РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук
Кафедра прикладной информатики и теории вероятностей

Презентация №5

Система Инициализации OS X / macOS

Студент: Эйвази Мани

Группа: НПИбд-03-24

Студенческий билет №: 1032245107

Система Инициализации OS X / macOS

Эволюция и архитектура инициализации ОС

Историческая Справка: От SystemStarter к launchd

1 До Mac OS X 10.4 Tiger

*Ранние версии OS X использовали **SystemStarter** для системных демонов и **mach_init** для пользовательских процессов. Эти системы были раздельными и менее интегрированными, унаследовав черты UNIX.*

- ***SystemStarter:*** *Скриптовый запуск сервисов при загрузке (похож на SysVinit).*
- ***mach_init:*** *Базовая инициализация, связанная с микроядром Mach.*

2 Mac OS X 10.4 Tiger (2005)

*Представлена **launchd** как унифицированная система инициализации. Она заменила SystemStarter, inetd, cron и интегрировалась с Mach и CoreFoundation.*

- ***Унификация:*** *Одна система для управления всеми процессами.*
- ***Простота:*** *Конфигурация через .plist файлы.*
- ***Эффективность:*** *Запуск процессов по требованию для снижения потребления ресурсов.*

3 Последующие Версии macOS

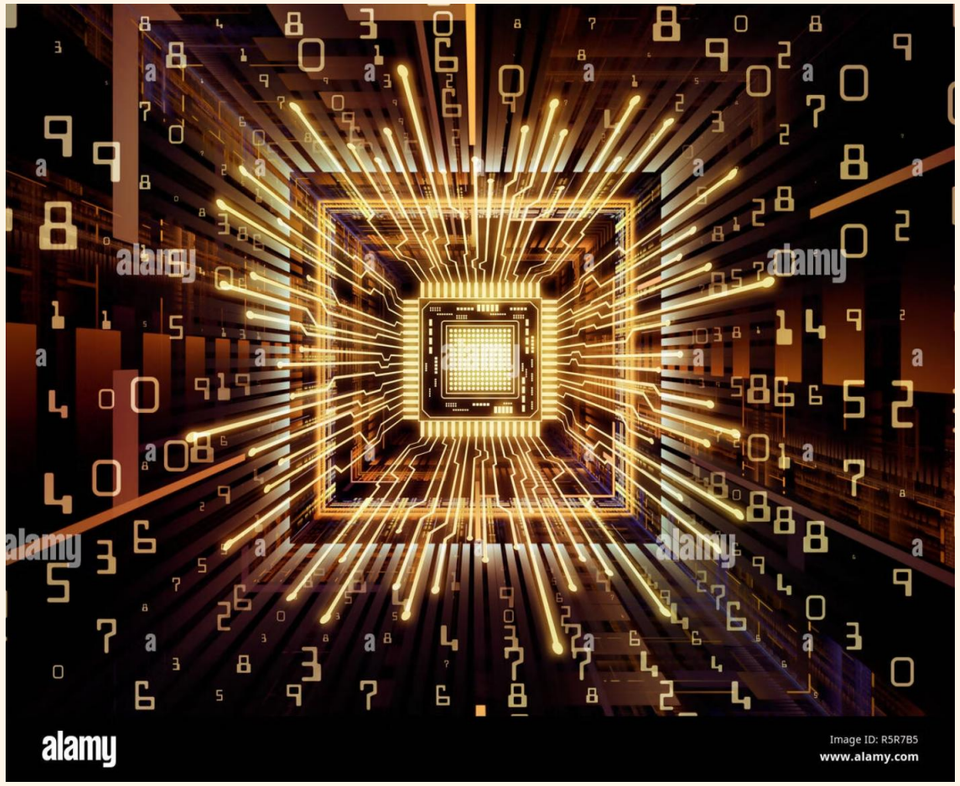
launchd продолжает развиваться как центральный элемент системы. В новых версиях macOS улучшаются стабильность, производительность и добавляются функции, но его фундаментальная архитектура неизменна.

- ***macOS Sierra (10.12):*** *Улучшения безопасности и изоляции процессов.*
- ***Современные macOS:*** *Интеграция с функциями безопасности, например, System Integrity Protection (SIP).*

Переход к launchd стал значительным шагом в модернизации системы инициализации Apple, сделав её более гибкой, эффективной и соответствующей уникальным требованиям macOS.

launchd как PID 1: Первый Процесс Системы

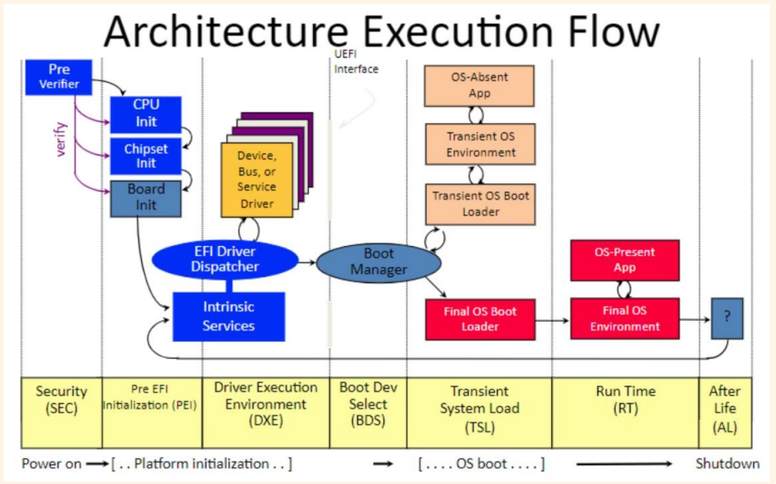
*В UNIX-подобных ОС, включая macOS, **PID 1** — это первый процесс, запускаемый ядром после инициализации.*



Запуск Ядром

*После загрузки и базовой настройки, ядро macOS передает управление **launchd**, делая его **корневым процессом** системы.*

- ***Ядро:*** *Управляет низкоуровневыми операциями, памятью и процессорным временем.*
- ***Запуск:*** *После инициализации ядро запускает **launchd** и ждет его работы.*



Роль PID 1

- ***Инициализация Системы:*** *Запускает все системные демоны и службы.*
- ***Управление Процессами:*** *Является родительским для всех процессов; "усыновляет" дочерние процессы при завершении родителя.*
- ***Завершение Системы:*** *Отвечает за корректное завершение работы, отправляя сигналы всем процессам.*

*launchd** незаменим для стабильности и работы macOS, действуя как "дирижёр" для всех программ и служб.*

*Понимание роли **launchd** как PID 1 критически важно для диагностики проблем с загрузкой и управления системными службами macOS.*

Архитектура launchd: Системный и Пользовательский Уровни

*Архитектура **launchd** в macOS разделена на системный и пользовательский уровни для гибкого и безопасного управления процессами.*

Системный Уровень: LaunchDaemons

- ***Расположение:*** `/Library/LaunchDaemons` *и* `/System/Library/LaunchDaemons`.
- ***Права:*** *Запускаются от имени* ***root****, работают независимо от входа пользователя. Предоставляют базовые системные службы.*
- ***Примеры:*** *Службы сетевого доступа, логирования, управление файловой системой.*
- ***Назначение:*** *Фундаментальная функциональность ОС для всех пользователей.*

Пользовательский Уровень: LaunchAgents

- ***Расположение:*** `/Library/LaunchAgents` *и* `~/Library/LaunchAgents`.
- ***Права:*** *Запускаются от имени текущего пользователя, работают только при его входе. Обеспечивают пользовательские службы.*
- ***Примеры:*** *Обновления приложений, синхронизация облачных сервисов, пользовательские настройки.*
- ***Назначение:*** *Функциональность, специфичная для пользователя или его сессии.*

Такое разделение обеспечивает эффективное управление ресурсами, изоляцию процессов и повышает общую безопасность системы.

Формат Конфигурации: Property List (.plist)

launchd использует .plist файлы для определения параметров служб. Это структурированные XML-документы.

Пример .plist файла

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
              <key>Label</key>
              <string>com.example.myservice</string>
              <key>ProgramArguments</key>
              <array>

              <string>/usr/local/bin/myservice</string>
                            <string>--config</string>

              <string>/etc/myservice.conf</string>
              </array>
              <key>RunAtLoad</key>
              <true/>
              <key>KeepAlive</key>
              <true/>
              <key>StartCalendarInterval</key>
              <dict>
                            <key>Hour</key>
                            <integer>3</integer>
                            <key>Minute</key>
                            <integer>0</integer>
              </dict>
</dict>
</plist>
```

Ключевые Параметры

1. ### Label

   Уникальный обязательный идентификатор службы (например, `com.example.myservice`).

2. ### ProgramArguments

   Массив строк: путь к исполняемому файлу и его аргументы. Первая строка — путь к программе.

3. ### RunAtLoad

   `true` *для запуска при загрузке,* `false` *для запуска по требованию.*

4. ### KeepAlive

   `true` *для автоматического перезапуска службы при сбое/завершении.*

5. ### StartCalendarInterval

   Словарь для запуска службы по расписанию (аналог cron). Указывает час, минуту, день и т.д.

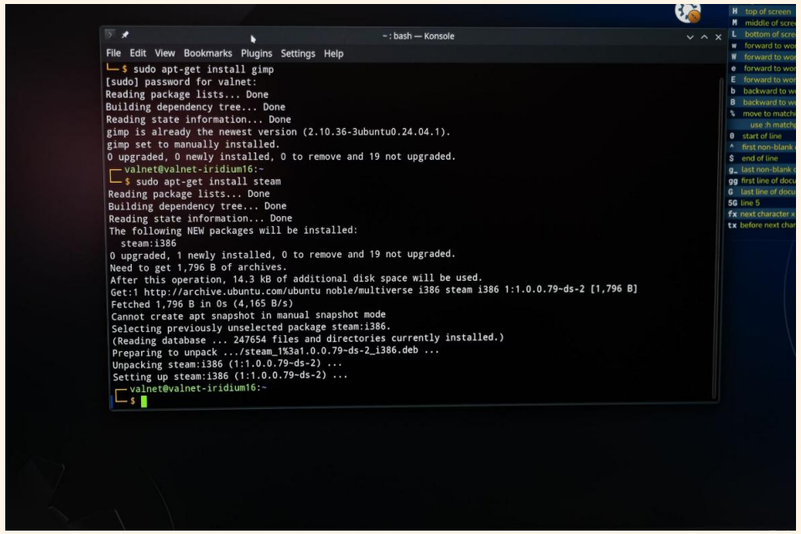
6. ### Other Keys

   Множество других параметров, таких как `StandardOutPath`, `UserName`, `Sockets`, *для тонкой настройки.*

Правильное использование этих параметров позволяет создавать надёжные и эффективные службы в macOS.

Управление Службами: Команда launchctl

launchctl *— основной инструмент командной строки для взаимодействия с* ***launchd*** *и управления его службами.*



Основные Команды

- ***launchctl load <path/to/plist>:*** *Загружает службу из .plist файла.*
- ***launchctl unload <path/to/plist>:*** *Выгружает службу.*
- ***launchctl list:*** *Отображает список всех загруженных служб со статусом и PID.*
- ***launchctl enable <label>:*** *Включает службу для автозапуска при загрузке системы или входе пользователя.*
- ***launchctl disable <label>:*** *Отключает службу, предотвращая её запуск.*
- ***launchctl start <label>:*** *Запускает загруженную службу по метке (Label).*
- ***launchctl stop <label>:*** *Останавливает запущенную службу.*
- ***launchctl kill <signal> <label>:*** *Отправляет сигнал процессу, управляемому launchd.*

Знание ***launchctl*** *является ключевым для тонкой настройки и отладки macOS.*

Примеры Использования

```
# Загрузить агент пользователя
launchctl load ~/Library/LaunchAgents/com.example.myapp.plist

# Выгрузить системный демон
sudo launchctl unload
/Library/LaunchDaemons/com.example.myservice.plist

# Посмотреть список всех служб
launchctl list

# Включить системный демон (для автозапуска)
sudo launchctl enable system/com.example.myservice

# Запустить пользовательский агент
launchctl start com.example.myapp
```

Для системных демонов (`/Library/LaunchDaemons`*) требуется* `sudo` *и префикс* `system/`*. Для пользовательских агентов (*`~/Library/LaunchAgents`*)* `sudo` *не нужен.*

Механизм Запуска По Требованию (On-Demand) и Socket Activation

***launchd** отличается от традиционных систем инициализации, запуская процессы только по необходимости, используя механизмы запуска по требованию и активации по сокету.*

Запуск По Требованию

launchd запускает службы только при наступлении определённых событий, экономя системные ресурсы (CPU, RAM).

- ***Файловые Пути (WatchPaths):** Запуск при изменении файла.*
- ***Запросы IPC:** Активация при межпроцессном взаимодействии.*
- ***Условия Окружения:** Запуск при выполнении условий.*

Socket Activation

Особая форма запуска по требованию для сетевых служб. launchd "слушает" сетевой порт и запускает службу при входящем соединении, передавая ей открытый сокет.

- ***Экономия Ресурсов:** Служба запускается только по необходимости.*
- ***Улучшенная Безопасность:** launchd может фильтровать соединения.*
- ***Простота Настройки:** Определяется в .plist с ключом* `Sockets`.

Пример: SSH-сервер с активацией по сокету запускается только при подключении к порту 22, что делает macOS более отзывчивой и энергоэффективной.

Размещение Файлов Конфигурации и Приоритеты Загрузки

***launchd** .plist файлы имеют разные расположения с собственным назначением и приоритетом.*

1	/System/Library/LaunchDaemons *Системные демоны Apple, защищены SIP. Не изменяются. Загружаются первыми при старте системы.*
2	/System/Library/LaunchAgents *Системные агенты Apple, защищены SIP. Запускаются при входе любого пользователя.*
3	/Library/LaunchDaemons *Демоны сторонних приложений/администраторов. Запускаются при старте системы (после системных). Для всех пользователей.*
4	/Library/LaunchAgents *Агенты сторонних приложений/администраторов. Запускаются при входе любого пользователя (после системных).*
5	~/Library/LaunchAgents *Агенты для конкретного пользователя. Запускаются при входе этого пользователя. Могут быть изменены пользователем.*

Приоритет загрузки идёт сверху вниз. Если метки совпадают, загружается файл с более высоким приоритетом (например, из `/System/Library`*).*

> *Используйте* `/Library/LaunchDaemons` *или* `/Library/LaunchAgents` *для общесистемных служб и* `~/Library/LaunchAgents` *для пользовательских. Никогда не изменяйте содержимое* `/System/Library`*.*

Отличия launchd от Классических Систем

***launchd** значительно отличается от традиционных UNIX-подобных систем инициализации, объединяя функции нескольких компонентов.*

Традиционные UNIX-системы

→ init (SysVinit, systemd)

Основная система инициализации. SysVinit — последовательные скрипты; systemd — юниты, параллельный запуск. Оба запускают большинство сервисов при загрузке.

→ inetd/xinetd

*Суперсервер для сетевых служб, запускающий демоны по требованию. Менее гибок, чем **launchd**.*

→ cron

Планировщик задач, выполняющий команды по расписанию. Отдельный демон с собственной конфигурацией.

Преимущества launchd

Унификация

Объединяет функции init, inetd и cron в едином механизме управления процессами.

Запуск По Требованию

Запускает процессы по запросу, а также по событиям (файл, IPC, таймер).

Гибкость Конфигурации

Использует .plist файлы для простой настройки поведения служб.

Экономия Ресурсов

Запуск процессов по необходимости снижает нагрузку и повышает отзывчивость.

Надёжность

Встроенный механизм `KeepAlive` обеспечивает автоматический перезапуск служб.

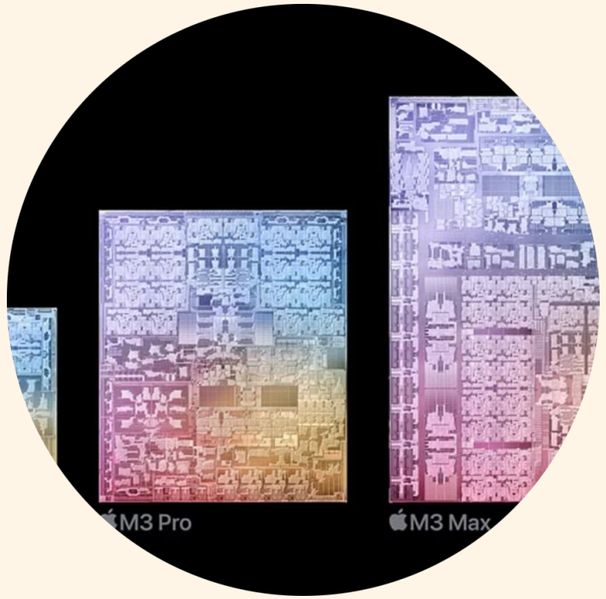
Интеграция с macOS

Глубокая интеграция с фреймворками macOS для создания эффективных и безопасных служб.

*Хотя **systemd** в Linux также стремится к унификации, **launchd** является зрелым и специализированным решением для экосистемы Apple.*

Заключение и Итоги: Современное Состояние launchd в macOS

launchd *остаётся ключевым элементом macOS, обеспечивая её стабильность и эффективность. Его архитектура постоянно адаптируется к новым требованиям.*



Эффективность и Производительность

Запуск по требованию и активация по сокету позволяют macOS экономить ресурсы, повышая общую производительность, особенно на мобильных устройствах.

Безопасность

Разделение демонов, агентов и интеграция с SIP повышают безопасность системы, защищая критические компоненты от несанкционированных изменений.

Интеграция и Гибкость

Унифицированный подход к управлению процессами упрощает администрирование macOS. Файлы .plist обеспечивают гибкую настройку.

Глубокое понимание ***launchd*** *— ключевой навык для IT-администраторов и инженеров. Оно позволяет эффективно управлять процессами macOS и разрабатывать надёжные интегрированные решения.*

launchd продолжит эволюционировать с развитием macOS, адаптируясь к новым требованиям безопасности, производительности и пользовательского опыта, сохраняя свою центральную роль.

Спасибо за внимание