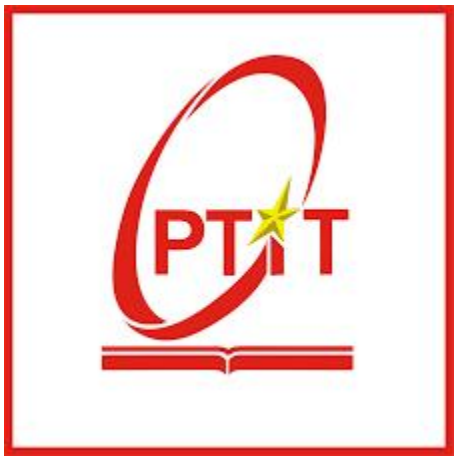
BỘ THÔNG TIN VÀ TRUYỀN THÔNG

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

KHOA CÔNG NGHỆ THÔNG TIN 1



BÁO CÁO BÀI TẬP LỚN

MÔN: HỆ ĐIỀU HÀNH

Đề tài: Tìm hiểu chung về hai họ hệ điều hành MacOS và iOS.

Giảng viên hướng dẫn: Đỗ Tiến Dũng

Nhóm thực hiện: Nhóm 12

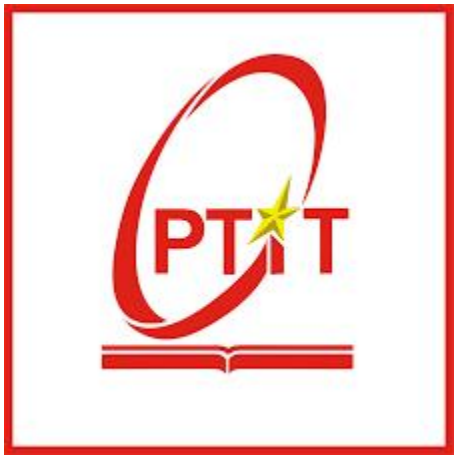
Khóa: 2022 – 2027

Hệ: Đại học chính quy

BỘ THÔNG TIN VÀ TRUYỀN THÔNG

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

KHOA CÔNG NGHỆ THÔNG TIN 1



BÁO CÁO BÀI TẬP LỚN

MÔN: HỆ ĐIỀU HÀNH

Đề tài: Tìm hiểu chung về hai họ hệ điều hành MacOS và iOS.

Thành viên	**Mã sinh viên**
Đỗ Đào Đông	B22DCAT088
Nguyễn Đăng Hiếu	B22DCAT120
Nguyễn Văn Hùng	B22DCAT136
Đặng Văn Phúc	B22DCAT223
Nguyễn Đình Tú	B22DCCN746

Mục lục

I. Giới thiệu

Apple cho ra mắt iOS lần đầu tiên vào ngày 29/6/2007 với tên gọi iPhone OS, đó cũng là lúc chiếc iPhone thế hệ thứ nhất xuất hiện trước toàn thế giới. Tại thời điểm đó, iOS chỉ được dùng trên iPhone. Năm 2010, Apple đổi tên hệ điều hành thành iOS để phản ánh việc mở rộng nền tảng sang iPad. Từ đó iOS đã trải qua nhiều phiên bản nâng cấp với các tính năng ngày càng đa dạng, tính bảo mật, giao diện, khả năng tương tác ngày càng phát triển.
Một số mốc quan trọng của iOS:

- **2007**: iPhone OS 1.0 ra mắt cùng iPhone đầu tiên.
- **2010**: iOS 4 giới thiệu đa nhiệm và hỗ trợ iPad.
- **2013**: iOS 7 ra mắt với thiết kế phẳng.
- **2017**: iOS 11 giới thiệu khả năng tăng cường thực tế (AR) và hỗ trợ cho iPad Pro.
- **2023**: iOS 17 ra mắt với các tính năng mới về FaceTime, cải tiến Siri và ứng dụng Messages.

Hệ điều hành MacOS đã trải qua một thời gian dài với nhiều tên gọi khác nhau trước khi được biết đến rộng rãi với cái tên hiện tại. Thời gian ban đầu, Apple công bố hệ điều hành với tên MacOS X. Sau đó vào năm 2012, tên dần được rút ngắn lại thành OS X. Và đến năm 2016 thì MacOS mới xuất hiện và được dùng cho tới hiện tại. Hệ điều hành MacOS được xây dựng trên nền tảng UNIX, mang lại tính ổn định cao và bảo mật mạnh mẽ. Giao diện trực quan của MacOS đã trở thành một trong những điểm nhấn của Apple, mang lại trải nghiệm dễ sử dụng và mượt mà cho người dùng.
Một số mốc quan trọng của macOS:

- **2000:** MacOS X Public Beta là phiên bản beta công khai đầu tiên của MacOS X.
- **2001**: MacOS X 10.0 Cheetah ra mắt, đánh dấu sự thay đổi từ MacOS cổ điển.
- **2011**: OS X Lion ra mắt, giới thiệu tính năng giống iOS như đa nhiệm và Launchpad.
- **2016**: Apple đổi tên OS X thành MacOS, bắt đầu với phiên bản MacOS Sierra.
- **2020**: macOS Big Sur ra mắt với một thiết kế mới và hỗ trợ chip Apple M1.

Cho đến nay, macOS và iOS đã phát triển cho mình một thị phần to lớn với lượng người dùng đông đảo. Thống kê thị phần tháng 10 năm 2024:

Hệ điều hành	Thị phần
Android	44.69%
Windows	26.81%
iOS	18.47%
macOS	5.66%
Linux	1.58%
Unknown	1.77%

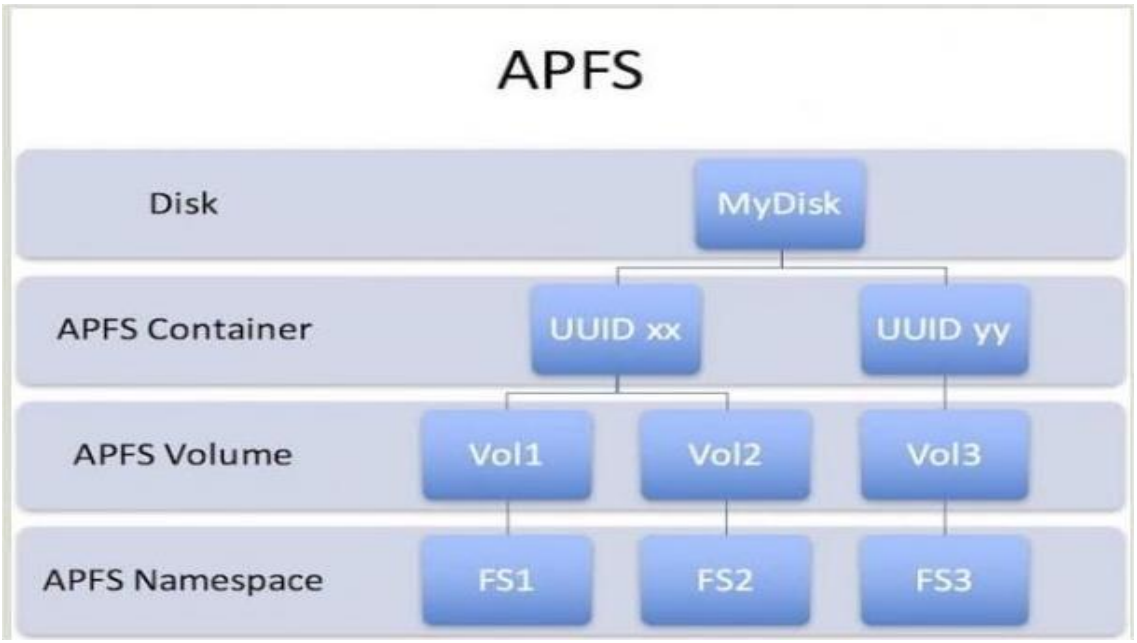
Cả hai hệ điều hành đều được được thiết kế để tối ưu hóa trải nghiệm người dùng, cung cấp giao diện trực quan và dễ sử dụng. Chúng được phát triển đồng bộ hóa với hệ sinh thái Apple, giúp dữ liệu, thông tin giữa các thiết bị Apple được trao đổi, chuyển tiếp liền mạch dễ dàng, thoải mái. Đi đôi với đó là sự bảo mật và quyền riêng tư mạnh mẽ.

II. Quản lý File

1. Hệ thống file APFS

- Trước khi APFS ra đời, hệ thống tệp mặc định của iOS và MacOS là MacOS Extended (HFS+), nó tồn tại nhiều vấn đề như nhiều tiến trình có thể truy cập hệ thống tệp cùng một lúc, thiếu dấu thời gian (timestamp) nano giây.
- Hệ thống tệp mặc định hiện tại được trên MacOS và iOS là Apple File System (APFS) được giới thiệu tại sự kiện WWDC vào năm 2016. Nó bắt đầu được sử dụng từ phiên bản iOS 10.3 và MacOS 10.13 High Sierra.
- APFS được thiết kế để tối ưu hóa cho lưu trữ trên SSD và hỗ trợ mã hóa, snapshot, và tính toàn vẹn dữ liệu.
- Ngoài ra macOS và iOS cũng hỗ trợ một số hệ thống tệp khác như ExFAT, FAT32, FAT, UDF, WebDAV,...

a) Kiến trúc của APFS



- **Container (Bộ chứa)**

  Đây là cấp độ cao nhất trong cấu trúc APFS, tương đương với một vùng đĩa (Volume group) trong các hệ thống tệp khác. Mỗi container có thể chứa nhiều volume, chia sẻ cùng một không gian vật lý. APFS sử dụng một đĩa ảo duy nhất để quản lý không gian cho tất cả các volume bên trong container, cho phép các volume trong cùng một container chia sẻ dung lượng lưu trữ một cách linh hoạt mà không cần phải đặt kích thước cố định từ đầu.

- **Volume (Ổ đĩa logic)**

  Trong APFS tương đương với các phân vùng (partitions) trong các hệ thống tệp truyền thống. APFS hỗ trợ tính năng volume cloning, cho phép tạo ra các bản sao của volume một cách nhanh chóng mà không sao chép toàn bộ dữ liệu, giúp tiết kiệm dung lượng. Mỗi volume trong APFS có thể được sử dụng như một hệ thống tệp độc lập với các thuộc tính và cấu trúc riêng.

- **Namespace**

  Namespace để quản lý và tổ chức các tệp và thư mục trong hệ thống file. Namespace trong APFS cho phép tạo ra các không gian riêng biệt để lưu trữ và quản lý các tệp và thư mục. Mỗi namespace trong APFS có thể được xem như một không gian độc lập với các tệp và thư mục của nó. Các namespace có thể được sử dụng để tạo ra các phân vùng ảo hoặc phân chia không gian lưu trữ trong hệ thống tệp.

b) Các tính năng của APFS

- **Tối ưu hóa cho Flash và SSD:** APFS được thiết kế để tối ưu hóa cho lưu trữ trên SSD và Flash.
- **Space Sharing (Chia sẻ dung lượng):** Giúp các volume trong một container có thể chia sẻ không gian trống với nhau.
- **Copy-on-Write (COW):** Kỹ thuật bảo vệ tính toàn vẹn của dữ liệu. Khi ghi dữ liệu, APFS không ghi đè trực tiếp lên các khối dữ liệu cũ. Thay vào đó, nó tạo một bản sao của dữ liệu đó và lưu trữ tại một vị trí mới, sau đó cập nhật các con trỏ đến bản sao mới này. Điều này đảm bảo rằng nếu hệ thống gặp sự cố trong quá trình ghi, dữ liệu cũ vẫn không bị thay đổi.
- **Snapshots:** APFS cho phép tạo một bản sao chỉ đọc của volume tại một thời điểm cụ thể mà không chiếm thêm nhiều không gian lưu trữ vì nó chỉ lưu các thay đổi từ lần snapshot gần nhất, sử dụng cơ chế copy-on-write. Ngoài ra, snapshots cũng giúp kiểm tra trạng thái hệ thống tại một thời điểm cụ thể.
- **Clones (Nhân bản):** Cho phép tạo ra bản sao của file hoặc thư mục mà không cần sao chép thực sự dữ liệu. Sử dụng phương pháp copy-on-write, khi bạn nhân bản một file, chỉ metadata của file được sao chép và dữ liệu chỉ được sao chép khi có thay đổi đối với file gốc hoặc bản sao.
- **Partition Scheme:** Partition scheme là một khái niệm liên quan đến cách phân chia và quản lý các phân vùng trên ổ đĩa. APFS hỗ trợ các partition scheme sau:

- GUID Partition Map (GPT): Đây là partition scheme phổ biến được sử dụng trên các máy tính Mac. GPT cho phép tạo ra nhiều phân vùng trên ổ đĩa và hỗ trợ dung lượng lớn.
- Master Boot Record (MBR): MBR là một partition scheme cũ hơn, hạn chế trong việc hỗ trợ dung lượng lớn và số lượng phân vùng. APFS hỗ trợ MBR để tương thích với các hệ thống cũ hơn hoặc hệ điều hành khác.
- Apple Partition Map (APM): APM là một partition scheme cũng được sử dụng trên các máy tính Mac cũ hơn. Tuy nhiên, APFS không hỗ trợ APM và khuyến nghị sử dụng GPT thay thế.
- Partition scheme quyết định cách ổ đĩa được phân chia và quản lý, và nó cũng ảnh hưởng đến khả năng tương thích với các hệ thống khác.

- **Crash Protection (Bảo vệ sự cố):**

  APFS cung cấp khả năng bảo vệ dữ liệu thông qua cơ chế copy-on-write và các snapshots. Giúp dễ dàng khôi phục dữ liệu từ các snapshots đồng thời cũng đảm bảo tính toàn vẹn dữ liệu.

- **Encryption (Mã hóa):**

  APFS hỗ trợ mã hóa mạnh mẽ ở mức độ volume hoặc từng file với chuẩn AES-256. Mã hóa có thể sử dụng một hoặc nhiều khóa để bảo vệ dữ liệu. Tích hợp chặt chẽ với các tính năng bảo mật như FileVault, giúp mã hóa toàn bộ ổ đĩa một cách an toàn và nhanh chóng.

- **Efficient File System Management (Quản lý tệp tin hiệu quả)**:

  APFS tối ưu hóa việc quản lý tệp tin với khả năng hỗ trợ tệp tin có kích thước lớn và số lượng tệp tin khổng lồ mà không ảnh hưởng đến hiệu suất.

- **Futureproofing**:

  APFS hỗ trợ các số 64-bit và hơn inode và hơn 9 triệu tệp trên 1 ổ đĩa. Inode là một cấu trúc mô tả một đối tượng hệ thống dưới dạng một tệp hoặc một thư mục.

- **TRIM Support**:

  APFS hỗ trợ lệnh TRIM, giúp tăng hiệu suất cho ổ SSD (Solid State Drive) bằng cách báo cho ổ đĩa biết những khối nào không còn sử dụng và có thể được xóa bỏ. Điều này giúp tối ưu hóa tuổi thọ và hiệu suất của SSD.

- **Sparse Files (Tệp thưa)**:

  APFS hỗ trợ Sparse Files, một tính năng cho phép tạo ra các tệp chứa nhiều khoảng trống (tệp thưa) mà không tiêu tốn dung lượng vật lý. Chỉ khi dữ liệu thực sự được ghi vào tệp thì không gian lưu trữ mới được sử dụng. Điều này hữu ích cho các ứng dụng yêu cầu phân bổ không gian tệp lớn nhưng không sử dụng hết không gian ngay lập tức.

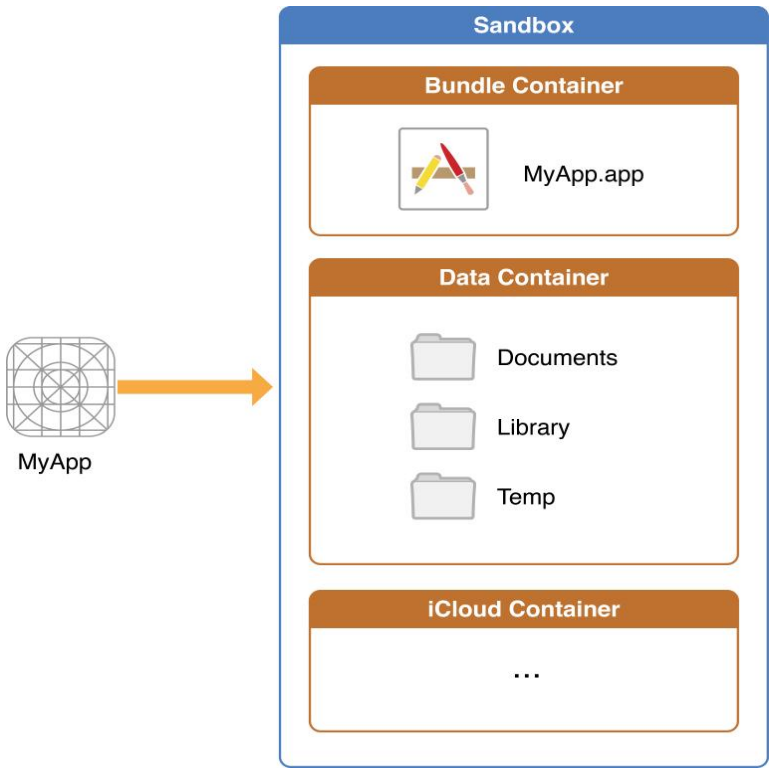
- **Atomic Safe-Save (Ghi an toàn nguyên tử):**Chức năng đảm bảo các thao tác ghi tệp toàn bộ ghi thành công hoặc không ghi gì cả, tránh được tình trạng tệp bị ghi dở dang khi có sự cố.

c) Các định dạng của APFS

Định dạng	Giới thiệu
APFS	Đây là phiên bản tiêu chuẩn của APFS, không có bất kỳ mã hóa nào được áp dụng cho dữ liệu. Định dạng này phù hợp với những người dùng không yêu cầu bảo vệ dữ liệu bằng mã hóa.
APFS (Encrypted)	Đây là định dạng tất cả dữ liệu trong volume đều được mã hóa bằng công nghệ mã hóa tích hợp của APFS. Người dùng sẽ cần nhập mật khẩu hoặc khóa để giải mã và truy cập vào dữ liệu.
APFS (Case-sensitive)	Đây là định dạng hệ thống tệp sẽ phân biệt giữa các tệp có tên giống nhau nhưng khác biệt về chữ hoa và chữ thường.
APFS (Case-sensitive, Encrypted)	Đây là định dạng kết hợp giữa phân biệt chữ hoa/chữ thường và mã hóa dữ liệu. Hệ thống tệp không chỉ phân biệt giữa các tên tệp có chữ hoa/chữ thường, mà dữ liệu của các tệp cũng được mã hóa để đảm bảo tính bảo mật.
APFS (Shared Space)	APFS cho phép các volume trong cùng một container chia sẻ không gian lưu trữ chung, các volume không bị giới hạn bởi kích thước cố định. Thay vì phân bổ trước dung lượng cho từng volume, không gian có thể được sử dụng động giữa các volume, giúp tối ưu hóa lưu trữ và tránh lãng phí dung lượng trống.

2. Cấu trúc thư mục trên iOS

- iOS sử dụng mô hình sandbox để đảm bảo rằng mỗi ứng dụng chỉ có thể truy cập vào không gian riêng biệt của nó trên hệ thống file.
- Trong quá trình cài đặt ứng dụng mới, trình cài đặt sẽ tạo ra một số thư mục chứa (container) cho ứng dụng bên trong thư mục sandbox. Mỗi container có một vai trò cụ thể.Ứng dụng cũng có thể yêu cầu quyền truy cập vào các thư mục chứa bổ sung—ví dụ như thư mục iCloud—tại thời điểm chạy.



Một ứng dụng iOS hoạt động trong sandbox của riêng nó

- Điều này bảo vệ hệ thống khỏi việc ứng dụng có thể xâm phạm dữ liệu của ứng dụng khác hoặc truy cập vào dữ liệu hệ thống.
- Một ngoại lệ cho quy tắc này là khi một ứng dụng sử dụng giao diện hệ thống công cộng để truy cập vào những thứ như danh bạ hoặc nhạc của người dùng. Trong trường hợp đó, hệ thống sử dụng các ứng dụng trợ giúp để xử lý mọi hoạt động liên quan đến việc đọc, sửa đổi các kho lưu trữ dữ liệu phù hợp.
- **Cấu trúc thư mục**:
  - **Document/**

    Chứa các tệp mà người dùng tạo hoặc quan trọng đối với ứng dụng. Các tệp trong thư mục này sẽ được sao lưu bởi iTunes và iCloud.
  - **Library/**

    Đây là thư mục cấp cao nhất cho bất kỳ tệp nào không phải là tệp dữ liệu người dùng. Các tệp thường được đặt trong các thư mục con tiêu chuẩn, như Caches. Bạn cũng có thể tạo thư mục con tùy chỉnh.

    Ứng dụng của bạn không nên sử dụng các thư mục này cho các tệp dữ liệu người dùng. Nội cung của thư mục Library (ngoại trừ thư mục con Caches) được hỗ trợ bởi iTunes và iCloud.
  - **tmp/**

Chứa các tệp tạm thời, có thể bị xóa khi hệ thống cần bộ nhớ hoặc khi ứng dụng không còn chạy. Nội dung của các thư mục này không được iTunes hoặc iCloud sao lưu.

3. Cấu trúc thư mục trên macOS

- Trong macOS, hệ thống tệp được chia thành nhiều miền (domain) để tổ chức và quản lý các tệp, giúp hệ thống áp dụng quyền truy cập phù hợp và cung cấp sự đơn giản cho người dùng. Việc sắp xếp các tệp theo miền cũng cho phép hệ thống áp dụng các quyền truy cập chung cho các tệp trong miền đó, ngăn người dùng không được phép thay đổi tệp một cách cố ý hoặc vô ý.

a) Các miền chính

- **User Domain (Miền người dùng)**

  Chứa các tệp và tài nguyên liên quan đến người dùng hiện tại, bao gồm thư mục chính (home directory) của họ.

  Người dùng có quyền kiểm soát các tệp trong thư mục này.

  Thư mục chính có thể nằm trên ổ khởi động của máy tính hoặc trên một ổ mạng.

- **Local Domain (Miền cục bộ)**

  Chứa các ứng dụng và tài nguyên được cài đặt trên máy tính và chia sẻ cho tất cả người dùng trên máy tính đó.

  Không tương ứng với một thư mục duy nhất mà bao gồm nhiều thư mục trên ổ khởi động.

  Người dùng có quyền quản trị có thể thêm, xóa, hoặc sửa đổi các tệp trong miền này.

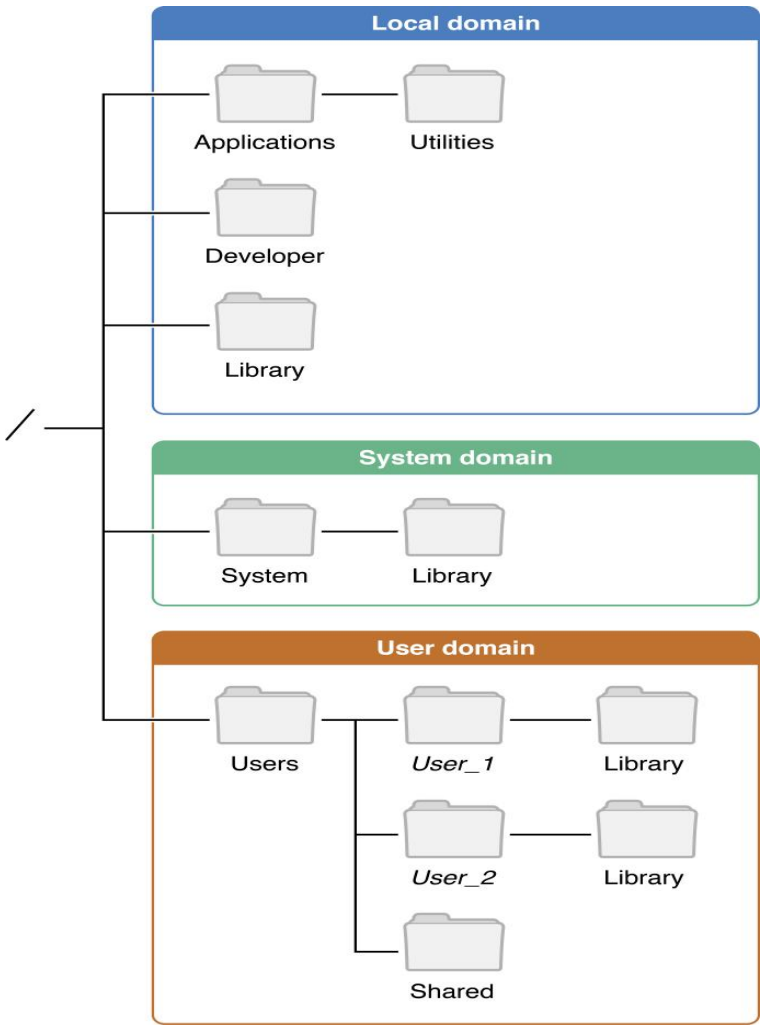
- **Network Domain (Miền mạng)**

  Chứa các tài nguyên được chia sẻ giữa nhiều người dùng trên một mạng cục bộ.

  Các tài nguyên này thường được lưu trữ trên máy chủ mạng và được quản lý bởi quản trị viên mạng.

- **System Domain (Miền hệ thống)**

  Chứa phần mềm hệ thống và tài nguyên do Apple cài đặt, cần thiết để hệ điều hành hoạt động. Người dùng không thể thay đổi các tệp trong miền này.



3.1. Hệ thống tệp macOS cục bộ

b) Các thư mục chính trong macOS

- **/Applications**

  Chứa các ứng dụng cài đặt trên hệ thống. Các ứng dụng có thể được truy cập bởi tất cả người dùng trên máy tính. Người dùng có thể thêm hoặc xóa ứng dụng tại đây nếu có quyền quản trị. Thư mục này là một phần của miền địa phương.

- **/Library**Chứa các tệp hỗ trợ chung cho các ứng dụng và hệ thống, chẳng hạn như tệp hỗ trợ ứng dụng, phông chữ, plugin, và cấu hình**.** Các thư mục con chính của Library:
  - **Application Support**: Sử dụng thư mục này để lưu trữ tất cả các tệp dữ liệu ứng dụng ngoại trừ các tệp được liên kết với các tài liệu của người dùng.
  - **Caches**: Sử dụng thư mục này lưu các tệp hỗ trợ dành riêng cho ứng dụng nào mà ứng dụng của người dùng có thể tạo lại dễ dàng. Ứng dụng của người dùng thường chịu trách nhiệm quản lý nội dung của thư mục này và thêm, xóa các tệp khi cần thiết.
  - **Frameworks**: Lưu trữ các frameworks để tạo ứng dụng macOS của mình.
  - **Preferences**: Chứa các tệp ưu tiên dành riêng cho ứng dụng.

- **/Network:** Thư mục chứa danh sách các máy tính trong mạng cục bộ.
- **/System:** Thư mục chứa các tệp hệ điều hành cốt lõi của macOS, bao gồm các tệp hệ thống cần thiết để hệ thống hoạt động. Người dùng không có quyền thay đổi nội dung của thư mục này.
- **/User**

  Thư mục này chứa một hoặc nhiều thư mục của tất cả người dùng trên máy tính. Mỗi người dùng có thư mục riêng, chứa tên tệp cá nhân và cài đặt người dùng.

  Các thư mục con cơ bản trong Users:

  - **Applications**: Chứa các ứng dụng dành riêng cho người dùng.
  - **Desktop**: Chứa các mục trên máy tính để bàn của người dùng.
  - **Documents**: Chứa tài liệu và tệp người dùng.
  - **Downloads**: Chứa các tệp được tải xuống từ Internet.
  - **Library**: Chứa các tệp ứng dụng dành riêng cho người dùng (được ẩn trong MacOS 10.7 trở lên).
  - **Movies**: Chứa các tệp video của người dùng.
  - **Music**: Chứa các tệp nhạc của người dùng.
  - **Pictures**: Chứa ảnh người dùng.
  - **Public**: Chứa nội dung mà người dùng muốn chia sẻ.
  - **Sites**: Chứa các trang web được sử dụng bởi trang web cá nhân của người dùng. (Chia sẻ web phải được bật để hiển thị các trang này.)
- **/bin, /sbin, /usr:**

  Các thư mục này chứa các lệnh hệ thống và các chương trình thực thi cần thiết cho việc quản lý và vận hành hệ điều hành. Chúng chủ yếu được sử dụng bởi hệ thống và các quản trị viên.
- **/tmp**

  Thư mục tạm thời, chứa các tệp tạm do hệ điều hành và ứng dụng tạo ra. Các tệp trong thư mục này thường bị xóa sau khi khởi động lại hệ thống.
- **/var**

  Chứa các tệp log, tệp tạm, và dữ liệu biến động của hệ thống, chẳng hạn như các tệp database, log hệ thống, và cache.
- **/Volumes**

  Thư mục này chứa các thiết bị lưu trữ được gắn kết vào hệ thống, chẳng hạn như ổ cứng ngoài, ổ đĩa USB, hoặc các phân vùng khác.

4. Kiểm soát quyền truy cập

- **Access Control Lists (ACLs)**

  Trên macOS, hệ thống quyền truy cập các tệp truyền thống dựa trên mô hình quyền của Unix, bao gồm 3 loại quyền cơ bản (đọc, ghi, thực thi) dành cho 3 đối tượng (chủ sở hữu, nhóm, người khác). Ngoài ra, macOS còn hỗ trợ ACLs.

  ACLs là tập hợp các điều khiển chi tiết, cho phép chỉ định chính xác những gì người dùng có thể và không thể làm đối với một tệp hoặc thư mục. ACLs cho phép bạn cấp quyền truy cập chi tiết cho từng người dùng cụ thể, cho phép hoặc từ chối những hành động như đọc, ghi, hoặc thực thi tệp.

- **File Protection (Bảo vệ file trên iOS)**

  iOS sử dụng cơ chế File Protection để mã hóa các tệp và dữ liệu ứng dụng. Mỗi tệp được lưu trữ trên thiết bị iOS có thể được gắn với một mức độ bảo vệ cụ thể, ví dụ:

  ➢ **Complete Protection**: Tệp chỉ có thể được truy cập khi thiết bị được mở khóa.
  ➢ **Protected Unless Open**: Các tệp chỉ được bảo vệ khi chúng chưa được mở.
  ➢ **No Protection**: Tệp không được mã hóa và có thể truy cập mọi lúc.

5. Icloud và quản lý file qua đám mây

- iCloud là dịch vụ đám mây của Apple, giúp người dùng lưu trữ và đồng bộ dữ liệu giữa các thiết bị Apple như iPhone, iPad, Mac và cả trên Windows. iCloud không chỉ giúp lưu trữ các file và dữ liệu cá nhân mà còn cung cấp các tính năng khác như đồng bộ hóa lịch, danh bạ, ảnh và hơn thế nữa.
- Quản lý file qua iCloud với File Provider API (iOS): Apple cung cấp API File Provider cho phép các ứng dụng bên thứ ba tích hợp với hệ thống quản lý file của iOS và macOS. Điều này giúp các ứng dụng lưu trữ và truy xuất file từ đám mây như iCloud, Dropbox hoặc Google Drive một cách liền mạch trong ứng dụng Files của iOS.

  Tóm lại, iCloud cung cấp một giải pháp hoàn chỉnh để lưu trữ và đồng bộ hóa file, sao lưu thiết bị, chia sẻ dữ liệu và bảo mật thông tin cá nhân. Tính năng quản lý file qua đám mây của iCloud giúp bạn dễ dàng truy cập và lưu trữ tài liệu, ảnh, video và các dữ liệu khác giữa các thiết bị Apple của mình.

6. Bảo mật hệ thống file

- **Mã hóa Dữ liệu**
  ➢ macOS sử dụng FileVault để thực hiện mã hóa toàn bộ ổ đĩa (Full Disk Encryption - FDE). FileVault sử dụng thuật toán mã hóa XTS-AES-128 với khóa 256-bit, giúp bảo vệ dữ liệu người dùng khỏi truy cập trái phép. Ngoài ra, FileVault cung cấp tùy chọn

lưu trữ khóa phục hồi (recovery key) trên iCloud hoặc cục bộ để khôi phục dữ liệu trong trường hợp người dùng quên mật khẩu.

➢ iOS sử dụng mã hóa dựa trên phần cứng cho cả dữ liệu tệp và hệ thống tệp. Mỗi tệp được mã hóa bằng một khóa riêng biệt. Tất cả dữ liệu lưu trữ trên bộ nhớ trong của thiết bị đều được mã hóa và chỉ có thể truy cập khi thiết bị được mở khóa. Ngoài ra, iOS áp dụng Data Protection Classes, các lớp bảo vệ này kiểm soát quyền truy cập dữ liệu tùy thuộc vào trạng thái thiết bị (khóa hay mở khóa), giúp tăng cường bảo mật ngay cả khi thiết bị bị đánh cắp.

- **Bảo vệ Tính toàn vẹn của Hệ thống**

➢ System Integrity Protection (SIP) trên macOS bảo vệ các tệp và thư mục hệ thống khỏi sự thay đổi hoặc truy cập trái phép, ngay cả từ người dùng có quyền root. SIP cũng ngăn chặn việc cài đặt hoặc tải các driver (kext) từ bên thứ ba mà không có sự phê duyệt, giảm nguy cơ bị tấn công từ cấp độ kernel.

- **Secure Enclave**

➢ Trên cả iOS và các máy Mac sử dụng Apple Silicon, Secure Enclave là một bộ xử lý đồng bộ chuyên biệt chịu trách nhiệm quản lý các hoạt động bảo mật như mã hóa dữ liệu và xác thực sinh trắc học (Touch ID/Face ID). Secure Enclave hoạt động tách biệt với phần còn lại của hệ thống, đảm bảo rằng các thông tin nhạy cảm vẫn được bảo mật ngay cả khi hệ điều hành chính bị xâm nhập.

- **Secure Boot**

➢ Trên các thiết bị iOS và máy Mac có Apple Silicon, Secure Boot đảm bảo rằng chỉ phiên bản hợp lệ và đáng tin cậy của hệ điều hành mới có thể khởi động, ngăn chặn việc cài đặt firmware giả mạo hoặc hệ điều hành độc hại. Cơ chế này giúp bảo vệ hệ thống ngay từ giai đoạn khởi động.

- **Minh bạch và Quyền đồng ý của người dùng**

➢ Transparency and Consent Framework (TCF): Cả macOS và iOS yêu cầu người dùng cung cấp sự đồng ý trước khi ứng dụng có thể truy cập các dữ liệu cá nhân nhạy cảm. Điều này đảm bảo rằng người dùng có quyền kiểm soát và nhận thức về việc ứng dụng có thể truy cập vào thông tin nào.

➢ App Tracking Transparency (ATT): Trên iOS, các ứng dụng phải xin phép người dùng trước khi có thể theo dõi hoạt động của họ trên các ứng dụng hoặc trang web khác. ATT cho phép người dùng kiểm soát quyền riêng tư và hạn chế việc thu thập dữ liệu theo dõi.

- **Quản lý khóa và thông tin nhạy cảm**

➢ Cả macOS và iOS sử dụng Keychain để quản lý và bảo vệ mật khẩu, khóa mã hóa và các thông tin đăng nhập nhạy cảm khác. Keychain được mã hóa bằng AES-256 và yêu

cầu xác thực trước khi có thể truy cập dữ liệu. Điều này đảm bảo rằng thông tin nhạy cảm của người dùng luôn được bảo vệ.

- **Cập nhật bảo mật và vá lỗi tự động**
  - ➢ macOS và iOS đều cung cấp cơ chế cập nhật bảo mật tự động, giúp bảo vệ hệ thống khỏi các lỗ hổng mới. Apple thường xuyên phát hành các bản cập nhật để vá lỗi và nâng cao bảo mật, đảm bảo rằng hệ điều hành và phần cứng luôn an toàn trước các mối đe dọa mới nhất.

III. Quản lý bộ nhớ

1. Các phương thức quản lý bộ nhớ trong MacOS và iOS

a) Bộ nhớ ảo (Virtual Memory)

- **Mục tiêu:** Cả MacOS và iOS đều sử dụng bộ nhớ ảo để tạo ra một không gian địa chỉ lớn hơn khả năng thực tế của bộ nhớ vật lý. Điều này giúp các ứng dụng hoạt động trong không gian bộ nhớ riêng biệt, tăng cường bảo mật và giảm xung đột.
- **Quản lý bằng phân trang (Paging):**
  - Bộ nhớ ảo được chia thành các trang có kích thước cố định. Hệ điều hành sẽ lưu trữ các trang này trong RAM khi cần thiết và di chuyển chúng sang ổ đĩa (swap) hoặc xóa khỏi RAM khi không còn cần thiết.
  - Mỗi ứng dụng hoạt động trong một không gian địa chỉ ảo độc lập, và bộ quản lý bộ nhớ của hệ điều hành chịu trách nhiệm ánh xạ các địa chỉ ảo này đến địa chỉ vật lý khi ứng dụng thực hiện lệnh truy cập bộ nhớ.

b) Bộ nhớ được bảo vệ (Protected Memory)

- **Mục tiêu:** Giúp ngăn chặn các ứng dụng hoặc tiến trình truy cập vào bộ nhớ của nhau, tránh làm rò rỉ dữ liệu và bảo vệ hệ điều hành khỏi các cuộc tấn công.
- **Cách hoạt động:** Mỗi ứng dụng chỉ được phép truy cập vào vùng bộ nhớ mà nó sở hữu. Hệ điều hành quản lý và xác định quyền truy cập, nếu ứng dụng truy cập vào vùng không được phép, hệ thống sẽ dừng hoạt động của ứng dụng đó.

c) ARC (Automatic Reference Counting)

- **Mục tiêu:** Tự động quản lý các đối tượng trong bộ nhớ mà không cần can thiệp thủ công của lập trình viên, giúp giảm bớt tình trạng rò rỉ bộ nhớ (memory leak).
- **Cách hoạt động:** Mỗi đối tượng trong bộ nhớ đều có một bộ đếm tham chiếu (reference count). Khi một biến hoặc đối tượng tạo ra một tham chiếu đến đối tượng khác, bộ đếm tăng thêm 1. Khi tham chiếu này kết thúc, bộ đếm giảm. Khi bộ đếm bằng 0, bộ nhớ của đối tượng sẽ được giải phóng.

d) Memory Warnings (Cảnh báo bộ nhớ) trong iOS

- **Mục tiêu:** Trong môi trường hạn chế về tài nguyên như iOS, cảnh báo bộ nhớ giúp các ứng dụng giải phóng tài nguyên không cần thiết.
- **Cách hoạt động:** iOS sẽ gửi thông báo khi bộ nhớ gần đầy, để các ứng dụng giải phóng tài nguyên đang giữ hoặc lưu trạng thái hiện tại và chuẩn bị đóng ứng dụng.

2. Quản lý bộ nhớ ảo (Virtual Memory Management)

a) Quản lý bộ nhớ ảo trong MacOS

- **Swapping và Paging:** MacOS sử dụng cả phân trang (paging) và "swap" để quản lý bộ nhớ ảo hiệu quả. Hệ điều hành sẽ di chuyển các trang không sử dụng từ RAM ra ổ cứng khi bộ nhớ vật lý đầy và đưa chúng trở lại RAM khi cần. Điều này giúp MacOS có khả năng mở rộng bộ nhớ thực tế lên nhiều lần, hỗ trợ tốt cho các ứng dụng yêu cầu bộ nhớ lớn.
- **Unified Buffer Cache (Bộ nhớ đệm hợp nhất):** Hệ điều hành MacOS sử dụng một bộ nhớ đệm hợp nhất cho cả dữ liệu của các ứng dụng và hệ thống file. Điều này cho phép hệ điều hành quản lý và tối ưu hóa bộ nhớ nhanh chóng, giảm tải I/O và tăng cường hiệu năng.
- **Memory Compression:** Khi bộ nhớ vật lý gần đầy, MacOS có thể nén dữ liệu thay vì "swap" ra ổ đĩa. Điều này giúp tăng cường hiệu quả sử dụng bộ nhớ và tránh giảm hiệu suất do việc đọc/ghi từ ổ cứng

b) Quản lý bộ nhớ ảo trong iOS

- **Không sử dụng "Swap":** Do bộ nhớ lưu trữ và tài nguyên hệ thống hạn chế, iOS không hỗ trợ cơ chế "swap" bộ nhớ ra ổ đĩa như MacOS. Điều này có nghĩa là nếu bộ nhớ đầy, hệ điều hành sẽ bắt buộc đóng ứng dụng hoặc giải phóng bộ nhớ. Các ứng dụng cần tối ưu hóa việc sử dụng bộ nhớ và giải phóng tài nguyên khi không còn cần thiết.
- **Trang bộ nhớ và Bộ nhớ đệm:** Mặc dù không có cơ chế "swap", iOS vẫn sử dụng phân trang và bộ nhớ đệm để tối ưu hóa bộ nhớ cho các tác vụ. Khi bộ nhớ cần được giải phóng, hệ điều hành sẽ tự động xóa bộ nhớ đệm và các trang không quan trọng để nhường chỗ cho các tác vụ ưu tiên.
- **Quản lý năng lượng**: iOS tối ưu hóa bộ nhớ cùng với việc tiết kiệm năng lượng. Hệ điều hành có thể đóng các ứng dụng nền và giải phóng tài nguyên để tiết kiệm pin, một yếu tố quan trọng trong các thiết bị di động.

3. Cơ chế thay thế bộ nhớ (Memory Replacement)

a) Cơ chế thay thế bộ nhớ trong MacOS

- **Thuật toán Least Recently Used (LRU):** Hệ điều hành sử dụng LRU để xác định các trang nào trong RAM ít được sử dụng nhất và có thể được "swap" ra ổ đĩa khi cần bộ

nhớ trống. Điều này giúp MacOS duy trì hiệu suất ổn định khi phải xử lý nhiều tiến trình cùng lúc.

- **Inactive List và Active List:** MacOS duy trì hai danh sách quản lý bộ nhớ là Inactive List và Active List. Các trang trong Active List là các trang đang được sử dụng, còn trong Inactive List là các trang không còn được sử dụng. Các trang trong Inactive List sẽ là các trang đầu tiên bị "swap" ra nếu bộ nhớ đầy.

b) Cơ chế thay thế bộ nhớ trong iOS

- **Không sử dụng LRU truyền thống:** iOS không thực hiện cơ chế "swap" bộ nhớ, vì vậy không có LRU như trong MacOS. Thay vào đó, hệ điều hành sẽ gửi cảnh báo bộ nhớ đến các ứng dụng để giải phóng tài nguyên hoặc đóng các ứng dụng nền.
- **Ứng dụng nền bị đóng:** Nếu không đủ bộ nhớ cho một ứng dụng đang chạy, iOS sẽ đóng các ứng dụng nền, giải phóng bộ nhớ để tối ưu hiệu năng cho ứng dụng đang chạy.

4. Kết luận

- **MacOS**: Được tối ưu hóa cho máy tính để bàn, MacOS sử dụng bộ nhớ ảo với khả năng hoán đổi và nén bộ nhớ, cung cấp không gian địa chỉ ảo lớn và cho phép nhiều tiến trình chạy cùng lúc. Hệ điều hành này còn sử dụng các thuật toán thay thế trang và quản lý bộ nhớ động.

- **iOS**: Được tối ưu hóa cho các thiết bị di động, iOS cũng sử dụng bộ nhớ ảo nhưng không có khả năng hoán đổi. Để khắc phục giới hạn RAM, iOS sử dụng cảnh báo bộ nhớ, nén bộ nhớ và quản lý bộ nhớ tự động với ARC. iOS cũng áp dụng các biện pháp như giải phóng ứng dụng ít dùng và giảm bộ nhớ cache để duy trì hiệu năng cao trên các thiết bị di động có tài nguyên hạn chế.Quản lý bộ nhớ trong MacOS và iOS được thiết kế phù hợp với từng loại thiết bị, giúp tối ưu hóa hiệu năng và tài nguyên hệ thống, đảm bảo hệ điều hành chạy mượt mà và ổn định

IV. Quản lý tiến trình

1. Hệ điều hành MacOS

- Mỗi một chương trình là một tệp thực thi và một tiến trình làm một khoảnh khắc của chương trình được thực hiện theo trục thời gian. Tiến trình bao gồm:
  - Mã trình thực thi
  - Dữ liệu(data)của tiến trình
  - Program (user) stack
  - CPU program counter
  - Kernel stack
  - CPU registers
  - Thông tin khác cần thiết để chạy trình.
- Các dữ liệu này tạo ra bối cảnh của tiến trình, mỗi tiến trình có bối cảnh riêng biệt. Có rất nhiều tiến trình được thực hiện đồng thời trên MacOS (đặc tính này còn gọi là đa trình - multiprogramming hay đa nhiệm - multitasking) theo nguyên lí phân chia thời gian, mà tổng số các tiến trình về logic là không có giới hạn. Các tiến trình có thể tạo ra các tiến trình mới, kết thúc các tiến trình, đồng bộ các giai đoạn thực hiện tiến trình, kiểm soát phản ứng với các sự kiện khác nhau.
- Người dùng có thể viết các chương trình thực hiện các thao tác rất tinh tế mà bản thân kernel không cần có nhiều chức năng hơn là cần thiết. Có thể đề cập tới một số các chức năng, chẳng hạn các trình thông dịch, bộ soạn thảo thuộc lớp các chương trình cấp người dùng và quan trọng hàng đầu là shell, là trình thông dịch mà người dùng sử dụng ngay sau khi login vào hệ thống: shell thông dịch các từ trong dòng lệnh thành tên lệnh máy, phát sinh tiến trình con và tiến trình con thực hiện lệnh đưa vào, xử lí các từ còn lại trong dòng lệnh như các thông số của lệnh. Shell thực hiện ba kiểu lệnh:
  - Lệnh là tệp có thể thực hiện được chứa mã máy phát sinh do bộ dịch tạo ra từ mã nguồn (chương trình C chẳng hạn).
  - Lệnh là tệp chứa một xâu các dòng lệnh của shell.
  - Là các lệnh bên trong của shell. Các lệnh bên trong này làm cho shell trở thành một ngôn ngữ lập trình rất mạnh trong MacOS.
- Shell là chương trình thuộc lớp người dùng, không phải là phần của kernel, cho nên có thể dễ dàng biến đổi cho mỗi môi trường đặc thù. Bản thân shell cũng có ba loại khác nhau thích hợp cho các nhu cầu sử dụng khác nhau và hệ thống có thể chạy các shell đó đồng thời. Sức mạnh của mỗi kiểu shell thể hiện ở khả năng lập trình của mỗi kiểu. Mỗi tiến

trình được thực hiện trong MacOS có một môi trường (execution environment) thực hiện, bao gồm cả thư mục hiện hành. Thư mục hiện hành của tiến trình là thư mục dùng để chỉ đường dẫn không bắt đầu bằng "/". Người dùng có thể thực hiện nhiều tiến trình cùng một lúc, và các tiến trình lại có thể tạo ra các tiến trình khác một cách động, và đồng bộ việc thực hiện các tiến trình đó. Đặc tính này tạo ra một môi trường thực hiện chương trình rất mạnh trong MacOS.

- Kernel thực hiện vô số các thao tác cơ bản thay mặt cho các tiến trình của người dùng để hỗ trợ cho giao diện người dùng, bao gồm:
  - Kiểm soát việc thực hiện các tiến trình gồm có: cho phép tiến trình tạo tiến trình mới, kết thúc tiến trình, treo việc thực hiện và trao đổi thông điệp giữa các tiến trình.
  - Lập thời gian biểu để các tiến trình được thực hiện trên CPU. Các tiến trình chia sẻ CPU theo phương thức phân chia thời gian, một tiến trình sẽ bị treo sau khi thời gian phân bổ đã hết, kernel lấy tiến trình khác đưa vào thực hiện. Sau này kernel sẽ lại lựa chọn tiến trình bị treo để đưa vào thực hiện trở lại.
  - Cấp phát bộ nhớ cho tiến trình đang thực hiện, cho phép tiến trình chia sẻ không gian địa chỉ của tiến trình dưới những điều kiện nhất định, bảo vệ miền địa chỉ riêng của tiến trình đối với các tiến trình khác. Nếu hệ thống chạy trong hoàn cảnh thiếu bộ nhớ, kernel sẽ giải phóng bộ nhớ bằng cách ghi lại các tiến trình tạm thời vào bộ nhớ dự phòng (còn gọi là thiết bị swap). Nếu tồn bộ tiến trình được ghi vào swap, thì hệ MacOS gọi là hệ tráo đổi (swapping system). Nếu kernel ghi các trang của bộ nhớ lên swap, thì hệ đó gọi là hệ lưu trang.
  - Cấp phát bộ nhớ thứ cấp để cất và tìm lại dữ liệu của người dùng có hiệu quả. Dịch vụ này cấu tạo nên hệ thống tệp. Kernel cấp vùng nhớ thứ cấp cho tệp của người dùng, khôi phục lại vùng nhớ, xây dựng cấu trúc tệp theo một cách thức hiểu được, bảo vệ tệp của người dùng trước các truy nhập bất hợp pháp.

2. Hệ điều hành iOS

Tiến trình iOS trải qua các trạng thái như sau:

- Trạng thái khởi tạo (Create):

  Khi mà một tiến trình mới được tạo, nó nhận vùng stack riêng của mình và vào trạng thái mới (new). Tiến trình có thể di chuyển đến trạng thái điều chỉnh (Modify). Nếu không có thay đổi cần thiết, thì tiến trình chuyển sang trạng thái thực thi (Execute).
- Trạng thái điều chỉnh (Modify):

Không giống như hầu hết các hệ điều hành, iOS không tự động truyền tải các tham số khởi tạo hoặc gán một giao tiếp đến một tiến trình mới khi nó được tạo, bởi vì nó cho rằng hầu hết các tiến trình không cần tài nguyên này. Nếu một tiến trình cần nguồn tài nguyên này, tiến trình mà tạo nó có thể điều chỉnh để thêm vào.

- Trạng thái thực thi (Execute):
  - Sau khi một tiến trình mới được tạo thành công và điều chỉnh, nó chuyển sang trạng thái sẵn sàng (Ready) và vào trạng thái thực thi (Execute).
  - Trong suốt trạng thái này, một tiến trình có thể truy cập CPU và chạy. Một tiến trình có thể là một trong 3 trạng thái: sẵn sàng, chạy và rỗi (Idle).
  - Một tiến trình ở trạng thái sẵn sàng sẽ đợi chuyển sang trạng thái truy cập CPU và bắt đầu thực thi lệnh. Một tiến trình ở trạng thái rỗi là đang ngủ, đợi sự kiện bên ngoài xuất hiện trước khi nó có thể chạy. Một tiến trình chuyển từ trạng thái sẵn sàng sang trạng thái chạy khi mà nó được lập lịch để chạy.
  - Với đa tác vụ mà không ưu tiên (non-preemptive multitasking), một tiến trình được lập lịch chạy trên CPU cho đến khi tạm ngừng hoặc kết thúc. Một tiến trình có thể tạm dừng theo 2 cách: nó có thể tự dừng bởi việc báo cho kernel, nó muốn nhường cho CPU và chuyển sang trạng thái sẵn sàng, và đợi đến lúc chạy lại. Tiến trình cũng có thể dừng bởi một hoạt động bên ngoài xảy ra. Khi mà một tiến trình đợi một sự kiện, kernel dừng tiến trình này và chuyển nó sang trạng thái rỗi. Sau khi một sự kiện xảy ra rồi thì kernel chuyển tiến trình trở lại trạng thái sẵn sàng để đợi chạy lại.
- Trạng thái kết thúc (Terminal):
  - Trạng thái cuối cùng trong vòng đời của tiến trình là trạng thái kết thúc. Một tiến trình vào trạng thái kết thúc khi nó hoàn thành chức năng của mình và đóng lại hoặc khi một tiến trình khác đóng nó. Khi một tiến trình bị đóng hoặc tự động, tiến trình chuyển sang trạng thái chết (Dead). Tiến trình này trạng thái chết (không hoạt động) cho đến khi kernel thu hồi tất cả các tài nguyên của nó. Sau khi tài nguyên được thu hồi, tiến trình bị kết thúc thoát khỏi trạng thái chết và xóa khỏi hệ thống.
- Độ ưu tiên tiến trình iOS:

  iOS thực hiện chế độ ưu tiên để lập lịch các tiến trình trên CPU. Tại thời điểm tạo, mỗi tiến trình được gán một trong 4 độ ưu tiên dựa trên mục đích của tiến trình. Độ ưu tiên là không đổi, chúng được gán khi một tiến trình được tạo và không bao giờ thay đổi. Các độ ưu tiên:
  - Critical: Dành riêng cho những tiến trình hệ thống thiết yếu mà giải quyết những vấn đề cấp phát tài nguyên.
  - High: Được gán cho những tiến trình mà cung cấp đáp ứng nhanh, như tiến trình nhận gói trực tiếp từ giao tiếp mạng.

- Medium: Độ ưu tiên mặc định sử dụng bởi hầu hết các tiến trình.
- Low: Được gán cho những tiến trình cung cấp những tác vụ mang tính định kỳ như bảng ghi lỗi... Độ ưu tiên các tiến trình cung cấp sự ưu đãi cho một vài tiến trình để truy cập CPU dựa trên sự quan trọng của nó đối với hệ thống và iOS không thực hiện quyền ưu tiên. Một tiến trình có sự ưu tiên cao hơn không thể ngắt một tiến trình có độ ưu tiên thấp hơn, thay vào đó, tiến trình có độ ưu tiên cao hơn thì có nhiều cơ hội hơn để truy cập CPU hơn.

V. So sánh theo các đề tài

1. Cấu trúc hệ thống.

Cấu trúc hệ thống của MacOS và iOS đều dựa trên nhân Darwin từ Unix.

	MacOS	iOS
Kernel	Tập trung vào đa nhiệm, hiệu năng trên chip Intel	Tối ưu hóa để tiết kiệm pin và hiệu suất trên chip ARM
Core OS	Hỗ trợ phần cứng	Tập trung tính toán hiệu suất cao, hỗ trợ xử lý song song.
Core Services	Có các dịch vụ cơ bản cho ứng dụng máy tính	Thêm các dịch vụ như GPS, iCloud, đồng bộ hóa dữ liệu
Graphics & Media	Quartx, OpenGL, QuickTime	Core Animation, Core Image, AVFoundation
Application Layer	Hỗ trợ Classic, Carbon, Cocoa cho ứng dụng máy tính	Cocoa Touch, UIKit xây dựng giao diện cảm ứng
Giao diện người dùng	Giao diện Aqua	Dùng UIKit

2. Hàm Shell.

- Hàm shell là một cách để nhóm các lệnh để thực hiện sau bằng cách sử dụng một tên duy nhất cho nhóm. Chúng được thực hiện giống như một lệnh thông thường. Khi tên của một hàm shell được sử dụng như một tên lệnh đơn giản, danh sách các lệnh liên quan đến tên hàm đó sẽ được thực hiện. Hàm shell thường được lưu tại .bashc, .zshrc.
- Trên hệ điều hành iOS, các lệnh shell thông thường không thể được sử dụng trực tiếp như trên hệ điều hành macOS hoặc Linux. iOS là một hệ điều hành có tính bảo mật cao và hạn chế người dùng truy cập trực tiếp vào shell.

	MacOS	iOS
Quản lý file		
open	Mở một tệp hoặc thiết bị và trả về mô tả tệp.	Mở một tệp hoặc thiết bị và trả về mô tả tệp.
close	Đóng tệp đã mở, giải phóng tài nguyên.	Đóng tệp đã mở, giải phóng tài nguyên.
read, write	Đọc và ghi dữ liệu giữa tệp và bộ nhớ.	Đọc và ghi dữ liệu giữa tệp và bộ nhớ.
lseek	Đặt lại vị trí đọc/ghi trong tệp.	Đặt lại vị trí đọc/ghi trong tệp.
unlink	Xóa tệp khỏi hệ thống tệp.	Xóa tệp khỏi hệ thống tệp.
rename	Đổi tên hoặc di chuyển tệp.	Đổi tên hoặc di chuyển tệp.
mkdir, rmdir	Tạo hoặc xóa thư mục.	Tạo hoặc xóa thư mục.
stat, fstat	Lấy thông tin chi tiết về tệp (kích thước, quyền truy cập).	Lấy thông tin chi tiết về tệp (kích thước, quyền truy cập).
chmod, chown	Thay đổi quyền truy cập và chủ sở hữu tệp.	Thay đổi quyền truy cập và chủ sở hữu tệp.
link, symlink	Tạo liên kết cứng hoặc liên kết mềm đến tệp.	Tạo liên kết cứng hoặc liên kết mềm đến tệp.
Quản lý tiến trình		
fork	Tạo tiến trình con bằng cách nhân bản tiến trình cha.	Hạn chế với ứng dụng người dùng (chỉ các dịch vụ hệ thống có quyền truy cập).
execve	Thay thế tiến trình hiện tại bằng tiến trình mới.	Hạn chế với ứng dụng người dùng, chỉ cho phép ứng dụng hệ thống.
wait, waitpid	Đợi tiến trình con kết thúc và lấy mã thoát.	Hạn chế truy cập, chỉ các dịch vụ nội bộ của hệ thống có thể sử dụng.

kill	Trả về PID của tiến trình hiện tại và tiến trình cha.	Trả về PID của tiến trình hiện tại và tiến trình cha.
setpgid	Gửi tín hiệu đến tiến trình khác (dừng, tiếp tục, hoặc yêu cầu thực hiện hành động).	Hạn chế với ứng dụng người dùng, chủ yếu dành cho các tiến trình hệ thống.
setsid	Đặt PID của tiến trình vào nhóm tiến trình hoặc tạo một nhóm mới.	iOS không cho phép quản lý nhóm tiến trình, chỉ dành cho hệ thống.
getpgrp	Trả về ID của nhóm tiến trình hiện tại.	Hạn chế với ứng dụng người dùng, chỉ dành cho hệ thống.
Quản lý bộ nhớ		
nmap, munmap	Ánh xạ hoặc gỡ bỏ ánh xạ tệp vào bộ nhớ, giúp truy cập nhanh hơn đến các tệp lớn.	Ánh xạ hoặc gỡ bỏ ánh xạ tệp vào bộ nhớ (cũng có thể chia sẻ bộ nhớ giữa các tiến trình).
mprotect	Thay đổi quyền truy cập trên vùng bộ nhớ đã ánh xạ.	Thay đổi quyền truy cập trên vùng bộ nhớ đã ánh xạ.
msync	Đồng bộ hóa dữ liệu trong bộ nhớ với đĩa (quan trọng để bảo vệ dữ liệu không bị mất).	Đồng bộ hóa dữ liệu trong bộ nhớ với đĩa.
mlock, munlock	Khóa hoặc mở khóa bộ nhớ trong RAM, ngăn trao đổi bộ nhớ ra đĩa.	Khóa hoặc mở khóa bộ nhớ trong RAM (chỉ trong các ứng dụng đặc biệt).
shmget, shmat, shmdt, shmctl	Quản lý bộ nhớ chia sẻ, cho phép các tiến trình chia sẻ vùng nhớ chung.	iOS hạn chế sử dụng bộ nhớ chia sẻ, chủ yếu cho nội bộ hệ thống.
sbrk. brk	Điều chỉnh kích thước vùng heap của tiến trình.	Điều chỉnh kích thước vùng heap (được hệ thống quản lý chặt chẽ).

3. Quản lý bộ nhớ.

		MacOS	iOS
Phương thức quản lý bộ nhớ	**Bộ nhớ ảo**	Sử dụng bộ nhớ ảo, hỗ trợ phân trang và swap để lưu trữ các trang trên ổ cứng khi cần.	Sử dụng phân trang nhưng không hỗ trợ swap; bộ nhớ hạn chế khiến iOS cần tối ưu và đóng các ứng dụng khi bộ nhớ gần đầy.
	Bộ nhớ được bảo vệ	Ngăn chặn truy cập vào vùng nhớ không được phép; mỗi ứng dụng hoạt động trong không gian riêng biệt.	Mỗi ứng dụng hoạt động trong vùng nhớ riêng; hệ thống dừng ứng dụng khi cố gắng truy cập vùng không được phép.
	Automatic Reference Counting	Áp dụng cho cả macOS, giúp giải phóng bộ nhớ tự động khi đối tượng không còn tham chiếu.	Sử dụng ARC để giảm rò rỉ bộ nhớ; quản lý bộ đếm tham chiếu và tự động giải phóng bộ nhớ khi không còn cần thiết.
	Memory Warning	Không có cảnh báo bộ nhớ cụ thể vì macOS có swap, giúp tránh tình trạng đầy bộ nhớ vật lý.	Gửi cảnh báo bộ nhớ cho ứng dụng để giải phóng tài nguyên khi bộ nhớ sắp đầy.
Quản lý bộ nhớ ảo	**Swapping & Paging**	Sử dụng cả phân trang và swap để quản lý bộ nhớ ảo, giúp mở rộng bộ nhớ thực tế và hỗ trợ các ứng dụng yêu cầu nhiều bộ nhớ.	Không hỗ trợ swap do hạn chế tài nguyên; yêu cầu ứng dụng tối ưu và giải phóng bộ nhớ khi không cần.
	Unified Buffer Cache	Sử dụng bộ nhớ đệm hợp nhất cho cả dữ liệu ứng dụng và hệ thống, giúp tối ưu hóa bộ nhớ và giảm tải I/O.	Không có bộ nhớ đệm hợp nhất; bộ nhớ đệm và các trang không quan trọng sẽ tự động xóa khi bộ nhớ cần được giải phóng.
Cơ chế thay	**Least**	Sử dụng LRU để quản lý bộ	Không dùng LRU do không có

thế bộ nhớ	**Recently Used**	nhớ, cho phép swap ra ổ đĩa các trang ít được sử dụng nhất khi cần bộ nhớ trống.	swap; khi cần bộ nhớ trống, hệ thống gửi cảnh báo và đóng các ứng dụng nền để giải phóng bộ nhớ.
	Inactive List & Active List	Quản lý bộ nhớ với hai danh sách Inactive và Active; các trang trong Inactive List sẽ bị swap ra nếu bộ nhớ đầy.	Đóng các ứng dụng nền khi không đủ bộ nhớ cho ứng dụng hiện tại, giúp tối ưu hiệu năng và bộ nhớ cho ứng dụng chính.

4. **Quản lý file.**

	MacOS	**iOS**
Hệ thống	APFS _ Apple File System	APFS
Mô hình truy cập	Dùng hệ thống miền để tổ chức và kiểm soát quyền truy cập (User, Local, Network, System). Các quyền được chia thành nhiều cấp độ khác nhau.	Sử dụng mô hình sandbox, mỗi ứng dụng chỉ có thể truy cập vào không gian riêng của nó, hạn chế khả năng truy cập tệp của ứng dụng khác hoặc hệ thống.
Quản lý không gian lưu trữ	Hỗ trợ Space Sharing: các volumes có thể chia sẻ không gian vật lý trong cùng một container, giúp tối ưu hóa không gian lưu trữ.	Cũng hỗ trợ Space Sharing giữa các volumes trong container.
Các loại thư mục chính	- /Applications: Chứa các ứng dụng của hệ thống. - /Library: Chứa các tệp hệ thống và dữ liệu ứng dụng cho tất cả người dùng. - /Users: Chứa thư mục của từng người dùng với các thư mục con như Desktop,	- Document/: Lưu các tệp do người dùng tạo hoặc cần thiết cho ứng dụng, được sao lưu trên iTunes/iCloud. - Library/: Lưu dữ liệu hỗ trợ ứng dụng.

	Documents, Downloads. - /System: Chứa tệp hệ điều hành cốt lõi. - /tmp: Chứa tệp tạm thời của hệ thống.	- tmp/: Lưu tệp tạm thời, không được sao lưu.
Mã hóa	Hỗ trợ mã hóa AES-256 ở cấp độ volume hoặc file thông qua FileVault, bảo mật toàn bộ ổ đĩa.	Mã hóa dữ liệu ứng dụng tự động do môi trường sandbox của iOS; mỗi ứng dụng được bảo vệ bởi các khóa mã hóa riêng.
Snapshot & Cloning	Hỗ trợ snapshots (ảnh chụp nhanh của volume) và cloning (nhân bản file/thư mục mà không sao chép dữ liệu thực sự).	Hỗ trợ snapshots và cloning ở mức hệ thống nhưng bị hạn chế đối với ứng dụng do mô hình sandbox.
Quyền truy cập	Sử dụng ACLs và quyền BSD để kiểm soát quyền truy cập tệp chi tiết và theo nhóm người dùng.	Quyền truy cập tự động kiểm soát bởi hệ thống với môi trường sandbox của từng ứng dụng, hạn chế quyền người dùng khác truy cập.
Tính an toàn	Sử dụng ACLs, quyền BSD, và Journal để bảo vệ dữ liệu trong trường hợp gặp sự cố hệ thống.	Ứng dụng luôn hoạt động trong sandbox; hạn chế truy cập vào các tệp của ứng dụng khác hoặc hệ thống, bảo vệ dữ liệu người dùng tự động.

VI. Kết luận

- Qua quá trình tìm hiểu về hai hệ điều hành MacOS và iOS, chúng ta có thể thấy rõ những đặc điểm nổi bật và sự khác biệt giữa chúng. Cả hai đều là sản phẩm của Apple, được xây dựng dựa trên nền tảng Unix và chia sẻ nhiều yếu tố chung trong thiết kế, tính năng và cấu trúc bảo mật. MacOS với tính năng mạnh mẽ và sự ổn định dành cho các thiết bị máy tính để bàn và máy tính xách tay, cung cấp cho người dùng một hệ điều hành linh hoạt và hiệu quả trong công việc sáng tạo và chuyên nghiệp. Ngược lại, iOS được thiết kế chuyên biệt cho các thiết bị di động như iPhone và iPad, chú trọng vào tính bảo mật, giao diện đơn giản và tối ưu hóa trải nghiệm người dùng trong môi trường di động.
- Hai hệ điều hành này cũng cho thấy sự phát triển mạnh mẽ và cải tiến liên tục từ Apple, với các phiên bản mới được ra mắt đều đặn, không chỉ để nâng cao hiệu năng mà còn để đồng bộ hóa trải nghiệm giữa các thiết bị trong hệ sinh thái Apple. Sự kết hợp giữa MacOS và iOS đã tạo nên một môi trường công nghệ liền mạch, đáp ứng nhu cầu đa dạng của người dùng và mở ra nhiều cơ hội cho sự phát triển công nghệ tương lai. Việc hiểu rõ về hai hệ điều hành này giúp người dùng khai thác tối đa lợi ích từ các sản phẩm của Apple, đồng thời tạo cơ sở cho các nhà phát triển ứng dụng trong việc xây dựng các phần mềm hiệu quả, tiện dụng và an toàn hơn.

Tài liệu tham khảo

https://gs.statcounter.com/os-market-share/?form=MG0AV3

https://yeuphancung.com/tim-hieu-apfs/#0

https://developer.apple.com/library/archive/documentation/FileManagement/Conceptual/FileSystemProgrammingGuide/FileSystemOverview/FileSystemOverview.html#//apple_ref/doc/uid/TP40010672-CH2-SW2

https://docs.swift.org/swift-book/documentation/the-swift-programming-language/automaticreferencecounting/

https://viblo.asia/p/co-che-quan-ly-bo-nho-trong-ios-arc-va-mrc-Ljy5VY7blra