Аннотация к лекциям

В. Карпов, К. Коньков "Основы операционных систем. Практикум"

Этот практикум является приложением к курсу "Основы операционных систем".
Материалы практических занятий дополняют лекционный курс и используются для иллюстрации реализации теоретических положений на примере операционной системы UNIX. На практике рассматриваются организация процессов, различные способы их взаимодействия, устройство файловой системы, системы ввода-вывода, начала сетевого программирования. Текст, размещенный в практической части курса, содержит многочисленные ссылки на лекционный материал.

Лекция 1

Введение в курс практических занятий. Знакомство с операционной системой UNIX

Введение в курс практических занятий. Краткая история операционной системы UNIX, ее структура. Системные вызовы и библиотека libc. Понятия login и password. Упрощенное понятие об устройстве файловой системы в UNIX. Полные имена файлов. Понятие о текущей директории. Команда pwd. Относительные имена файлов. Домашняя директория пользователя и ее определение. Команда man – универсальный справочник. Команды cd – смены текущей директории и ls – просмотра состава директории. Команда cat и создание файла. Перенаправление ввода и вывода. Простейшие команды для работы с файлами – cp, rm, mkdir, mv. История редактирования файлов – ed, vi. Система Midnight Commander – mc. Встроенный mc редактор и редактор joe. Пользователь и группа. Команды chown и chgrp. Права доступа к файлу. Команда ls с опциями -al. Использование команд chmod и umask. Системные вызовы getuid и getgid. Компиляция программ на языке C в UNIX и запуск их на счет.

- Введение в курс практических занятий
- Краткая история операционной системы UNIX, ее структура
- Системные вызовы и библиотека libc
- Понятия login и password
- Вход в систему и смена пароля
- Упрощенное понятие об устройстве файловой системы в UNIX. Полные и относительные имена файлов
- Понятие о текущей директории. Команда pwd. Относительные имена файлов
- Домашняя директория пользователя и ее определение
- Команда man – универсальный справочник
- Команды cd – для смены текущей директории и ls – для просмотра состава директории
- Путешествие по структуре файловой системы
- Команда cat и создание файла. Перенаправление ввода и вывода
- Создание файла с помощью команды cat
- Простейшие команды работы с файлами – cp, rm, mkdir, mv
- История редактирования файлов – ed, vi
- Система Midnight Commander – mc. Встроенный mc редактор и редактор

joe
- Пользователь и группа. Команды chown и chgrp. Права доступа к файлу
- Команда ls с опциями -al. Использование команд chmod и umask
- Системные вызовы getuid и getgid
- Компиляция программ на языке C в UNIX и запуск их на счет

Написание, компиляция и запуск программы с использованием системных вызовов getuid() и getgid()

Лекция 2

Процессы в операционной системе UNIX

Понятие процесса в UNIX, его контекст. Идентификация процесса. Состояния процесса. Краткая диаграмма состояний. Иерархия процессов. Системные вызовы getpid(), getppid(). Создание процесса в UNIX. Системный вызов fork(). Завершение процесса. Функция exit(). Параметры функции main() в языке C. Переменные среды и аргументы командной строки. Изменение пользовательского контекста процесса. Семейство функций для системного вызова exec().

- Понятие процесса в UNIX. Его контекст
- Идентификация процесса
- Состояния процесса. Краткая диаграмма состояний
- Иерархия процессов
- Системные вызовы getppid() и getpid()
- Написание программы с использованием getpid() и getppid()
- Создание процесса в UNIX. Системный вызов fork()
- Прогон программы с fork() с одинаковой работой родителя и ребенка
- Системный вызов fork() (продолжение)
- Написание, компиляция и запуск программы с использованием вызова fork() с разным поведением процессов ребенка и родителя
- Завершение процесса. Функция exit()
- Параметры функции main() в языке C. Переменные среды и аргументы командной строки
- Написание, компиляция и запуск программы, распечатывающей аргументы командной строки и параметры среды
- Изменение пользовательского контекста процесса. Семейство функций для системного вызова exec()
- Прогон программы с использованием системного вызова exec()

Написание, компиляция и запуск программы для изменения пользовательского контекста в порожденном процессе

Лекция 3

Организация взаимодействия процессов через pipe и FIFO в UNIX

Понятие потока ввода-вывода. Представление о работе с файлами через системные вызовы и стандартную библиотеку ввода-вывода. Понятие файлового дескриптора. Открытие файла. Системный вызов open(). Системные вызовы close(), read(), write(). Понятие pipe. Системный вызов pipe(). Организация связи через pipe между процессом-родителем и процессом-потомком. Наследование файловых дескрипторов при вызовах fork() и exec(). Особенности поведения вызовов read() и write() для pip'a. Понятие FIFO. Использование системного вызова mknod() для создания FIFO. Функция mkfifo(). Особенности поведения вызова open() при открытии FIFO.

- Понятие о потоке ввода-вывода

- Понятие о работе с файлами через системные вызовы и стандартную библиотеку ввода-вывода для языка C
- Файловый дескриптор
- Открытие файла. Системный вызов open()
- Системные вызовы read(), write(), close()
- Прогон программы для записи информации в файл
- Написание, компиляция и запуск программы для чтения информации из файла
- Понятие о pipe. Системный вызов pipe()
- Прогон программы для pipe в одном процессе
- Организация связи через pipe между процессом-родителем и процессом-потомком. Наследование файловых дескрипторов при вызовах fork() и exec()
- Прогон программы для организации однонаправленной связи между родственными процессами через pipe
- Написание, компиляция и запуск программы для организации двунаправленной связи между родственными процессами через pipe
- Особенности поведения вызовов read() и write() для pip'a
- Понятие FIFO. Использование системного вызова mknod() для создания FIFO. Функция mkfifo()
- Особенности поведения вызова open() при открытии FIFO
- Прогон программы с FIFO в родственных процессах
- Написание, компиляция и запуск программы с FIFO в неродственных процессах

Неработающий пример для связи процессов на различных компьютерах

Лекция 4

Средства System V IPC. Организация работы с разделяемой памятью в UNIX. Понятие нитей исполнения (thread)

Преимущества и недостатки потокового обмена данными. Понятие System V IPC. Пространство имен. Адресация в System V IPC. Функция ftok(). Дескрипторы System V IPC. Разделяемая память в UNIX. Системные вызовы shmget(), shmat(), shmdt(). Команды ipc и ipcrm. Использование системного вызова shmctl() для освобождения ресурса. Разделяемая память и системные вызовы fork(), exec() и функция exit(). Понятие о нити исполнения (thread) в UNIX. Идентификатор нити исполнения. Функция pthread_self(). Создание и завершение thread'a. Функции pthread_create(), pthread_exit(), pthread_join(). Необходимость синхронизации процессов и нитей исполнения, использующих общую память.

- Преимущества и недостатки потокового обмена данными.
- Понятие о System V IPC
- Пространство имен. Адресация в System V IPC. Функция ftok()
- Дескрипторы System V IPC
- Разделяемая память в UNIX. Системные вызовы shmget(), shmat(), shmdt()
- Прогон программ с использованием разделяемой памяти
- Команды ipcs и ipcrm
- Использование системного вызова shmctl() для освобождения ресурса
- Разделяемая память и системные вызовы fork(), exec() и функция exit()
- Самостоятельное написание, компиляция и запуск программы для организации связи двух процессов через разделяемую память.
- Понятие о нити исполнения (thread) в UNIX. Идентификатор нити

исполнения. Функция pthread_self()
- Создание и завершение thread'a. Функции pthread_create(), pthread_exit(), pthread_join()
- Прогон программы с использованием двух нитей исполнения

Написание, компиляция и прогон программы с использованием трех нитей исполнения.

Лекция 5

Семафоры в UNIX как средство синхронизации процессов

Семафоры в UNIX. Отличие операций над UNIX-семафорами от классических операций. Создание массива семафоров или доступ к уже существующему массиву. Системный вызов semget(). Выполнение операций над семафорами. Системный вызов semop(). Удаление набора семафоров из системы с помощью команды ipcrm или системного вызова semctl(). Понятие о POSIX-семафорах.

- Семафоры в UNIX. Отличие операций над UNIX-семафорами от классических операций
- Создание массива семафоров или доступ к уже существующему. Системный вызов semget()
- Выполнение операций над семафорами. Системный вызов semop()
- Прогон примера с использованием семафора
- Изменение предыдущего примера
- Удаление набора семафоров из системы с помощью команды ipcrm или системного вызова semctl()
- Написание, компиляция и прогон программы с организацией взаимоисключения с помощью семафоров для двух процессов, взаимодействующих через разделяемую память
- Написание, компиляция и прогон программы с организацией взаимной очередности с помощью семафоров для двух процессов, взаимодействующих через pipe

Понятие о POSIX-семафорах

Лекция 6

Очереди сообщений в UNIX

Сообщения как средства связи и средства синхронизации процессов. Очереди сообщений в UNIX как составная часть System V IPC. Создание очереди сообщений или доступ к уже существующей. Системный вызов msgget(). Реализация примитивов send и receive. Системные вызовы msgsnd() и msgrcv(). Удаление очереди сообщений из системы с помощью команды ipcrm или системного вызова msgctl(). Понятие мультиплексирования. Мультиплексирование сообщений. Модель взаимодействия процессов клиент-сервер. Неравноправность клиента и сервера. Использование очередей сообщений для синхронизации работы процессов.

- Сообщения как средства связи и средства синхронизации процессов
- Очереди сообщений в UNIX как составная часть System V IPC
- Создание очереди сообщений или доступ к уже существующей. Системный вызов msgget()
- Реализация примитивов send и receive. Системные вызовы msgsnd() и msgrcv()
- Удаление очереди сообщений из системы с помощью команды ipcrm или системного вызова msgctl()

- Прогон примера с однонаправленной передачей текстовой информации
- Модификация предыдущего примера для передачи числовой информации
- Написание, компиляция и прогон программ для осуществления двусторонней связи через одну очередь сообщений
- Понятие мультиплексирования. Мультиплексирование сообщений. Модель взаимодействия процессов клиент-сервер. Неравноправность клиента и сервера
- Написание, компиляция и прогон программ клиента и сервера

Использование очередей сообщений для синхронизации работы процессов

Лекция 7

Контрольная работа проводится на практических занятиях преподавателями практикумов в середине семестра. Ее задачи содержат материал только лекционных занятий. Примерный вид задач приводится после каждой лекции. Каждая задача контрольной работы оценивается определенным количеством баллов (как правило, от 3 до 18). Всего работа содержит 7–10 задач и рассчитана на два академических часа. Мы не стали опускать в книге этот семинар, чтобы не нарушать нумерацию семинаров, которые должны следовать за соответствующими лекциями.

Лекция 8

Организация файловой системы в UNIX. Работа с файлами и директориями. Понятие о memory mapped файлах

Разделы носителя информации (partitions) в UNIX. Логическая структура файловой системы и типы файлов в UNIX. Организация файла на диске в UNIX на примере файловой системы s5fs. Понятие индексного узла (inode). Организация директорий (каталогов) в UNIX. Понятие суперблока. Операции над файлами и директориями. Системные вызовы и команды для выполнения операций над файлами и директориями. Системный вызов open(). Системный вызов close(). Операция создания файла. Системный вызов creat(). Операция чтения атрибутов файла. Системные вызовы stat(), fstat() и lstat(). Операции изменения атрибутов файла. Операции чтения из файла и записи в файл. Операция изменения указателя текущей позиции. Системный вызов lseek(). Операция добавления информации в файл. Флаг O_APPEND. Операции создания связей. Команда ln, системные вызовы link() и symlink(). Операция удаления связей и файлов. Системный вызов unlink(). Специальные функции для работы с содержимым директорий. Понятие о файлах, отображаемых в память (memory mapped файлах). Системные вызовы mmap(), munmap().

- Введение
- Разделы носителя информации (partitions) в UNIX
- Логическая структура файловой системы и типы файлов в UNIX
- Организация файла на диске в UNIX на примере файловой системы s5fs. Понятие индексного узла (inode)
- Организация директорий (каталогов) в UNIX
- Понятие суперблока
- Операции над файлами и директориями
- Системные вызовы и команды для выполнения операций над файлами и директориями
- Практическое применение команд и системных вызовов для операций над файлами

- Специальные функции для работы с содержимым директорий
- Написание, прогон и компиляция программы, анализирующей содержимое директории
- Понятие о файлах, отображаемых в память (memory mapped файлах). Системные вызовы mmap(), munmap()
- Анализ, компиляция и прогон программы для создания memory mapped файла и записи его содержимого

Изменение предыдущей программы для чтения из файла, используя его отображение в память

Лекция 9

Организация ввода-вывода в UNIX. Файлы устройств. Аппарат прерываний. Сигналы в UNIX

Понятие виртуальной файловой системы. Операции над файловыми системами. Монтирование файловых систем. Блочные, символьные устройства. Понятие драйвера. Блочные, символьные драйверы, драйверы низкого уровня. Файловый интерфейс. Аппаратные прерывания (interrupt), исключения (exception), программные прерывания (trap, software interrupt). Их обработка. Понятие сигнала. Способы возникновения сигналов и виды их обработки. Понятия группы процессов, сеанса, лидера группы, лидера сеанса, управляющего терминала сеанса. Системные вызовы getpgrp(), setpgrp(), getpgid(), setpgid(), getsid(), setsid() Системный вызов kill и команда kill(). Системный вызов signal(). Установка собственного обработчика сигнала. Восстановление предыдущей реакции на сигнал. Сигналы SIGUSR1 и SIGUSR2. Использование сигналов для синхронизации процессов. Завершение порожденного процесса. Системный вызов waitpid(). Сигнал SIGCHLD. Возникновение сигнала SIGPIPE при попытке записи в pipe или FIFO, который никто не собирается читать. Понятие о надежности сигналов. POSIX-функции для работы с сигналами.

- Понятие виртуальной файловой системы
- Операции над файловыми системами. Монтирование файловых систем
- Блочные, символьные устройства. Понятие драйвера. Блочные, символьные драйверы, драйверы низкого уровня. Файловый интерфейс
- Аппаратные прерывания (interrupt), исключения (exception), программные прерывания (trap, software interrupt). Их обработка
- Понятие сигнала. Способы возникновения сигналов и виды их обработки
- Понятия группы процессов, сеанса, лидера группы, лидера сеанса, управляющего терминала сеанса. Системные вызовы getpgrp(), setpgrp(), getpgid(), setpgid(), getsid(), setsid()
- Системный вызов kill() и команда kill
- Изучение особенностей получения терминальных сигналов текущей и фоновой группой процессов
- Изучение получения сигнала SIGHUP процессами при завершении лидера сеанса
- Системный вызов signal(). Установка собственного обработчика сигнала
- Прогон программы, игнорирующей сигнал SIGINT
- Модификация предыдущей программы для игнорирования сигналов SIGINT и SIGQUIT
- Прогон программы с пользовательской обработкой сигнала SIGINT
- Модификация предыдущей программы для пользовательской обработки сигналов SIGINT и SIGQUIT

- Восстановление предыдущей реакции на сигнал
- Сигналы SIGUSR1 и SIGUSR2. Использование сигналов для синхронизации процессов
- Завершение порожденного процесса. Системный вызов waitpid(). Сигнал SIGCHLD
- Прогон программы для иллюстрации обработки сигнала SIGCHLD
- Возникновение сигнала SIGPIPE при попытке записи в pipe или FIFO, который никто не собирается читать

Понятие о надежности сигналов. POSIX функции для работы с сигналами

Лекция 10

Семейство протоколов TCP/IP. Сокеты (sockets) в UNIX и основы работы с ними

Краткая история семейства протоколов TCP/IP. Общие сведения об архитектуре семейства протоколов TCP/IP. Уровень сетевого интерфейса. Уровень Internet. Протоколы IP, ICMP, ARP, RARP. Internet-адреса. Транспортный уровень. Протоколы TCP и UDP. UDP и TCP сокеты (sockets). Адресные пространства портов. Понятие encapsulation. Уровень приложений/программ. Использование модели клиент-сервер при изучении сетевого программирования. Организация связи между удаленными процессами с помощью датаграмм. Сетевой порядок байт. Функции htons(), htonl(), ntohs(), ntohl(). Функции преобразования IP-адресов inet_ntoa(), inet_aton(). Функция bzero(). Системные вызовы socket(), bind(), sendto(), recvfrom(). Организация связи между процессами с помощью установки логического соединения. Системные вызовы connect(), listen(), accept(). Использование интерфейса сокетов для других семейств протоколов. Файлы типа "сокет".

- Краткая история семейства протоколов TCP/IP
- Общие сведения об архитектуре семейства протоколов TCP/IP
- Уровень сетевого интерфейса
- Уровень Internet. Протоколы IP, ICMP, ARP, RARP. Internet-адреса
- Транспортный уровень. Протоколы TCP и UDP. TCP и UDP сокеты. Адресные пространства портов. Понятие encapsulation
- Уровень приложений/процессов
- Использование модели клиент-сервер для взаимодействия удаленных процессов
- Организация связи между удаленными процессами с помощью датаграмм
- Сетевой порядок байт. Функции htons(), htonl(), ntohs(), ntohl()
- Функции преобразования IP-адресов inet_ntoa(), inet_aton()
- Функция bzero()
- Создание сокета. Системный вызов socket()
- Адреса сокетов. Настройка адреса сокета. Системный вызов bind()
- Системные вызовы sendto() и recvfrom()
- Определение IP-адресов для вычислительного комплекса
- Пример программы UDP-клиента
- Пример программы UDP-сервера
- Организация связи между процессами с помощью установки логического соединения
- Установление логического соединения. Системный вызов connect()
- Пример программы TCP-клиента
- Как происходит установление виртуального соединения

Системный вызов listen()