Лекции lite

1 марта 2019 г.

Содержание

Лекция 1

1.1 Введение

- **Структура лекций:**

  Будет 2 курса: классический и более современный. Мы не будем писать, к примеру, свои реализации ядер, а будем писать софт под ОС.

- **Структура практик:**

  Bash, последняя лаба под Windows, т.п.

1.2 Эволюция ОС (этапы)

Когда появляются операционные системы?

1.2.1 Диспетчеры

- Архитектура фон Неймана (**RAM, input, output, storage**), данные и код никак не разделены в памяти
- Разумная идея - создать в **RAM** область памяти часто используемого кода (подпрограммы). Появилась **линковка** (вызов подпрограммы и возврат в код). Осуществлялась посредством набора конкретных указателей.
- Идея буфера (разделение больших объемов данных на блоки) и параллельности.
- **SPOOL (-ing)** : появляется контроллер связанный с **RAM, CPU, storage**.
- **interrupt (INT)** - сигнал контроллера к **CPU**, заставляющий его вызвать обработчик прерывания.
- Раньше все было ограничено физической оперативной памятью. Проблема с большим кодом, откачка, перевычисление адресов. Теперь мы не ограничены виртуальной памятью.
- Пакеты (модификация **storage**).

1.2.2 Мультипрограммы

- **N.B.** Всегда псевдо-параллельность.
- Концепция разделения времени.

  Передача управления другому процессу, добавление таймера, умеющего создавать прерывания.
- Сохранение контекста (состояние регистров) + атомарность.
- Проблема с разделением памяти между процессами (раньше однопрограммность подразумевала начало с одного и того же места в памяти).
- Появление концепции виртуальной памяти (виртуальные адреса отсчитываются от нуля).
- Проблема: запрос "не своей"памяти. Появление привилегированного режима. Прерывание SYSVIOLATION.
- Концепция библиотек (хранящихся в пакетах?) : не хотим хранить в **RAM**, а подгружать из **storage** когда нужно.
- Регуляция доступа к библиотекам (так появилось имя файла?, карта размещения, таблицы соответствия — control lists)
- Общая концепция : виртуальная машина (некоторый процесс оказывается изолированным, абстрагированным). С позиции процесса он существует один (но хотя он "не существует когда работают другие процессы). Из-за этого костыли, потому что процессы не знают друг о друге.
- 1963 - **MCP (Main Control Program)** - назвали операционной системой. Мультипрограммирование, мультиядерность ?, концепции виртуальной памяти и виртуальной машины.

1.2.3 Сети

- Кругом суперкомпьютеры, продажа машинного времени. 1 час — 120$.
- Способ модуляции данных, аналогово-цифровое преобразование и обратно, передача данных.
- Появление терминалов. Концепция:
  - **Идентификация** — получил карту.
  - **Аутентификация** — вставил карту и система узнала.
  - **Авторизация** — предоставление доступа.
- Появление компаний, специализирующихся на расчетах.
- Модем ?

1.2.4 Мобильные ОС

- До этого каждая ОС создавалась под конкретный компьютер. Быть программистом было непросто, так как очень часто появлялись новые компьютеры, под которые нужно было почти с нуля писать ОС.
- **UNICS, Bell Laboratories, MULTIX?**.
- Курица и яйцо : что было раньше : ОС или высокоуровневые языки?
- Томпсон, Керниган, Ричи последовательными шагами решают эту проблему.
  - 1 edition - assembler. Создают язык B. Потом переписывается ядро на B.
  - 2 edition - B. Создают Си. Создают компилятор под Си. Переписывают ядро на Си. Теперь имеют универсальную ОС.
  - Последняя редакция - седьмая.
  - Из-за разногласий, **UNICS** → **UNIX**.
- История **BSD** (Университет Беркли взял исходный код **UNIX** и разработал ОС).
- Монтекки и Капуллети - Университеты Беркли и Стэнфорд
- Стэнфорд создает **SUN** в противовес. **Solaris**.
- **UNIX** коммерциализируется
- **NextStep**(startup from **BSD**) → **Darvin?** → **MacOS**.
- Проприетарные ОС - тормозят развитие.
- Ричард Столлман - свободный код. Если ты используешь свободный код, то твой код тоже должен быть свободным. **GCC**.
- Танненбаум - **MINIX**. Торвальдс (аспирант) высказывается за модульность, споры в сети, Линус "на слабо"пишет ядро.
- Линус стал участвовать в проекте **GNU/Linux?** и потребовал чтобы его имя было частью имени проекта и чтобы он "maintain - ил"ядро.