Operating Systems

Introduction

Me

September 6, 2016

Структура курса

- Лекции
  - посещаемость не проверяю;
  - контрольных не устраиваю;
  - возможность задавать вопросы - не задаете вопросы, значит все понятно.
- Практики и домашние задания
  - 5 домашних заданий - пишем простую ОС;
  - домашние задания не независимы - пропустили одну будут проблемы со следующим;
  - практики отведены для обсуждения домашних заданий (поэтому последняя пара).

Темы лекций 1/3

- Прерывания и внешние устройства
  - тема привязана к первому домашнему заданию;
- Управление памятью
  - алгоритмы аллокации памяти (Buddy, SLOB/SLAB);
  - защита памяти, сегментация, страничная организация и page fault;
  - интерфейс ОС для процессов (как работает malloc?);
  - NUMA?
- Многозадачность, планирование и конкуретность:
  - понятие процесса и потока исполнения;
  - диспетчеризация и планирование потоков;
  - многоядерность, когерентность кешей и барьеры памяти;
  - примитивы синхронизации потоков;

Темы лекций 2/3

- Межпроцессное взаимодействие
  - существующие примитивы IPC, без рассмотрения реализации;
- Файловые системы
  - иерархические ФС: файлы, каталоги, символьные и "жесткие" ссылки;
  - классические файловые системы: FAT и ext2;
  - индексные структуры данных: B-деревья и модификации, LSM;
  - транзакции;
  - устройство диска, избыточность и помехоустойчивое кодирование?

Темы лекций 3/3

- Краткое введениме в распределенные системы:
  - консенсус, разрешимые и неразрешимые задачи, протоколы консенсуса.
- Краткое введение в виртуализацию:
  - история вопроса, назначение;
  - критерии эффективной виртуализации;
  - QEMU и динамическая трансляция;
  - аппаратная поддержка виртуализации.

Темы домашних заданий

- прерывания (3 основных + 2 дополнительных)
- аллокация (3 + 2)
- потоки (3 + 2)
- ФС (2 + 0)
- системные вызовы (3 + 0)

Условия выполнения заданий

- Правила выполнения домашних заданий:
  - задания сдаются в виде ссылки на git репозиторий;
  - для каждого задания есть жесткий дедлайн (опоздали на 1 минуту или на 1 час не важно);
  - чтобы задание было засчитано оно должно компилироваться и работать *у меня*;
  - по вашему решению будут задаваться вопросы.
- Критерии оценки:
  - "отлично" - 11 баллов;
  - "хорошо" - 8 баллов;
  - "удовлетворительно" - 5 баллов.

Необходимые утилиты

- QEMU (виртуальная машина, далее просто ВМ)
  - тестировать ОС быстрее в ВМ;
  - к QEMU можно подключить отладчик;
- GNU ld, gcc или clang
  - владельцы Windows могут использовать ВМ с Linux;
  - владельцы Mac OS имееют две возможности:
    - собрать toolchain из исходников;
    - использовать ВМ с Linux;
- задание №0 - настроить окружение, собрать пример и запустить;

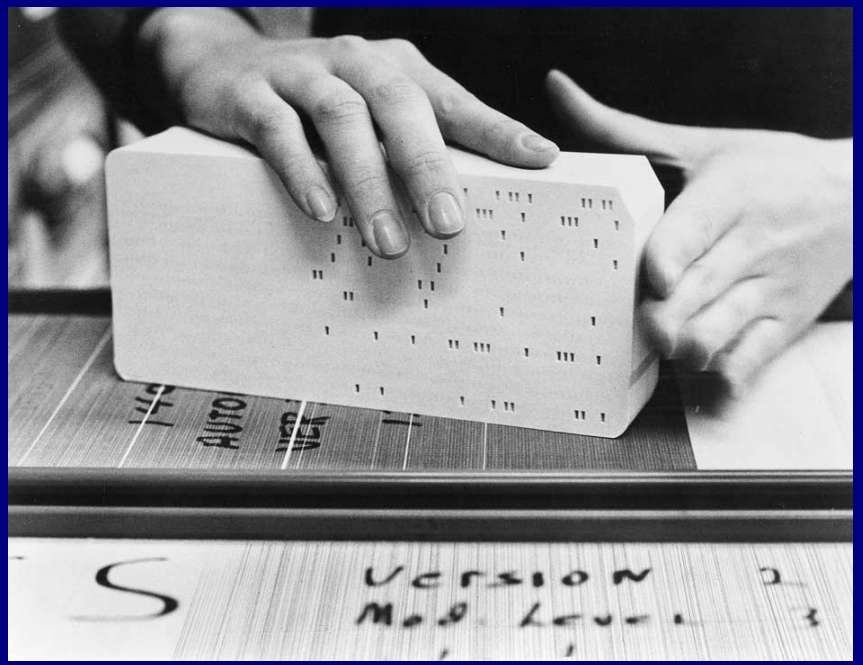
Open Shop

- IBM 701 - первый коммерческий компьютер от IBM;
- одна машина стоила $2 млн. (50-ые)[1];
- месячная аренда 15 000;
- из 15 минут работы - 10 настройка, и 5 вычисления;

[1] http://motherboard.vice.com/blog/the-rise-of-ibm-s-immortal-mainframe

Пакетная обработка

- Человек - слабое звено, удалить человека!
- Машина сама может планировать задачи;
- основной критерий - уменьшение времени простоя;
- отдаете программу оператору, а потом пару дней ждете результата.



Многозадачность (Multiprogramming)

- Больше оперативной памяти - больше можно потратить на ерунду;
- энергонезависимая память произвольного доступа - нет строгой необходимости обрабатывать все последовательно;
- аппаратные прерывания позволяют выполнять ввод/вывод паралельно с расчетами и симулировать конкуретное исполнение (но об этом далее);
- примеры: B5000 MCP (60-ые, первая известная мне система с виртуальной памятью), Exec II (60-ые, позволяла засылать задачи удаленно, по телефону);

Разделение времени

- Когда компьютеры становятся мощнее, инженеры начинают распускать руки;
- компьютерное время становится дешевле
  - ждать пару дней завершения задачи на 5 мину - не целесообразно;
  - хочется больше интерактивности - все пользователи получают свою долю;
- примеры: CTSS (считается первой системой с разделением времени), Multics (академическое достижение и инженерный провал);

Unix

- После провала Multics инженеры из Bell Labs стали искать альтернативы:
  - Multics была слишком сложной поэтому проект и провалился;
  - создатели Unix руководствовались простотой и проект взлетел;
  - есть мнение, что популярность Unix стала препядствием для ее дальнейшего развития;

Concurrent Programming

- В начале 60-ых конкурентное исполнение уже существовало, а понимания как с ним работать правильно нет:
  - первый систематический подход к созданию ОС был продемонстрирован Дейкстрой в его "THE multiprogramming system";
  - Дейкстра же предложил концепцию семафора, как примитива синхронизации и взаимодействия;
  - другим пример систематичного подхода к дизайну ОС стала RC 4000.

Personal Computers

- 20 лет работы насмарку, ОС системы вновь стали:
  - однопользовательскими;
  - однопроцессными;
  - простыми;
- зато появился графический интерфейс и компьютерная мышь.

Q&A