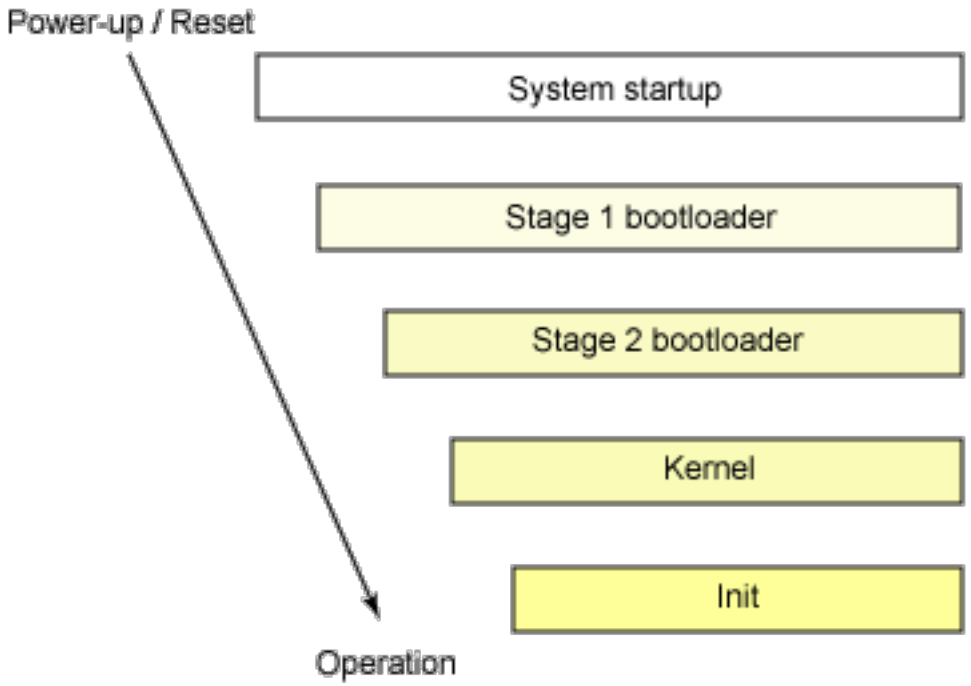
Конструирование ядра операционных систем (III)

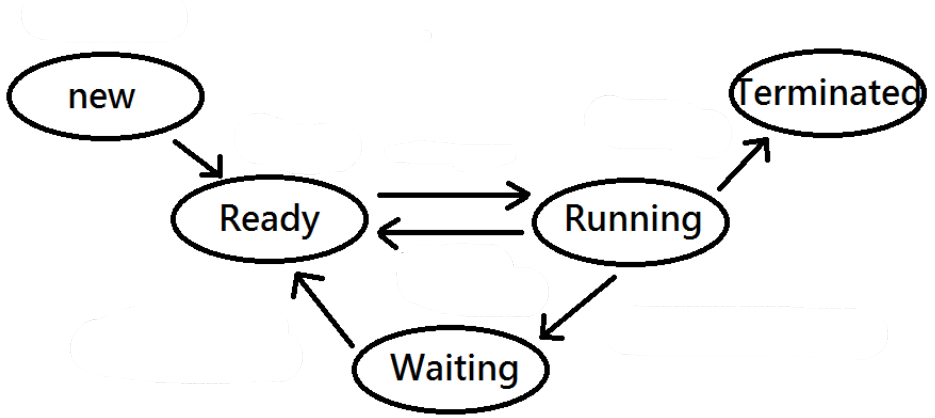
Организация процессов

План

- Создание и запуск процессов
- Библиотечные и системные интерфейсы
- Планирование процессов
- Проблемы изоляции и производительности
- Практическая часть лабораторной работы

Запуск процессов в ОС





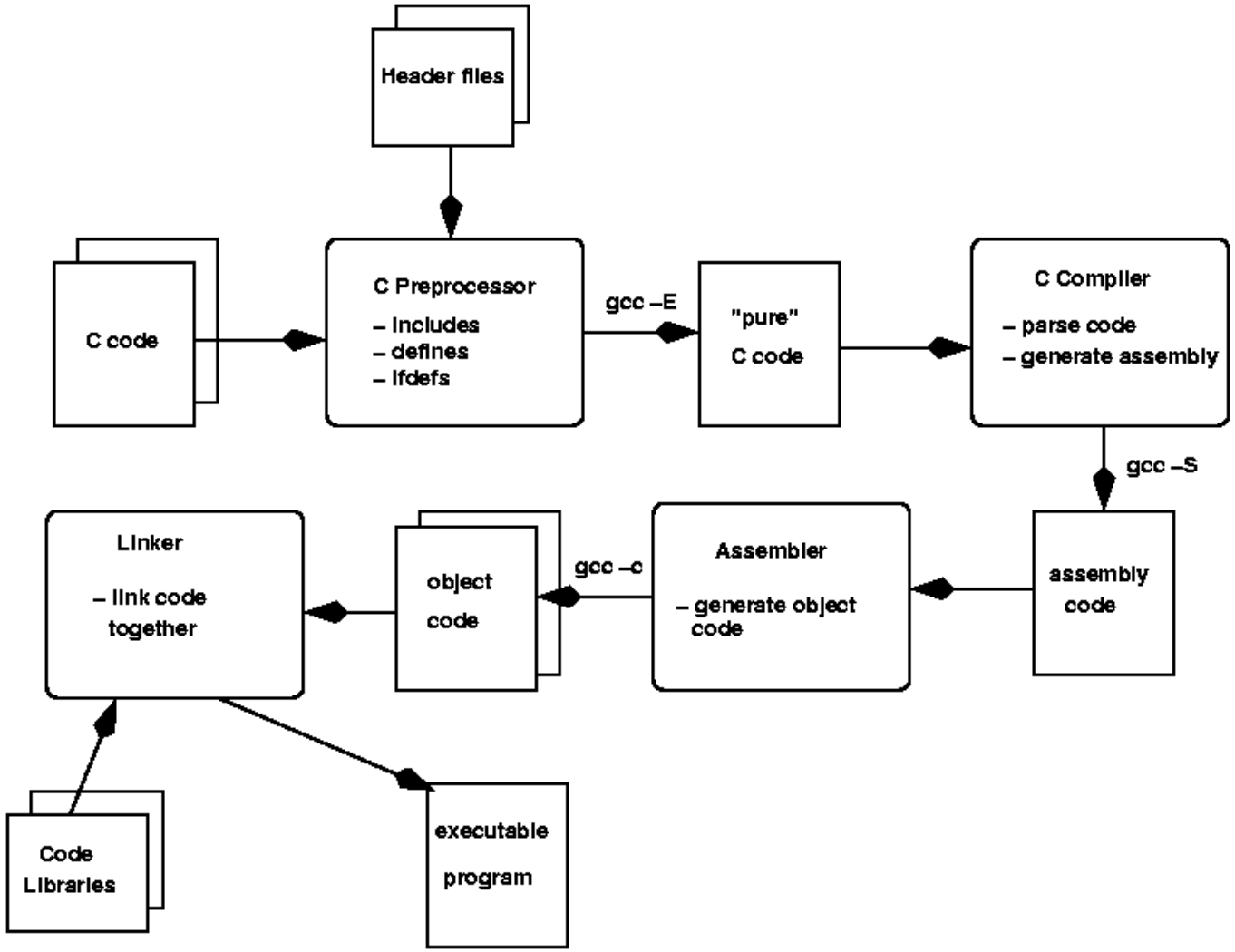
- Для UEFI загрузчик (bootloader) нередко один.
- В ядре могут быть дополнительные сервисы, например, отладчик уровня ядра/монитор.
- Init стадия в ОС общего назначения чаще всего представлена исполняемым файлом в пользовательском пространстве, в рамках которого запускается shell или GUI.

Элементы структуры процесса

- PID, PPID, тип, состояние, контекст
- Адресное пространство (e.g. vmap)
- Owner и Security Credentials
- Приоритеты (пользователя, планировщика…)
- Флаги (режим работы, отладка, ЭЦП…)
- Статистика и отладка (e.g. dtrace)
- POSIX группа, обработчики сигналов, дескрипторы файлов

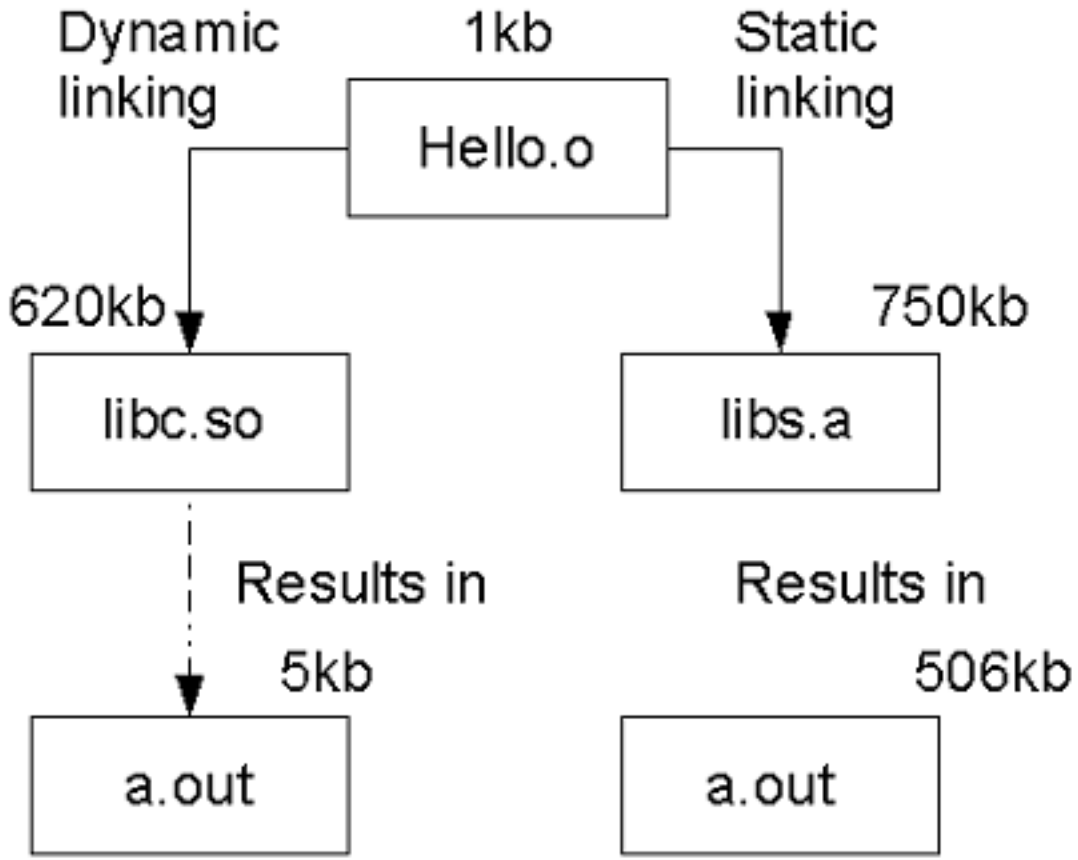
macOS/BSD: **proc**, Linux: **task_struct**, JOS: **Env**.

Генерация исполняемого файла



* В современных компиляторах трансформация С кода в машинный производится в два шага:
— Генерация и оптимизация IR
— Генерация машинного кода
При этом генерация ассемблерных мнемоник кода опциональна.

Виды линковки



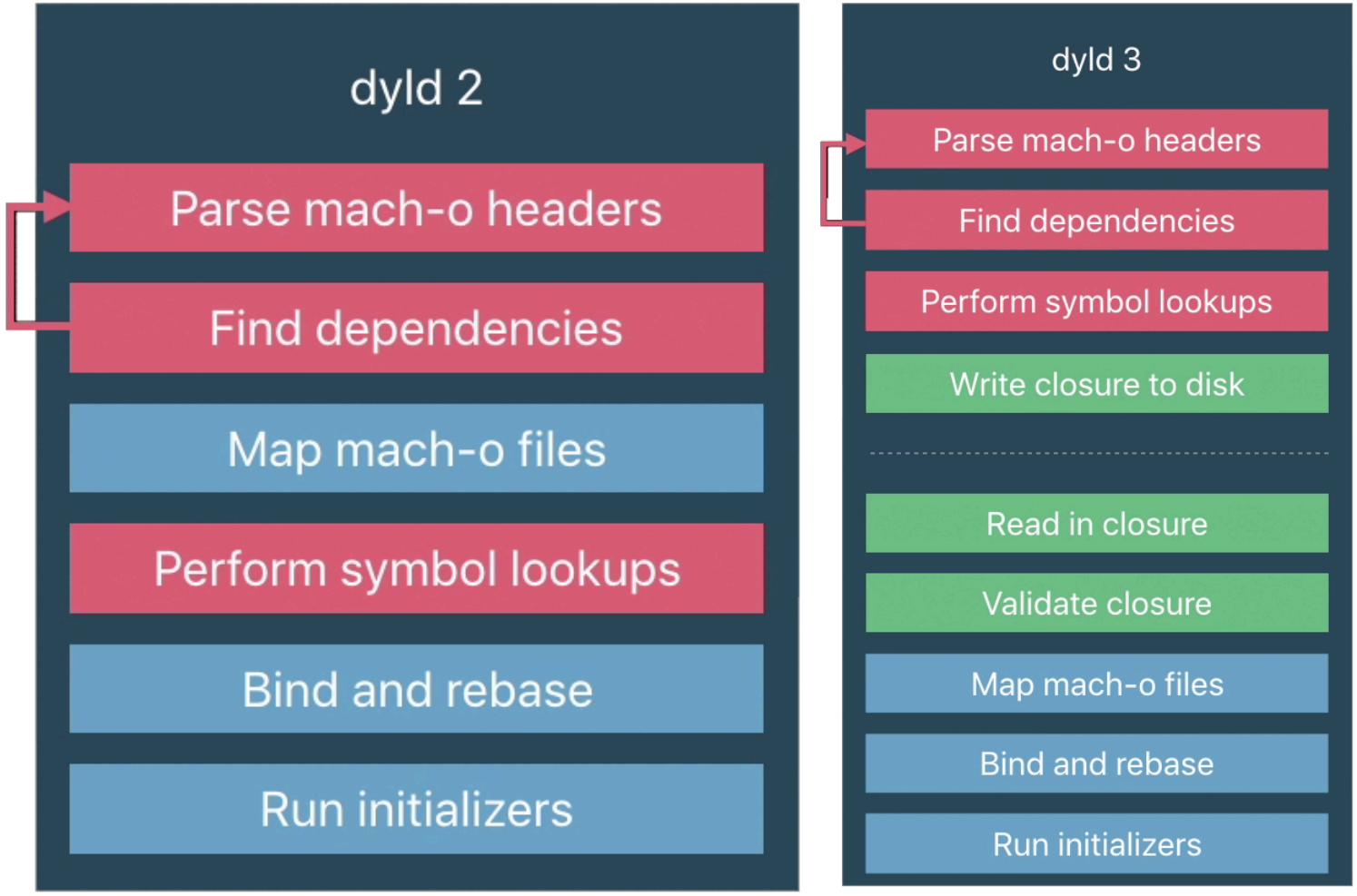
Преимущества динамической линковки:

- Меньшее потребление RAM
- Меньшее потребление ROM
- Отдельный ABI интерфейс с ОС
- Потенциально быстрее сборка
- Упрощение поставки обновлений
- Опциональные интерфейсы

Недостатки:

- Обязательна поддержка ОС
- Проблемы версионирования библиотек
- Потенциально сложнее модель памяти
- Проблемы пространства имён

Динамический линквощик



Памятка:

- Кэш библиотек (например, в macOS это dyld_shared_cache)
- ASLR (рандомизация адресного пространства) и модель памяти
- Цифровая подпись
- Права и разрешения (chmod, suid, entitlemenents...)

Планировщики процессов

- Long-term scheduling (e.g. cron, launchd)
- Medium-term scheduling (e.g. swapping, page out)
- Short-term scheduling (CPU scheduling)
- Dispatcher (e.g. Grand Central Dispatch)

В JOS реализован исключительно short-term scheduling по алгоритму Round-robin.

Алгоритмы планирования

- Вытесняющие/невытесняющие
- Периодические/непериодические
- Со статическими/динамическим приоритетами
- С онлайн/оффлайн расписанием

- Fairness (процесс выбран справедливо)
- Liveness (процесс выполняет задачу)

Ref: Lab3-RTOS.pdf

Алгоритмы планирования

- Вытесняющие/невытесняющие
- Периодические/непериодические
- Со статическими/динамическим приоритетами
- С онлайн/оффлайн расписанием

- Fairness (процесс выбран справедливо)
- Liveness (процесс выполняет задачу)

Ref: Lab3-RTOS.pdf

Изоляция процессов

У разных процессов разные:

- Критичные данные (sensitive data)
- Привилегии доступа
- Степень влияния на ОС в целом

Примеры технических защит:

- Virtual Memory
- Stack Canary
- ASLR slide
- Authenticated Pointers
- Memory/TLB/Cache Flushing
- Obfuscation / Encryption
- Code Signing/Entitlements
- W^X
- NX/DEP (Stack/Data)
- Control Flow Integrity
- (GNU) RELRO
- …

Спасибо за внимание!

Вопросы?