

# LINUX. УРОВЕНЬ 1.

ОСНОВЫ АДМИНИСТРИРОВАНИЯ



# Программа

**Модуль 1. Знакомство с Linux и установка операционной системы**

**Модуль 2. Файловая система Linux**

**Модуль 3. Настройка системы**

**Модуль 4. Управление устройствами и подсистемой хранения данных**

**Модуль 5. Процессы**

**Модуль 6. Системы инициализации и управление конфигурацией**

**Модуль 7. Управление пользователями**

**Модуль 8. Управление доступом**

**Модуль 9. Управление программным обеспечением**

**Модуль 10. Резервное копирование и восстановление**

**Модуль 11. Мониторинг производительности**

**Модуль 12. Подключения к графической подсистеме Linux**

# Глава 1

Начальные сведения для работы с системой

# Подготовка стенда

- Свободного пространства в ОЗУ для виртуальной машины - не менее 512 Mb
- Свободное пространство на диске не менее 8 Gb
- Установленный Oracle VirtualBox (<https://www.virtualbox.org>)
- Для 64 разрядной версии дистрибутива поддержка виртуализации CPU (Intel-VT или AMD-V)
- Для 32-разрядной поддержка виртуализации не требуется.
- Установочный образ Debian 10: <https://www.debian.org>

# Процедура установки



## Software selection

At the moment, only the core of the system is installed. To tune the system to your needs, you can choose to install one or more of the following predefined collections of software.

*Choose software to install:*

- ☒ Debian desktop environment
  - ☐ ... GNOME
  - ☐ ... Xfce
  - ☒ ... KDE Plasma
  - ☐ ... Cinnamon
  - ☐ ... MATE
  - ☐ ... LXDE
  - ☐ ... LXQt
- ☐ web server
- ☐ print server
- ☒ SSH server
- ☒ standard system utilities

[Screenshot](#) [Continue](#)

# UNIX

В 1969 году Кен Томпсон создал первую версию новой операционной системы для мини-компьютера DEC PDP-7. Эта версия получила название «первая редакция» (Edition 1) и была первой официальной версией.

Системное время все реализации Unix отсчитывают с 1 января 1970 года.

Первым официальным релизом считается вышедшая в 1971 году версия UNIX для компьютеров DEC PDP-11.

В 1975 вышла пятая редакция, полностью переписанная на Си.



*Ken Thompson (за терминалом телетайп 33) и Dennis Ritchie. 1972 год*

POSIX (Portable Operating System Interface for Unix) - переносимый интерфейс операционных систем Unix.

Набор стандартов, описывающих интерфейс между операционной системой и прикладной программой.

**Основные задачи POSIX:**

- Упрощать перенос кода прикладных программ на различные платформы;
- Способствовать унификации интерфейсов на этапе проектирования, до начала реализации;
- Обеспечивать условия для сохранения и использования созданных ранее прикладных программ;
- Определять необходимый минимум интерфейсов прикладных программ, для ускорения создания, одобрения и утверждения документов;

# Проект GNU

1983 год. Richard Stallman основал проект GNU (логотип проекта антилопа гну), цель которого создание открытой и свободной операционной системы.

GNU - аббревиатура фразы "GNU - это не UNIX" (GNU is Not UNIX).

Разработал лицензию GNU General Public License (GPL) или Открытое лицензионное соглашение GNU



<https://www.gnu.org/>







1991 год. Linus Torvalds опубликовал исходный код ядра (версия 0.01)

1996 год. Выбран пингвин Tux (расшифровывается как Torvalds UniX) в качестве эмблемы

Исходники ядра Linux можно взять на:

<http://kernel.org/> и

<https://github.com/torvalds/linux>



# Знакомство с системой

- Ядро (Linux)
- Оболочка (TUI, GUI)
- Утилиты (набор стандартных программ)
- ПО



# Архитектура операционной системы

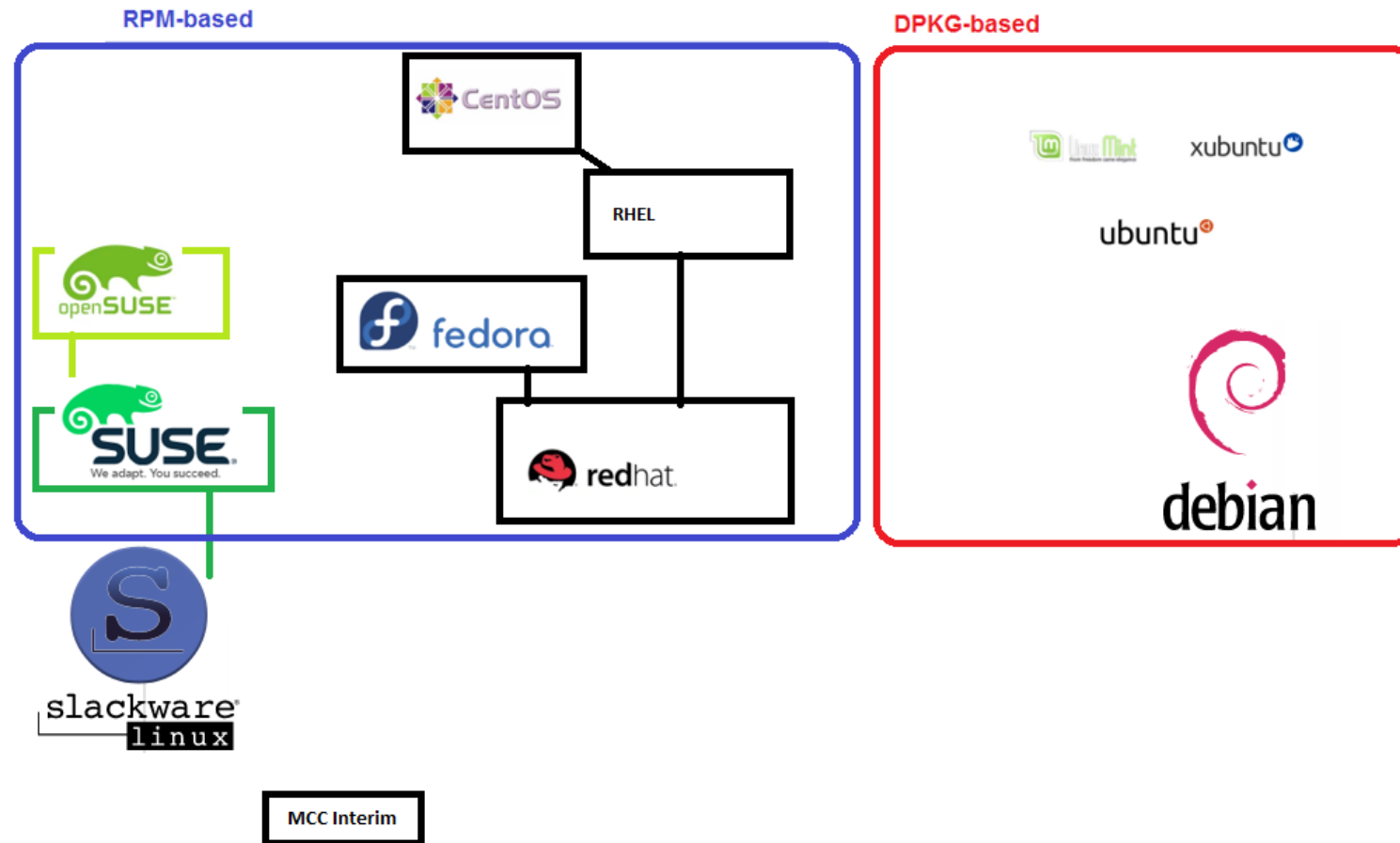


# Обзор дистрибутивов Linux

- Дистрибутив - форма распространения программного обеспечения.
- Содержит программы для начальной инициализации системы (инициализация аппаратной части, загрузка урезанной версии системы), программу-установщик и набор файлов компонентов ОС и дополнительного ПО (пакеты)



# Обзор дистрибутивов Linux



# Основы интерфейса командной строки

# Вход в систему



PDP-7. Рабочее место(терминал) пользователя.



Teletype ASR 33. 1978 год

# Вход в систему

- Консоль и Виртуальные терминалы
- Ctrl + Alt + Fn
- Псевдотерминал



# Командная оболочка (shell)

- В Unix-подобных системах этот вид интерфейса всегда был основным, а поэтому и хорошо развитым
- Стандартом де-факто в Linux является bash (Bourne Again Shell)
- `echo $SHELL` – отобразит используемую оболочку
- Поддерживается автоматическое дополнение длинных названий команд или имён файлов, поиск и повторное выполнение исполнявшейся ранее команды, подстановка имён файлов по шаблону и др

# Командная оболочка (shell)

- Вы должны увидеть символ \$, если вы вошли в систему как обычный пользователь. Это визуальный сигнал о том, что вы входите в систему как обычный пользователь.
- Административный пользователь в linux называется root и символ # в приглашении указывает на то, что Вы вошли в систему как административный пользователь.
- ВНИМАНИЕ: Для Linux как и для Unix характерна регистрозависимость.

# Командная оболочка (shell)

- Командами могут быть как встроенные директивы самой оболочки (`cd`, `pwd`, `echo`), так и вызываемые из под нее утилиты (`ls`, `cat`, `grep`, `vi`)
- Командная строка в `bash` составляется из имени команды, за которым могут следовать ключи (опции) - указания, модифицирующие поведение команды. Ключи начинаются с символа `-` или `--`
- Пример: `ls --all` и `ls -a`
- Далее могут следовать аргументы (параметры) - имена объектов, над которыми должна быть выполнена команда

# Командная оболочка (shell)

- Ctrl-A - переход на начало строки (также, нажатие кнопки Home)
- Ctrl-U - удаление текущей строки;
- Ctrl-C - прервать выполнение команды.
- Символ «;» для ввода нескольких команд в одной строке

# SSH

- Стандартным способом отправки команд является использование удаленной оболочки. Для этого используется протокол ssh
- Ssh означает «Secure Shell» (безопасная оболочка) и является протоколом для предоставления безопасного подключения

# Получение справочной информации

- man
  - Примеры:
    - `man -L ru man`
    - `man file`
    - `man filesystems`
    - `man proc`
- info
  - установка: `apt install info`
- Опция `--help`
  - `ls --help`

# Работа с историей команд (history)

- Ctrl + r — шаблон поиска — поиск по истории
- для просмотра списка ранее введенных команд в bash — имеется команда history. По умолчанию пишется в файл ~/.bash\_history
- Если нам надо повторить команду под номером 28, то набираем в терминале !28
- history -c — очистить историю команд, удалив все записи
- history -d n — удалить из истории запись под номером n

# Работа с историей команд (history)

- Так же можно выводить дату и время для каждой команды в истории, для этого в конец `.bashrc` дописываем:

```
export HISTTIMEFORMAT="%h/%d-%H:%M:%S "
```



# Лабораторная работа 1

Установка системы

Подключение и вход в систему

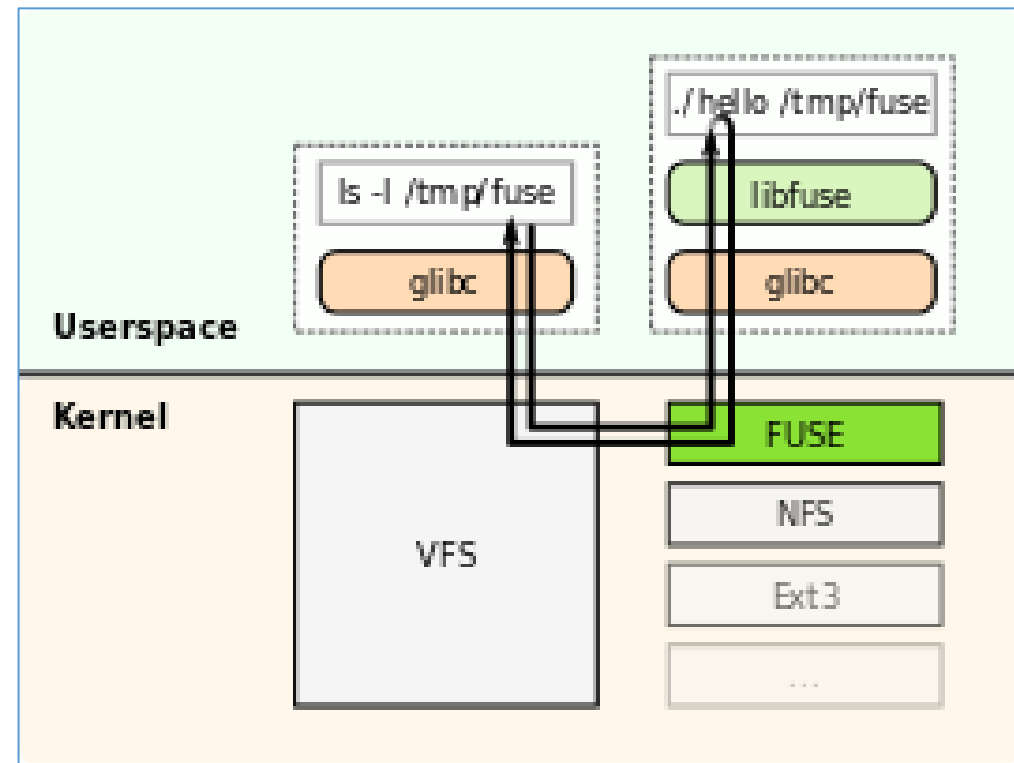
Работа в командной строке

# Глава 2

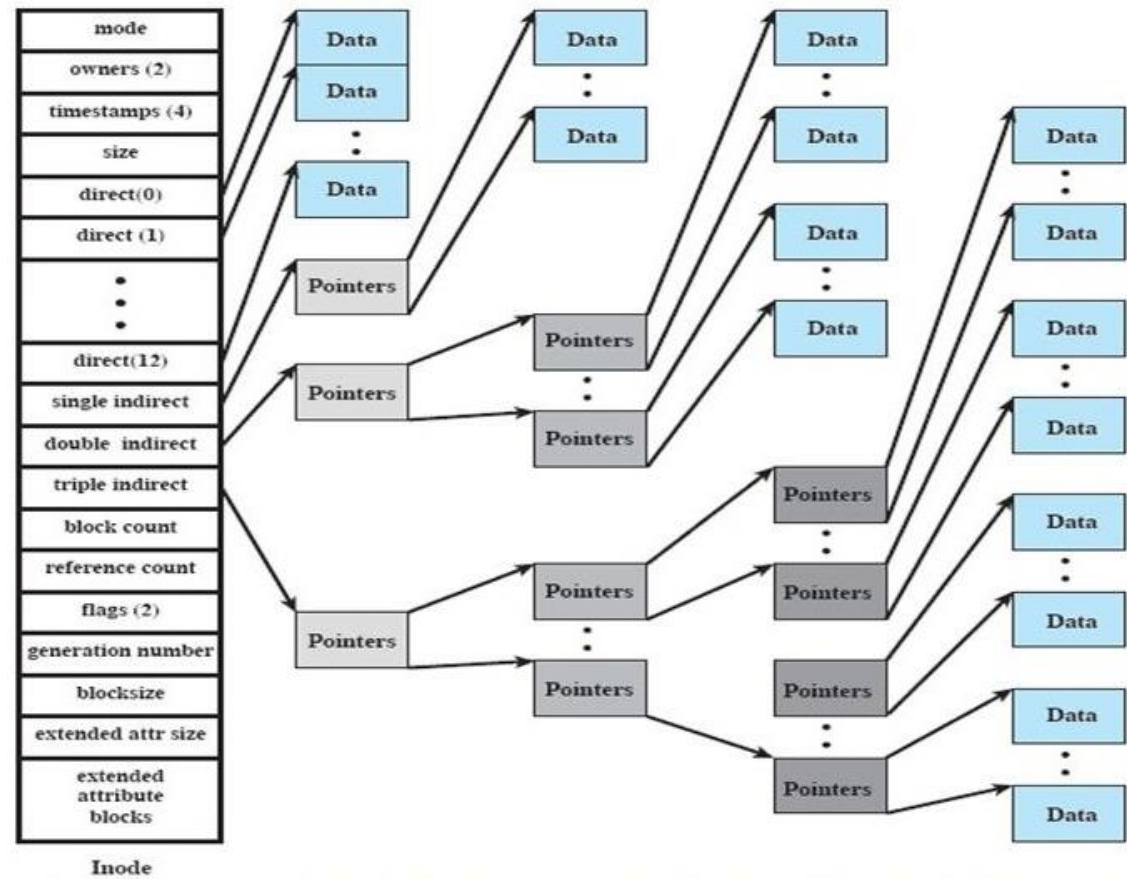
Структура ФС в Linux

# VFS

- Взаимодействие в различными, поддерживаемыми файловыми системами происходит через посредничество VFS.
- Виртуальная файловая система (англ. virtual file system — VFS) или виртуальный коммутатор файловой системы (англ. virtual filesystem switch) — служит интерфейсом для работы с конкретной файловой системой



# inode



# Получение информации о файловых системах

- `df -T` - показывает тип файловой системы
- `du -h` - подсчитывает и выводит размер, занимаемый директорией

# Стандарт FHS

- <https://www.pathname.com/fhs/>

## / — the root directory

— bin	Essential command binaries
— boot	Static files of the boot loader
— dev	Device files
— etc	Host-specific system configuration
— lib	Essential shared libraries and kernel modules
— mnt	Mount point for mounting a filesystem temporarily
— opt	Add-on application software packages
— sbin	Essential system binaries
— tmp	Temporary files
— usr	Secondary hierarchy
— var	Variable data

## /usr — Secondary Hierarchy

— bin	Most user commands
— include	Header files included by C programs
— lib	Libraries
— local	Local hierarchy (empty after main installation)
— sbin	Non-vital system binaries
— share	Architecture-independent data

/usr-это второй основной раздел файловой системы. /usr-это общие данные, доступные только для чтения. Это означает, что /usr может совместно использоваться различными хостами, совместимыми с FHS, и не должен записываться на них.

# Типы файлов

- ls -l

```
user1@debian:~$ ls -l
-rw-r--r-- 1 user1 user1    8 Nov  4 11:08 myhardlink
drwxr-xr-x 3 user1 user1 4096 Nov  4 12:13 myproject
lrwxrwxrwx 1 user1 user1   30 Nov  4 10:24 mysoftlink -> /home/user1/myfolder/testfile1
```

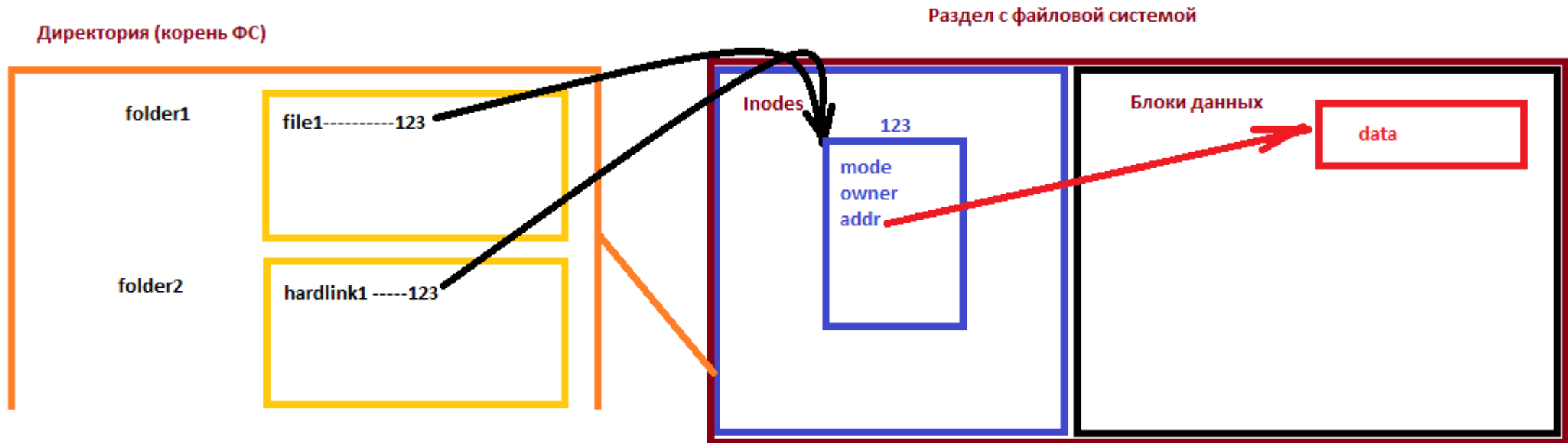
# Утилиты для работы с файлами

- cat
- touch
- cp
- mv
- rm
- ln
- mkdir
- rmdir



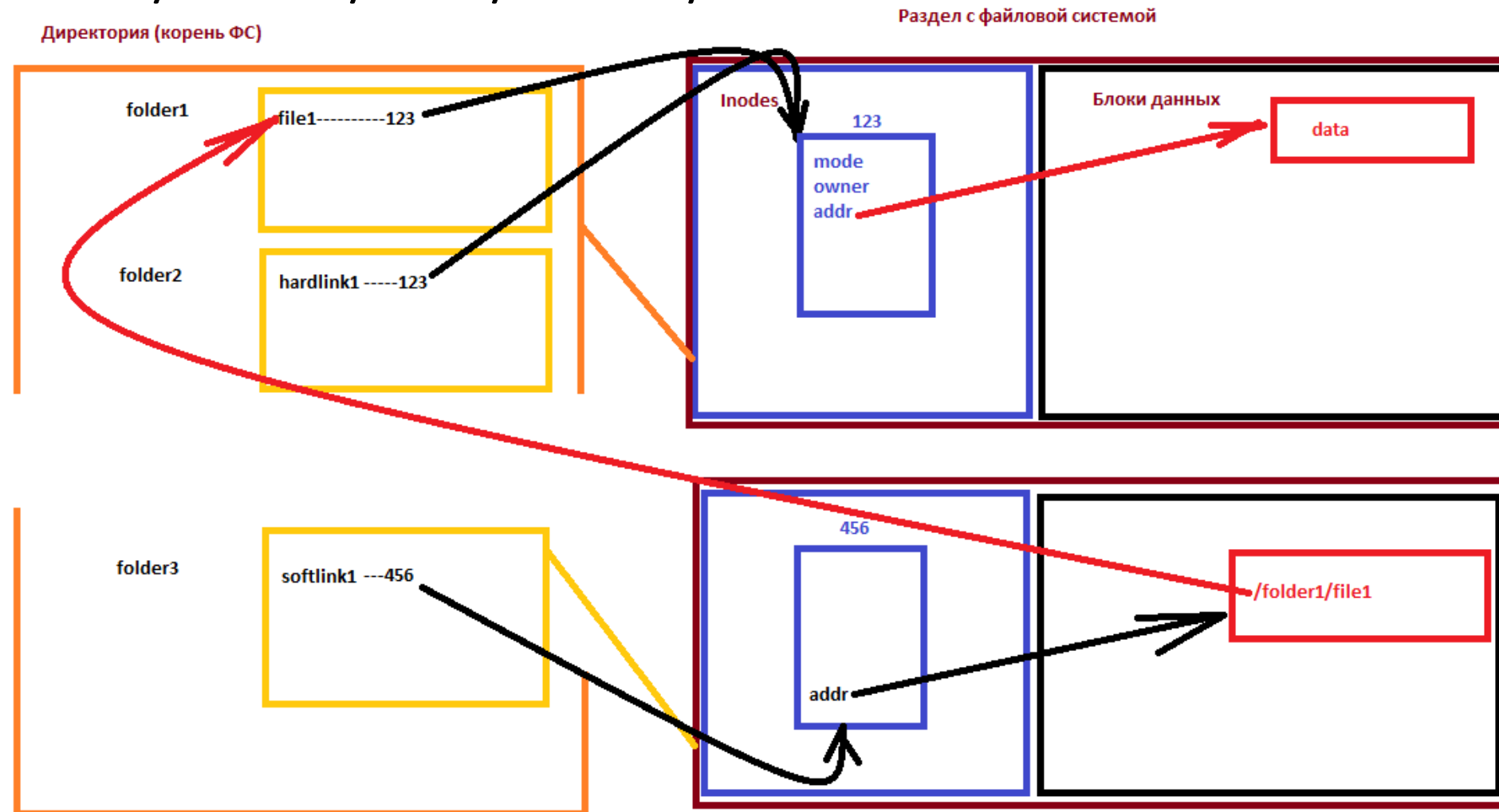
# Утилита ln. Жесткие ссылки

- In /folder1/file1 /folder2/hardlink1



# Утилита ln. Символьные ссылки

- `ln -s /folder1/file1 /folder3/softlink1`



# Поиск файлов

- find [search base] -[критерий поиска]
- locate [pattern]

# Регулярные выражения

- `man 7 regex`, `man re_format`
- Набор символов. Это символы, сохраняющие свои буквальные значения. Простейший пример регулярного выражения состоит только из набора символов и не содержит метасимволов.
- Якорь. Он (якорь) обозначает позицию текста в строке, которая совпадает с регулярным выражением. Например, символы `^` и `$` являются якорями.
- Модификаторы. Они расширяют или сужают (модифицируют) блок текста, совпадающего с регулярным выражением

# Утилита grep

- Grep акроним фразы «search globally for lines matching the regular expression, and print them» — «искать везде строки, соответствующие регулярному выражению, и выводить их».

# Консольные редакторы

- vi, vim
- vimtutor – встроенный учебник по vim
- sed – потоковый редактор
- mc и mcedit
- nano
- Установка: `sudo apt install vim mc -y`

# Лабораторная работа 2

Работа с файлами

Работа с текстовыми редакторами

# Глава 3

Настройка ОС



# Настройка сети

- ifupdown - пакет содержит утилиты ifup и ifdown, которые могут быть использованы для настройки сетевых интерфейсов, согласно параметрам в файле `/etc/network/interfaces`
- Демон NetworkManager пытается сделать настройку и работу сети максимально простой и автоматической, управляя сетевыми интерфейсами, такими как Ethernet, Wi-Fi и мобильные широкополосные устройства
- systemd-networkd - системный демон для управления сетевыми настройками

# Локализация окружения пользователя

- `locale -a`
- `sudo dpkg-reconfigure locales`
- `echo export LANG=ru_RU.utf8 >> .bashrc`

# Настройка представления системного времени

- `timedatectl set-timezone Europe/Moscow`

# Настройка имени системы

- `sudo hostnamectl set-hostname vm1`

- `sudo vi /etc/hosts`

127.0.0.1      localhost vm1

127.0.1.1      vm1

# Управление модулями ядра

# Ядро

- Ядро является сердцем операционной системы Linux и отвечает за планирование запущенных программ, управление файлами и безопасность. Драйверы устройств, реализация сетевого взаимодействия, файловые системы – все это реализуется в ядре. Именно это имеется в виду, когда говорится о пространстве ядра.
- Задача ядра также заключается в поддержке пользовательских программ, которые выполняются в пользовательском пространстве, например, в оболочке, веб-браузере или подобных программах. Программы пользовательского пространства взаимодействуют с ядром через специальные устройства или системные вызовы, которые они делают.

# Управление модулями ядра

- Модули хранятся в каталоге `"/lib/modules/<версия ядра>"`
- `lsmod` вывод всех загруженных модулей в виде таблицы.
- `modinfo` вывод информации о модуле: файл модуля, краткое описание, авторы, лицензия, параметры.
- `insmod` утилита для загрузки модулей ядра. Повторяет функционал `modprobe` название\_модуля.
- `rmmod` простая программа для выгрузки модулей. Повторяет функционал `modprobe -r` название\_модуля.
- `modprobe` утилита для загрузки и выгрузки модулей.

# Управление параметрами ядра

- Команда `sysctl` используется для изменения параметров ядра во время выполнения



# Лабораторная работа 3

Постинсталляционная настройка системы

Управление модулями ядра

# Глава 4

Управление подсистемой хранения данных

# Создание и форматирование разделов

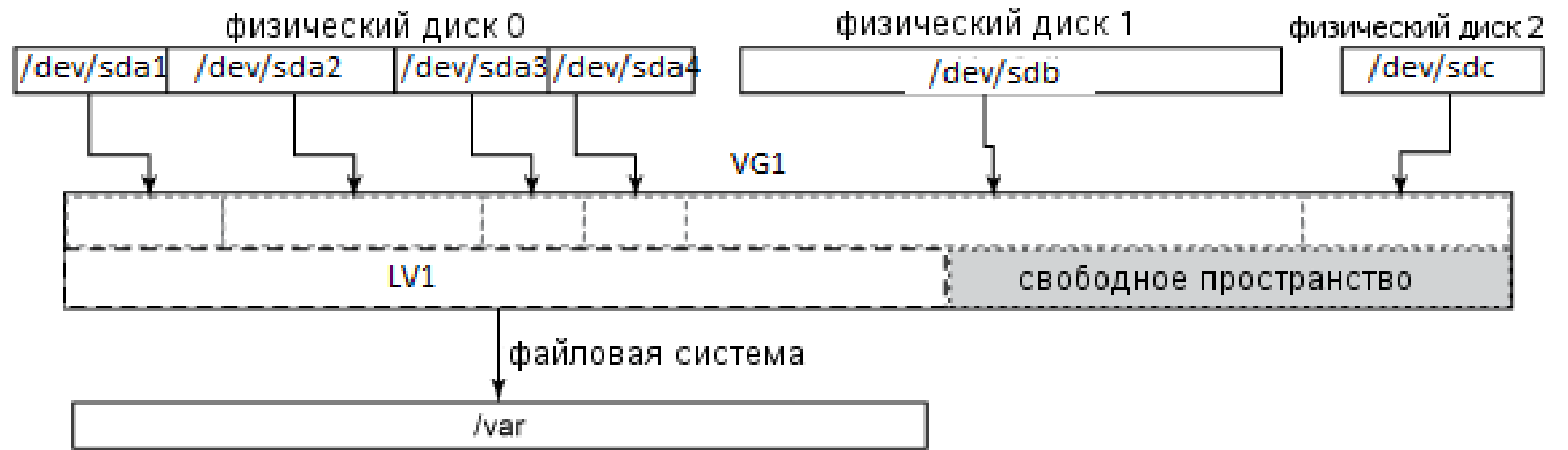
- `fdisk -l /dev/sdx`
- `fdisk /dev/sdx`
- `mkfs.<filesystem_name> [options] /dev/sdxN`
- `file -s /dev/sdxN`
- `blkid`
- `lsblk`

# Монтирование файловых систем

- `mkdir /var/disk1`
- `mount /dev/sdxN /var/disk1`
- `umount /var/disk1`
  
- *`vim /etc/fstab`*
- *`mount -o remount <mountpoint>`*

# LVM2

- pvcreate
- vgcreate
- lvcreate



# Лабораторная работа 4

Создание разделов

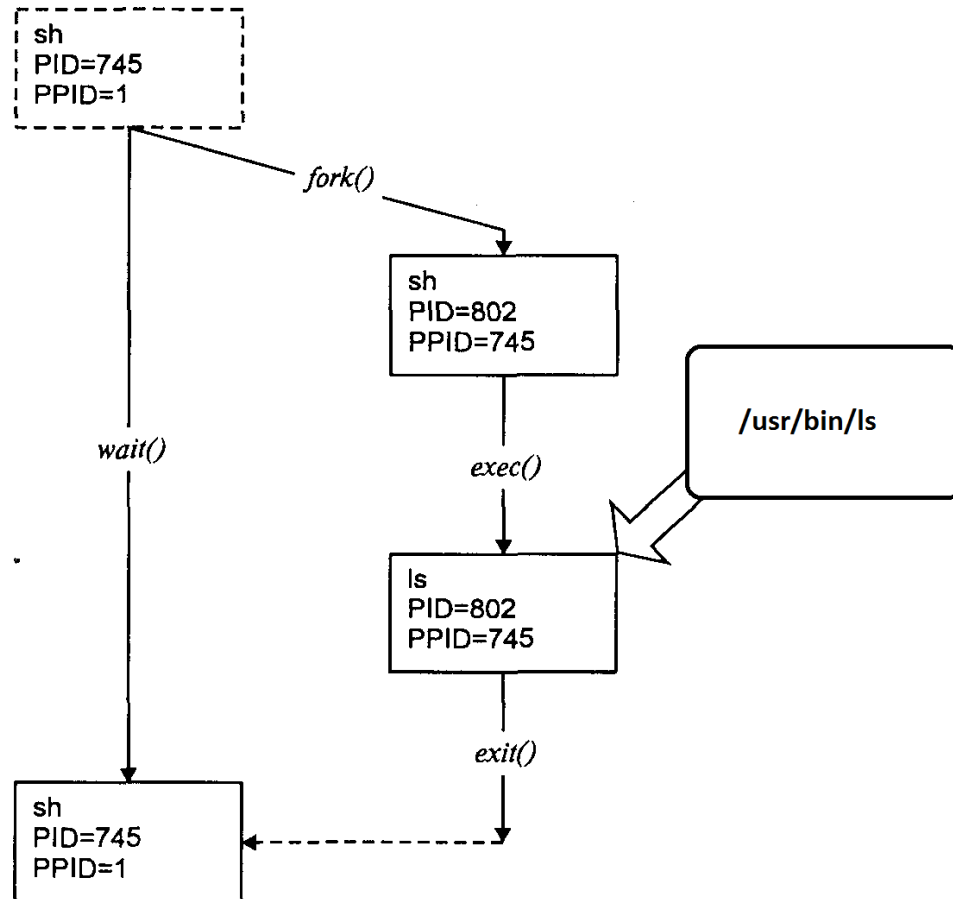
Создание и монтирование файловых систем

Работа с LVM2

# Глава 5

Процессы в Linux

# Процессы.





# Мониторинг процессов

- ps
- top
- *htop*
- *nice*
- *renice*

# Сигналы

- *kill -l*
- *kill -[SIGNAL] [PID]*

# Работа процесса в фоновом режиме

- `sudo tail -fn0 /var/log/syslog`

- `Ctrl + Z`

- `jobs`

```
user1@debian:~$ sudo tail -fn0 /var/log/syslog
```

```
^Z
```

```
[1]+  Остановлен      sudo tail -fn0 /var/log/syslog
```

- `bg` запуск в фоновом режиме

```
user1@debian:~$ bg
```

```
[1]+ sudo tail -fn0 /var/log/syslog &
```

- `fg` переключение на передний план

- `user1@debian:~$ fg`

```
sudo tail -fn0 /var/log/syslog
```

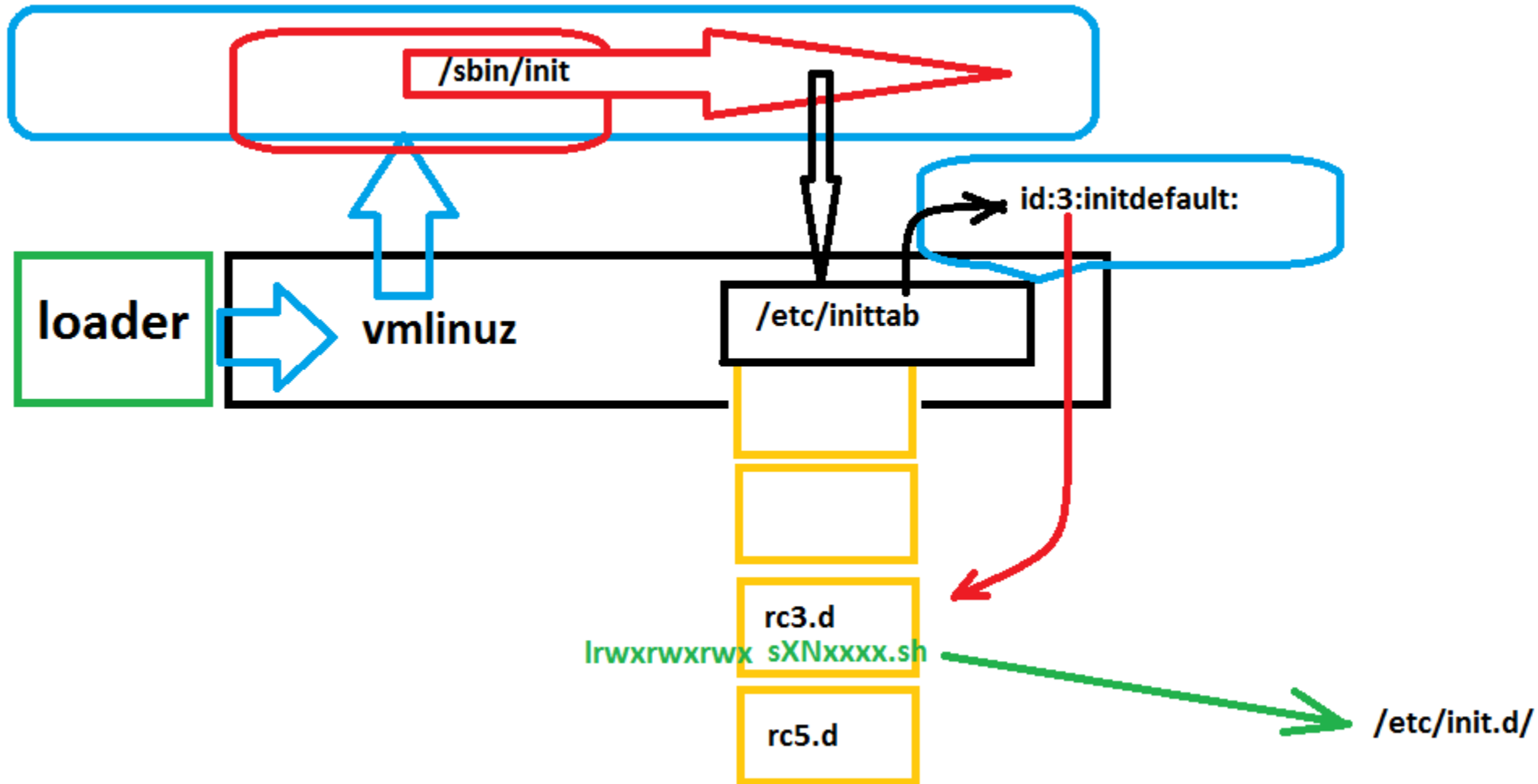
# Лабораторная работа 5

Мониторинг процессов

# Глава 6


Управление конфигурацией

# Системы инициализации




# Systemd

- `systemctl set-default graphical.target`
- `systemctl get-default`
- `ls -l /etc/systemd/system/default.target`

 user1@debian: ~

```
root@debian:~# ls -l /lib/systemd/system | grep target
```

```
lrwxrwxrwx 1 root root 15 anp 27 2020 runlevel0.target -> poweroff.target
lrwxrwxrwx 1 root root 13 anp 27 2020 runlevel1.target -> rescue.target
drwxr-xr-x 2 root root 4096 anp 27 2020 runlevel1.target.wants
lrwxrwxrwx 1 root root 17 anp 27 2020 runlevel2.target -> multi-user.target
drwxr-xr-x 2 root root 4096 anp 27 2020 runlevel2.target.wants
lrwxrwxrwx 1 root root 17 anp 27 2020 runlevel3.target -> multi-user.target
drwxr-xr-x 2 root root 4096 anp 27 2020 runlevel3.target.wants
lrwxrwxrwx 1 root root 17 anp 27 2020 runlevel4.target -> multi-user.target
drwxr-xr-x 2 root root 4096 anp 27 2020 runlevel4.target.wants
lrwxrwxrwx 1 root root 16 anp 27 2020 runlevel5.target -> graphical.target
drwxr-xr-x 2 root root 4096 anp 27 2020 runlevel5.target.wants
lrwxrwxrwx 1 root root 13 anp 27 2020 runlevel6.target -> reboot.target
```

 user1@debian: ~

```
root@debian:~# ls -l /sbin/init
```

```
lrwxrwxrwx 1 root root 20 anp 27 2020 /sbin/init -> /lib/systemd/systemd
```

```
root@debian:~# ls -l /sbin/runlevel
```

```
lrwxrwxrwx 1 root root 14 anp 27 2020 /sbin/runlevel -> /bin/systemctl
```

```
root@debian:~# ls -l /sbin/telinit
```

```
lrwxrwxrwx 1 root root 14 anp 27 2020 /sbin/telinit -> /bin/systemctl
```

# Targets

```
# less /lib/systemd/system/multi-user.target
```

[Unit]

Description=Graphical Interface

Documentation=man:systemd.special(7)

**Requires=multi-user.target**

Wants=display-manager.service

Conflicts=rescue.service rescue.target

After=multi-user.target rescue.service rescue.target display-manager.service

AllowIsolate=yes



# Units

```
# less /lib/systemd/system/ssh.service
```

```
user1@debian: ~  
[Unit]  
Description=OpenBSD Secure Shell server  
Documentation=man:sshd(8) man:sshd_config(5)  
After=network.target auditd.service  
ConditionPathExists=!/etc/ssh/ssh_not_to_be_run  
  
[Service]  
EnvironmentFile=-/etc/default/ssh  
ExecStartPre=/usr/sbin/sshd -t  
ExecStart=/usr/sbin/sshd -D $SSHD_OPTS  
ExecReload=/usr/sbin/sshd -t  
ExecReload=/bin/kill -HUP $MAINPID  
KillMode=process  
Restart=on-failure  
RestartPreventExitStatus=255  
Type=notify  
RuntimeDirectory=sshd  
RuntimeDirectoryMode=0755  
  
[Install]  
WantedBy=multi-user.target  
Alias=sshd.service
```

# systemctl

- `systemctl [stop|start|restart|status] <unitname>`
- `systemctl [enable|disable] <unitname>`
- `systemctl list-units`
- `systemctl get-default`

# Настройка окружения

- /etc/environment
- /etc/bash.bashrc
- /etc/profile
- ~/.bashrc
- ~/.profile

```
user1@debian: ~  
# /etc/profile: system-wide .profile file for the Bourne shell (sh(1))  
# and Bourne compatible shells (bash(1), ksh(1), ash(1), ...).
```

```
if [ "`id -u`" -eq 0 ]; then  
    PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"  
else  
    PATH="/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games"  
fi  
export PATH
```

```
user1@debian: ~  
# ~/.bashrc: executed by bash(1) for non-login shells.  
# You may uncomment the following lines if you want `ls` to be colorized:  
export LS_OPTIONS='--color=auto'  
eval "`dircolors`"  
alias ls='ls $LS_OPTIONS'  
alias ll='ls $LS_OPTIONS -l'  
alias l='ls $LS_OPTIONS -la'
```

# Настройка окружения

- Команда `export` является встроенной директивой оболочки `bash`, позволяющей устанавливать значения переменных окружения.
- Для ознакомления с полным списком переменных окружения и их значений может использоваться команда `env`.

## Синтаксис:

`$ export [параметры] ПЕРЕМЕННАЯ=ЗНАЧЕНИЕ`

## Параметры:

- f использовать функции командного процессора
- n удалить свойство экспорта из всех имён
- p вывести список всех экспортированных переменных и функций

# Сценарии оболочки

- `cat > myscript.sh` создаем файл
- `#!/bin/bash` сигнатура (sha-bang) определяет интерпретатор, который вызывается для исполнения сценария
- `echo "Hello World!"` команды сценария
- `chmod +x myscript.sh` делаем исполняемым
- `./myscript.sh` запускаем

# Лабораторная работа 6

Управление конфигурацией

Использование сценариев оболочки

# Пользователи и группы в Linux

## Глава 7

Управление пользователями



# Управление пользователями и группами

- `useradd backupuser -m -s /bin/bash`
- `passwd backupuser`
- `chage -d 0 backupuser`
- `groupadd -g 1500 group1`
- `gpasswd -M user1,backupuser group1`



# Криптографические ключи и аутентификация

- ssh-keygen
- ssh-copy-id <hostname>

# Модули PAM

- `ls /etc/pam.d/`
- `ls /lib/x86_64-linux-gnu/security`

# Лабораторная работа 7

Управление пользователями

Настройка использования ключей для аутентификации

# Глава 8

Управление доступом

# Управление привилегиями

- `sudo visudo`
- Файл `/etc/sudoers`
- `# cat <<EOF> /etc/sudoers.d/backupuser`  
    `backupuser    ALL= NOPASSWD: /usr/bin/tar`

Проверка:

```
backupuser$ tar -cvf etcbak /etc
```

# Режим доступа

- `umask`
  - `ls -l`
  - `chown user <file | directory>`
  - `chmod [ugo|a] [+|-|=] [rwx] <file | directory>`
- 
- `r`     чтение      $= 1 * 2^2 = 4$
  - `w`     запись      $= 1 * 2^1 = 2$
  - `x`     исполнение  $= 1 * 2^0 = 1$

# Расширенные атрибуты

## lsattr and chattr

- Extended attributes may be set for files
  - i: immutable flag
  - a: append-only flag
  - d: no dump flag
  - A: No atime update flag
- Flags are viewed with **lsattr** command
- Flags are set with **chattr** command
- Flags are stored in the file's inode flags
- May only be set by root

# Использования дополнительных битов

- SUID
- `chmod u+s <file>`
- `chmod 4755 <file>`
- SGID
- `chmod g+s <file>`
- sticky
- `chmod +t <directory>`
- `chmod 1755 <directory>`



# POSIX ACL

- `apt install -y acl`
- `getfacl <filename>`
- `setfacl -m [u|g]:`
- `setfacl -x [u|g]:`
- `setfacl -m u:username:rwX <file>`

# Лабораторная работа 8

Управление доступом и привилегиями

# Глава 9

Управление программным обеспечением

# Системы управления пакетами

Пакетные менеджеры низкого уровня

- dpkg (Debian, Ubuntu)
- rpm (RedHat, CentOS, SUSE)

# Системы управления пакетами

`dpkg -I` #установленные в системе пакеты

`dpkg -L openssh-server` #Содержимое пакета

`dpkg -S /etc/init/ssh.conf` #В какой пакет входит файл

# Системы управления пакетами

Высокоуровневые пакетные менеджеры

Debian, Ubuntu, Astra:

- apt-get, apt-cache
- apt

RedHat, CentOS:

- yum, dnf

SUSE, OpenSUSE:

- zypper

# Системы управления пакетами

apt - объединяет в одной утилите функционал команд: apt-get, apt-cache

Основные команды APT:

- list - список пакетов
- search - поиск пакетов по имени
- show - показать подробную информацию о пакете
- update - обновить списки доступных пакетов
- install - установить пакет
- remove - удалить пакет
- upgrade - установить доступные новые версии пакетов
- full-upgrade - полное обновление системы
- edit-sources - редактировать файл источников программного обеспечения

# Системы управления пакетами

Synaptic — графический интерфейс для системы управления пакетами apt или проекта Debian

Aptitude — интерфейс для Advanced Packaging Tool, части системы управления пакетами в Debian и производных



# Репозитории ПО

## **/etc/apt/sources.list**

- deb <http://deb.debian.org/debian/> buster main
- deb-src <http://deb.debian.org/debian/> buster main
- deb <http://security.debian.org/debian-security> buster/updates main
- deb-src <http://security.debian.org/debian-security> buster/updates main

# Структура репозитория

pool/

- В этом каталоге лежат пакеты в иерархии по веткам (main, restricted).

dists/

- отражает разбиение на релизы, компоненты и архитектуры.
- Содержит файлы, которые извлекаются и кэшируются локально (когда выполняется команда `sudo apt-get update`)

Пример: <http://ru.archive.ubuntu.com/ubuntu/>

# Сборка и установка ПО из исходников

`fakeroot` создаёт фиктивное окружение суперпользователя. Предполагается, что данная программа будет использована с `dpkg-buildpackage -rfakeroot`, чтобы для сборки пакета не требовалось прав суперпользователя

## Дебианизация

создание директории `debian` в корне исходников, с нужными файлами конфигурации и скриптами.

Описание пакета создается в файле `debian/control`

# Создание локального репозитория

```
sudo apt install reprepro  
mkdir -p /var/repo/debian/conf  
mkdir -p /var/repo/debian/pool/main
```

```
sudo nano /var/repo/debian/conf/distributions  
cd /var/repo/debian  
reprepro createsymlinks  
reprepro -b /var/repo/debian --ask-passphrase includedeb buster  
<путь к файлу пакета .deb>
```

# Добавление локального репозитория

```
sudo nano /etc/apt/sources.list
```

```
...
```

```
deb [trusted=yes] file:/mnt/repo/debian buster main
```

```
...
```

```
sudo apt update
```

# Настройка автоматического обновления

`dpkg-reconfigure -plow unattended-upgrades...`

`ls /etc/apt/apt.conf.d`

# Лабораторная работа 9

Управление пакетами ПО

# Глава 10

Управление периодическими заданиями

Резервное копирование

Журналирование событий и аудит



# Утилиты резервного копирования

Tape archiver – tar. Архивирует каталоги.

-C #использовать родительский относительно архивируемого каталог

-c #create (создать)

-t #type (просмотреть архив)

-x #extract (распаковать)

-j или -z #сжимать

-f #(файл архива) - далее указываются каталоги для архивирования.

-v #(verbose)

# Управление периодическими заданиями

Для настройки пользовательских задач по расписанию в linux доступен crontab

- crontab -e
- Для системных – системный /etc/crontab

```
# Example of job definition:
# .----- minute (0 - 59)
# | .----- hour (0 - 23)
# | | .----- day of month (1 - 31)
# | | | .----- month (1 - 12) OR jan,feb,mar,apr ...
# | | | | .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fr
# | | | | |
# * * * * * user-name command to be executed
25 6 * * * root cd / && run-parts --report /etc/cron.hourly
```

# Управление периодическими заданиями

Для настройки пользовательских задач по расписанию в linux доступен crontab

- crontab -e
- Для системных – системный /etc/crontab

# Журналирование событий

- /var/log
  - syslog
  - auth.log
- journalctl

# Лабораторная работа 10

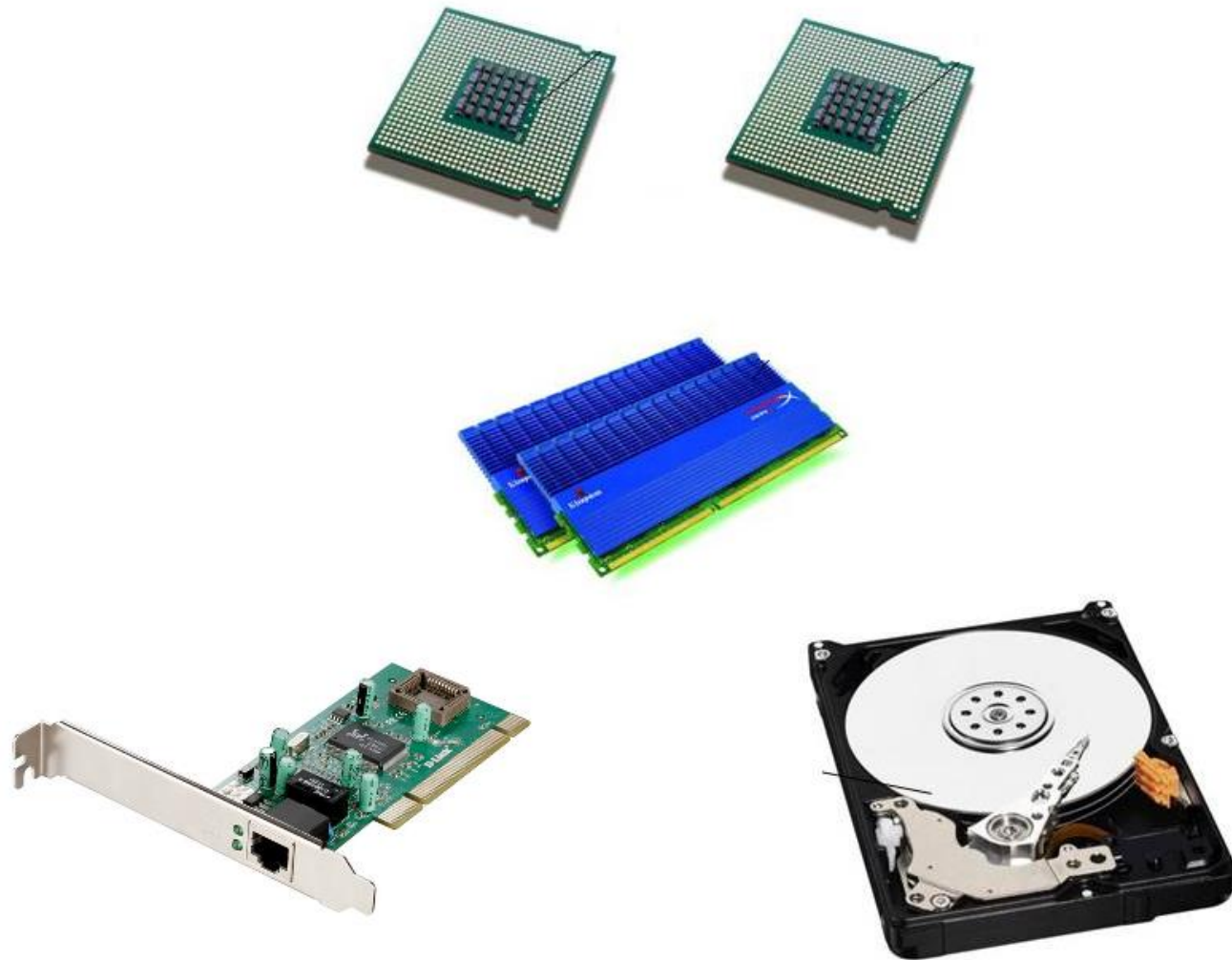
Настройка задачи резервного копирования.

# Глава 11

Мониторинг производительности

# Мониторинг производительности

- Подсистемы компьютера



# Мониторинг производительности

- Загрузка ЦП определяется как процент использования ЦП
- Понимание того, насколько хорошо работает процессор, зависит от состояния очереди выполнения
- Очереди выполнения – Очередь выполнения должна иметь не более 1-3 потоков в очереди на процессор.
- В идеальной среде процессор делит процент своего времени между пользователем (65%), ядром (30%) и простоем (5%).



# Лабораторная работа 11

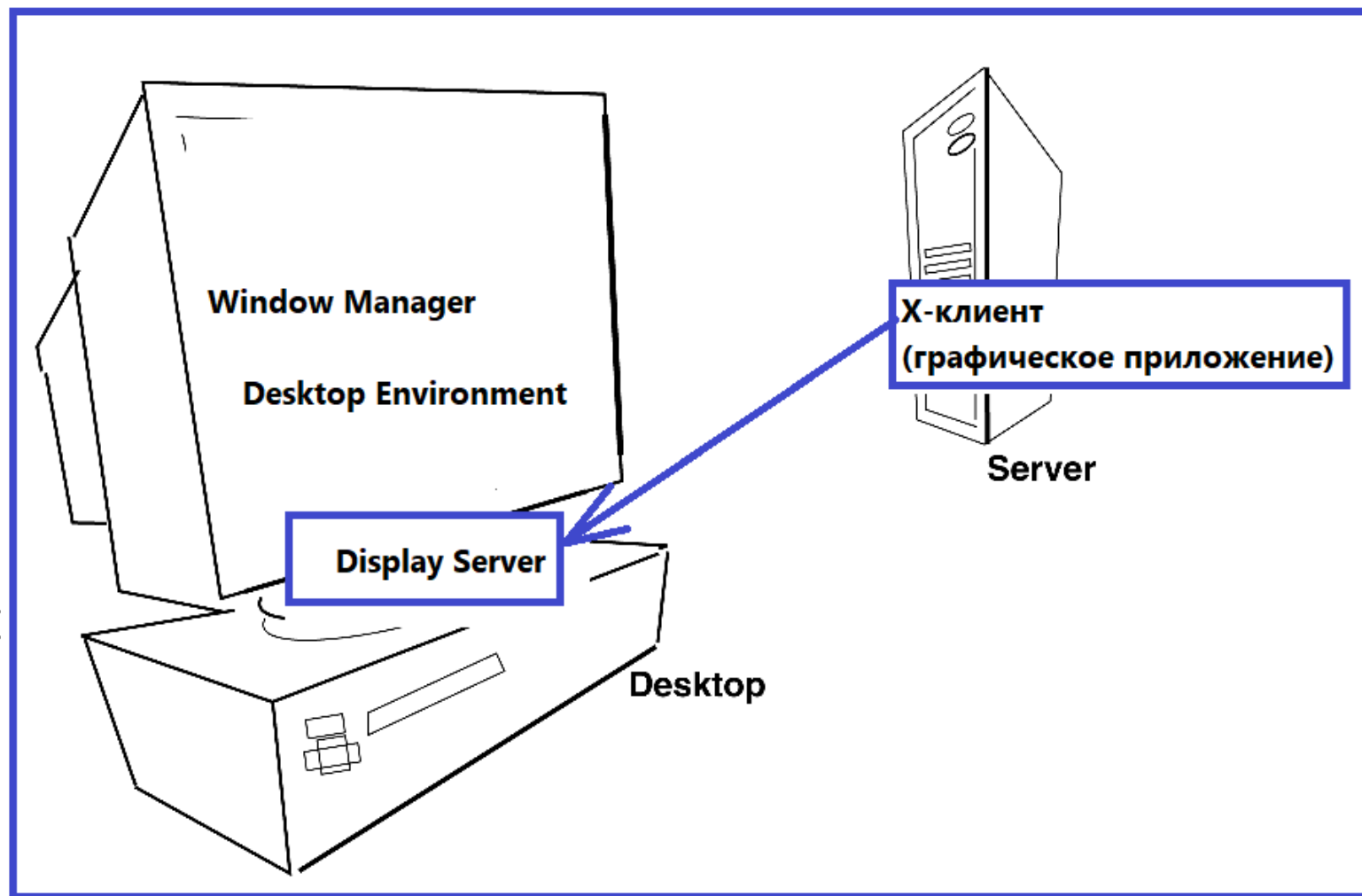
Упражнение 1. Использование утилит мониторинга

# Глава 12

Подключение к графической подсистеме Linux

# Графическая подсистема Linux

- Display Manager
- (sddm/lightdm)
- Display Server
- (X/Wayland)
- Desktop Environment
- (GNOME, KDE...)



# Лабораторная работа 12

Настройка задачи резервного копирования.