Архитектура Linux и других Unix-подобных систем.

Linux – это одна из доминирующих ОС для рабочих станций и серверов. Используется от смартфонов (ОС Android основана на Linux) до суперкомпьютеров.

Архитектура Linux является многослойной рис. 1.

Интерфейс пользователя

Пользователи

Режим пользо-вателя

Оболочка, компиляторы и т.д.

Интерфейс системных вызовов

Стандартные библиотеки (open, close,fork,read,write и т.п.)

Режим ядра

ОС Unix в основе. Управление процессами. Памятью, вводом-выводом

Аппаратное обеспечение(центральный процессор, память, диски и т.д.)

Рис. 1 – Уровни ОС Linux

Самый нижний слой – уровень ядра. Здесь выделяют след. компоненты (рис.2):

Системные вызовы (open, close…)

Компонент I/O

Компонент управления памятью

Компо-нент управле-ния процес-сами

Терминалы, сокеты, файловые системы

Прерывания Диспетчер

Рис.2 – Структура ядра ОС Linux

1. Управление памятью
2. Управление процессами и потоками
3. Ввод-вывод (I/O).

Диспетчер осуществляет выбор программы обработки прерываний и запуск нужного драйвера. Драйвер - программный модуль, кот. обслуживает некоторое внешнее устройство (мышь, клавиатуру, принтер, жесткий диск и т.д.) по номеру прерывания. Прерывание - это сигнал, идущий от внешнего устройства (например, нажатие клавиши мыши и т.д.). Это внешнее прерывание, а бывает и внутреннее (программное), например, запись в файл- это программное прерывание.

Диспетчер повышает/понижает приоритет обработчику прерываний. Самая приоритетная программа получает наибольший квант процессорного времени.

**Уровень управления процессами/потоками**. Здесь решаются такие задачи: запуск процессов (по команде exec), запуск потоков, обработка и передача сигналов, уничтожение процессов/потоков. Здесь же находится планировщик процессов, кот. выбирает, какой процесс запустить.

**Управление памятью:**

- динамическое выделение памяти под переменные, структуры, массивы (куча);

- управление кэш-памятью – используется для подкачки команд и данных, т.е. программа работает и в то же время подкачивается след. порция команд, чтобы совместить и увеличить скорость работы программы, чтобы процессор все время не лазил в оперативную память за данными в процессе работы, а обращался за ними в кэш.

- виртуальная память. Под вирт. памятью понимается пространство памяти (оперативной и дисковой (страничный файл)). В этом смысле можно говорить, что диски являются «продолжением» ОП на внешних устройствах.

Здесь управление сводится к процессу перемещения областей (страниц) из ОП на диск и обратно, т.е. когда не хватает ОП, то на диск выгружается часть данных (в страничный файл), а на освобожденное место записываются нужные данные (это есть механизм свопинга).

**Компонент I/O:** работа с файлами (поиск, копирование, запись, перемещение, чтение и т.д. – за это отвечает файловая система), терминалами и сокетами.

Работа с консолью – это работа с клавиатурой и монитором.

Сокеты позволяют работать в сети. Сокет – это порт (ячейка), через кот. взаимодействуют машины в сети. Одна из них выступает как клиент, другая- как сервер. Взаимодействие осуществляется на основании протоколов (TCP, IP). Протокол – это набор инструкций и форматов сообщений, которые посылаются между сервером и клиентом и назад.

С этими 3мя частями (терминалами, сокетами, файлами) работают драйверы: символьных устройств, сетевых устройств и блочных устройств соответственно.

След. уровень за ядром – **Shell** – оболочка (командная строка Linux). На основе ее можно создавать скрипты наподобие батников в DOS. Называется bash, но могут использоваться и другие оболочки - ksh, csh и т.д.

Когда оболочка запускается, она инициализируется и выводит на экран символ приглашения к вводу, ждет, когда пользователь введет команду. Если пользователь вводит командную строку, оболочка извлекает из нее первое слово, предполагая, что это имя запускаемой программы, ищет эту программу и запускает ее, если находит. При этом работа оболочки приостанавливается на время запущенной программы.

Оболочка представляет собой обычную пользовательскую программу, кот. способна читать с клавиатуры, выводить на монитор, а также запускать другие программы.

Итак Shell выполняет след .действия (рис.3):

Выполнение команды

Проверка правильности задания пара-метров

Лексичес-кий анализ

Рис. 3 – Структура Shell

1. Лексический анализ заключается в выделении лексем – структурных единиц командной строки, таких как код команды, операнды, флаги и т.д:

cp f1 f2 - В этом примере 3 лексемы: cp, f1,f2.

1. Проверка правильности заданных параметров. Т.е. правильно ли заданы файлы, флажки, каналы и т.д.
2. Собственно выполнение команды.