Содержание

1 Обзор ОС UNIX: архитектура, вход в систему, файлы и каталоги, ввод и вывод. 4

1.1 Архитектура UNIX 4
1.1.1 Определение ОС 4
1.1.2 Интерфес ядра 4
1.1.3 Коммандая оболочка 4
1.1.4 Обобщение 4
1.1.5 Linux 4
1.1.6 Схема, отражающая структуру UNIX 5
1.2 Вход в систему 5
1.3 Файлы и каталоги 5
1.3.1 Файловая система 5
1.3.2 Имя файла 6
1.3.3 Путь к файлу 6
1.3.4 Рабочий каталог 6
1.3.5 Домашний каталог 6
1.4 Ввод и вывод 7
1.4.1 Дескрипторы файлов 7
1.4.2 Стандартный ввод, стандартный вывод,Стандартный вывод сообщений об ошибках 7
1.4.3 Небуферизованный ввод-вывод 7
1.4.4 Стандартные функции ввода-вывода 7

2 Обзор ОС UNIX: программы и процессы, обработка ошибок, идентификация пользователя. 8

3 Обзор ОС UNIX: сигналы, представление времени, системные вызовы и библиотечные функции. 10

4 Стандарты и реализации ОС UNIX: пределы ISO C, пределы POSIX, функции sysconf(), pathconf() и fpathconf(), элементарные системные типы данных. 10

5 Файловый ввод-вывод: дескрипторы файлов, функция open(), функция creat(), функция close(). 10

6 Файловый ввод-вывод: Функция lseek(), функция read(), функция write()/ 10

7 Файловый ввод-вывод: эффективность операций ввода-вывода 10

8 Файловый ввод-вывод: совместное использование файлов, атомарные операции, функции dup() и dup2() 10

9 Файловый ввод-вывод: функции sync(), fsync(), fdatasync(), fcntl(), ioctl(), /dev/fd 10

10 Файлы и каталоги: функции stat(), fstat(), lstat(), содержимое struct stat. 10

11 Файлы и каталоги: типы файлов, права доступа к файлу, функция umask(). 10

12 Файлы и каталоги: функции chmod(), fchmod(), chown(), fchown(), lchown(). 10

13 Файлы и каталоги: размер файла, дырки в файлах, усечение файлов, файловые системы, функции link(), unlink(), remove(), rename(). 10

14 Файлы и каталоги: символические ссылки, функции symlink() и readlink(). 10

15 Файлы и каталоги: временные характеристики файлов, функция utime(). 12

16 Файлы и каталоги: функции mkdir() и rmdir(), чтение каталогов, функции chdir(), fchdir(), getcwd(). 12

17 Стандартная библиотека ввода-вывода: потоки и объекты FILE, стандартные потоки ввода, вывода и сообщений об ошибках, буферизация. 12

18 Стандартная библиотека ввода-вывода: открытие потока, чтение из потока и запись в поток, функции ввода, функции вывода. 12

19 Стандартная библиотека ввода-вывода: эффективность стандартных операций ввода-вывода, позиционирование в потоке. 12

20 Стандартная библиотека ввода-вывода: форматированный вывод, форматированный ввод, временные файлы. 12

21 Управление процессами: идентификаторы процесса, функция fork(), совместное использование файлов. 12

22 Управление процессами: функция exit(), функции wait() и waitpid(). 12

23 Управление процессами: семейство функций exec(). 12

24 Управление процессами: изменение идентификаторов пользователя и группы, функции setuid(), setgid(), seteuid(), setegid(). 12

25 Управление процессами: интерпретируемые файлы, функция system(). 12

26 Сигналы: концепция сигналов, функция signal(), ненадежные сигналы. 12

27 Сигналы: прерванные системные вызовы, реентерабельные функции. 12

28 Сигналы: функции kill(), raise(), alarm(), pause(). 12

29 Сигналы: надежные сигналы, терминология и семантика, наборы сигналов. 12

30 Сигналы: маска сигналов процесса и функция sigprocmask(), функция sigpending(), 12

31 Сигналы: функция sigaction(). 12

32 Сигналы: функция sigsuspend(). 12

1 Обзор ОС UNIX: архитектура, вход в систему, файлы и каталоги, ввод и вывод.

1.1 Архитектура UNIX

1.1.1 Определение ОС

Операционная система (ОС) – это программное обеспечение (ПО), которое управляет аппаратными ресурсами компьютера и предоставлет среду выполнения прикладных программ (*application*). Обычно это ПО называют ядром (kernel), т.к. оно имеет относительно небольшой объем и состовляет основу системы (см. рисунок ниже).

1.1.2 Интерфес ядра

Интерфес ядра – это слой ПО, называемый системными вызовами (*system calls*). Библиотеки функций общего пользования (*library routines*) строятся на основе интерфеса системных вызовов, но прикладная программа (*application*) может свободно пользоваться как теми, так и другими. Т.е. прикладная программа может использовать как системные вызовы, так и библиотечные функции.

1.1.3 Коммандая оболочка

Коммандая оболочка (*shell*) – это особое приложение, которое предоставлет интерфес для запуска других приложений.

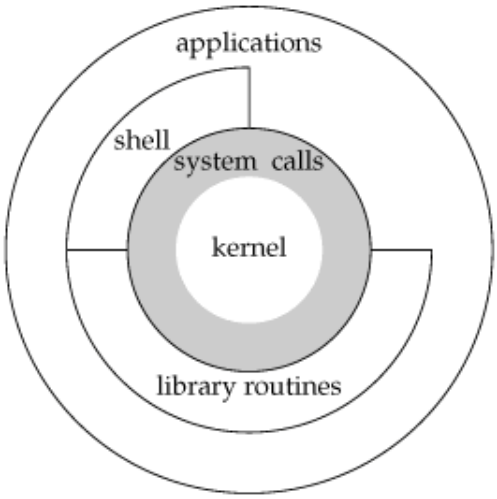
1.1.4 Обобщение

В общем, ОС – это ядро и всё остальное ПО, которое делает компьютера пригодным к использованию. В состов этого ПО входят системные утилиты (например ls, cat), прикладные программы (например GIMP), коммандые оболочки (например bash, fish), библиотеки функций общего пользования (например stdio для Си) и т.п..

1.1.5 Linux

Linux – ядро ОС GNU (или же GNU/Linux). Отдельно Linux – это ядро, называть его ОС не совсем корректно.

1.1.6 Схема, отражающая структуру UNIX



1.2 Вход в систему

При входе в систему UNIX мы вводим имя пользователя и пароль. Послеэ того система отыскивает введенное имя в файле паролей; обычно это файл `/etc/passwd`. Файл паролей содержит записи, каждая из которых состоит из семи полей, разделенных двоеточиями: имя пользователя, зашифрованный пароль, числовой идентификатор пользователя (`205`), числовой идентификатор группы (`105`), поле комментария, домашний каталог (`/home/sar`) икомандный интерпретатор (`/bin/ksh`).
Пример записи:
`sar:x:205:105:Stephen Rago:/home/sar:/bin/ksh`
Все современные системы хранят пароли в отдельном файле.
После входа в систему пользователь получает доступ к коммандной оболочке (например к bash или sh).

1.3 Файлы и каталоги

1.3.1 Файловая система

- Файловая система UNIX представляет собой иерархическую древовидную структуру, состоящую из каталогов и файлов. Начинается она с каталога, который называется корнем (`root`), а имя этого каталога представлено единственным символом - `/`.

- Каталог представляет собой файл, в котором содержатся каталожные записи. Логически каждую такую запись можно представить в виде струк-

туры,состоящей из имени файла и дополнительной информации, описывающей атрибуты файла.

- Атрибуты файла - это такие характеристики, как тип файла (обычный файл или каталог), размер файла, владелец файла, права доступа к файлу (есть ли у других пользователей доступ к файлу), время последней модификации файла.
  Для получения атрибутов файла используются функции `stat()` и `fstat()`.

1.3.2 Имя файла

Имена элементов каталога называются именами файлов. Имя файла не может содержать слеш `/` или нулевой символ `0`. Файл «точка» `.` – это текущий каталог, файл «точка-точка» `..` – родительский. При создании нового каталога в нём создаются файлы «точка» и «точка-точка». Для корневого каталога «точка-точка» – это то же самое что и «точка».
Современные ОС позволяют создавать файлы с длинной имени не менее 255 символов.

1.3.3 Путь к файлу

Последовательность одного или более имен файлов (каталог – это тоже файл), разделенных слэшами, образует строку пути к файлу. Эта строка может также начинаться с символа слэша, и тогда она называется строкой **абсолютного пути**, в противномслучае – строкой **относительного пути**. В случае относительного пути маршрут начинается от текущего каталога.
Также в пути могут присутсвовать «точка» и «точка-точка», например путь `../qwe` – это файл `qwe` в родительском каталоге.

1.3.4 Рабочий каталог

У каждого процесса имеется свой рабочий каталог, который иногда называют текущим рабочим каталогом. Это каталог, от которого отсчитываются все относительные пути, используемые в программе. Процесс может изменить свой рабочий каталог с помощью функции `chdir`.

1.3.5 Домашний каталог

Когда пользователь входит в систему, рабочим каталогом становится его домашний каталог. Домашний каталог пользователя определяется всоответствии с записью в файле паролей.

1.4 Ввод и вывод

1.4.1 Дескрипторы файлов

Дескрипторы файлов - это, как правило, небольшие целые положительные числа, используемые ядром для идентификации файлов, к которым обращается конкретный процесс. Всякий раз, когда процесс открывает существующий или создает новый файл, ядро возвращает его дескриптор, который затем используется для выполнения над файлом операций чтения или записи.

1.4.2 Стандартный ввод, стандартный вывод,Стандартный вывод сообщений об ошибках

По принятым соглашениям все командные оболочки при запуске новой программы открывают для нее три файловых дескриптора: файл стандартного ввода (`stdin`), файл стандартного вывода (`stdout`) и файл стандартного вывода сообщений об ошибках (`stderr`).
`ls > file.out` – перенаправление вывода `stdout` в файл `file.out`
`ls 2> file.err` – перенаправление вывода ошибок `stderr` в файл `file.err`

1.4.3 Небуферизованный ввод-вывод

Небуферизованный ввод-вывод осуществляется функциями `open` , `read` , `write` , `lseek` и `close` . Все эти функции работают с файловыми дескрипторами.
Заголовочный файл `<unistd. h>` , подключаемый из файла `apue.h` , и константы `STDIN_FILENO` и `STDOUT_FILENO` являются частью стандарта **POSIX.**
Константы `STDIN_FILENO` и `STDOUT_FILENO` , определенные в файле `<unistd. h>` ,устанавливают дескрипторы файлов стандартного ввода и стандартного вывода.Обычные значения этих констант - соответственно 0 и 1.

1.4.4 Стандартные функции ввода-вывода

Стандартные функции ввода-вывода предоставляют буферизованный интерфейс к функциям небуферизованного ввода-вывода. Использование стандартных функций ввода-вывода избавляет нас от необходимости задумываться о выборе оптимального размера буфера.
Другое преимущество стандартных функций ввода-вывода в том, что они значительно упрощают обработку пользовательского ввода (например `fgets()` или `printf()`).
`<stdio.h>` содержит прототипы всех стандартных функций ввода-вывода.

2 Обзор ОС UNIX: программы и процессы, обработка ошибок, идентификация пользователя.

qwe

3 Обзор ОС UNIX: сигналы, представление времени, системные вызовы и библиотечные функции.

4 Стандарты и реализации ОС UNIX: пределы ISO C, пределы POSIX, функции sysconf(), pathconf() и fpathconf(), элементарные системные типы данных.

5 Файловый ввод-вывод: дескрипторы файлов, функция open(), функция creat(), функция close().

6 Файловый ввод-вывод: Функция lseek(), функция read(), функция write()/

7 Файловый ввод-вывод: эффективность операций ввода-вывода

8 Файловый ввод-вывод: совместное использование файлов, атомарные операции, функции dup() и dup2()

9 Файловый ввод-вывод: функции sync(), fsync(), fdatasync(), fcntl(), ioctl(), /dev/fd

10 Файлы и каталоги: функции stat(), fstat(), lstat(), содержимое struct stat.

11 Файлы и каталоги: типы файлов, права доступа к файлу, функция umask().

12 Файлы и каталоги: функции chmod(), fchmod(), chown(), fchown(), lchown().

13 Файлы и каталоги: размер файла, дырки в файлах, усечение файлов, файловые системы, функции link(), unlink(), remove(), rename()

15 Файлы и каталоги: временные характеристики файлов, функция utime().

16 Файлы и каталоги: функции mkdir() и rmdir(), чтение каталогов, функции chdir(), fchdir(), getcwd().

17 Стандартная библиотека ввода-вывода: потоки и объекты FILE, стандартные потоки ввода, вывода и сообщений об ошибках, буферизация.

18 Стандартная библиотека ввода-вывода: открытие потока, чтение из потока и запись в поток, функции ввода, функции вывода.

19 Стандартная библиотека ввода-вывода: эффективность стандартных операций ввода-вывода, позиционирование в потоке.

20 Стандартная библиотека ввода-вывода: форматированный вывод, форматированный ввод, временные файлы.

21 Управление процессами: идентификаторы процесса, функция fork(), совместное использование файлов.

22 Управление процессами: функция exit(), функции wait() и waitpid().

23 Управление процессами: семейство функций exec().

24 Управление процессами: изменение идентификаторов пользователя и группы, функции setuid(), setgid(), seteuid(), setegid().