отправке сообщения Postfix генерирует адрес отправителя из имени пользователя и собственного домена (или значения *myorigin*). Даже если почтовый клиент выставил заголовок `From:`, этот адрес попадает в служебную информацию сообщения и может быть использован получателем, что не всегда желательно. Преобразование адресов отправителей к глобальным адресам можно задать в таблице типа `canonical` (см. `canonical(5)`):

```
sender_canonical_maps = hash:/etc/postfix/sender_canonical
```

Аналогичная таблица `recipient_canonical` и соответствующий параметр `recipient_canonical_maps` могут быть использована для преобразования адресов назначения. Для актуализации изменений таблиц используйте команду `postmap имя_таблицы`.

Борьба со спамом и почтовыми вредителями

Противодействие спаму (массовым рассылкам непрошенной корреспонденции) — отдельная большая тема, которую невозможно полностью раскрыть в этом руководстве; здесь даны лишь несколько практических советов применительно к конфигурации Postfix. По умолчанию сервер сконфигурирован так, что отвергает попытки переслать сообщения извне на другие удалённые серверы. Со спамом, адресованным локальным получателям, дело обстоит сложнее. Хорошо зарекомендовали себя служба *MAPS RBL* и ей подобные, организованные по принципу “чёрного списка” IP-адресов; чтобы задействовать эти сервисы, предварительно ознакомившись с условиями их использования, занесите имена доменов, работающих по принципу RBL, в конфигурацию:

```
smtpd_client_restrictions = permit_mynetworks, reject_maps_rbl
```

```
maps_rbl_domains = relays.ordb.org, blackholes.mail-abuse.org
```

В некоторых случаях требуется адресная работа с отдельными нарушителями почтового этикета. Адресная работа заключается в блокировании SMTP-соединений с их адресов, сетей либо доменов. Для этого предусмотрены таблицы типа `access` (см. `access(5)`):

```
smtpd_client_restrictions = permit_mynetworks,  
                             hash:/etc/postfix/access
```

Пример таблицы:

1.2.3.4 550 No more canned meat, please

1.2.5 REJECT

goodguy.generallybad.com ОК

.generallybad.com REJECT

Как и с другими таблицами, после редактирования приведите файлы конфигурации в действие командой

```
postmap /etc/postfix/access.
```

Прочие настройки

По умолчанию размер файла почтового ящика при локальной доставке ограничен 51200000 байтами. Это ограничение можно изменить с помощью параметра *mailbox_size_limit*. Установка параметра в 0 снимает ограничение.

Использование Postfix

После того, как Postfix настроен и запущен как сервис с предсказуемым именем postfix, в настройках почтовых клиентов можно указывать имя или адрес машины (например, localhost) как SMTP-сервер. Программа fetchmail работает в связке с Postfix, опрашивая внешние почтовые ящики пользователей по протоколам POP3 или IMAP и передавая полученные сообщения системному MTA для локальной доставки. Лог-файлы Postfix находятся в каталоге /var/log/mail.

Простой pop3-сервер pop3d

Не всегда нужна установка и настройка больших pop3/imap серверов. Иногда почта настраивается для 3-5 человек, которые меняются раз в год, а то и реже. Поэтому городить систему с авторизацией из внешних источников (отличных от системной авторизации) нет никакого смысла.

pop3d – простой и надежный почтовый сервер. Он работает только по протоколу pop3, отдавать почту умеет только из mboxов и поддерживает только pam авторизацию.

В принципе, наличие pam авторизации уже предусматривает возможность подключения внешних источников авторизации, но с pop3d это делать нецелесообразно.

Установка проводится, как обычно, с помощью synaptic.

у pop3d есть небольшой диалог настройки опций сборки:

```
Options for pop3d 1.0.2_1
```

```
[ ] SMTP_AFTER_POP3  Enable SMTP-after-POP mode
```

```
[ ] STANDALONE_POP3  Enable standalone server mode
```

```
[X] SETPROCTITLE    Enable setproctitle mode
```

[OK] Cancel

Дополнительные опции позволяют включить pop3 before smtp авторизацию. Это когда для отправки почты требуется предварительно пройти pop3 авторизацию.

Вторая опция, Standalone pop3 сервер, позволит pop3d запускаться самому, без использования inetd. Это теоретически должно повысить производительность, но по факту, большого смысла не имеет. Дело в mbox, этот формат хранения почты, сам по себе очень медленный.

И последняя опция setproctitle. Ее включение приведет к тому, что в выводе команды ps можно увидеть, чем в данный момент занимается процесс pop3d. Полезно при поиске причин сбоев.

Настройка проста. В /etc/inetd.conf нужно внести такую строчку:

```
pop3 stream tcp nowait root /usr/local/libexec/pop3d pop3d
```

И нужно заставить inetd перечитать свой конфигурационный файл, для этого можно послать ему сигнал HUP:

```
killall -HUP inetd
```

На этом настройка окончена. По умолчанию sendmail принимает почту для пользователей присутствующих в системе, а pop3d для системных пользователей отдает почту. Добавляйте пользователей с помощью команды adduser и они будут получать почту. Простейшая почтовая система готова к работе.

Лабораторная работа № 13

Тема: ТРАНСЛЯЦИЯ С ВЕБ-КАМЕРЫ НА САЙТ В ИНТЕРНЕТ

Цель: Трансляция с веб-камеры в на свой сайт в Интернет в условиях динамического IP-адреса (фото и потоком).

Задание:

1. Создать на бесплатном хостинге свой сайт. Хостинг выбирать с возможностью установки на страницу видео-плеера.
2. Создать контент для сайта — рекомендуется использовать результаты своих предыдущих лабораторных работ.
3. Сконфигурировать передачу видеотрафика видеоплееру на сайте.

Указания к выполнению работы:

1. Требуется создать техническое решение, то есть:
 - найти бесплатный hosting, на котором можно реализовать нужную функцию (hosting должен позволять установить на странице сайта плеер);
 - оформить нужным образом страницу сайта;
 - настроить домашний компьютер на работу с web-камерой;
 - сконфигурировать ПО на передачу видеотрафика с домашнего компьютера в плеер на страничке своего сайта, а плеер, соответственно, на приём видеотрафика с домашнего компьютера.
2. Рекомендуется показать «жизнь за окном».

Порядок сдачи лабораторной работы:

1. По требованию преподавателя повторить работу в лаборатории и объяснить, что, собственно, делал.
2. Продемонстрировать в лаборатории видеотрансляцию, путем подключения браузером к своему сайту.
3. Представить отчёт.

Общие требования к отчёту указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчет должен содержать следующую информацию:

- а) задание на работу;
- б) описание использованного ПО;
- в) описание конфигурации — конфигурационные файлы с копиями экрана.

Дополнительная справочная информация:

Подробности смотреть на форуме altlinux.ru.

Настройка трансляции видео с веб-камеры

При помощи веб-камеры вы можете транслировать видео в прямом эфире для многочисленной аудитории со всего мира.

Метод 1: Трансляция посредством сервера потокового видео

Шаг 1. В интернете найдите сервер потокового видео, функционал и условия работы которого соответствуют вашим потребностям. Воспользовавшись таким сервером, вы упростите себе задачу, так как вам не придется беспокоиться о написании кода или иных технических деталях. В интернете множество платных и бесплатных серверов потокового видео, при помощи которых вы просто настроите трансляцию с вашей веб-камеры. Для поиска сервера в поисковике введите «сервер потокового видео». Подумайте, какие возможности вам нужны. Большая аудитория? Возможность трансляции видео высокого разрешения? Совместимость с мобильными устройствами? Отсутствие рекламы? В этом случае остановите ваш выбор на платном сервере. Если же вам нужно настроить трансляцию на ограниченную аудиторию и вас не заботит наличие рекламы, воспользуйтесь услугами бесплатного сервера.

Для поиска бесплатного сервера в поисковике введите «бесплатный сервер потокового видео». Но будьте готовы к тому, что бесплатных серверов намного меньше, чем платных.

Перед окончательным выбором протестируйте работу нескольких бесплатных серверов. Также многие платные серверы предоставляют возможность бесплатного тестирования их работы.

Популярными серверами являются: Ustream (платный, но можно бесплатно протестировать его работу), Dacast (платный), Livestream (бесплатный плюс бесплатное тестирование), Vambuser (бесплатный).

Шаг 2. Выбрав сервер, создайте учетную запись. Скорее всего, кнопка для создания аккаунта будет расположена в правом верхнем углу сайта. Создайте премиум-аккаунт, если вам нужна более

широкая аудитория, высокое качество потокового видео и отсутствие рекламы. Однако, премиум-аккаунт довольно дорогой – ежемесячная плата может составлять более \$100 (6500 руб.). Если вам не нужны перечисленные особенности или у вас нет средств на оплату премиум-аккаунта, создайте бесплатный аккаунт.

Шаг 3. Войдите в вашу учетную запись и настройте трансляцию потокового видео. Для этого нажмите кнопку «Прямой эфир» или «Трансляция». Так вы сможете транслировать видео непосредственно с веб-камеры без необходимости устанавливать какое-либо программное обеспечение.

Шаг 4. Откройте серверу доступ к вашей веб-камере. Сделайте это только один раз; для этого в открывшемся сообщении отметьте опцию «Запомнить мой выбор». Возможно, вам придется обновить Adobe Flash. Если сервер получил доступ к веб-камере, она автоматически включится. Если этого не произошло, поменяйте веб-камеру (возможно, она сломана) или обновите драйверы веб-камеры. Для обеспечения наилучшего качества видео обновите прошивку веб-камеры или купите новую камеру.

Шаг 5. Запустите трансляцию потокового видео. На этот счет у каждого сервера свои инструкции и инструменты, но они не сильно отличаются друг от друга. Просто нажмите соответствующую кнопку для запуска трансляции; для ее остановки еще раз нажмите на эту кнопку. Запустив трансляцию потокового видео, вам выделят канал, на который будет настраиваться ваша аудитория. Ссылку на ваш канал выложите в социальных сетях или в блоге. Также вы можете присвоить имя вашему каналу и изменить его настройки. Потоковое видео можно разместить на вашем веб-сайте. У потокового видео есть определенный код, который можно вставить в код вашего сайта. Свяжитесь с разработчиком вашего сайта, если у вас нет доступа к его коду. Как правило, вставить код потокового видео в код сайта или блога довольно просто.

Шаг 6. Для улучшения качества потокового видео с выбранного вами сервера скачайте и установите соответствующее программное обеспечение. На некоторых сайтах оно есть, а на других нет. Также вы можете воспользоваться бесплатными сторонними программами, например, Open Broadcaster Software. Такое программное обеспечение позволит вам снизить зависимость от сервера потокового видео и Wi-Fi сети.

Метод 2: Трансляция посредством YouTube

Шаг 1. Откройте настройки YouTube. Перед этим войдите в ваш аккаунт Google+.

Шаг 2. У опции «Прямые трансляции» нажмите «Включить». Чтобы сделать это, ваша учетная запись должна быть в полном порядке. Прочитайте условия пользования сервисом и нажмите «Я согласен».

Шаг 3. Нажмите «Создать прямую трансляцию». Теперь вы можете присвоить имя потоковому

видео и добавить описание и теги. Задайте время начала потоковой трансляции или запустите ее прямо сейчас. В выпадающем меню выберите требуемые настройки конфиденциальности. Если вы выберете «Публичный», то ваше видео сможет посмотреть любой пользователь. Если вы выберете «Приватный», то ваше видео посмотрят только определенные пользователи.

Шаг 4. Отметьте опцию «Быстрый», а не «Пользовательский», чтобы активировать функцию «Google Hangouts On Air». Так плагин Hangouts получит доступ к вашей веб-камере. Выберите опцию «Пользовательский», если вы умеете программировать, так как в этом случае необходимо использовать собственный программный код.

Шаг 5. Нажмите «Запустить прямую трансляцию». Веб-камера автоматически включится и запустится Google+ Hangouts; вам будет предложено установить плагин Google+, если у вас его нет. Также для Google+ необходимо открыть доступ к вашей веб-камере. Откроется окно Hangouts, но вам придется дождаться буферизации видео; затем вы можете запустить потоковую трансляцию. На буферизацию уйдет около минуты.

Шаг 6. Нажмите «Запустить трансляцию», чтобы начать трансляцию потокового видео с веб-камеры. Для подтверждения вашего решения нажмите «ОК». Вы можете транслировать потоковое видео в течение 8 часов. Нажмите «Диспетчерская», чтобы управлять вашей аудиторией, например, отключить невоспитанного пользователя.

Шаг 7. Поделитесь и вставьте потоковое видео. Для этого нажмите «Ссылки» в нижней части окна Hangouts; отобразится ссылка, которой можно поделиться, и код, который можно вставить. Потоковое видео будет автоматически транслироваться на вашем канале YouTube.

Метод 3: Кодирование потокового видео

Шаг 1. Скачайте и установите программу-кодировщик. Кодировщик преобразует трансляцию с вашей веб-камеры в потоковое видео более высокого качества (по сравнению с программами, предлагаемыми на серверах потокового видео). Программа-кодировщик имеет расширенный функционал, при помощи которого вы получите лучшее изображение и звук; также вы сможете лучше контролировать потоковую трансляцию. Вот список отличных программ-кодировщиков:

- Open Broadcaster Software (OBS) – это бесплатный, простой в использовании и легко настраиваемый кодировщик, хотя и не такой мощный, как некоторые другие программы. При помощи этого кодировщика вы можете транслировать потоковое видео на Twitch, YouTube, Hitbox. Этот кодировщик является одним из лучших бесплатных кодировщиков.

- Flash Media Live Encoder (FMLE) – это бесплатный кодировщик с функционалом, аналогичным OBS. Но это довольно ресурсоемкая программа, поэтому устанавливайте ее только в том случае, если владеете мощным компьютером.

- Wirecast – это профессиональный кодировщик, также используемый при производстве видеороликов. Эта программа имеет значительно больший функционал по сравнению с OBS и

FMLE, но она весьма дорогая.

- Windows Media Encoder (WME) – этот кодировщик предназначен для установки на компьютеры под управлением Windows, но у многих пользователей компьютеры управляются другими операционными системами. Если на вашем компьютере установлена Windows, остановите ваш выбор именно на WME, так как это мощный и бесплатный кодировщик, при помощи которого можно работать с любимым мультимедийным файлом (то есть даже без наличия веб-камеры).

Для каждого кодера есть определенные инструкции по скачиванию и установке.

Шаг 2. Определитесь с сервером потокового видео. Для правильной работы кодировщика необходимо выбрать соответствующий сервер потокового видео (смотрите первый раздел). Хотя некоторые серверы предлагают скачать собственный кодировщик, лучше работать со сторонними программами. Выполните действия, описанные в первом разделе, чтобы создать учетную запись и настроить канал.

Шаг 3. Определите скорость вашего подключения к интернету. Для этого в поисковике введите «тестирование скорости интернет-соединения». Запустите проверку и запишите скорость загрузки.

- Скорость загрузки должна измеряться в Мбит/с (мегабит в секунду).

Шаг 4. Откройте настройки кодера и оптимизируйте его производительность. Нужно настроить кодер так, чтобы потоковое видео транслировалось с максимальной эффективностью.

Битрейт задайте равным 64% от скорости загрузки вашего интернет-соединения. Согласно исследованиям, эта цифра позволяет соблюсти оптимальный баланс между качеством видео и степенью его сжатия. Битрейт определяет величину информации в каждом видеокadre, поэтому с увеличением битрейта увеличивается качество видео. Если скорость загрузки вашего интернет-соединения равна 5 Мбит/с, битрейт установите равным 3,2 Мбит/с. Вы можете увеличить битрейт до 80%, но в этом случае видеоизображение может искажаться из-за проблем со сжатием.

Величину буферизации установите равной битрейту.

Базовое разрешение видео изображения задайте равным разрешению вашего рабочего стола (монитора). Для определения разрешения монитора откройте настройки или свойства экрана.

Задайте выходное разрешение, основываясь на величине битрейта: 480p для 1-2 Мбит/с, 720p для 2-3 Мбит/с, 1080p для 3-5 Мбит/с, 1080p для 5 Мбит/с и выше.

Величину FPS (число кадров в секунду) задайте равной 60 (если возможно). Убедитесь, что выбранный вами сервер потокового видео поддерживает трансляцию с FPS = 60, так как многие серверы ограничиваются 30 кадрами в секунду.

Протестируйте ваше потоковое видео при различных настройках, чтобы добиться оптимального качества и скорости трансляции.

Шаг 5. Свяжите кодировщик с выбранным вами сервером потокового видео. В большинстве кодировщиков вы найдете длинные списки серверов потокового видео. Если вашего сервера в

списке кодировщика нет, выберите «Указать». Кодировщик предоставит вам код, который вы скопируете и вставите на выбранном сервере потокового видео, чтобы связать кодировщик с потоковым видео.

Процедура связывания немного разнится для каждого кодировщика и сервера. Но, как правило, вам предоставят код или веб-адрес, который нужно скопировать и вставить на выбранном сервере. Если вы столкнулись с проблемами, откройте справочный раздел вашего кодировщика.

Шаг 6. Сначала запустите трансляцию на кодировщике, а затем – на сервере потокового видео. Трансляция только через кодировщик не будет являться трансляцией в прямом эфире. Поэтому необходимо передать трансляцию с кодировщика на сервер потокового видео.

Убедитесь, что трансляция через кодировщик проходит в штатном режиме, и только потом запускайте трансляцию через сервер потокового видео.

Вы можете изменить источник потокового видео в настройках кодировщика. Для того, чтобы транслировать потоковое видео с веб-камеры, выберите опцию «Устройства видеосъемки» (или аналогичную).

Метод 4: Трансляция посредством приложений

Шаг 1. Найдите приложение, при помощи которого можно транслировать потоковое видео с веб-камеры. Это простейший способ трансляции потокового видео, но у него множество недостатков. В большинстве своем приложения имеют весьма ограниченный функционал, они не поддаются тонкой настройке и позволяют транслировать видео поток в худшем качестве (по сравнению с серверами и кодировщиками). Приложения не подойдут для трансляции потокового видео на широкую аудиторию, но они сгодятся для домашнего пользования или простых бизнес-презентаций.

Самыми популярными бесплатными приложениями для трансляции потокового видео являются My Webcam Broadcaster (для Mac OS) и Yawcam (для Windows).

На многих серверах потокового видео вы найдете компьютерные и мобильные приложения, которые можно использовать для трансляции потокового видео.

Шаг 2. Скачайте и установите выбранное вами приложение.

Шаг 3. Приложению откройте доступ к веб-камере. Возможно, вам придется обновить Flash.

Шаг 4. Запустите трансляцию потокового видео. Для этого в окне приложения нажмите «Начать трансляцию»; веб-камера подключится к видеоплееру по назначенному веб-адресу. Вы можете определить этот веб-адрес через браузер или приложение.

Протестируйте потоковое видео при различных настройках приложения, чтобы добиться оптимального качества и разрешения. Помните, что приложение не обеспечит такого качества видеоизображения, как кодировщик или сервер.

Шаг 5. Разместите ссылку на потоковое видео в социальных сетях, чтобы увеличить вашу

аудиторию. Это можно сделать при помощи приложения. Вы можете транслировать потоковое видео в течение нескольких минут.

При помощи приложения вы можете просмотреть список пользователей, которые смотрят ваше видео, и даже отключить невоспитанных пользователей.

У вас есть возможность сделать ваше потоковое видео приватным, чтобы просматривать его самому/самой.

Советы

Перед трансляцией потокового видео в прямом эфире проверьте, что все в порядке. Для этого просмотрите поток в приватном режиме.

Для увеличения вашей аудитории размещайте ссылку на потоковое видео на других сайтах и расскажите о нем друзьям и родным.

Протестируйте потоковое видео при различных настройках, чтобы добиться оптимального качества.

Отрепетируйте то, что вы хотите показать в видеоролике, чтобы избежать оплошностей в прямом эфире.

Лабораторная работа № 14

Тема: УСТАНОВКА DNS ДЛЯ INTRANET ФИРМЫ

Цель: Создание модели корпоративной сети фирмы.

Задание:

В некоторой виртуальной фирме локальная сеть состоит из 10 сеток подразделений (192.168.101.0 – 192.168.110.0) и общей сетки 192.168.99.0, в которой располагаются общие серверы.

Написать конфигурационные файлы внутреннего primary DNS для этой фирмы в предположении:

- в каждом подразделении в настоящий момент имеется по 7 ПЭВМ,
- общих серверов – 4,
- сервер primary DNS располагается в серверной сетке;
- выход в Internet обеспечивается не для всех подразделений.

Написать конфигурационные файлы внутреннего slave DNS для этой фирмы и расположить его в одной из сеток подразделений. Обеспечить взаимодействие серверов DNS.

Порядок сдачи лабораторной работы:

1. По требованию преподавателя повторить работу в лаборатории и объяснить, что, собственно, делал.
2. Продемонстрировать протокол работы.
3. Представить отчет.

Общие требования к отчету указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчёт должен содержать следующую информацию:

- а) задание на работу;
- б) копии окна xterm с выполненной командой ps -ax, показывающей, что сервер DNS запущен;
- в) описание порядка запуска и настройки серверов primary и slave DNS, в том числе порядок изменения конфигурации и перезапуска сервера;
- в) конфигурационные файлы обоих серверов.

Дополнительная справочная информация:

Как решить данную задачу смотреть в документе <http://docs.altlinux.org/archive/p7/school-server/>

Лабораторная работа № 15

Тема: КОРПОРАТИВНАЯ СЕТЬ ФИРМЫ И ЕЕ ЗАЩИТА

Цель: Изучить способы защиты корпоративной сети (Intranet) фирмы.

Задание:

1. Нужно настроить защищённый выход в Интернет из локальной сети виртуальной фирмы (или из домашней локальной сети).
2. Для этого в качестве gw (gateway) на отдельной ПЭВМ установить smoothwall-express-3.0. Настроить: красный интерфейс смотрит в общеуниверситетскую сеть, зелёный — модель корпоративной сети фирмы.
- 3. На gw в smoothwall установить дополнительно proxy и snort.**
4. Построить модель сети в лаборатории.

Внимание! Возможная проблема: поскольку университетская сеть защищена fw+проху, которые «режут» всё, кроме http, а некоторые пакеты требуют прямого доступа в Интернет уже при установке, то . . . предложить возможные решения и реализовать.

Порядок сдачи лабораторной работы:

1. По требованию преподавателя повторить работу в лаборатории и объяснить, что, собственно, делал.
2. Продемонстрировать модель в лаборатории. Модель строится одна на всю группу.
3. Представить отчет.

Общие требования к отчету указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчет должен содержать следующую информацию:

- а) задание на работу;
- б) описание порядка запуска и настройки серверов со скринами;
- в) описание порядка установки proxy и snort на smoothwall;
- г) конфигурационные файлы серверов.

Дополнительная справочная информация:

Как решить данную задачу смотреть в документе <http://docs.altlinux.org/archive/p7/school-server/>

Ниже приведена информация о SmoothWall Express Firewall.

SmoothWall Express Firewall

Межсетевой экран на базе OS Linux SmoothWall Express Firewall является программным решением, позволяющим сделать из старого компьютера полноценный межсетевой экран, по возможностям и производительности не уступающий многим аппаратным версиям. Smoothwall Express распространяется под лицензией GPL и основывается на Linux, однако, в отличие от множества других решений на базе Linux, Smoothwall может похвастаться графическим интерфейсом настройки. Используя браузер, вы можете настраивать даже самые продвинутые функции абсолютно без каких-либо проблем.

Smoothwall нацелен на любых пользователей, от домашних до администраторов сетей, однако сразу отметим, что Express будет особенно привлекателен именно домашним пользователям. Онлайн-контекстная помощь способна даже неопытных пользователей превратить в технических гуру. Smoothwall Express бесплатен, однако Smoothwall предлагает также версию Corporate Server, которая, конечно стоит денег, но предлагает больше функций.

Коротко о главном: Этому файрволу не нужна OS, он сам является самостоятельной OS на базе linux с удаленным управлением через web интерфейс, устанавливается на чистый винчестер.

Функциональность. Smoothwall является программным решением, позволяющим сделать из старого компьютера полноценный межсетевой экран, по возможностям и производительности не уступающий многим аппаратным версиям. Smoothwall Express распространяется под лицензией GPL и основывается на Linux, однако, в отличие от множества других решений на базе Linux, Smoothwall имеет графический интерфейс настройки. Используя браузер, вы можете настраивать даже самые продвинутые функции абсолютно без каких-либо проблем.

Smoothwall нацелен на любых пользователей, от домашних до администраторов сетей, однако сразу отметим, что Express будет особенно привлекателен именно домашним пользователям. Онлайн-контекстная помощь способна даже неопытных пользователей превратить в технических гуру. Smoothwall Express бесплатен, однако Smoothwall предлагает также версию Corporate Server, которая, конечно стоит денег, но предлагает больше функций.

Установка. Вы можете скачать образ в формате iso, который можно использовать для записи CD. После записи вы получите загрузочный CD, откуда и будет выполняться установка - достаточно просто загрузиться с диска. Если на выбранной машине нет оптического привода, то предусмотрена возможность установки по сети.

В целом, установка Smoothwall Express 2.0 логична. Отчасти это связано с тем, что опций для выбора немного. Впрочем, с одной стороны это хорошо, но с другой - плохо, здесь всё зависит

от того, что вам нужно. Во время установки нельзя выбрать раздел. Установка Smoothwall полностью очистит и переформатирует основной жёсткий диск, и, как мы думаем, это вряд ли можно обойти. Конечно, при установке вы увидите несколько предупреждений, но согласитесь, что такой подход не совсем оправдан, особенно если вы не уверены, что Smoothwall вас полностью устроит.

После предупреждений об очистке жёсткого диска вы сможете выбрать сетевые интерфейсы. На этом этапе хорошо бы знать, какие сетевые карты установлены в ваш ПК и какие драйверы Linux они используют. Если у вас этой информации нет, то придётся поэкспериментировать, чтобы выяснить, какая карта к чему подключается. Smoothwall окрашивает интерфейсы в разные цвета - зелёный выбран для локального сегмента, красный - для Интернета и оранжевый - для DMZ (демилитаризованная зона, где обычно располагаются различные серверы). Как только локальная сеть настроена, инсталлятор скопирует файлы на жёсткий диск и продолжит установку. Затем будет произведена настройка остальных сетевых адаптеров, а также ввод базовой информации типа имени узла. Кроме того, существует возможность обновить конфигурацию с предыдущей установки - очень удобно при выполнении установки на новую версию брандмауэра. Как видите, большая часть установки проходит автоматически.

Настройка. Smoothwall может работать с несколькими типами интернет-подключений, например, ISDN, ADSL и даже коммутируемое соединение. Ниже приведено описание использования всех трех зон Smoothwall при установке web- и почтового серверов в оранжевой зоне. После завершения установки вы сможете подключаться к web-интерфейсу администрирования Smoothwall с любого компьютера из зелёной зоны. При подключении вы увидите следующую страницу на рис.18:



Рис. 18. Главная страница web-интерфейса (Control: Home)

Если вы только что установили Smoothwall, то на домашней странице появится сообщение о наличии обновлений. Smoothwall обновляется весьма своеобразно. Сначала придётся скачать обновления на локальный компьютер, а затем, используя web-интерфейс Smoothwall, нужно загрузить файл на Smoothwall, как показано ниже на рис.19.

The screenshot shows the 'Updates' page of the SmoothWall Express 2.0 web interface. The page title is 'Updates' and it includes a sub-header: 'See the latest updates and fixes available for your SmoothWall, and an installation history of updates previously applied.' Below this, there is a table of installed updates with the following data:

ID	Title	Description	Released	Installed
001	fixes1 update	This update contains an updated kernel to version 2.4.24 to correct recently discovered, locally exploitable, vulnerabilities. It also corrects known issues and a problem with dynamic DNS.	2004-01-12	2004-11-25
002	fixes2 update	This update contains an updated kernel to version 2.4.25 to correct recently discovered, locally exploitable vulnerabilities.	2004-02-26	2004-11-25
003	fixes3 update	This update contains an updated kernel to version 2.4.26 to correct recently discovered, locally exploitable vulnerabilities. It also updates Apache and OpenSSL to correct several recently discovered vulnerabilities. In addition, it adds support for the latest SpeedTouch modem (revision 4), it also corrects issues with custom dyndns.org accounts.	2004-05-26	2004-11-25
004	fixes4 update	This update contains an updated kernel to version 2.4.27 to correct recently discovered, locally exploitable vulnerabilities. It also updates mod_ssl to correct a recently discovered vulnerability.	2004-08-10	2004-11-25

Below the table, there is a section for 'Available updates' which states 'All updates installed'. There is also a section for 'Install new update:' with a text input field for the update file, a 'Browse...' button, and an 'upload' button. A 'Refresh update list' button is located below the upload section. At the bottom of the page, there is a footer with the text: 'Produced in association with U.S. Robotics njuju © 2000 - 2003 The SmoothWall Team'.

Рис. 19. Обновление (Maintenance: Updates)

После обновления можно перейти к настройке Smoothwall. Брандмауэр имеет встроенный сервер DHCP (dhcpd), web-прокси (squid) и систему обнаружения атак (snort). Кроме того, поддерживается динамическая служба имён DDNS, которая работает, к примеру, с Dyndns.org, сервер SSH и даже IPsec VPN (FreeSWAN). Каждый сервис легко настраивается через web-интерфейс. Ниже на рис.20 показан пример настройки сервера DHCP:

File Edit View Go Bookmarks Tools Help

http://192.168.1.1:81/cgi-bin/dhcp.cgi

SmoothWall Express 2.0 connection status

control about your smoothie **services** networking vpn logs tools maintenance

web proxy | dhcp | dynamic dns | intrusion detection system | remote access | time

shutdown | help

DHCP

Configure and enable your SmoothWall's DHCP service, to automatically allocate LAN IP addresses to your network clients.

DHCP:

Start address: End address:

Primary DNS: Secondary DNS:

Primary WINS: Secondary WINS:

Default lease time (mins): Max lease time (mins):

Domain name suffix: * Enabled:

* This field may be blank.

Add a new static assignment:

Description: MAC address:

IP address:

Current static assignments:

Description	MAC address	IP address	Mark
<input type="button" value="Remove"/>		<input type="button" value="Edit"/>	

Note:

Produced in association with U.S. Robotics Fujitsu

express 2.0 p2 ui-3.6.1

SmoothWall™ is a trademark of SmoothWall Limited.

© 2000 - 2003 The SmoothWall Team
Credits - Portions © original authors

Done

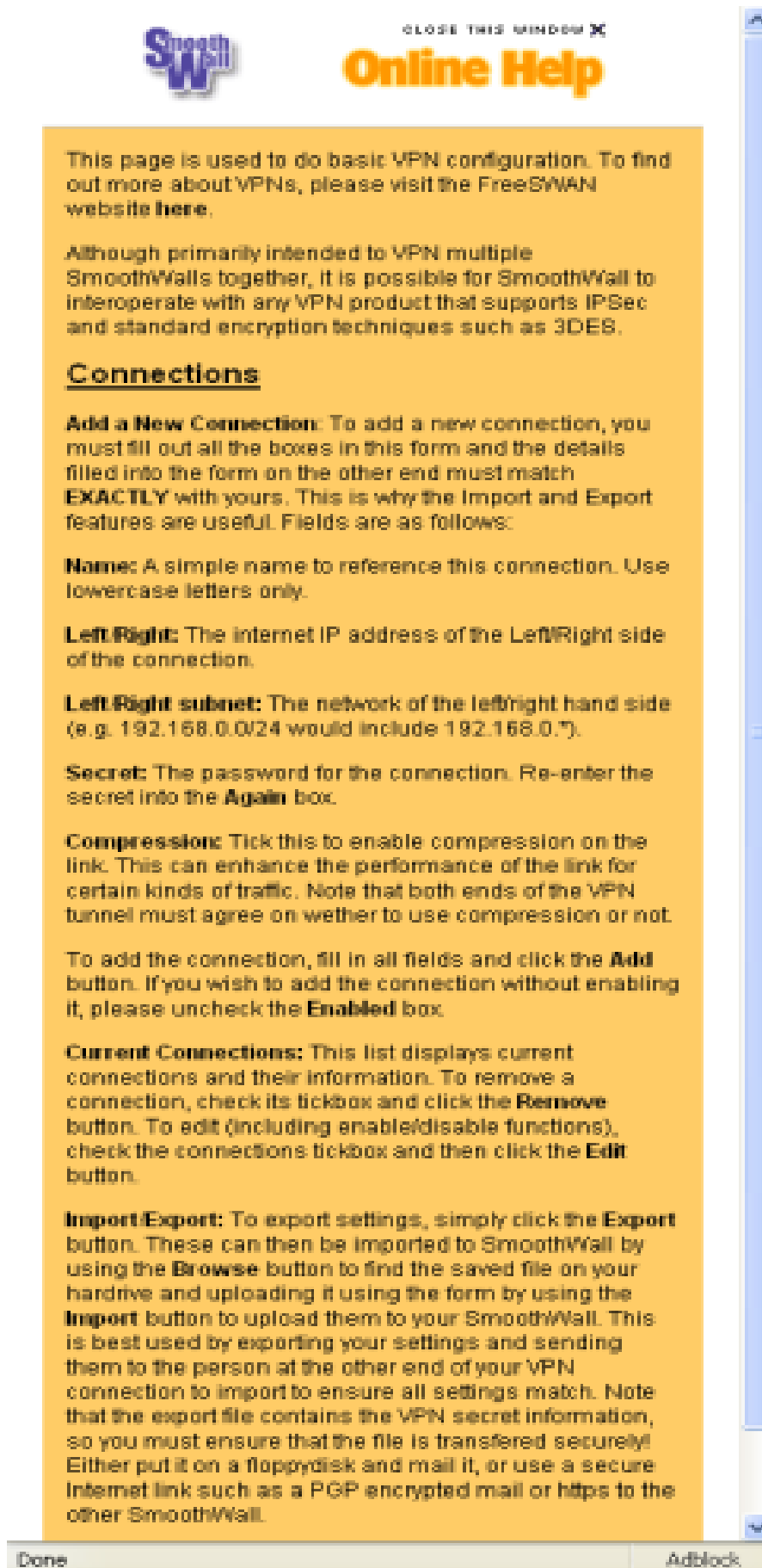
Рис. 20. Сервисы: DHCP (Services: DHCP)

Некоторые службы предлагают настроить достаточно большой набор параметров, а у других, например SSH, число опций ограничено всего одной-двумя галочками (см. рис.21).



Рис. 21. Сервисы: Удалённый доступ (Services: Remote Access).

Все сервисы легко настраиваются при помощи web-интерфейса, который, хотя и не обеспечивает доступ ко всему богатству параметров, достаточен для запуска и работы службы в большинстве конфигураций. Ещё одна великолепная особенность web-интерфейса - это онлайн-система помощи. Прочитав подсказку вы можете легко выполнить настройку даже тех служб, которые раньше были для вас "белым пятном". Помощь выводится в отдельном окне, позволяя одновременно и читать, и настраивать систему (см. рис.22).



SmoothWall CLOSE THIS WINDOW X **Online Help**

This page is used to do basic VPN configuration. To find out more about VPNs, please visit the FreeSWAN website [here](#).

Although primarily intended to VPN multiple SmoothWalls together, it is possible for SmoothWall to interoperate with any VPN product that supports IPsec and standard encryption techniques such as 3DES.

Connections

Add a New Connection: To add a new connection, you must fill out all the boxes in this form and the details filled into the form on the other end must match **EXACTLY** with yours. This is why the Import and Export features are useful. Fields are as follows:

Name: A simple name to reference this connection. Use lowercase letters only.

Left/Right: The internet IP address of the Left/Right side of the connection.

Left/Right subnet: The network of the left/right hand side (e.g. 192.168.0.0/24 would include 192.168.0.*).

Secret: The password for the connection. Re-enter the secret into the **Again** box.

Compression: Tick this to enable compression on the link. This can enhance the performance of the link for certain kinds of traffic. Note that both ends of the VPN tunnel must agree on whether to use compression or not.

To add the connection, fill in all fields and click the **Add** button. If you wish to add the connection without enabling it, please uncheck the **Enabled** box.

Current Connections: This list displays current connections and their information. To remove a connection, check its tickbox and click the **Remove** button. To edit (including enable/disable functions), check the connections tickbox and then click the **Edit** button.

Import/Export: To export settings, simply click the **Export** button. These can then be imported to SmoothWall by using the **Browse** button to find the saved file on your harddrive and uploading it using the form by using the **Import** button to upload them to your SmoothWall. This is best used by exporting your settings and sending them to the person at the other end of your VPN connection to import to ensure all settings match. Note that the export file contains the VPN secret information, so you must ensure that the file is transferred securely! Either put it on a floppydisk and mail it, or use a secure Internet link such as a PGP encrypted mail or [https](#) to the other SmoothWall.

Done Adblock

Рис. 22. Онлайн-помощь.

Как мы считаем, онлайн-система помощи является одним из наиболее серьезных

преимуществ Smoothwall. Многим пользователям часто не хватает терпения, чтоб узнать в Интернете или в руководстве о той или иной функции, однако почти все "кликнут" по нужной кнопке, если возникнет необходимость. Эта возможность, по нашему мнению, даёт Smoothwall огромное преимущество над многими другими web-интерфейсами, которые нам доводилось встречать ранее, особенно для тех пользователей, кто впервые настраивает межсетевой экран.

Smoothwall также имеет основанный на Java web-интерфейс SSH, который показан ниже на рис.23.

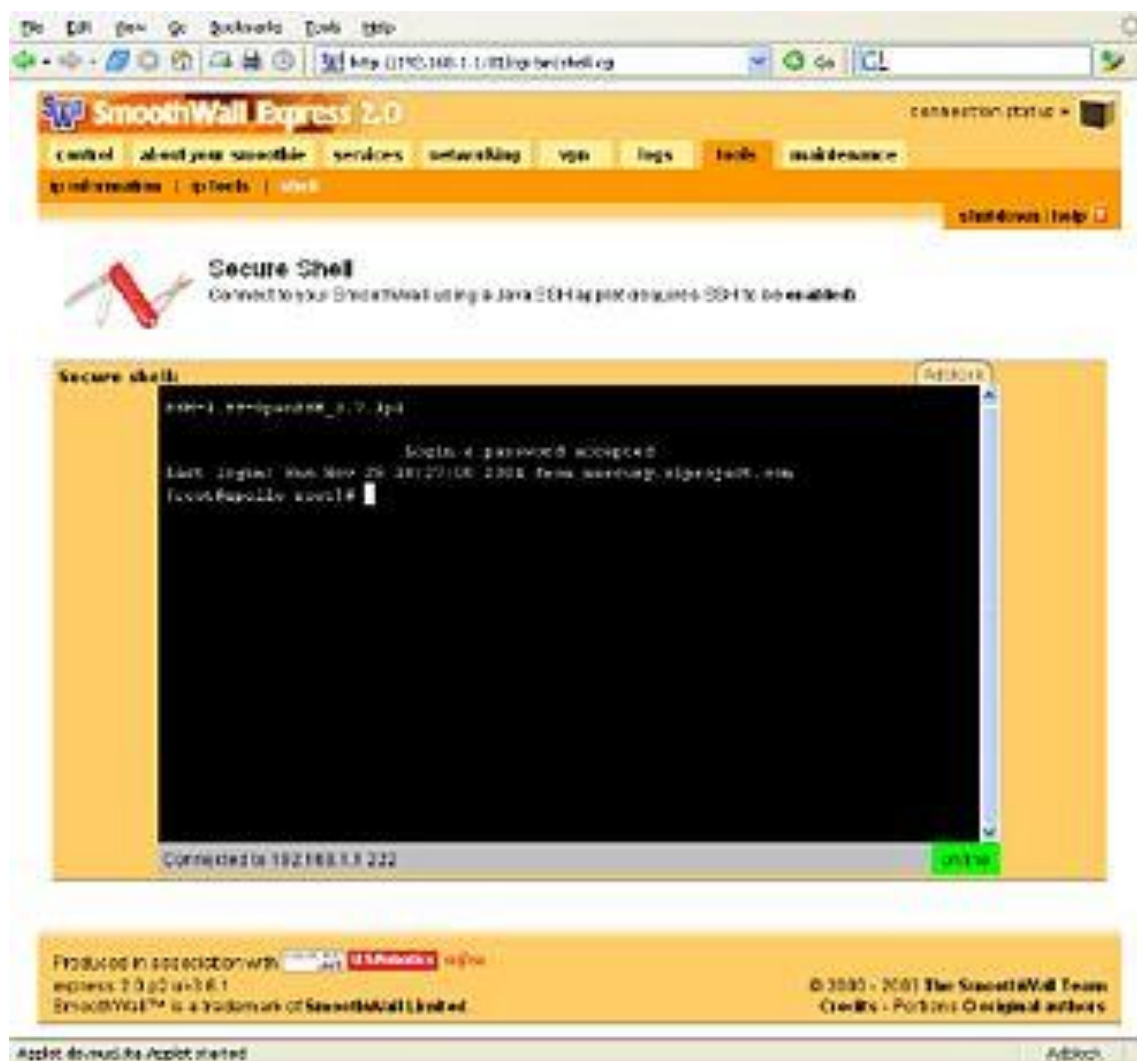


Рис. 23. Оболочка SSH (Tools: Shell).

Используя встроенный SSH, а также и любого другого клиента SSH, вы получите доступ к интерфейсу командной строки, где сможете редактировать файлы конфигурации вручную или производить настройки, не представленные в web-интерфейсе.

По умолчанию Smoothwall разрешает доступ в Интернет для всех систем из зелёной и оранжевой зон, позволяет отправлять запросы из зелёной зоны в оранжевую, но не наоборот. Все

входящие запросы из Интернета (кроме порта 113) отбрасываются. Правила можно поменять, перейдя на закладку "Сеть/Networking" web-интерфейса. Например, поскольку у нас в оранжевой зоне работают почтовый и web-серверы, мы разрешили перенаправлять пакеты по портам 80 и 25, как показано ниже на рис.24.

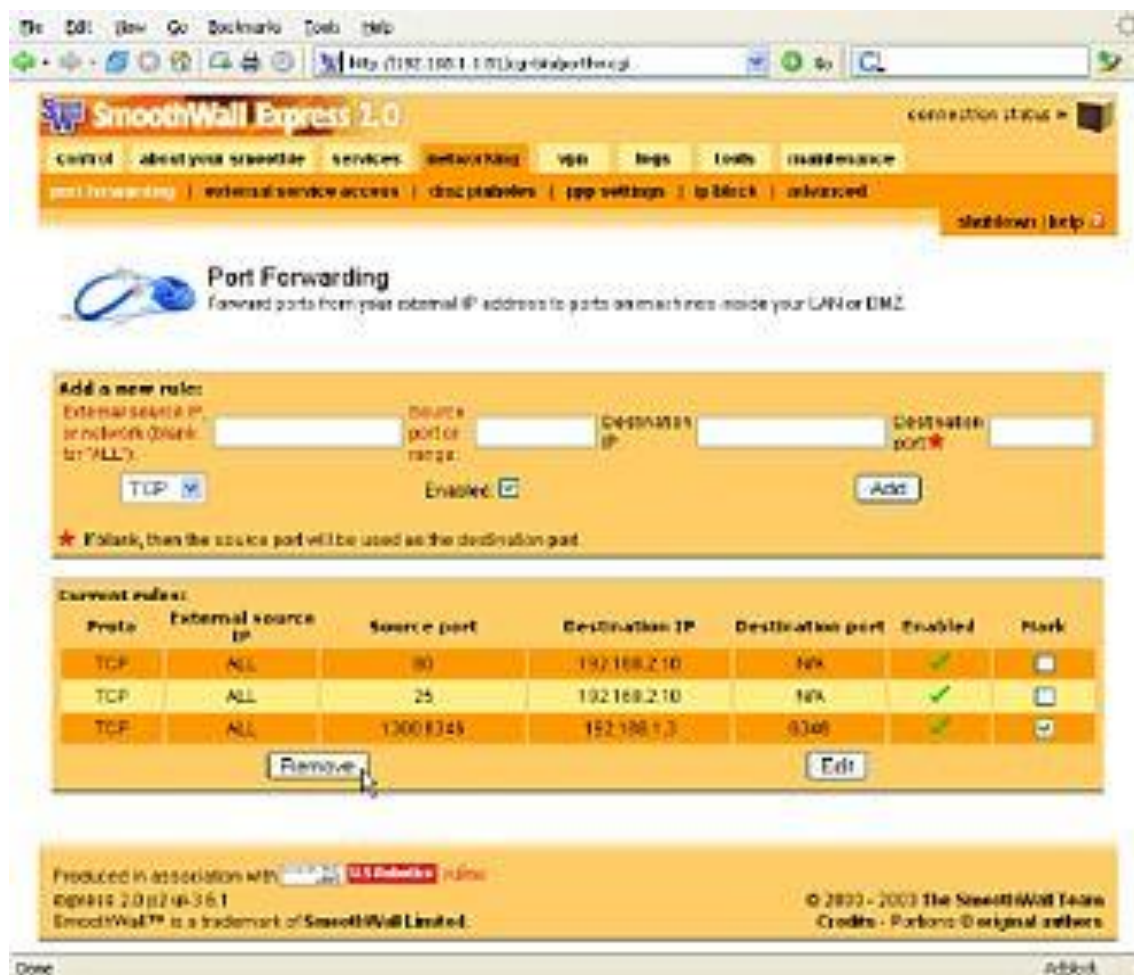


Рис. 24. Перенаправление портов (Networking: Port Forwarding).

Добавление и удаление правил выполняется в один щелчок мыши. Выше показано, как удаляется временное правило, созданное для тестирования производительности Gnutella.

Журналирование. После того, как Smoothwall будет настроен можно периодически проверять состояние системы: просматривать не только страницы состояния, но и информацию о трафике, представленную графически. Ниже на рис.25 показан внешний вид страницы "Traffic Graphs".

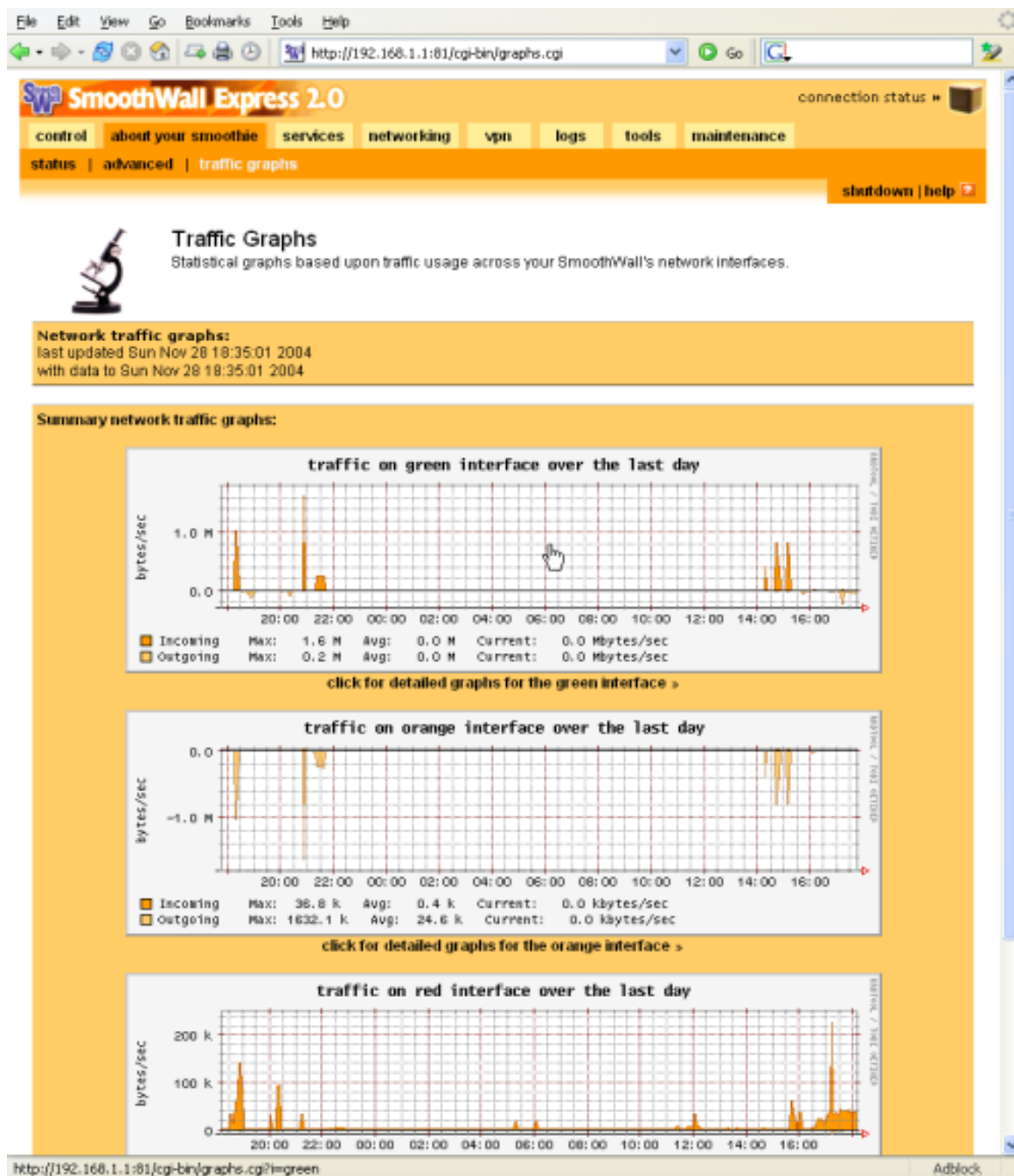


Рис. 25. Отображение трафика (About: Traffic Graphs).

Можно "кликнуть" по графику любого интерфейса и посмотреть детализацию графика за день, неделю, месяц или год. Кроме страниц статуса и графиков, здесь присутствует и страница просмотра журналов. Некоторые журналы отображаются в открытом тексте, но другие, например "Межсетевой экран/Firewall" и "Обнаружение атак/Intrusion Detection System", отформатированы для более удобного просмотра. Страница "Межсетевой экран/Firewall" на рис.26 даже включает галочки и кнопки для просмотра и блокирования IP-адресов.

SmoothWall Express 2.0

control | about your smoothie | services | networking | vpn | logs | tools | maintenance

other | web proxy | firewall | intrusion detection system

Firewall Log Viewer

Check logs for attempted access to your network from outside hosts. Connections listed here **have** been blocked.

Settings:
 Month: Day:

Time	In	Out	Proto	Source	Src Port	Destination	Dst Port
18:01:25	eth1	-	TCP	221.153.34.44	1445	24.163.31.32	2745
18:02:25	eth1	-	UDP	65.19.169.197	1332	24.163.31.32	1434
18:03:43	eth1	-	TCP	218.255.130.22	4443	24.163.31.32	5554
18:03:44	eth1	-	TCP	218.255.130.22	4919	24.163.31.32	9898
18:11:54	eth1	-	UDP	202.83.199.117	1096	24.163.31.32	1434
18:37:18	eth1	-	UDP	24.123.176.142	1090	24.163.31.32	2649
18:37:21	eth1	-	UDP	24.123.176.142	1090	24.163.31.32	2649
18:37:24	eth1	-	UDP	24.123.176.142	1090	24.163.31.32	2649
19:02:15	eth2	eth0	TCP	192.168.2.10	59603	192.168.1.20	113(AUTH)
19:14:49	eth1	-	UDP	209.146.248.82	8348	24.163.31.32	1026
19:14:49	eth1	-	UDP	210.133.147.73	4200	24.163.31.32	1027
19:28:35	eth1	-	TCP	69.158.85.208	4550	24.163.31.32	9898
19:39:27	eth1	-	UDP	149.159.29.49	3053	24.163.31.32	1434
19:54:50	eth1	-	TCP	24.163.133.85	4507	24.163.31.32	2745
19:54:50	eth1	-	TCP	24.163.133.85	4738	24.163.31.32	2082
19:54:50	eth1	-	TCP	24.163.133.85	4910	24.163.31.32	1025
19:54:50	eth1	-	TCP	24.163.133.85	3240	24.163.31.32	3127
19:54:50	eth1	-	TCP	24.163.133.85	3289	24.163.31.32	6129
19:54:53	eth1	-	TCP	24.163.133.85	3289	24.163.31.32	6129
19:54:53	eth1	-	TCP	24.163.133.85	3425	24.163.31.32	1433
19:54:53	eth1	-	TCP	24.163.133.85	4910	24.163.31.32	1025
19:54:53	eth1	-	TCP	24.163.133.85	3240	24.163.31.32	3127
19:54:53	eth1	-	TCP	24.163.133.85	4507	24.163.31.32	2745
19:54:53	eth1	-	TCP	24.163.133.85	4738	24.163.31.32	2082
19:54:58	eth1	-	TCP	24.163.133.85	3425	24.163.31.32	1433

Рис. 26. Журналы: межсетевой экран (Logs: Firewall).

В журнале "Обнаружение атак/Intrusion Detection System" на рис.27, показанном ниже, содержатся все попытки атак на систему.

The screenshot displays the SmoothWall Express 2.0 IDS Log Viewer interface. At the top, there is a navigation menu with options like 'control', 'about your smoothie', 'services', 'networking', 'vpn', 'logs', 'tools', and 'maintenance'. Below this, the 'IDS Log Viewer' section provides instructions: 'Check logs for potentially malicious attempted access to your network from outside hosts. Connections listed here have not necessarily been blocked — use the Firewall Log Viewer to confirm blocked access.'

The 'Settings' section allows filtering logs by month (currently 'November') and day (currently '28'). There are 'Update' and 'Export' buttons. The main 'Log' section contains a table of intrusion detection events:

Date	Priority	Name	Type	IP info	References
11/28 04:43:11	2	ICMP PING NMAP	Attempted Information Leak	24.215.140.222:n/a -> 24.163.31.32:n/a	1
11/28 06:41:40	2	MS-SQL Worm propagation attempt	Misc Attack	202.134.128.132:2286 -> 24.163.31.32:1434	1 2 3
11/28 07:21:11	2	MS-SQL Worm propagation attempt	Misc Attack	221.215.99.180:39446 -> 24.163.31.32:1434	1 2 3
11/28 09:45:01	2	ICMP PING NMAP	Attempted Information Leak	65.75.141.190:n/a -> 24.163.31.32:n/a	1
11/28 12:45:11	n/a	spp_portscan: PORTSCAN DETECTED from 24.115.3.145 (THRESHOLD 4 connections exceeded in 3 seconds)	n/a	n/a:n/a -> n/a:n/a	none found
11/28 12:45:16	2	WEB-MISC webdav search access	access to a potentially vulnerable web application	24.115.3.145:1470 -> 24.163.31.32:80	1
11/28 12:50:00	n/a	spp_portscan: portscan status from 24.115.3.145: 5 connections across 1 hosts: TCP(5), UDP(0)	n/a	n/a:n/a -> n/a:n/a	none found

Рис. 27. Журналирование: обнаружение атак (Logs: IDS).

Следует учитывать, что система обнаружения атак позволяет лишь обнаруживать атаки. Smoothwall их не блокирует, если только они не подпадают под правила запрета межсетевого экрана. Если вы настроили перенаправление портов, убедитесь, что система, на которую перенаправляете трафик (надеемся, она находится в оранжевой зоне), не подвержена уязвимостям, которых сегодня множество.

Smoothwall позволяет быстро и просто изменять пароли, выполнять резервирование, а также запускать мелкие утилиты (whois, ping и traceroute). В целом, система достаточно

функциональна и позволяет удовлетворить потребности множества пользователей.

Можно отметить несколько недостатков. Во-первых, очистка всего жёсткого диска при установке является слишком уж резким решением, особенно с учётом того, что Smoothwall основывается на Redhat, которая всегда предлагала превосходные средства для работы с разделами во время установки. Во-вторых, нас не порадовало отсутствие возможности редактировать через web-интерфейс файл /etc/hosts, хотя, конечно, можно воспользоваться SSH и откорректировать его вручную. В-третьих, Smoothwall использует для синхронизации времени NTP, но не поддерживает эту службу для зелёной и оранжевой зон. И, в заключение, Smoothwall обеспечивает разрешение имён DNS, но только для зелёной зоны. Компьютерам, находящимся в оранжевой зоне, для разрешения имён придётся использовать внешние серверы DNS (например, провайдера). Отметим, что все эти проблемы можно запросто решить, лишь немного поработав с системой. Помните, что внутри работает Linux, поэтому можно изменить всё что угодно, главное только захотеть.

Производительность. Как упоминалось ранее, поскольку Smoothwall основан на Linux, то его производительность достаточно высока, даже на относительно медленных системах. Конечно, некоторые ресурсоёмкие функции, например web-прокси и VPN, могут работать не так быстро, как хотелось бы, упираясь в производительность жёсткого диска и процессора, но обычная передача пакетов будет происходить достаточно быстро даже на древнем "железе". Для тестирования производительности сети можно использовать систему с Smoothwall Express 2.0 на базе ПК с процессором Pentium II 400 МГц с 256 Мбайт ОЗУ и Pentium II 333 МГц с 256 Мбайт памяти (в качестве конечной точки) в оранжевой зоне. С нашего ПК, находящегося в зелёной сети, Pentium 4 2,6 ГГц с 512Мбайт памяти (тестовая система), запущенные несколько тестов через Smoothwall на систему в оранжевой зоне дали следующие результаты.

При использовании конечной точки Windows 2000 Pro и тестовой системы под управлением Windows XP Home, тестовая утилита QCheck, которую использовали для измерения пропускной способности TCP, показала 93,023 Мбит/с при размере файла 1000 килобайт. Для UDP пропускная способность составила 27,778 Мбит/с при том же размере файла 1000 килобайт. При использовании на конечной точке и тестовой системе Mandrake Linux 10.1 Official и утилиты IPerf, для TCP мы получили пропускную способность TCP 93,6 Мбит/с для окна 16 кбайт.

На практике пропускная способность 100Base-TX Ethernet обычно составляет от 60 до 95 процентов теоретических 100 Мбит/с. Таким образом, результаты, определённо, находятся ближе к верхней границе. Это означает, что ограничивающим фактором здесь, по крайней мере, когда речь идёт о "сырой" пропускной способности сети, будет, без сомнения, не программное обеспечение.

Smoothwall Express 2.0 представляет собой прекрасный межсетевой экран, который будет сразу же отлично работать, даже после минимальной настройки. Опций конфигурации с лихвой хватит для большинства пользователей. Благодаря web-интерфейсу настройку может проводить даже начинающий пользователь, а система онлайн-помощи придётся как нельзя кстати.

Продвинутым пользователям, а также для сетей сложной конфигурации рекомендуется проводить настройку через командную строку SSH.

При сравнении с ClarkConnect, тоже межсетевой экран с открытым исходным кодом, наибольшее различие состоит в том, что Smoothwall разрабатывался для работы только в качестве межсетевого экрана, а ClarkConnect может быть как межсетевым экраном, так и файловым сервером, или и тем и другим одновременно. С другой стороны, системы очень похожи и предусматривают администрирование через web-интерфейс.

Считается, что web-интерфейс Smoothwall более продуман и прост в навигации, чем у ClarkConnect. Кроме того, система помощи у Smoothwall показалась более полезной и подробной, чем у Clarkconnect, где представлены только общие описания и параметры. В общем, оба дистрибутива достойны внимания и замечательно выполняют свою работу. Лёгкость настройки не поставит в тупик даже начинающего пользователя.

Лабораторная работа № 16

Тема: ПРОЕКТ СТРУКТУРИРОВАННОЙ КАБЕЛЬНОЙ СИСТЕМЫ ОРГАНИЗАЦИИ

Цель: Научиться проектировать структурированную кабельную систему организации

Задание: Согласно вариантам разработать проект структурированной кабельной системы

Указания к выполнению работы:

1. Составить Пояснительную Записку по проекту (описать назначение вычислительной сети, основные требования, исходные данные своего варианта, привязку к поэтажным планам зданий: на поэтажных планах указать, что где должно стоять (активное оборудование), количество портов по комнатам и их назначение).

Оформление ПЗ выполнить в соответствии с требованиями ГОСТ ЕСКД.

2. Зайти на сайт www.netwizard.ru. Прочсть инструкции: как работать с этим сервисом.

3. Зарегистрироваться, указав свой реальный e-mail. Выйти с сайта. Через некоторое время на указанный почтовый адрес придёт письмо с паролем.

4. Снова зайти на www.netwizard.ru

5. Войти как зарегистрированный пользователь.

6. Создать свой проект. Ввести необходимые данные, описывающие проект — свой вариант.

7. Поскольку проект на сайте сохраняется — его можно уточнять по мере необходимости.

8. Сохранить документацию, созданную системой netwizard по вашему проекту — это основа отчёта по лабе.

9. Составить Техническое Задание на проект вычислительной сети (свой вариант). Оформление ТЗ выполнить в соответствии с требованиями ГОСТ ЕСКД. В ТЗ должно быть описано

- распределение активного и пассивного оборудования с привязкой к поэтажным планам (таблицы и рисунки),

- расположение серверных комнат и этажных коммутационных центров (шкафов),

- оборудование серверных комнат и коммутационных шкафов,

- расположение розеток в комнатах,

- таблица длин кабельных лучей.

10. Приложением к ТЗ должна быть «Смета проекта», содержащая разделы:

«Активное оборудование»,

«Пассивное оборудование»,

«Работы по проектированию и реализации».

11. Разработка документации должна вестись по ГОСТ ЕСКД, в частности, ниже приведена обязательная информация в документах:

а) ПЗ – цель создания, основные функции (требования), общий вид корпуса (фото или аксонометрия), исходные данные своего варианта, привязку к поэтажным планам зданий;

б) ТЗ – введение (цель создания), назначение, основание для разработки, основные функции, исходные данные (распределение компьютеров по комнатам (таблица портов), поэтажные планы с указанием мест расположения розеток и прохождения кабелей/кабель-каналов), требования к проекту, требования к проектной документации;

в) спецификация – перечень комплектующих. Спецификация может быть приложением к ТЗ;

г) смета и ПЗ к ней. ПЗ на одном листе А4. Смета и ПЗ к смете также могут быть приложениями к ТЗ.

Примечание. Пункты в), г) могут быть оформлены как приложения к ТЗ.

Примечание. В стоимость проекта входит:

- стоимость сетевого оборудования (активное, пассивное),
- стоимость оборудования серверной комнаты,
- стоимость работ.

Стоимость ПЭВМ рабочих мест в состав проекта не входят.

Вариант 01. Проект вычислительной сети. Учебный корпус.

Здание 3 этажа., 15x70 м. Факультет 1, кафедр 5. 1 эт. = 20 комн., 2 эт. = 12 комн., 3 эт. = 18 комн.

Всего до 300 компьютеров, то есть, до 300 портов.

Серверный центр на 2 этаже в центре. Выделенные серверы: web, ftp, dns – 2 шт, mail. Сервер Информационной Системы «Факультет». Выход в Internet (gw, fw, squid на одной ЭВМ).

Некоторые кафедры имеют свои серверы, расположенные в помещении кафедры.

Лестничных клеток две, 3 входа в здание. Внутренние несущие стены полые.

1 этаж: фойе, вахта 1 комн., буфет 2 комн., раздевалка. На каждом этаже туалеты.

Вариант 02. Проект вычислительной сети. Учебный корпус.

Здание 3 этажа., 15x70 м. Факультет 1, кафедр 5. 1 эт. = 18 комн., 2 эт. = 16 комн., 3 эт. = 15 комн.

Всего до 250 компьютеров, то есть, до 250 портов.

Серверный центр на 3 этаже в центре. Выделенные серверы: web, ftp, dns – 2 шт, mail. Сервер

Информационной Системы «Факультет». Выход в Internet (gw, fw, squid на одной ЭВМ).

Некоторые кафедры имеют свои серверы, расположенные в помещении кафедры.

Лестничных клеток две, 4 входа в здание. Внутренние несущие стены полые.

1 этаж: фойе, вахта 1 комн., буфет 2 комн., раздевалка. На каждом этаже туалеты.

Вариант 03. Проект вычислительной сети. Учебный корпус.

Здание 3 этажа, 15x70 м. Факультет 1, кафедр 6. 1 эт. = 12 комн., 2 эт. = 18 комн., 3 эт. = 17 комн.

Всего до 220 компьютеров, то есть, до 220 портов.

Серверный центр на 2 этаже в центре. Выделенные серверы: web, ftp, dns – 2 шт, mail. Сервер Информационной Системы «Факультет». Выход в Internet (gw, fw, squid на одной ЭВМ).

Некоторые кафедры имеют свои серверы, расположенные в помещении кафедры.

Лестничных клеток две, 3 входа в здание. Внутренние несущие стены полые.

1 этаж: фойе, вахта 1 комн., буфет 2 комн., раздевалка. На каждом этаже туалеты.

Вариант 04. Проект вычислительной сети. Учебный корпус.

Здание 4 этажа., 15x70 м. Факультет 1, кафедр 6. 1 эт. = 21 комн., 2 эт. = 17 комн., 3 эт. = 18 комн., 4 эт. = 22 комн.

Всего до 280 компьютеров, то есть, до 280 портов.

Серверный центр на 3 этаже в центре. Выделенные серверы: web, ftp, dns – 2 шт, mail. Сервер Информационной Системы «Факультет». Выход в Internet (gw, fw, squid на одной ЭВМ).

Некоторые кафедры имеют свои серверы, расположенные в помещении кафедры.

Лестничных клеток две, 3 входа в здание. Внутренние несущие стены полые.

1 этаж: фойе, вахта 1 комн., буфет 2 комн., раздевалка, комендант здания – 1 комн. На каждом этаже туалеты.

Вариант 05. Проект вычислительной сети. Учебный корпус.

Здание 4 этажа., 15x50 м. Факультет 1, кафедр 4. 1 эт. = 8 комн., 2 эт. = 9 комн., 3 эт. = 12 комн., 4 эт. = 14 комн.

Всего до 200 компьютеров, то есть, до 200 портов.

Серверный центр на 3 этаже в центре. Выделенные серверы: web, ftp, dns – 2 шт, mail. Сервер Информационной Системы «Факультет». Выход в Internet (gw, fw, squid на одной ЭВМ).

Некоторые кафедры имеют свои серверы, расположенные в помещении кафедры.

Лестничных клеток одна, 2 входа в здание. Внутренние несущие стены полые.

1 этаж: фойе, вахта 1 комн., буфет 2 комн., раздевалка. На каждом этаже туалеты.

Вариант 06. Проект вычислительной сети. Учебный корпус.

Здание 5 этажей, 15x50 м. Факультет 1, кафедр 5. 1 эт. = 8 комн., 2 эт. = 10 комн., 3 эт. = 12 комн., 4 эт. = 14 комн., 5 эт. = 10 комн.

Всего до 280 компьютеров, то есть, до 280 портов.

Серверный центр на 3 этаже в центре. Выделенные серверы: web, ftp, dns – 2 шт, mail. Сервер Информационной Системы «Факультет». Выход в Internet (gw, fw, squid на одной ЭВМ).

Некоторые кафедры имеют свои серверы, расположенные в помещении кафедры.

Лестничных клеток одна, 2 входа в здание. Внутренние несущие стены полые.

1 этаж: фойе, вахта 1 комн., буфет 2 комн., раздевалка, комендант здания – 1 комн. На каждом этаже туалеты.

Вариант 07. Проект вычислительной сети. Учебный корпус.

Здание 4 этажа., 17x70 м. Факультет 1, кафедр 6. 1 эт. = 14 комн., 2 эт. = 17 комн., 3 эт. = 22 комн., 4 эт. = 21 комн.

Всего до 250 компьютеров, то есть, до 250 портов.

Серверный центр на 4 этаже в центре. Выделенные серверы: web, ftp, dns – 2 шт, mail. Сервер Информационной Системы «Факультет». Выход в Internet (gw, fw, squid на одной ЭВМ).

Некоторые кафедры имеют свои серверы, расположенные в помещении кафедры.

Лестничных клеток две, 4 входа в здание. Внутренние несущие стены полые.

1 этаж: фойе, вахта 1 комн., буфет 2 комн., раздевалка, комендант здания – 1 комн. На каждом этаже туалеты.

Вариант 08. Проект вычислительной сети. Учебный корпус.

Здание 4 этажа., 17x70 м. Факультет 1, кафедр 6. 1 эт. = 16 комн., 2 эт. = 18 комн., 3 эт. = 23 комн., 4 эт. = 24 комн.

Всего до 220 компьютеров, то есть, до 220 портов.

Серверный центр на 2 этаже в центре. Выделенные серверы: web, ftp, dns – 2 шт, mail. Сервер Информационной Системы «Факультет». Выход в Internet (gw, fw, squid на одной ЭВМ).

Некоторые кафедры имеют свои серверы, расположенные в помещении кафедры.

Лестничных клеток две, 4 входа в здание. Внутренние несущие стены полые.

1 этаж: фойе, вахта 1 комн., буфет 2 комн., раздевалка, комендант здания – 1 комн. На каждом этаже туалеты.

Вариант 09. Проект вычислительной сети. Учебный корпус.

Здание 5 этажей, 17х50 м. Факультет 1, кафедр 6. 1 эт. = 12 комн., 2 эт. = 23 комн., 3 эт. = 19 комн., 4 эт. = 15 комн., 5 эт. = 17 комн.

Всего до 300 компьютеров, то есть, до 300 портов.

Серверный центр на 4 этаже в центре. Выделенные серверы: web, ftp, dns – 2 шт, mail. Сервер Информационной Системы «Факультет». Выход в Internet (gw, fw, squid на одной ЭВМ).

Некоторые кафедры имеют свои серверы, расположенные в помещении кафедры.

Лестничных клеток две, 4 входа в здание. Внутренние несущие стены полые.

1 этаж: фойе, вахта 1 комн., буфет 2 комн., раздевалка, комендант здания – 1 комн. На каждом этаже туалеты.

Вариант 10. Проект вычислительной сети. Административный корпус ВУЗа.

Здание 3 этажа, 17х54 м. 1 эт. = 12 комн., 2 эт. = 18 комн., 3 эт. = 17 комн.

Всего до 200 компьютеров, то есть, до 200 портов.

Серверный центр на 2 этаже в центре. Выделенные серверы: web, ftp, dns – 2 шт, mail. Сервер КИС «Университет». Выход в Internet (gw, fw, squid на одной ЭВМ).

Лестничных клеток три, 3 входа в здание. Внутренние несущие стены полые.

1 этаж: фойе, вахта 1 комн., буфет 2 комн., раздевалка, комендант здания – 1 комн. На каждом этаже туалеты.

Вариант 11. Проект вычислительной сети. Административный корпус ВУЗа.

Здание 3 этажа, 17х54 м. 1 эт. = 10 комн., 2 эт. = 16 комн., 3 эт. = 18 комн.

Всего до 120 компьютеров, то есть, до 120 портов.

Серверный центр на 1 этаже. Выделенные серверы: web, ftp, dns – 2 шт, mail. Сервер КИС «ВУЗ». Выход в Internet (gw, fw, squid на одной ЭВМ).

Лестничных клеток три, 3 входа в здание. Внутренние несущие стены полые.

1 этаж: фойе, вахта 1 комн., буфет 2 комн., раздевалка. На каждом этаже туалеты.

Вариант 12. Проект вычислительной сети. Административный корпус ВУЗа.

Здание 3 этажа, 18х50 м. 1 эт. = 8 комн., 2 эт. = 17 комн., 3 эт. = 15 комн.

Всего до 140 компьютеров, то есть, до 140 портов.

Серверный центр на 3 этаже в центре. Выделенные серверы: web, ftp, dns – 2 шт, mail. Сервер КИС «Университет». Выход в Internet (gw, fw, squid на одной ЭВМ).

Лестничных клеток три, 3 входа в здание. Внутренние несущие стены полые.

1 этаж: фойе, вахта 1 комн., буфет 2 комн., раздевалка. На каждом этаже туалеты.

Вариант 13. Проект вычислительной сети. Городская библиотека.

Здание 3 этажа, 54х80 м. 1 эт. = 12 комн., 2 эт. = 10 комн., 3 эт. = 18 комн.

Всего до 80 компьютеров, то есть, до 80 портов.

Серверный центр на 3 этаже в центре. Серверы: web, ftp, dns – 2, mail. Сервер КИС «Библиотека». Выход в Internet (gw, fw, squid на одной ЭВМ).

Лестничных клеток три, 3 входа в здание. Внутренние несущие стены полые.

1 этаж: фойе, вахта 1 комн., раздевалка. На каждом этаже туалеты.

Вариант 14. Проект вычислительной сети. Городская библиотека.

Здание 3 этажа, 54х80 м. 1 эт. = 11 комн., 2 эт. = 12 комн., 3 эт. = 19 комн.

Всего до 70 компьютеров, то есть, до 70 портов.

Серверный центр на 2 этаже в центре. Выделенные серверы: web, ftp, dns – 2 шт, mail. Сервер КИС «Библиотека». Выход в Internet (gw, fw, squid на одной ЭВМ).

Лестничных клеток три, 3 входа в здание. Внутренние несущие стены полые.

1 этаж: фойе, вахта 1 комн., раздевалка. На каждом этаже туалеты.

Вариант 15. Проект вычислительной сети. ООО «Символ».

Здание 3 этажа, 14х30 м. 1 эт. = 5 комн., 2 эт. = 9 комн., 3 эт. = 15 комн.

Всего до 60 компьютеров, то есть, до 60 портов.

Серверный центр на 3 этаже в центре. Выделенные серверы: web, ftp, dns – 2 шт, mail. Сервер КИС «Галактика». Выход в Internet (gw, fw, squid на одной ЭВМ).

Лестничная клетка одна в центре, 2 входа в здание.

1 этаж: фойе, вахта 1 комн., туалеты.

Вариант 16. Проект вычислительной сети. ООО «СМ-Плюс».

Здание 3 этажа, 15х30 м. 1 эт. = 6 комн., 2 эт. = 10 комн., 3 эт. = 11 комн.

Всего до 50 компьютеров, то есть, до 50 портов.

Серверный центр на 2 этаже сбоку. Выделенные серверы: web, ftp, dns – 2 шт, mail. Сервер КИС «Атлант». Выход в Internet (gw, fw, squid на одной ЭВМ).

Лестничная клетка одна в центре, 2 входа в здание.

1 этаж: фойе, вахта 1 комн. На 1 и 2 этажах туалеты.

Вариант 17. Проект вычислительной сети. ООО «Комп».

Здание 2 этажа, 18х32 м. 1 эт. = 8 комн., 2 эт. = 17 комн.

Всего до 35 компьютеров, то есть, до 35 портов.

Серверный центр на 2 этаже сбоку. Выделенные серверы: web, ftp, dns – 2 шт, mail. Сервер КИС «Компас». Выход в Internet (gw, fw, squid на одной ЭВМ).

Лестничная клетка одна в центре, 2 входа в здание.

1 этаж: фойе, вахта 1 комн. На 1 и 2 этажах туалеты.

Вариант 18. Проект вычислительной сети. ООО «Е-сервис».

Здание 3 этажа, 15х32 м. 1 эт. = 5 комн., 2 эт. = 8 комн., 3 эт. = 10 комн.

Всего до 50 компьютеров, то есть, до 50 портов.

Серверный центр на 2 этаже сбоку. Выделенные серверы: web, ftp, dns – 2 шт, mail. Сервер КИС «Галактика». Выход в Internet (gw, fw, squid на одной ЭВМ).

Лестничная клетка одна в центре, 2 входа в здание.

1 этаж: фойе, вахта 1 комн. На 1 и 2 этажах туалеты.

Вариант 19. Проект вычислительной сети. ЗАО «Линк».

Арендует один этаж в офисном центре. Всего = 12 комн.

Всего до 60 компьютеров, то есть, до 60 портов.

Серверный центр в центре этажа. Выделенные серверы: web, ftp, dns – 2 шт, mail. Сервер КИС

«Компас». Выход в Internet (gw, fw, squid на одной ЭВМ).

Вариант 20. Проект вычислительной сети. ЗАО «ТК».

Арендует 6 комнат в офисном центре. Есть филиал. Филиал включается в проект.

Всего до 30 компьютеров, то есть, до 30 портов.

Серверная: выделенные серверы web, ftp, dns , mail. Сервер КИС «1С-Торговля». Выход в Internet (gw, fw, squid на одной ЭВМ). Сопряжение с филиалом.

Филиал: 3 комнаты в другом здании, 1 этаж (магазин), 3 компьютера, сопряжение с головным офисом.

Вариант 21. Проект вычислительной сети. ЗАО «Маркет».

Два отдельно стоящих двухэтажных здания, расстояние между ними 10 м.

Корпус 1. 12х25 м. 1 эт. = 5 комн., 2 эт. = 5 комн. 1 этаж – фойе, туалеты.

Корпус 2. 16х30 м. 1 эт. = 5 комн., 2 эт. = 8 комн. 1 этаж – фойе, туалеты, вахта.

Всего до 60 компьютеров, то есть, до 60 портов.

Серверный центр находится в 1-ом корпусе на 2-ом этаже. Выделенные серверы: web, ftp, dns , mail. Сервер КИС «1С-Предприятие». Выход в Internet (gw, fw, squid на одной ЭВМ).

Лестничные клетки в середине зданий.

Вариант 22. Проект вычислительной сети. Головной офис торгового дома «Вабо»

Здание 2 этажа, 22х28 м. 1 эт. = 8 комн., 2 эт. = 7 комн.

Всего до 30 компьютеров, то есть, до 30 портов.

Серверный центр на 2 этаже. Выделенные серверы: web, ftp, dns , mail. Сервер КИС «1С-Предприятие» с резервированием. Выход в Internet (gw, fw, squid на одной ЭВМ). Сопряжение с филиалами.

Лестничная клетка одна с торца здания, 1 вход в здание.

1 этаж: фойе, вахта 1 комн., туалеты.

Вариант 23. Проект вычислительной сети. Торговый дом «Алтын»

Здание 3 этажа, 50x120 м. 1 эт. = 10 комн., 2 эт. = 12 комн., 3 эт. = 32 комн.

Филиалы: 12 магазинов по городу, складская база.

Всего до 120 компьютеров и до 20 касс, то есть, до 140 портов в основном здании, 39 компьютеров и 35 касс в филиалах.

Серверный центр на 3 этаже в центре. Выделенные серверы: web, ftp, dns – 2 шт, mail. Сервер КИС «1С-Предприятие» с резервированием. Выход в Internet (gw, fw, squid на одной ЭВМ).

Сопряжение с филиалами.

3 входа в здание: 2 с фасада и один сзади. 1 и 2 этажи – торговые залы и склады. 3 этаж - административный.

Кассы в торговых залах подключены к серверу.

Вариант 24. Проект вычислительной сети. Торговый дом «РСТ»

Два здания далеко друг от друга, аренда канала у ОАО «Электросвязь».

Корпус 1. 18x30 м. 2 этажа. 1 эт. = 3 комн., в т. ч. магазин. 2 эт. = 7 комн.

Корпус 2. 20x35 м. 3 этажа. 1 эт. = 4 комн., в т. ч. магазин. 2 эт. = 5 комн. 3 эт. = 8 комн.

Лестничные клетки в середине зданий.

Всего до 60 компьютеров и до 5 касс, то есть, до 65 портов в двух зданиях.

Серверные: 1 корпус – на 2 этаже, 2 корпус – на 3 этаже. Выделенные серверы: web, ftp, dns – 2 шт, mail. Сервер КИС «1С-Предприятие» с резервированием. Выход в Internet (gw, fw, squid на одной ЭВМ). Сопряжение с филиалами.

Общие требования к отчету указаны в первом разделе пособия.

Дополнительные требования к отчёту:

Отчёт должен содержать следующую информацию:

- а) задание на работу;
- б) разработанные документы и смета проекта.

ИСПОЛЬЗОВАННАЯ И РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

13. Олифер В.Г., Олифер Н.А. Сетевые операционные системы. - Издательский дом «Питер», 2001.
16. Снейдер Й. "Эффективное программирование TCP/IP", Питер, 2001
17. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологии, протоколы: Учебник для ВУЗов. 4-е изд - Спб., Питер, 2010. - 944 с.
1. David Garlan and Mary Shaw, An Introduction to Software Architecture, Advances in Software Engineering and Knowledge Engineering, Volume I, World Scientific Publishing Company, 1993.
2. Боуман И. Концептуальная архитектура Ядра Linux. 1998 г. : Русский перевод: Шевченко Д., 2006 г. - URL: <http://>
3. URL: [//syscalls.kernelkrok.com](http://syscalls.kernelkrok.com) — дата доступа 06.01.14
4. URL: [//j00ru.vexillum.org/win32k_syscalls](http://j00ru.vexillum.org/win32k_syscalls) — дата доступа 06.01.14
5. URL: [//top500.org](http://top500.org) — дата доступа 07.01.14
6. Bach M.J.. The design of the UNIX Operating System.- Prentice-Hall, 1986
7. Tanenbaum A.S.. Modern Operating Systems. - Prentice Hall, 1992
8. Ахо В., Хопкрофт Д., Ульман Д. Структуры данных и алгоритмы. "Вильямс". 2001.
9. Беляков М.И., Рабовер Ю.И., Фридман А.Л. "Мобильная операционная система", М., Радио и связь, 1991.
10. Дейтел Г. Введение в операционные системы. М.: Мир. 1987.
11. Дунаев С. Unix. System V. Release 4.2. М.: Диалог МИФИ. 1996.
12. Керниган Б. В, Пайк Р. UNIX - универсальная среда программирования. М.:Финансы и статистика. 1992.
14. Робачевский Андрей. Операционная система UNIX. - BHV, 1999.
15. Цикритис Д.,Бернстайн Ф.. Операционные системы. М.: Мир. 1977.