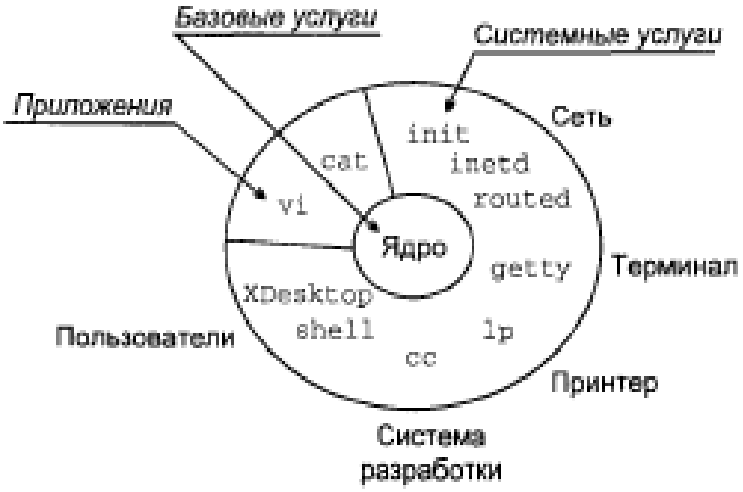
# Архитектура Linux

## Двухуровневая модель системы




*Ядро (kernel)* взаимоодействует с аппаратной частью компьютера, изолируя прикладные программы от особенностей ее архитектуры. Ядро имеет набор услуг, предоставляемых прикладным программам. К этим услугам ядра относятся операции ввода/вывода (открытия, чтения, записи и управления файлами), создания и управления процессами, их синхронизации и межпроцессного взаимодействия.
Все приложения запрашивают услуги ядра посредством *системных вызовов*.

Второй уровень составляют приложения или задачи, как *системные* (определяющие функциональность системы), так и *прикладные* (обеспечивающие пользовательский интерфейс UNIX). Несмотря на внешнюю разнородность приложений, схемы взаимодействия приложений с ядром одинаковы.

## Ядро системы

Ядро обеспечивает базовую функциональность операционной системы:
создает процессы и управляет ими,
распределяет память и
обеспечивает доступ к файлам и
периферийным устройствам.

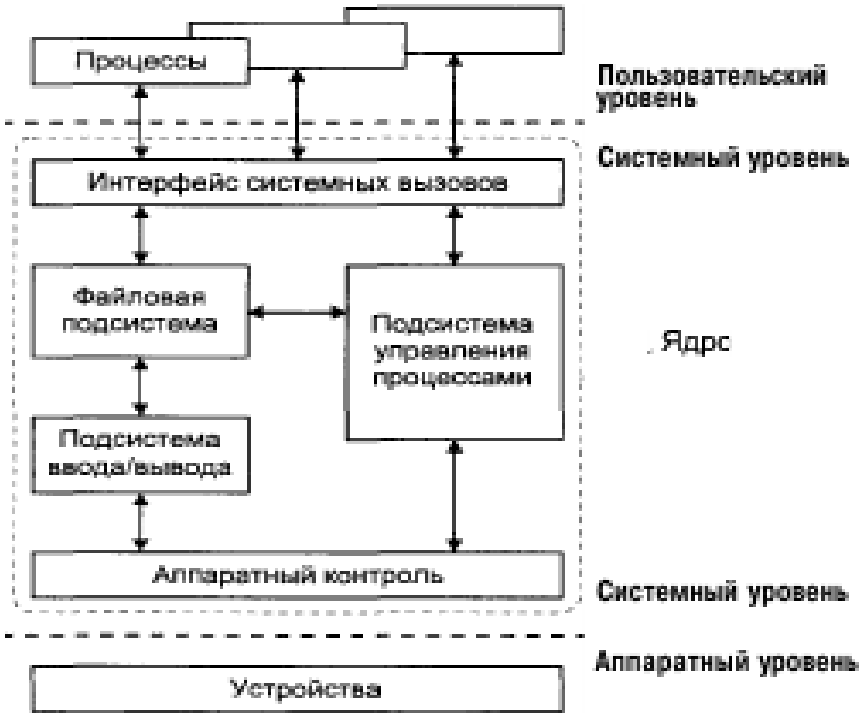
Взаимодействие прикладных задач с ядром происходит посредством стандартного *интерфейса системных вызовов*.
Интерфейс системных вызовов характеризуется:
набором услуг ядра и определением формата запросов на эти услуги.
Процесс запрашивает услугу посредством системного вызова определенной процедуры ядра (внешне это похоже на обычный вызов библиотечной функции).
Ядро от имени процесса выполняет этот запрос и возвращает процессу необходимые данные.

В примере :

```
main(void)
{
int fd;
char buffer [80];
/* Открываем файл - получаем ссылку (файловый дескриптор) fd  */
fd = open("file_1" , 0_RDONLY);
/* Считываем в буфер 80 символов */
read(fd,  buffer, sizeof(buffer));
/* Закрываем файл */
close(fd);
return 0;
}
```

программа открывает файл, считывает из него данные и закрывает его.
Операции открытия, чтения и закрытия файла выполняются ядром по запросу задачи,
а функции *open*, *read* и *close* являются системными вызовами.

**Структура ядра**



Три основные подсистемы ядра:
1. Файловая подсистема
2. Подсистема управления процессами и памятью
3. Подсистема ввода/вывода

**Файлы и файловая система**

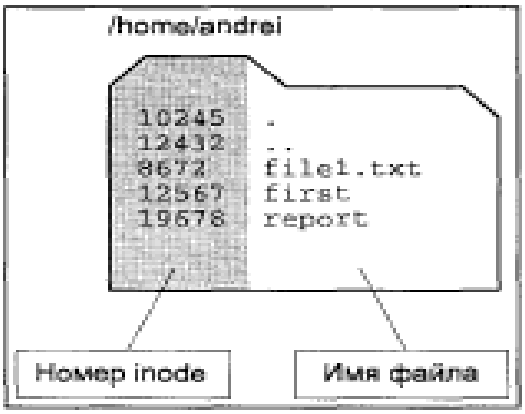
Как и во многих других системах, файлы в UNIX организованы в виде древовидной структуры.
Корнем этого дерева является корневой каталог (root directory), имеющий имя "/" .
Полное имя любого файла начинается с "/" и не содержит идентификатор устройства (дискового накопителя), на котором он фактически хранится.
Каждый файл имеет связанные с ним метаданные (хранящиеся в индексных дескрипторах - *inode*), содержащие все характеристики файла и позвооляющие операционной системе выполнять с ним операции, заказанные приикладной задачей:
открыть файл, прочитать или записать данные, создать или удалить файл.
В частности, метаданные содержат и указатели на дисковые блоки хранения данных файла

**Типы файлов**

В UNIX существуют 6 типов файлов, различающихся по функциональному назначению и действиям ОС при выполнении операций над файлами:

- regular file - обычный файл
- directory - каталог
- special device file - специальный файл устройства
- named pipe - FIFO или именованный канал
- link - связь
- socket – сокет

С помощью **каталогов** формируется логическое дерево файловой системы.
*Каталог* – это файл, содержащий имена находящихся в нем файлов, а также указатели на дополнительную информацию - *метаданные*, (информация для операционной системы при работе с файлом).
Каталоги определяют положение файла в дереве файловой системы, поскольку сам файл не содержит информации о своем местонахождении.
По существу каталог представляет собой таблицу, каждая запись которой соответствует некоторому файлу.
Первое поле каждой записи содержит указатель на метаданные (номер *индексного дескриптора inode*), а второе - определяет имя файла.

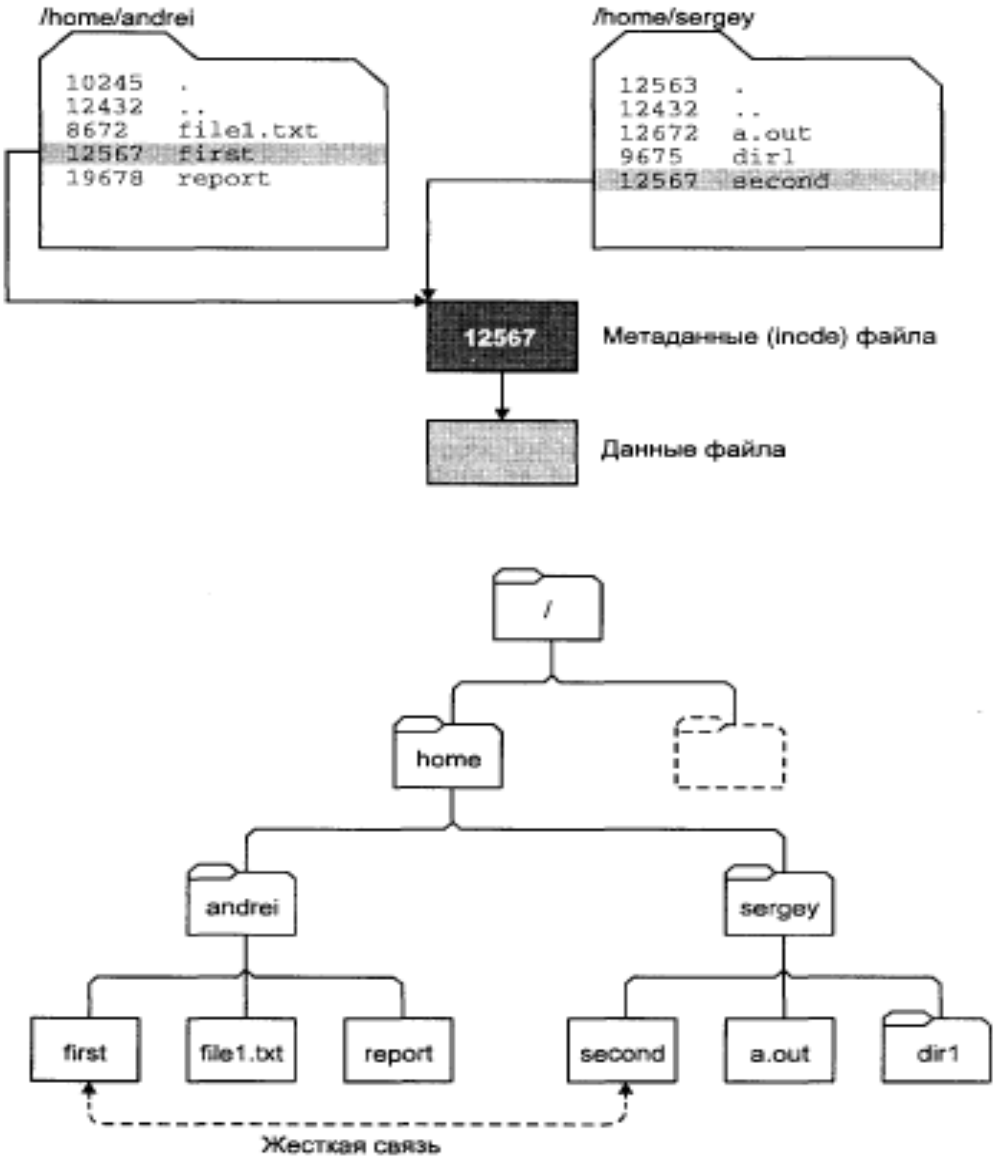



**Специальные файлы устройств** обеспечивают доступ
к физическим устройствам.
Различают *символьные* (character) и *блочные* (block) файлы устройств,
соответствующие разным типам устройств и режимам доступа.
Доступ к устройствам осуществляется посредством
открытия, чтения/записи и закрытия специального файла устройства.

**FIFO или именованный канал** – это тип файла,
используемый для связи между процессами.

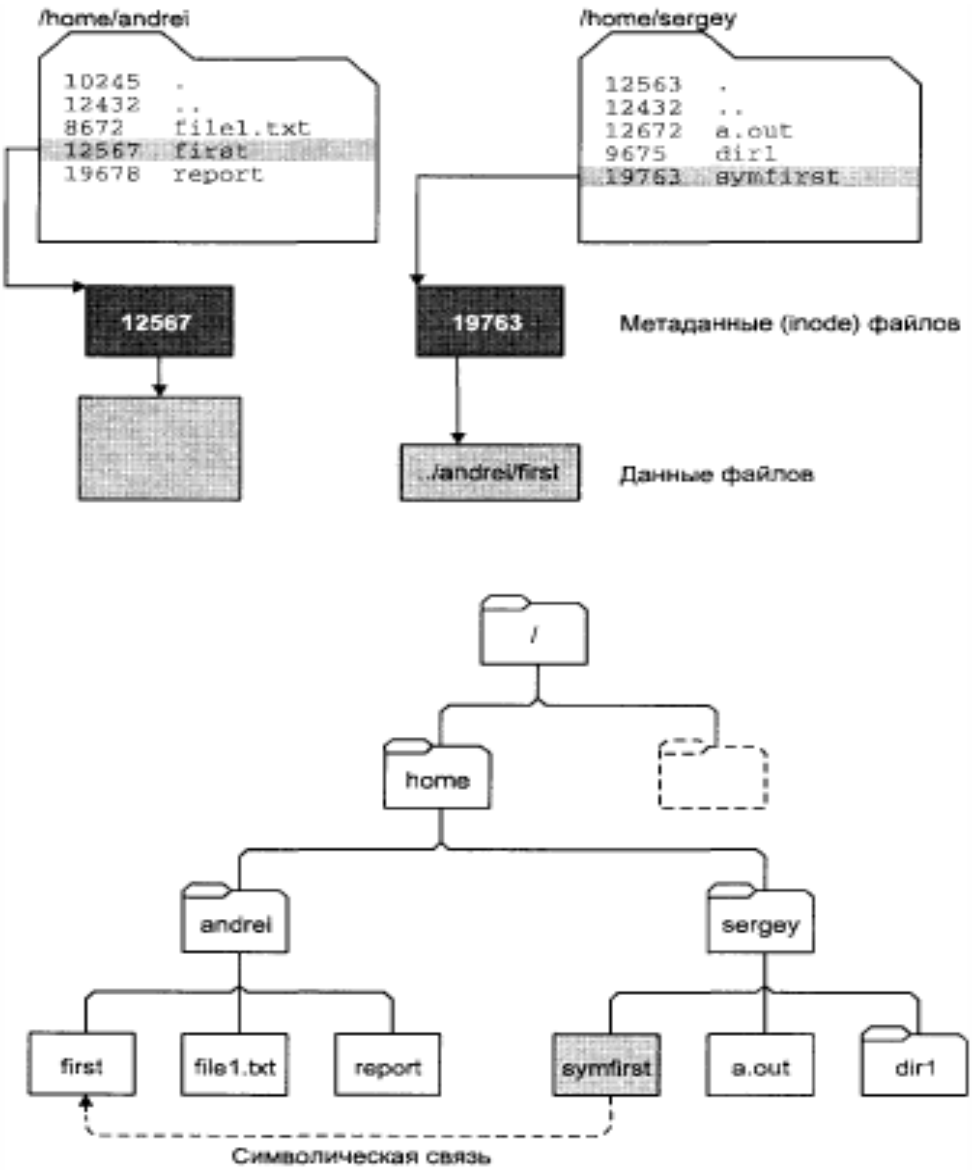
**Связь**. Каталог содержит имена файлов и указатели на их метаданные.
В то же время, сами метаданные не содержат
ни имени файла, ни указателя на это имя.
Такая архитектура позволяет одному файлу иметь
несколько имен в файловой системе.
Имена жестко связаны с метаданными,
в то время как сам файл существует независимо от того,
как его называют в файловой системе.
Можно создать еще одно (и более) имен одного и того же файла.
В списках файлов каталогов, файлы будут отличаться только именем.
Все остальные *атрибуты файла* будут абсолютно одинаковыми.
С точки зрения пользователя - это два разных файла, но
изменения, внесенные в любой из этих файлов, затронут и другой,
поскольку оба они ссылаются на одни и те же данные.
Удаление одного из файлов не приведет к удалению самого файла,
т. е. его метаданных и данных.

Пример структуры файловой системы в случае link-а :




Помимо организации *жесткой связи* возможно установление *символической связи* (*гибкой ссылки*) с помощью специального файла, который будет только косвенно адресовать целевой файл.
Данные специального файла (являющегося символической ссылкой) содержат только полное имя целевого файла, то есть ссылаются на него.

Пример структуры файловой системы в случае символической связи:

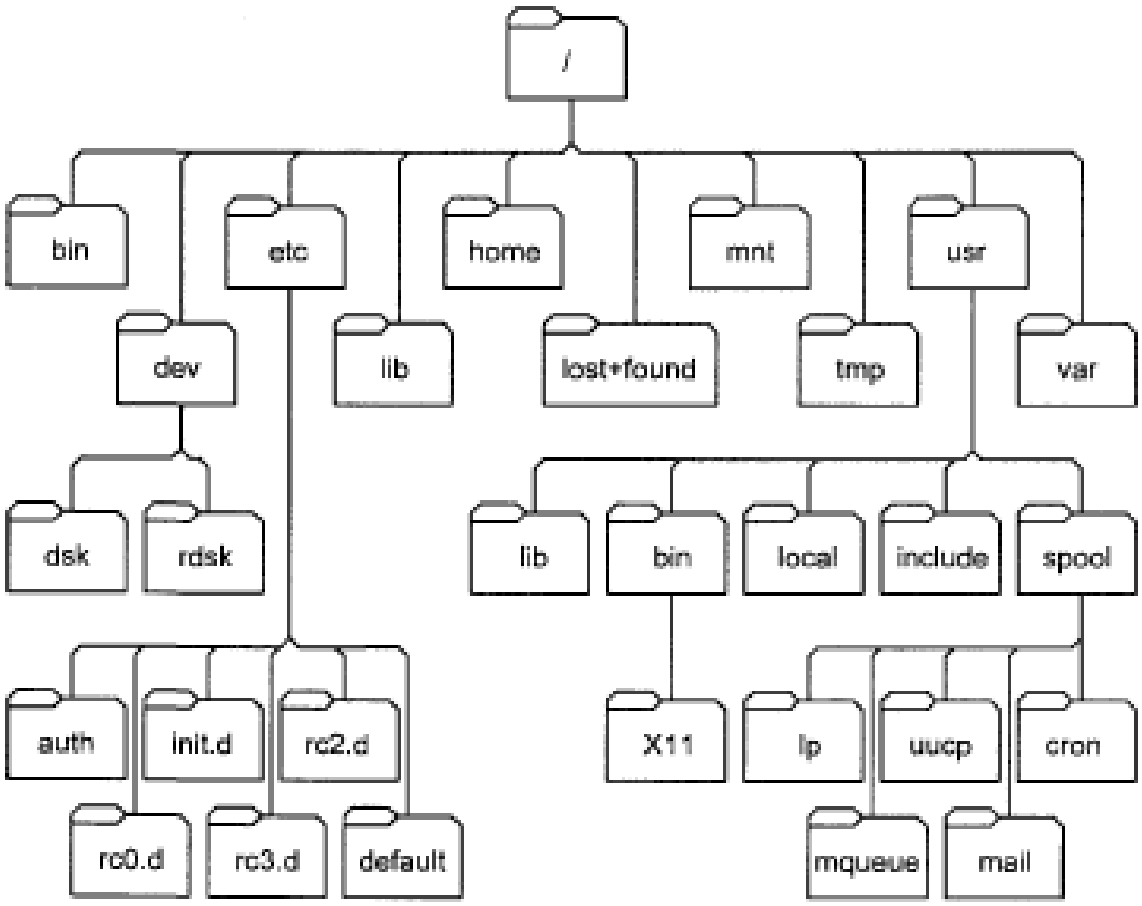



**Сокеты.** Этот тип файлов предназначен для организации взаимодействия между процессами исполняющимися на разных компьютерах. Интерфейс сокетов используется для доступа к сети на IP протоколе .

## Структура файловой системы UNIX

Использование общепринятых имен основных файлов и структуры каталогов существенно облегчает работу в UNIX, ее администрирование и переносимость.
Нарушение этой структуры может привести

к неработоспособности системы или отдельных ее компонентов.




Корневой каталог **/** является основой любой файловой системы UNIX. Все файлы и каталоги располагаются в рамках структуры, порожденной корневым каталогом, независимо от их физического местонахождения.

В каталоге **/bin** находятся наиболее часто употребляемые команды и утилиты системы.

Каталог **/dev** содержит специальные файлы устройств, являющиеся интерфейсом доступа к периферийным устройствам.

**/etc** . В этом каталоге находятся системные конфигурационные файлы и многие утилиты администрирования, например, скрипты инициализации системы.

В каталоге **/lib** находятся библиотечные файлы языка С и других языков программирования. Стандартные названия библиотечных файлов имеют вид типа libx.a или libx.so.

Часть библиотечных файлов также находится в каталоге **/usr/lib**
**/lost+found** каталог "потерянных" файлов.
При аппаратных сбоях могут возникать ошибки целостности файловой системы, что может приводить к появлению «безымянных» файлов. Структура и содержимое такого файла являются правильными, однако для него отсутствует имя в каком-либо из каталогов. Программы проверки и восстановления файловой системы помещают такие файлы в каталог /lost+found под системными именами (числовыми).

**/mnt с**тандартный каталог для временного связывания (монтирования) физических файловых систем к корневой для получения единого дерева логической файловой системы.

**/home** каталог для размещения домашних каталогов пользователей. Например, имя домашнего каталога пользователя alex будет называться /home/alex .

В каталоге **/usr** находятся подкаталоги различных сервисных подсистем, исполняемые файлы утилит UNIX, подкаталоги электронной почты, а также и дополнительные пользовательские программы, библиотеки т. д.

Каталог **/var** используется для хранения временных файлов различных сервисных подсистем системы печати и т. д.

Каталог **/tmp** предназначен для хранения временных файлов, необходимых для работы различных подсистем UNIX, а также файлов пользователей. Этот каталог открыт на запись для всех пользователей системы, в том числе для работающих под гостевым аккаунтом.

## Для практических занятий

Основные команды:

**pwd**
**ls**
**cd**
**mkdir**
**rm**
**cp**
**link**
**who**
**sort**

**man** *on-line reference manuals*

**sed** *stream editor*
**awk** *text processing*
**bash** *shell*