Ядро Linux

Ядро Linux содержит более 13 миллионов строк кода и является одним из самых крупных проектов с открытым исходным кодом в мире.

Ядро - это самый низкий уровень программного обеспечения, которое взаимодействует с аппаратными средствами компьютера. Оно отвечает за взаимодействие всех приложений, работающих в пространстве пользователя.

Типы ядер:

- ✓ микроядро;
- ✓ монолитное ядро (Linux);
- ✓ гибридное ядро (Windows)

Микроядро

Микроядро управляет только процессором, памятью и процессами. Все остальные функции обрабатывается в режиме пользователя.

Плюсы

- Портативность
- Небольшой размер
- Низкое потребление памяти
- Безопасность

Минусы

- Аппаратные средства доступны через драйверы
- Аппаратные средства работают медленнее потому что драйверы работают в пользовательском режиме
- Процессы должны ждать свою очередь чтобы получить информацию
- Процессы не могут получить доступ к другим процессам не ожидая

Монолитное ядро

Реализует расширенный набор функций : работу с процессором, памятью, процессами, драйверы устройств, управление файловой системой, систему ввода-вывода.

Плюсы:

- Более прямой доступ к аппаратным средствам
- Проще обмен данными между процессами
- Процессы реагируют быстрее
- **Минусы**:
- Большой размер
- Занимает много оперативной памяти
- Менее безопасно

Гибридное ядро

Гибридное ядро может выбирать с чем нужно работать в пользовательском режиме, а с чем в пространстве ядра.

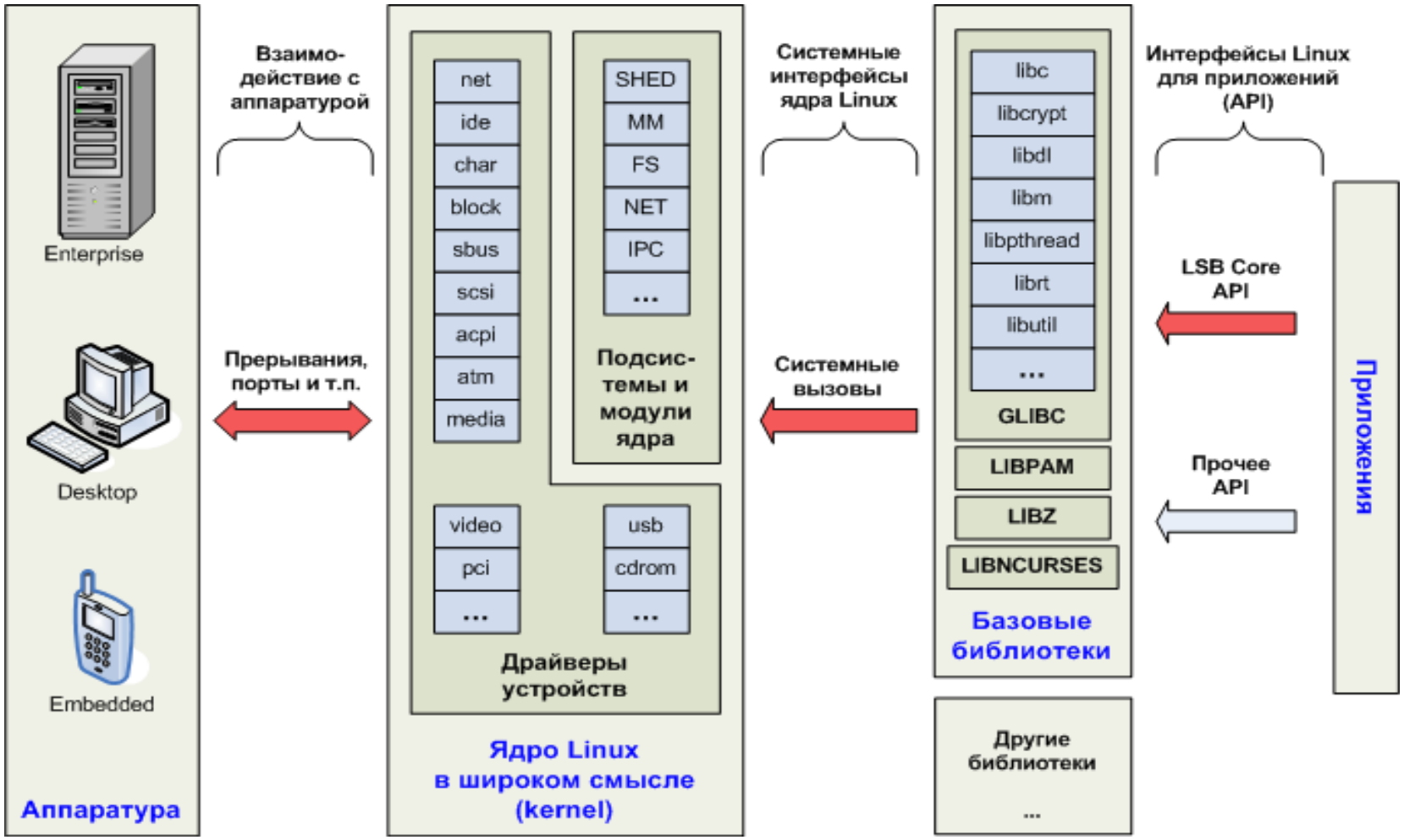
Плюсы

- Возможность выбора того что будет работать в пространстве ядра и пользователя
- Меньше по размеру чем монолитное ядро
- Более гибкое

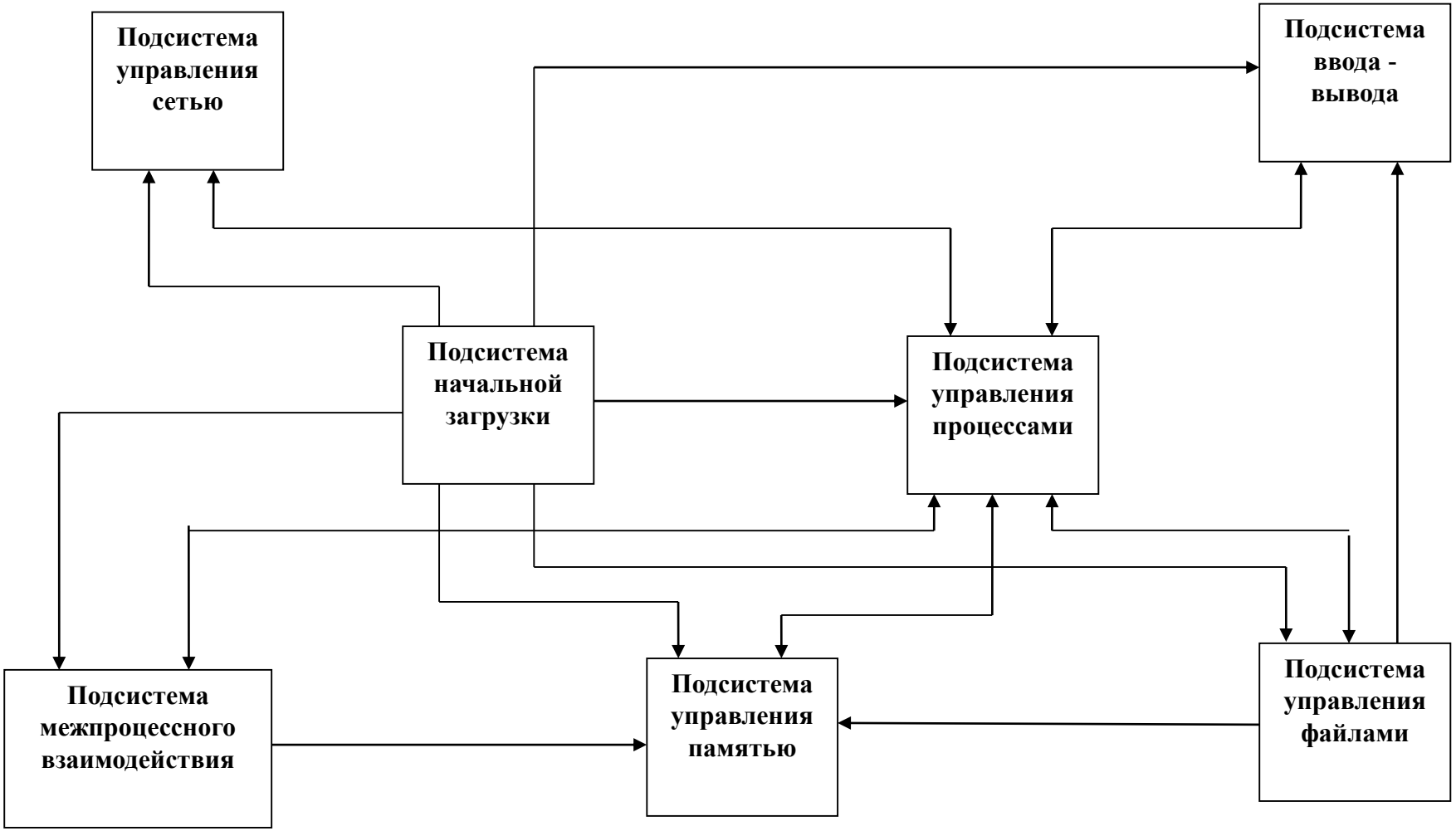
Минусы

- Может работать медленнее
- Драйверы устройств выпускаются производителями

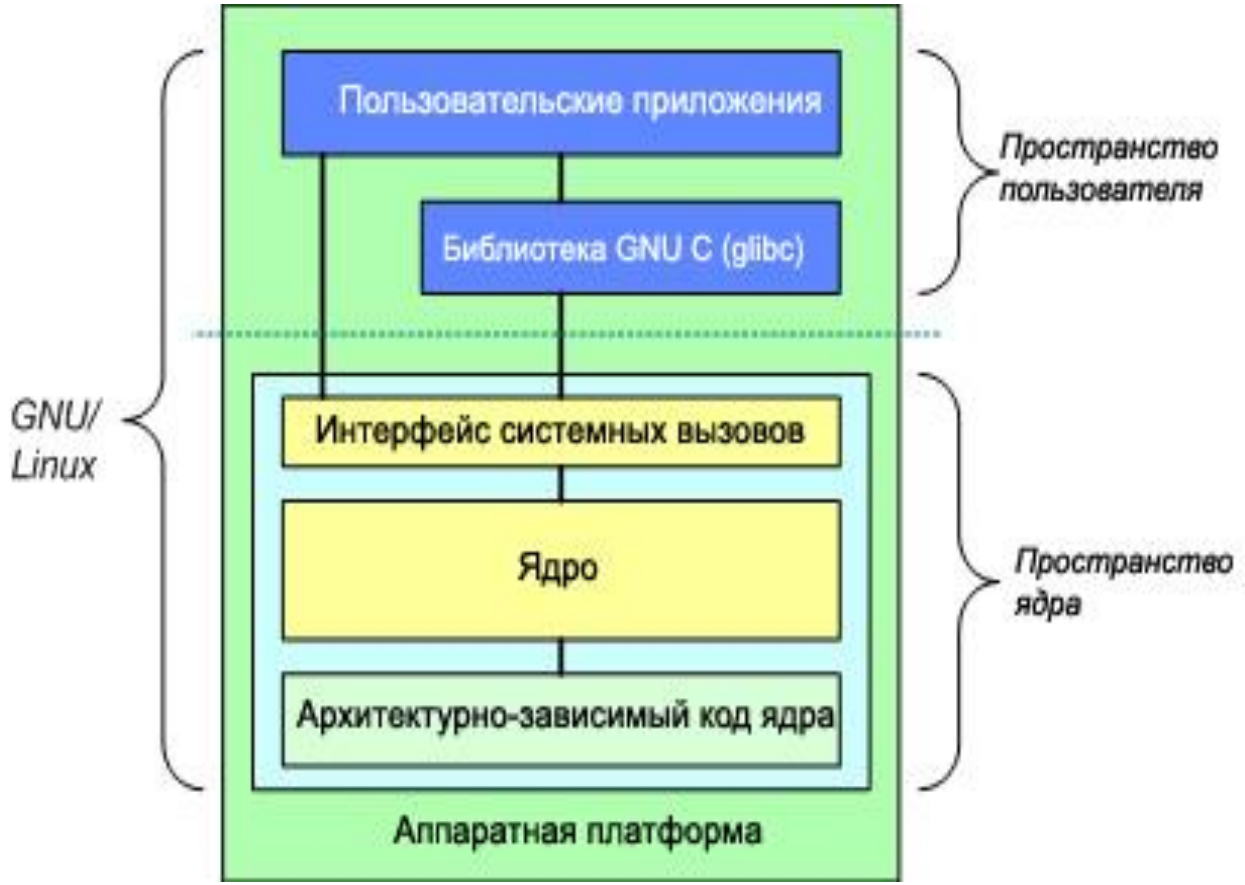
Архитектура ОС Linux



Состав подсистем



Взаимодействие между прикладными программами и ядром



Версии ядра Linux

Ядро Linux поставляется в двух вариантах:

- стабильном (stable);
- разрабатываемом (development).

Нумерация версий

старшая версия . младшая версия . редакция

редакция четная – стабильное ядро (для пользователей)

редакция нечетная – разрабатываемое ядро (для разработчиков)

https://www.kernel.org/

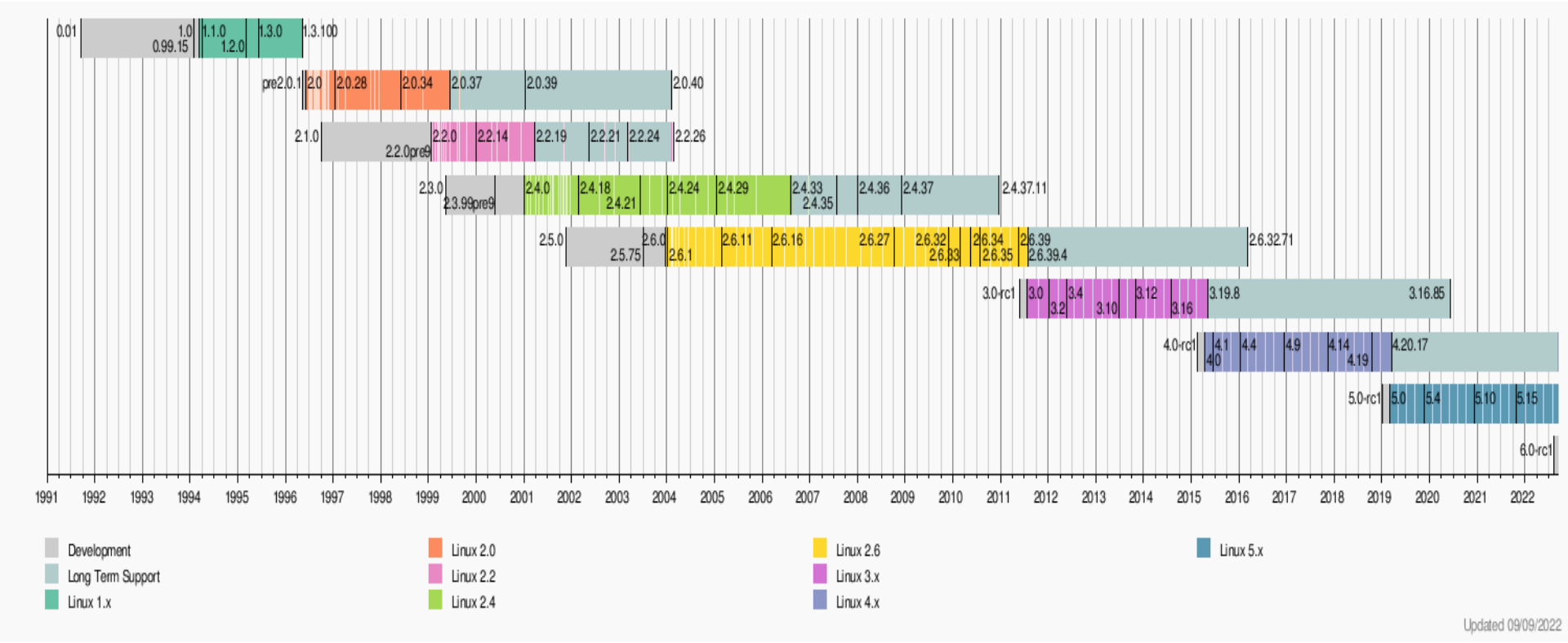
Узнать полную информацию о ядре можно с помощью команды

$ uname –a

Узнать номер установленной версии можно с помощью команды

$ lsb_release -a

История версий ядра Linux



Версии ядра Linux для Ubuntu

Версия Ubuntu	Версия ядра
4.10	2.6.9
5.04	2.6.11
5.10	2.6.13
6.06 LTS	2.6.15
6.10	2.6.18
7.04	2.6.19
7.10	2.6.20
8.04 LTS	2.6.24
8.10	2.6.27
9.04	2.6.28
9.10	2.6.31
10.04 LTS	2.6.32
10.10	2.6.35
11.04	2.6.38
11.10	3.0.4

Версия Ubuntu	Версия ядра
12.04 LTS	3.2
12.10	3.5
13.04	3.8
13.10	3.11
14.04 LTS	3.13
14.10	3.16
15.04	3.19
16.04.7 LTS	4.4.20
17.10.05	4.14.9
18.04.5 LTS	4.18
20.04.3 LTS	5.12
21.04	5.14
22.04.5 LTS	5.15

LTS (**long term support**) поддерживается обновлениями в течение трёх лет для настольной версии, а для серверной — пять.

Варианты конфигурирования ядра

Существует три основных способа:

- sudo make config

Способ для смелых и отважных. Конфигуратор задаст вам около четырехсот вопросов на которые нужно будет дать ответ.

- sudo make menuconfig

Предоставляет псевдографический конфигуратор, в котором все настройки разложены по пунктам.

- sudo make xconfig

Предоставляет графический конфигуратор.

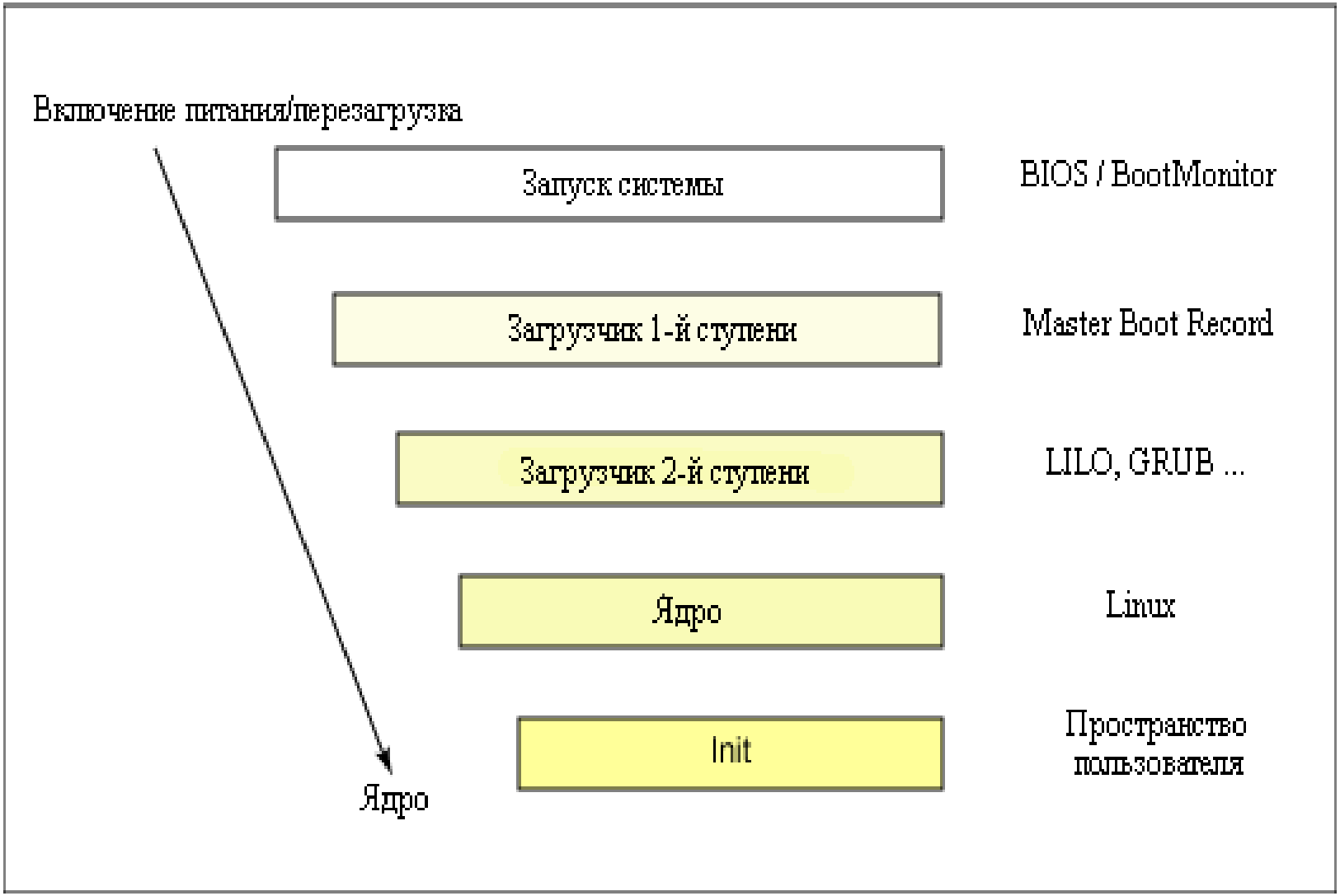
Каталоги исходных кодов ядра

Каталог	Описание
arch	Специфичный для аппаратной платформы исходный код
crypto	Криптографический API
Documentation	Документация исходного кода ядра
drivers	Драйверы устройств
fs	Подсистема VFS и отдельные файловые системы
include	Заголовочные файлы ядра
init	Загрузка и инициализация ядра
ipc	Код межпроцессного взаимодействия
kernel	Основные подсистемы, такие как планировщик
lib	Вспомогательные подпрограммы
mm	Подсистема управления памятью и поддержка виртуальной памяти
net	Сетевая подсистема
scripts	Сценарии компиляции ядра
security	Модуль безопасности Linux
sound	Звуковая подсистема
usr	Начальный код пространства пользователя (initramfs)

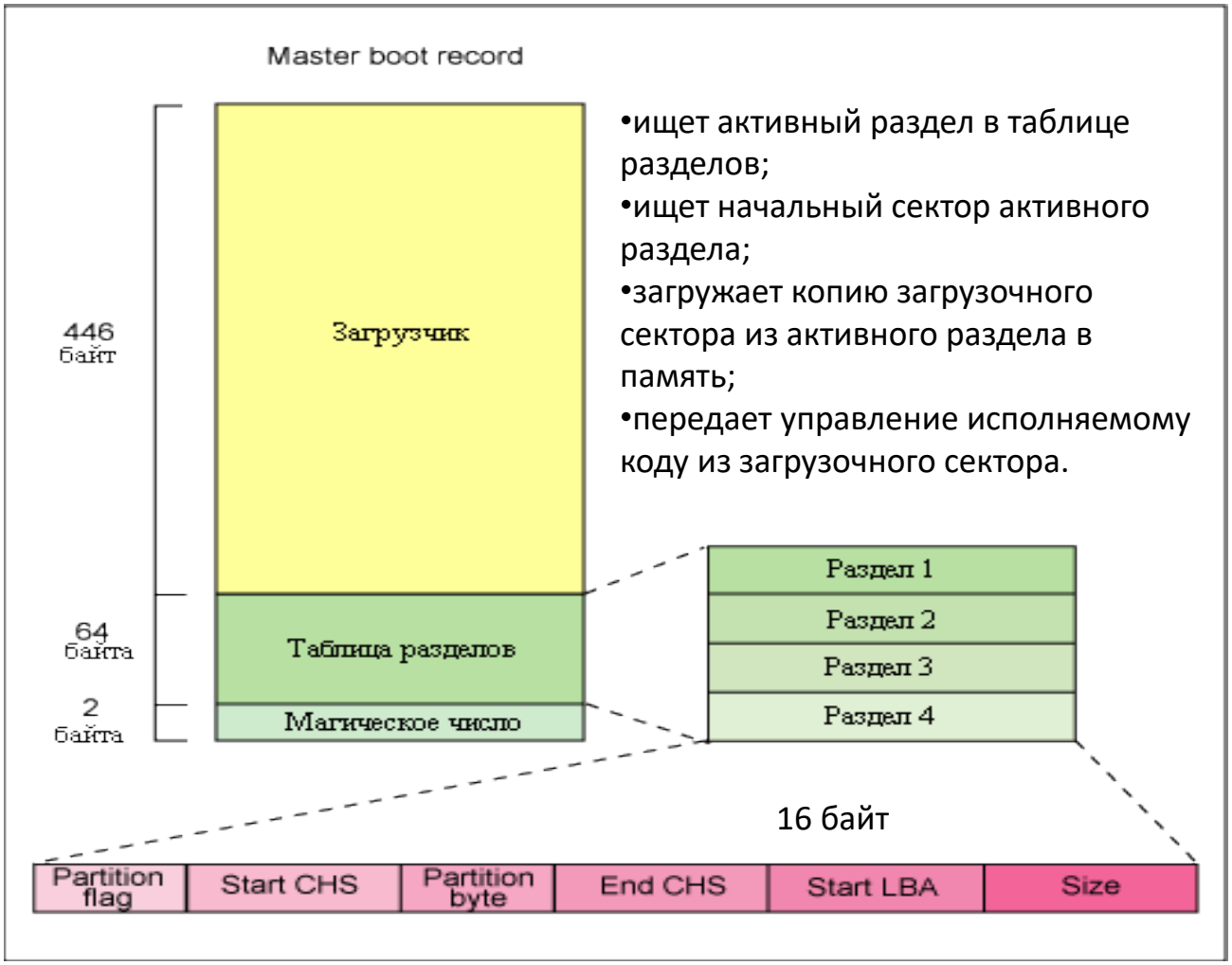
Функции ОС

1. Управление выполнением процессов посредством их создания, приостановки, организации взаимодействия и завершения.
2. Планирование очередности предоставления выполняющимся процессам процессорного времени (диспетчеризация).
3. Выделение процессу оперативной памяти, защищая его адресное пространство.
4. Управление внешней памятью и защиты пользовательских файлов от несанкционированного доступа.
5. Управление доступом процессов к периферийным устройствам.

Процесс загрузки Linux



Запуск системы



При включении питания выполняется проверочный тест BIOS и процессора — POST.

BIOS выбирает загрузочное устройство (обычно жесткий диск) и загружает первичный загрузчик (**First Boot Loader)**.

Признак активного раздела, код системы (файловая система FAT16, FAT32 или NTFS), номер начального сектора, номер последнего сектора, относительный номер сектора начала раздела, размер раздела в секторах. Магическое число используется для идентификации образа ядра.

Загрузчик 1-й ступени

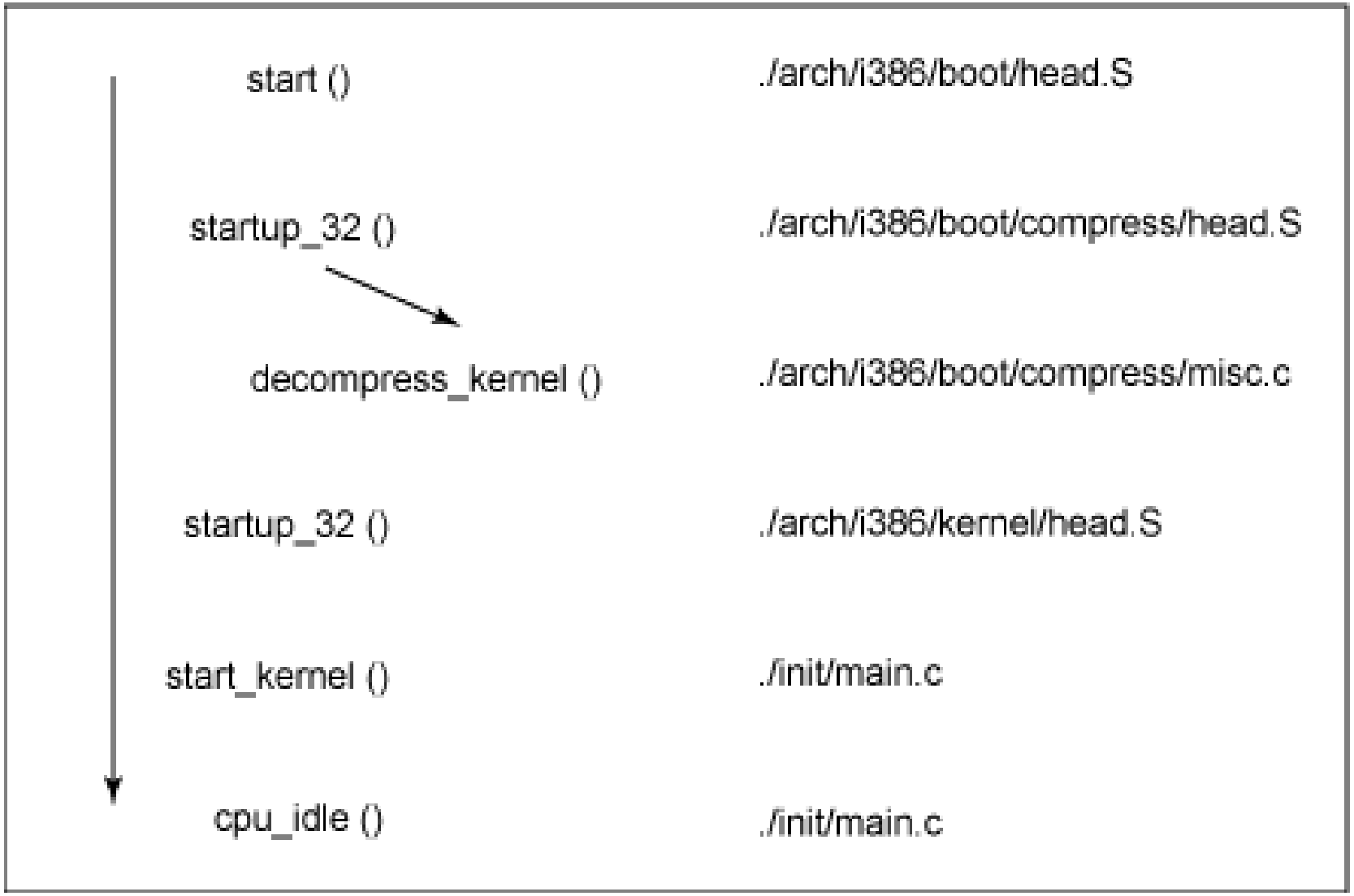
FBL выполняет следующие действия:

- ищет активный раздел в таблице разделов;
- ищет начальный сектор активного раздела;
- загружает копию загрузочного сектора из активного раздела в память;
- передает управление исполняемому коду из загрузочного сектора (SSB).

Загрузчик 2-й ступени GRUB

1. Обращается к файловой системе и выводит список вариантов загрузки (Загрузчиком может быть **LILO** (более старый) или **GRUB**. Загрузчик берет свои настройки из конфигурационного файла (*/etc/lilo.conf* - для **LILO** и */boot/grub/grub.conf* или */boot/grub/menu.lst* - для **GRUB**).
2. Из каталога /boot выполняет загрузку в память образа ядра
3. Передает управление процедуре start()

Функции, выполняемые на стадии загрузки ядра



Этапы загрузки на стадии развертывания ядра

1. Инициализация сегментных регистров и временного стека.
2. Декомпрессия ядра
3. Выполнение процесса 0 (настройка стека, инициализация таблицы прерываний, определение типа процессора, монтирование корневой файловой системы)
4. Запуск процесса init (инициализация таблицы страниц и системы распределения памяти, создает процесс 1, запускает /sbin/init как первый процесс пользовательского режима)

Типы процессов в Linux

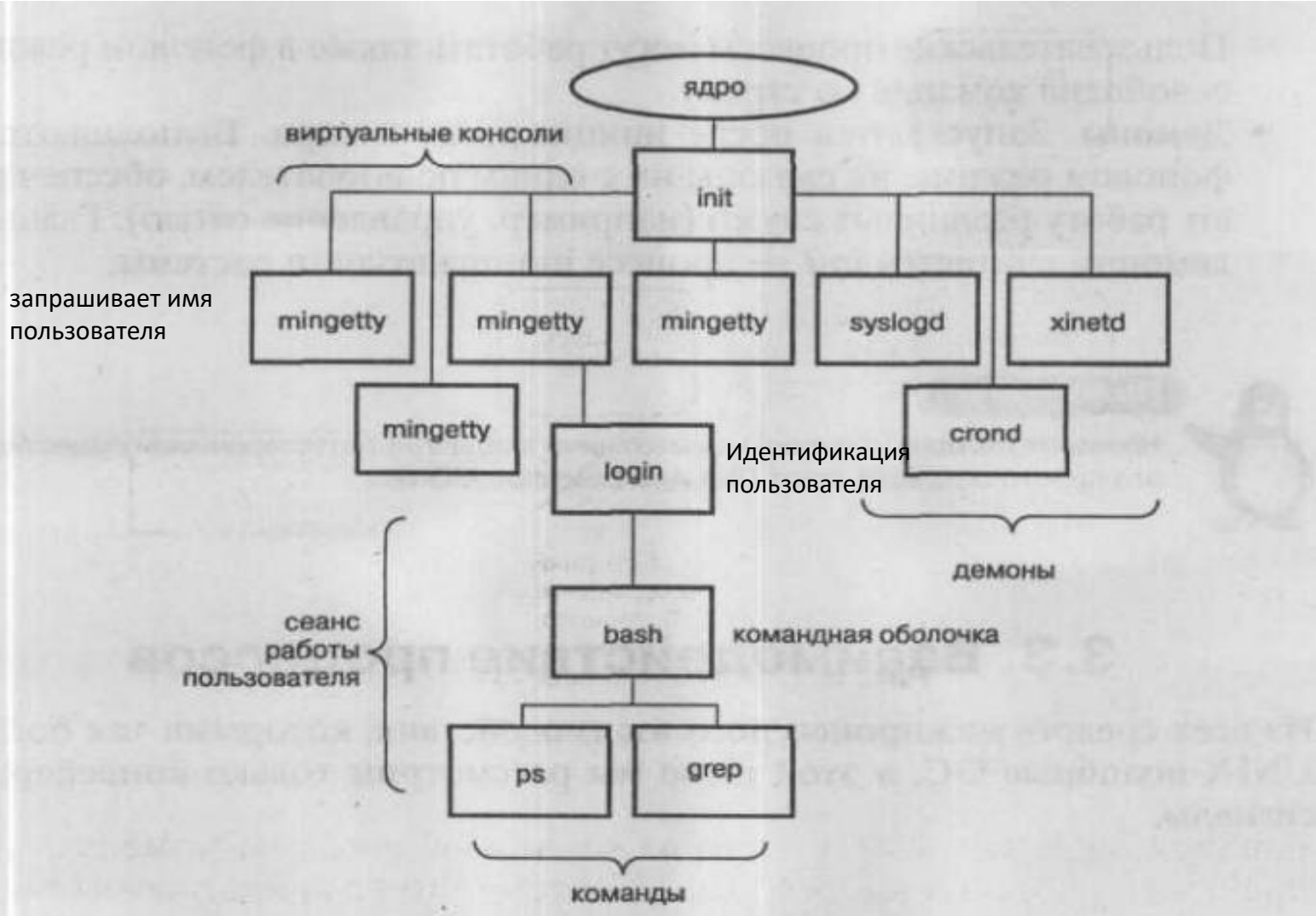
Системные процессы - являются частью ядра и всегда расположены в оперативной памяти. Системные процессы не имеют соответствующих им программ в виде исполняемых файлов и запускаются при инициализации ядра системы.

Демон (Daemon) — это программа, работающая в фоновом режиме без прямого общения с пользователем (предком демона является процесс init).

Сервисы (службы) - это программы, которые запускаются и останавливаются через инициализационные скрипты, расположенные в каталоге /etc/init.d. Есть два вида сервисов: те, которые выполняются и завершают свою работу — задачи (task) и те, которые стартуют и продолжают работать в фоне.

Прикладные процессы - это процессы, порожденные в рамках пользовательского сеанса работы.

Иерархия процессов



Уровни запуска сервисов

Термин уровень выполнения означает режим функционирования операционной системы компьютера. Загружается из файла **/etc/inittab**

№ Уровня	Функция	Описание уровня
0	Выключение системы	По команде завершения работы
1	Загрузка в однопользовательском режиме	Используется для диагностики и устранения неполадок в системе
2	Загрузка в многопользовательском режиме без поддержки сети	
3	Загрузка в многопользовательском режиме с поддержкой сети	Работа с NFS
4	Не используется	
5	Загрузка в многопользовательском режиме с поддержкой сети и графического входа в систему	XDM (графический менеджер дисплея X)
6	Перезагрузка	

Для того, чтобы скрипт запускался автоматически во время запуска системы, надо создать символическую ссылку на скрипт и разместить её в каталоге /etc/rc.d/rcN.d где N соответствует уровню выполнения скрипта.

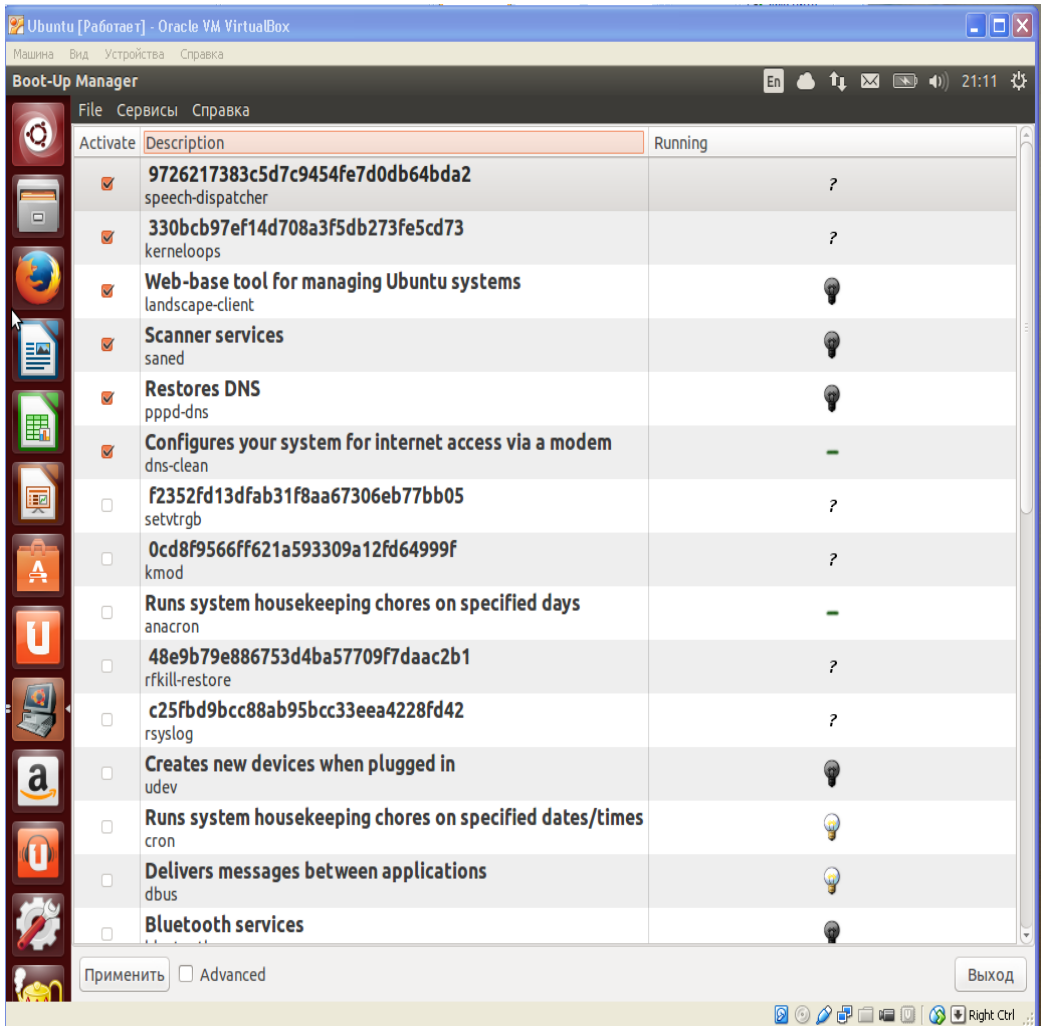
Примеры стандартных служб Linux

Сервис	**Описание**
autofs	Автомонтировщик файловых систем
gpm	Поддержка мыши в консоли операционной системы
keytable	Загружает раскладку клавиатуры, указанную в файле /etc/sysconfig/keyboard
klogd	Протоколирует сообщения ядра в файлы каталога /var/log/kernel (для диагностики ошибок)
network	Поддержка сети, необходима даже для работы графической системы и системы печати

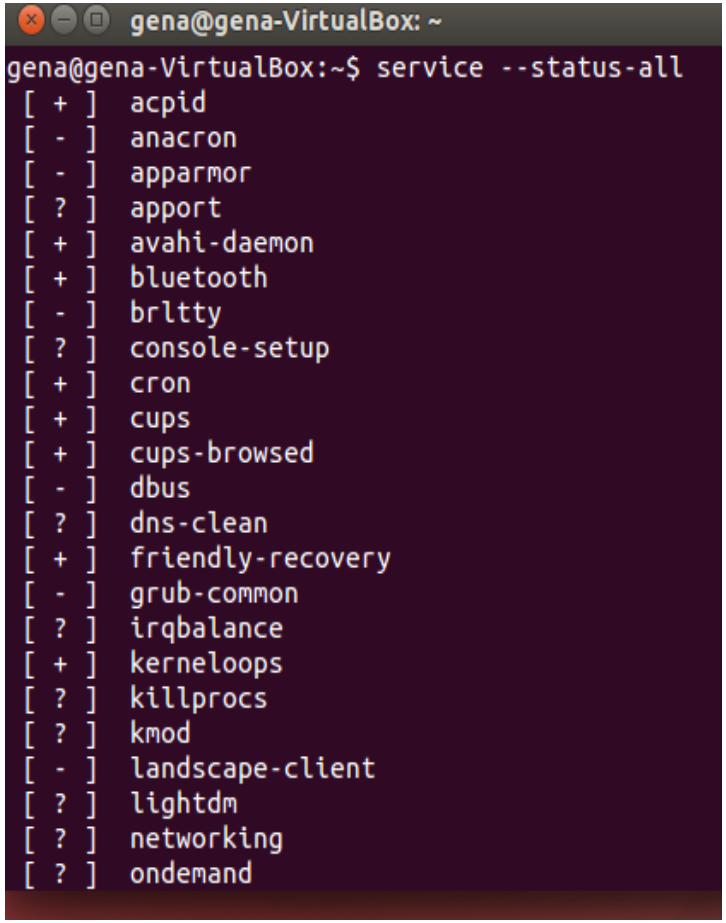
Управлять сервисами можно с помощью команд **service** или **systemctl**
Запустить службу: sudo service **имя службы** start
Остановить службу: sudo service **имя службы** stop
Перезапустить службу: sudo service **имя службы** restart
Проверить статус службы: sudo service **имя службы** status

Просмотр и администрирование сервисов

Boot-Up Manager



service --status-all



[+] - запущенный сервис.
[-] - остановленный сервис.
[?] - для данного сервиса отсутствует команда status.

Использование скриптов для автозапуска сервисов

1. Написать скрипт и поместить его в каталог /etc/init.d (например с именем myscript)

2. Назначить скрипту права доступа

Для файлов: **r** - право на чтение из файла; **w** - разрешает запись в файл (в частности перезапись или изменение); **x** - позволяет исполнить файл.

sudo chmod +x /etc/init.d/myscript

3. Включить скрипт в автозагрузку

sudo update-rc.d myscript start 99 2 3 4 5 . stop 01 0 1 6

Эта команда добавит скрипт «myscript» во все уровни загрузки. Будет выполнен запуск сервиса на уровнях со 2 по 5 с приоритетом 99 (в последнюю очередь) и остановка сервиса на 0, 1 и 6 уровнях с приоритетом 01 (самым первым).

4. Чтобы удалить скрипт из автозагрузки, воспользуйтесь командой:
sudo update-rc.d -f myscript remove

Сервис планирования заданий cron

Сервис **cron** позволяет выполнять сценарии или команды в определенное время, он запускается при старте системы и работает в фоновом режиме.
cron настраивается через записи в файле crontab. Файлы crontab разделяются на поля:
m h dom mon dow command
m: минуты запуска команды, от 0 до 59.
h: час запуска команды, от 0 до 23.
dom: день месяца для выполнения команды.
mon: месяц даты выполнения команды.
dow: день недели для выполнения команды, от 0 до 7. Воскресенье может быть обозначено как 0 так и 7, оба значения допустимы.
command: выполняемая команда.
Для добавления или изменения записей в файле crontab используется команда ***crontab -e***. Содержимое файла crontab можно просмотреть с помощью команды ***crontab -l***.
Пример: запуск команды каждый день в конце рабочего дня
0 18 * * 1-5 echo “Пора домой” >/dev/pts/2 переадресация вывода на управляющий терминал
Запуск программы prog каждые 15 минут
*/15 * * * * work/prog

Файлы загрузки системы

- /etc/rc.d0 …. /etc/rc.d6 – файлы с командами инициализации системы;
- /etc/lilo.conf - файл, определяющий конфигурацию загрузчика lilo;
- /etc/modules - файл, определяющий конфигурацию загружаемых модулей ядра;
- /etc/fstab - содержит информацию, необходимую для автоматического монтирования файловых систем;
- /etc/passwd - различная регистрационная информация, включая пароли;
- /etc/profile - глобальный файл профилей - устанавливает переменную $PATH и другие важнейшие переменные;
- /etc/bashrc - глобальный файл конфигурации bash, устанавливает синонимы (алиасы) и функции, и т.п.;
- /etc/issue - содержит сообщение, выдаваемое на терминал перед входом в систему (перед запросом имени и пароля); однако редактировать этот файл с целью изменения текста сообщения не стоит, потому что сам он формируется инициализационным скриптом /etc/rc.d/rc.local;