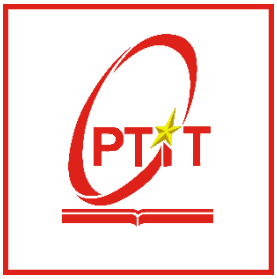
BỘ KHOA HỌC VÀ CÔNG NGHỆ
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
----------o0o----------



BÀI TIỂU LUẬN HỆ THỐNG PHÂN TÁN

PHÂN TÍCH CHUYÊN SÂU VỀ KIẾN TRÚC ỨNG DỤNG CHAT P2P PHI TẬP TRUNG

Người hướng dẫn: TS. Kim Ngọc Bách

Học viên: Nghiêm Quốc Việt – B24CHHT100

Đàm Văn Trung – B24CHHT095

Mai Việt Hùng – B24CHHT077

Lớp: M24CQHT02-B

Hà Nội - 2025

PHỤ LỤC

CHƯƠNG 1: GIỚI THIỆU MÔ HÌNH GIAO TIẾP PHI TẬP TRUNG

1.1. Bối cảnh chuyển dịch: Từ Client-Server đến mạng ngang hàng (P2P)

Trong lịch sử phát triển của Internet, kiến trúc Client-Server đã trở thành mô hình thống trị, định hình cách chúng ta tương tác với các dịch vụ kỹ thuật số. Trong mô hình này, một máy chủ trung tâm (server) đóng vai trò là cơ quan quyền lực, lưu trữ dữ liệu, xử lý logic và quản lý kết nối từ nhiều máy khách (client).1 Mặc dù mô hình này đã chứng tỏ hiệu quả trong việc cung cấp dịch vụ cho hàng tỷ người dùng, sự phụ thuộc vào một thực thể trung tâm cũng bộc lộ những điểm yếu cố hữu và ngày càng trở nên rõ rệt trong bối cảnh kỹ thuật số hiện đại.

Các điểm yếu chính của kiến trúc tập trung bao gồm:

- *Điểm lỗi đơn (Single Point of Failure):* Toàn bộ hệ thống phụ thuộc vào sự ổn định và hoạt động của máy chủ trung tâm. Bất kỳ sự cố nào tại máy chủ, dù là do lỗi phần cứng, phần mềm hay tấn công từ chối dịch vụ (DDoS), đều có thể làm tê liệt hoàn toàn dịch vụ, ảnh hưởng đến tất cả người dùng.
- *Rủi ro Kiểm duyệt và Giám sát:* Cơ quan trung ương sở hữu toàn quyền kiểm soát dữ liệu. Điều này tạo ra khả năng kiểm duyệt nội dung, chặn tài khoản hoặc giám sát các hoạt động của người dùng mà không có sự minh bạch hoặc đồng thuận rõ ràng. Người dùng buộc phải tin tưởng vào chính sách và đạo đức của nhà cung cấp dịch vụ.
- *Tập trung hóa Dữ liệu (Data Silos):* Dữ liệu người dùng bị khóa trong các hệ thống độc quyền, tạo ra các "hầm chứa dữ liệu" (data silos). Người dùng có ít quyền kiểm soát đối với thông tin cá nhân của mình, và dữ liệu này thường được các công ty khai thác cho mục đích thương mại. Việc di chuyển dữ liệu giữa các dịch vụ khác nhau trở nên khó khăn, làm giảm tính linh hoạt và quyền tự chủ của người dùng.

Để giải quyết những thách thức này, một mô hình kiến trúc thay thế đã nổi lên: mạng ngang hàng (Peer-to-Peer - P2P). Trong kiến trúc P2P, không có sự phân biệt rõ ràng giữa client và server. Mỗi nút trong mạng, được gọi là "peer"

(ngang hàng), hoạt động bình đẳng, vừa có khả năng yêu cầu dịch vụ (như một client) vừa có khả năng cung cấp dịch vụ (như một server). Bằng cách phân tán trách nhiệm lưu trữ, xử lý và truyền tải dữ liệu trên toàn bộ mạng lưới các peer, kiến trúc P2P giải quyết trực tiếp các vấn đề của mô hình tập trung. Nó loại bỏ điểm lỗi đơn, tăng cường khả năng chống chịu lỗi, và trao lại quyền kiểm soát dữ liệu cho chính người dùng. Sự chuyển dịch này không chỉ là một sự thay đổi về mặt kỹ thuật; nó còn đại diện cho một sự thay đổi triết học về quyền lực và quyền kiểm soát trong không gian kỹ thuật số, hướng tới một mô hình trao quyền cho cá nhân thay vì các tổ chức tập trung.

1.2. Định nghĩa và nguyên tắc cốt lõi của ứng dụng Phi tập trung (dApp)

Ứng dụng phi tập trung (Decentralized Application - dApp) là sự hiện thực hóa của triết lý P2P. Đây là các ứng dụng phần mềm chạy trên một mạng lưới P2P thay vì trên một máy chủ đơn lẻ. Nhiều dApp hiện đại tận dụng công nghệ blockchain và hợp đồng thông minh (smart contracts) để tự động hóa các quy tắc và hoạt động một cách tự chủ mà không cần sự can thiệp của con người hay một cơ quan trung gian. Một ứng dụng chat P2P được xây dựng dưới dạng dApp sẽ kế thừa các nguyên tắc nền tảng sau đây, mỗi nguyên tắc là một phản ứng trực tiếp đối với những thiếu sót của các ứng dụng tập trung truyền thống:

- *Mã nguồn mở (Open Source):* Mã nguồn của dApp thường được công khai. Điều này cho phép bất kỳ ai cũng có thể kiểm tra, xác minh logic hoạt động và đóng góp vào sự phát triển của ứng dụng. Tính minh bạch này là nền tảng để xây dựng lòng tin trong một môi trường vốn "không tin cậy" (trustless), nơi người dùng không cần phải tin vào nhà phát triển mà có thể tin vào mã nguồn đã được kiểm chứng.

- *Phi tập trung (Decentralization):* Đây là đặc tính cốt lõi. Không có một cá nhân hay tổ chức nào nắm giữ quyền kiểm soát đa số đối với ứng dụng. Các quyết định quan trọng, nếu có, thường được đưa ra thông qua cơ chế đồng thuận của cộng đồng người dùng. Đặc tính này giúp ứng dụng chống lại sự kiểm soát,

thao túng và đảm bảo hoạt động liên tục ngay cả khi một phần của mạng lưới gặp sự cố

- *Bảo mật mật mã (Cryptographic Security):* Dữ liệu và giao dịch trong dApp được bảo vệ bằng các thuật toán mật mã mạnh mẽ. Dữ liệu được lưu trữ trên một sổ cái phân tán (như blockchain) thường là bất biến, có nghĩa là một khi đã được ghi lại, nó không thể bị thay đổi hoặc xóa bỏ. Điều này đảm bảo tính toàn vẹn và chống giả mạo cho lịch sử giao tiếp.

- *Quyền tự chủ của người dùng (User Autonomy):* Trong một dApp, người dùng sở hữu và kiểm soát hoàn toàn dữ liệu và danh tính số của mình. Họ có quyền quyết định chia sẻ thông tin gì, với ai và khi nào. Mô hình này trái ngược hoàn toàn với các ứng dụng tập trung, nơi dữ liệu người dùng thuộc sở hữu của công ty cung cấp dịch vụ.

Mối quan hệ nhân quả giữa các thiếu sót của mô hình tập trung và các đặc tính của dApp là rất rõ ràng. Điểm lỗi đơn trong hệ thống tập trung được giải quyết bằng bản chất phân tán của P2P. Nguy cơ kiểm duyệt được hóa giải bằng một hệ thống không có cơ quan kiểm soát trung ương. Các "hầm chứa dữ liệu" bị phá vỡ bởi một mô hình nơi người dùng tự chủ về dữ liệu của mình. Do đó, kiến trúc phi tập trung không phải là một lựa chọn tùy ý, mà là một bước tiến hóa logic, được thúc đẩy bởi những hạn chế có thể quan sát được của mô hình tiền nhiệm.

1.3. Mục tiêu và phạm vi của báo cáo

Mục tiêu chính của báo cáo này là tiến hành một phân tích kỹ thuật chuyên sâu về một kiến trúc P2P thuần túy, khả thi cho một ứng dụng chat phi tập trung. Báo cáo sẽ không chỉ dừng lại ở việc mô tả các thành phần cấu thành hệ thống mà còn đi sâu vào việc làm sáng tỏ các nguyên lý hoạt động, sự tương tác phức tạp giữa chúng, và các quyết định thiết kế chiến lược đằng sau mỗi lựa chọn công nghệ.

Phạm vi của báo cáo sẽ tập trung vào việc phân tích chi tiết bốn module chính, được xem là bốn trụ cột cấu thành nên hệ thống:

+ Module định danh & xác thực người dùng: Nền tảng cho việc thiết lập danh tính và niềm tin

+ Module khám phá peer: Cơ chế để các nút tìm thấy và kết nối với nhau.

+ Module giao tiếp & lan truyền tin nhắn: Lõi của hệ thống, chịu trách nhiệm phân phối tin nhắn.

+ Module đồng bộ hóa trạng thái: Đảm bảo tính nhất quán cho lịch sử hội thoại.

Thông qua việc phân tích từng module và mối liên kết giữa chúng, báo cáo này sẽ cung cấp một cái nhìn toàn diện về cách xây dựng một môi trường giao tiếp bền vững, riêng tư và tự chủ trên nền tảng P2P.

CHƯƠNG 2: TỔNG QUAN KIẾN TRÚC HỆ THỐNG P2P

2.1. Mô hình mạng ngang hang thuần túy (Pure P2P)

Kiến trúc được phân tích trong báo cáo này dựa trên mô hình mạng ngang hàng thuần túy (Pure P2P). Đây là dạng phi tập trung triệt để nhất, trong đó tất cả các nút (peer) trong mạng đều hoàn toàn bình đẳng về vai trò và chức năng.7 Không tồn tại các nút có vai trò đặc biệt như "máy chủ trung tâm" (trong mô hình Napster đời đầu) hay "siêu nút" (super-peers) (trong các mạng lai như Gnutella 0.6). Mỗi peer trong mạng thuần túy tự chịu trách nhiệm cho các tác vụ cốt lõi như lưu trữ dữ liệu, định tuyến tin nhắn, và khám phá các peer khác.

Ưu điểm của mô hình Pure P2P:

- *Tính phân tán hoàn toàn:* Vì không có thành phần nào quan trọng hơn các thành phần khác, mạng lưới có khả năng chống chịu lỗi và chống kiểm duyệt ở mức độ cao nhất. Sự ra đi của một hoặc nhiều peer không làm sụp đổ toàn bộ hệ thống, miễn là vẫn còn các peer khác kết nối với nhau

- *Đơn giản về vai trò logic:* Việc phát triển và bảo trì trở nên đơn giản hơn ở cấp độ peer, vì tất cả các peer đều chạy cùng một bộ quy tắc và logic. Không cần phải quản lý các loại nút khác nhau với các hành vi khác nhau.

Thách thức cố hữu của mô hình Pure P2P:

- *Hiệu quả tìm kiếm:* Một trong những thách thức lớn nhất là làm thế nào để một peer có thể tìm thấy một tài nguyên cụ thể hoặc một peer khác một cách hiệu quả. Các cơ chế ban đầu như "flooding" (lũ lụt tin nhắn), trong đó một yêu cầu tìm kiếm được gửi đến tất cả các hàng xóm và lan truyền ra toàn mạng, có thể gây ra lưu lượng mạng khổng lồ và làm giảm hiệu suất.

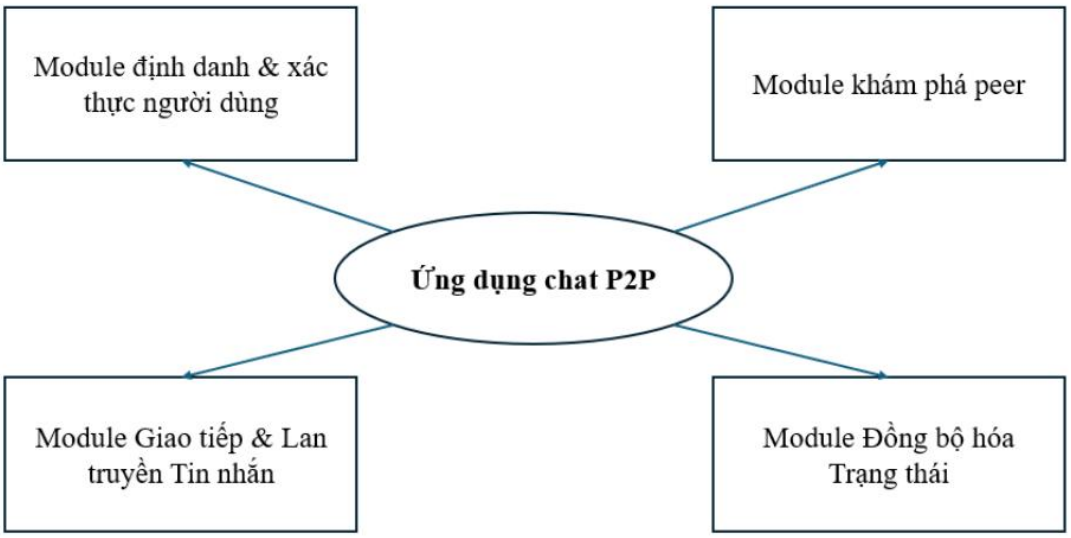
- *Vấn đề khởi động (Bootstrapping):* Khi một peer hoàn toàn mới muốn tham gia mạng lưới, nó phải đối mặt với câu hỏi cơ bản: "Làm thế nào để tìm thấy

peer đầu tiên để kết nối?". Nếu không có một danh sách các "điểm vào" được biết trước, việc khởi động có thể trở nên khó khăn.

Việc lựa chọn kiến trúc Pure P2P là một quyết định thiết kế có chủ đích. Nó buộc hệ thống phải giải quyết các vấn đề cơ bản của hệ thống phân tán từ những nguyên tắc đầu tiên, mà không dựa vào các "nạng đỡ" tập trung hóa như máy chủ theo dõi (tracker) hay các siêu nút. Điều này làm nổi bật sự đánh đổi cố hữu giữa mức độ phi tập trung tuyệt đối và hiệu suất/hiệu quả. Các giải pháp được trình bày trong các module tiếp theo chính là câu trả lời cho những thách thức do mô hình Pure P2P đặt ra.

2.2. Bốn trụ cột của hệ thống giao tiếp

Xây dựng một ứng dụng chat hoàn chỉnh trên nền tảng Pure P2P, kiến trúc hệ thống được cấu thành từ bốn module chính, hoạt động như bốn trụ cột hỗ trợ lẫn nhau. Mỗi module giải quyết một tập hợp các vấn đề cụ thể, và sự phối hợp nhịp nhàng giữa chúng tạo nên một hệ thống giao tiếp tự trị và bền vững.



- *Module định danh & xác thực người dùng:* Đây là nền tảng của sự tin cậy trong mạng. Nó cung cấp cơ chế để người dùng tạo và quản lý danh tính số của riêng mình và xác thực danh tính đó với các peer khác mà không cần dựa vào một bên thứ ba cấp chứng chỉ.

- *Module khám phá peer:* Đây là cơ chế "xã hội" của mạng lưới, cho phép các peer tìm thấy, kết nối và duy trì một danh sách các peer khác đang trực tuyến. Module này biến một tập hợp các nút riêng lẻ thành một mạng lưới kết nối và sống động.

- *Module Giao tiếp & lan truyền tin nhắn:* Đây là trái tim của ứng dụng, chịu trách nhiệm phân phối tin nhắn từ người gửi đến người nhận một cách hiệu quả, đáng tin cậy và có khả năng mở rộng trên toàn bộ mạng lưới.

- *Module đồng bộ hóa trạng thái:* Module này đảm bảo tính toàn vẹn và nhất quán của lịch sử trò chuyện. Nó cho phép các peer mới tham gia hoặc các peer quay trở lại sau một thời gian offline có thể cập nhật trạng thái cuộc hội thoại để có cùng một góc nhìn với phần còn lại của mạng.

2.3. Bảng so sánh kiến trúc: P2P với Client – Server

Để làm rõ hơn lý do lựa chọn kiến trúc P2P, bảng dưới đây tổng hợp và so sánh các đặc điểm chính của mô hình P2P và mô hình Client-Server truyền thống trên nhiều tiêu chí quan trọng.

Tiêu chí	**Kiến trúc Client-Server (Tập trung)**	**Kiến trúc Peer-to-Peer (Phi tập trung)**
Quản lý dữ liệu	Dữ liệu được lưu trữ và quản lý tập trung trên máy chủ.	Mỗi peer tự quản lý và lưu trữ dữ liệu của riêng mình.
Khả năng chịu lỗi	Thấp. Máy chủ trung tâm là một điểm lỗi đơn (single point of failure).	Cao. Sự cố của một hoặc nhiều peer không làm ảnh hưởng đến hoạt động của toàn mạng.
Khả năng mở rộng	Bị giới hạn bởi năng lực xử lý và băng thông của máy chủ trung tâm.	Mở rộng linh hoạt. Hiệu năng và khả năng cung cấp nội dung

Tiêu chí	Kiến trúc Client-Server (Tập trung)	Kiến trúc Peer-to-Peer (Phi tập trung)
		có thể tăng lên khi có thêm peer tham gia mạng.
Chi phí vận hành	Cao, đòi hỏi chi phí đầu tư và duy trì cơ sở hạ tầng máy chủ.	Thấp, vì không cần máy chủ chuyên dụng, tận dụng tài nguyên sẵn có của các peer.
Bảo mật	Dễ quản lý chính sách bảo mật tập trung, nhưng máy chủ là mục tiêu tấn công có giá trị cao ("honeypot").	Khó bị tấn công quy mô lớn, nhưng mỗi peer là một điểm yếu tiềm tàng và việc quản lý an ninh phức tạp hơn.
Độ phức tạp	Tương đối đơn giản để thiết lập và quản lý do có cấu trúc rõ ràng.	Phức tạp hơn trong việc triển khai các cơ chế khám phá peer, đồng bộ hóa trạng thái và định tuyến.

Bảng so sánh này cho thấy rõ ràng sự đánh đổi giữa hai mô hình. Trong khi Client-Server cung cấp sự đơn giản trong quản lý, P2P lại vượt trội về khả năng chịu lỗi, khả năng mở rộng và chi phí, những yếu tố then chốt để xây dựng một hệ thống giao tiếp bền vững và tự chủ.

CHƯƠNG 3: PHÂN TÍCH CHUYÊN SÂU CÁC THÀNH PHẦN HỆ THỐNG

3.1. Module Định danh & Xác thực Người dùng: Xây dựng Niềm tin trong Môi trường "Trustless"

Trong một hệ thống phi tập trung, không có cơ quan trung ương nào để xác thực danh tính người dùng. Do đó, việc xây dựng một cơ chế định danh và xác thực mạnh mẽ, không phụ thuộc vào bên thứ ba là yêu cầu cơ bản và quan trọng nhất. Module này giải quyết vấn đề "bạn là ai?" và "làm thế nào để chứng minh điều đó?" trong một môi trường "trustless" (không tin cậy).

3.1.1. Nguyên lý định danh phi tập trung (Decentralized Identity - DID)

Nền tảng của module này là khái niệm Định danh Phi tập trung (DID), một mô hình mới cho phép người dùng sở hữu và kiểm soát hoàn toàn danh tính số của mình.[6] Công nghệ cốt lõi đằng sau DID là Mật mã hóa Khóa công khai (Public Key Cryptography - PKC).

Trong hệ thống này, mỗi người dùng tự tạo ra một cặp khóa mật mã bất đối xứng:

- *Khóa riêng tư (Private Key):* Đây là một chuỗi dữ liệu ngẫu nhiên, được người dùng giữ bí mật tuyệt đối. Khóa riêng tư tương đương với "chìa khóa" của danh tính số. Bất kỳ ai sở hữu khóa riêng tư đều có toàn quyền kiểm soát danh tính tương ứng. Do đó, việc bảo vệ khóa riêng tư là tối quan trọng.
- *Khóa công khai (Public Key):* Được tạo ra từ khóa riêng tư thông qua một thuật toán toán học một chiều. Khóa công khai có thể được chia sẻ rộng rãi mà không gây rủi ro cho khóa riêng tư. Nó hoạt động như một định danh (identifier) duy nhất và có thể xác minh công khai cho người dùng, tương tự như một địa chỉ email hay số tài khoản ngân hàng trong thế giới tập trung.

Mô hình này hiện thực hóa khái niệm quyền tự chủ về danh tính (Self-Sovereign Identity). Người dùng không cần phải "đăng ký" tài khoản với một nhà cung cấp dịch vụ. Thay vào đó, họ tự tạo ra danh tính của mình, quản lý nó trong một "ví số" (digital wallet) và có thể sử dụng danh tính này trên nhiều nền tảng khác nhau mà không bị ràng buộc. Điều này tạo ra một sự tách biệt cơ bản và mạnh mẽ: hành động *chứng minh danh tính* được tách rời khỏi *dịch vụ đang được sử dụng*. Ứng dụng chat trong trường hợp này không còn là người "giữ" thông tin đăng nhập nhạy cảm (như mật khẩu); vai trò của nó được giảm xuống chỉ còn là người "xác minh" các bằng chứng mật mã công khai. Sự thay đổi này làm giảm đáng kể gánh nặng và rủi ro bảo mật cho chính ứng dụng.

3.1.2. Quy trình xác thực dựa trên chữ ký số

Với nền tảng DID, việc xác thực người dùng không còn dựa vào mật khẩu. Thay vào đó, hệ thống sử dụng một quy trình xác thực dựa trên chữ ký số, an toàn và hiệu quả hơn. Luồng hoạt động diễn ra như sau:

- *Thách thức (Challenge):* Khi một peer (Peer A) muốn xác thực một peer khác (Peer B), Peer A sẽ tạo ra một thông điệp ngẫu nhiên, duy nhất, gọi là "thông điệp thách thức" (challenge message) và gửi nó cho Peer B.

- *Ký (Signing):* Peer B nhận được thông điệp thách thức. Thay vì gửi lại bất kỳ thông tin bí mật nào, Peer B sử dụng khóa riêng tư của mình để "ký" lên thông điệp thách thức đó. Quá trình ký này tạo ra một chuỗi dữ liệu mới gọi là chữ ký số (digital signature). Chữ ký này là duy nhất cho cả thông điệp và khóa riêng tư đã sử dụng.

- *Phản hồi (Response):* Peer B gửi lại thông điệp thách thức ban đầu cùng với chữ ký số vừa tạo cho Peer A.

- *Xác minh (Verification):* Peer A nhận được phản hồi. Bây giờ, Peer A thực hiện một phép toán sử dụng ba yếu tố: thông điệp thách thức gốc, chữ ký số nhận được, và khóa công khai của Peer B (mà Peer A đã biết trước đó). Nếu chữ

ký hợp lệ, phép toán sẽ thành công. Điều này chứng minh một cách toán học rằng người sở hữu khóa riêng tư tương ứng với khóa công khai đó chính là người đã tạo ra chữ ký. Do đó, Peer A có thể tin rằng mình đang giao tiếp với Peer B thật sự, mà không cần Peer B phải tiết lộ bất kỳ thông tin bí mật nào.

Quy trình này không chỉ là một hệ thống đăng nhập. Nó biến danh tính thành một "nguyên thủy" (primitive) có thể kết hợp và sử dụng lại được trên toàn bộ web phi tập trung. Một DID có thể được liên kết với nhiều "chứng thực có thể xác minh" (Verifiable Credentials) khác nhau, chẳng hạn như bằng cấp do trường đại học cấp, hoặc giấy phép lái xe do chính phủ cấp. Ứng dụng chat sau này có thể yêu cầu người dùng chứng minh một thuộc tính cụ thể (ví dụ: "chứng minh bạn là thành viên của một tổ chức") mà không cần biết các thông tin cá nhân khác, thông qua cơ chế "tiết lộ có chọn lọc" (selective disclosure).[6] Điều này mở ra một mô hình tương tác phong phú, an toàn và bảo vệ quyền riêng tư vượt xa các hệ thống đăng nhập truyền thống.

3.2. Module Khám phá peer: Tìm kiếm và kết nối trong mạng lưới năng động

Sau khi có danh tính, một peer cần phải tìm thấy các peer khác để bắt đầu giao tiếp. Module Khám phá Peer giải quyết vấn đề cơ bản này: làm thế nào để xây dựng và duy trì một bản đồ mạng lưới trong một môi trường mà các peer liên tục tham gia và rời đi. Vấn đề này có thể được chia thành hai giai đoạn riêng biệt: khám phá ban đầu và duy trì trạng thái hiện diện.

3.2.1. Khám phá Ban đầu trên Mạng Cục bộ qua UDP Broadcast

Đối với một peer mới tham gia, cách đơn giản và hiệu quả nhất để tìm các peer khác trong cùng một mạng cục bộ (ví dụ, cùng một mạng Wi-Fi) là sử dụng UDP Broadcast.

- *Nguyên lý hoạt động:* Peer mới sẽ tạo một gói tin nhỏ chứa các thông tin cần thiết như khóa công khai (định danh) và số cổng mà nó đang lắng nghe kết

nối. Gói tin này sau đó được gửi đến địa chỉ broadcast của mạng (ví dụ: 255.255.255.255 hoặc địa chỉ broadcast cụ thể của subnet). Việc gửi đến địa chỉ này đảm bảo rằng gói tin sẽ được tất cả các thiết bị trong cùng mạng cục bộ nhận được.

- *Lắng nghe và phản hồi:* Tất cả các peer khác trong ứng dụng chat đang hoạt động trên mạng cục bộ sẽ liên tục lắng nghe trên một cổng UDP được quy định trước. Khi nhận được gói tin broadcast từ peer mới, chúng sẽ phân tích thông tin, thêm peer mới vào danh sách các peer đã biết ("danh bạ") của mình. Sau đó, chúng sẽ gửi lại một gói tin trực tiếp (unicast) đến địa chỉ IP và cổng của peer mới, chứa thông tin của chính chúng để peer mới cũng có thể cập nhật danh bạ.

- *Lựa chọn UDP:* Giao thức UDP (User Datagram Protocol) được lựa chọn cho mục đích này vì nó là "connectionless" (không yêu cầu thiết lập kết nối). Điều này hoàn toàn phù hợp với bản chất "gửi và quên" (fire-and-forget) của broadcast, nơi người gửi không cần biết ai đã nhận được tin nhắn. UDP cũng có overhead (dữ liệu phụ trợ) thấp hơn so với TCP, giúp cho quá trình khám phá diễn ra nhanh chóng và nhẹ nhàng.

3.2.2. Duy trì trạng thái hiện diện với cơ chế Heartbeat

Mạng P2P có tính chất rất năng động, với hiện tượng "churn" – các peer liên tục tham gia, rời đi, hoặc bị mất kết nối. Do đó, một danh sách peer được tạo ra từ quá trình khám phá ban đầu sẽ nhanh chóng trở nên lỗi thời. Để giải quyết vấn đề này, hệ thống cần một cơ chế để xác minh sự "sống" (liveness) của các peer đã biết.

- *Cơ chế Heartbeat:* Mỗi peer sẽ định kỳ, ví dụ mỗi 30 giây, gửi một gói tin nhỏ, nhẹ gọi là "heartbeat" (nhịp tim) đến một tập hợp hoặc tất cả các peer trong danh bạ của nó.[18] Gói tin này không chứa dữ liệu gì quan trọng ngoài việc báo hiệu rằng "tôi vẫn đang trực tuyến".

- *Xử lý lỗi và cập nhật danh sách:* Mỗi peer cũng theo dõi thời gian nhận được heartbeat cuối cùng từ các peer khác. Nếu một peer không nhận được heartbeat từ một peer cụ thể trong một khoảng thời gian chờ định trước (ví dụ, 3 lần chu kỳ heartbeat, tức 90 giây), nó sẽ giả định rằng peer đó đã ngoại tuyến (offline) hoặc không thể truy cập được. Peer bị coi là "chết" sẽ bị tạm thời hoặc vĩnh viễn loại bỏ khỏi danh sách các peer đang hoạt động.

Kiến trúc này sử dụng một giải pháp hai giai đoạn cho một vấn đề hai phần. UDP Broadcast là công cụ phù hợp cho việc tìm kiếm ban đầu trên phạm vi rộng nhưng chi phí thấp (trong mạng cục bộ), trong khi Heartbeat là công cụ lý tưởng cho việc xác minh trạng thái liên tục, có mục tiêu và chi phí thấp với các peer đã biết. Tuy nhiên, cần lưu ý rằng cơ chế UDP Broadcast chỉ giới hạn trong mạng cục bộ. Để ứng dụng có thể hoạt động trên quy mô toàn cầu, kết nối các peer qua Internet, module này cần được bổ sung một lớp khám phá toàn cầu, chẳng hạn như sử dụng Bảng băm Phân tán (Distributed Hash Table - DHT). Lớp khám phá cục bộ đóng vai trò là tầng đầu tiên, nhanh nhất, nhưng một tầng toàn cầu là không thể thiếu để hệ thống có thể mở rộng thực sự.

3.3. Module Giao tiếp & lan truyền tin nhắn: Hiệu quả phân phối thông qua giao thức Gossip

Khi các peer đã tìm thấy và xác thực lẫn nhau, chúng cần một cơ chế để trao đổi tin nhắn. Trong một mạng P2P lớn, việc một peer gửi tin nhắn đến tất cả các peer khác một cách trực tiếp (broadcasting) là không khả thi vì sẽ tạo ra một "cơn bão" lưu lượng mạng. Giao thức Gossip, hay còn gọi là Giao thức dịch bệnh (Epidemic Protocol), cung cấp một giải pháp thanh lịch, mạnh mẽ và có khả năng mở rộng cho vấn đề lan truyền thông tin này.

3.3.1. Nguyên lý Giao thức Gossip (Epidemic Protocol)

Giao thức Gossip hoạt động bằng cách mô phỏng cách tin đồn lan truyền trong một cộng đồng xã hội hoặc cách một dịch bệnh lây lan trong một quần thể.[37]

- *Lan truyền ngẫu nhiên:* Khi một peer (nút nguồn) có một tin nhắn mới cần lan truyền, nó không cố gắng gửi tin nhắn đó cho tất cả mọi người. Thay vào đó, nó chọn ngẫu nhiên một số lượng nhỏ các peer từ danh sách hàng xóm của mình (số lượng này được gọi là "fanout") và chỉ gửi tin nhắn cho họ.

- *Lây lan theo cấp số nhân:* Các peer lần đầu tiên nhận được tin nhắn này (trở thành "bị nhiễm") sẽ lặp lại chính xác quy trình trên: chúng cũng chọn ngẫu nhiên một số peer khác từ danh sách hàng xóm của mình và chuyển tiếp tin nhắn. Quá trình này tiếp diễn, tạo ra một hiệu ứng lan truyền theo cấp số nhân. Với mỗi "vòng" gossip, số lượng peer biết về tin nhắn tăng lên đáng kể, đảm bảo rằng tin nhắn sẽ nhanh chóng đến được gần như toàn bộ mạng lưới với xác suất rất cao.

Hệ thống này hoạt động dựa trên các đảm bảo xác suất thay vì các đảm bảo tất định. Nó không cam kết 100% tin nhắn sẽ được gửi đến tất cả mọi người trong một khoảng thời gian cố định, nhưng nó đảm bảo rằng với xác suất cực cao, tin nhắn sẽ lan truyền đến toàn mạng sau một số vòng lặp theo hàm logarit của số lượng nút. Điều này đòi hỏi các nhà phát triển phải thay đổi tư duy, thiết kế logic ứng dụng có khả năng xử lý các tình huống như tin nhắn đến không theo thứ tự hoặc có độ trễ không xác định.

3.3.2. Phân tích ưu và nhược điểm

Việc lựa chọn giao thức Gossip mang lại nhiều lợi ích nhưng cũng đi kèm với những sự đánh đổi cần được xem xét cẩn thận.

Ưu điểm:

- *Khả năng mở rộng (Scalability):* Đây là ưu điểm lớn nhất. Vì mỗi nút chỉ cần giao tiếp với một số lượng nhỏ các nút khác, tổng tải trên mạng được phân bổ đều. Giao thức hoạt động hiệu quả ngay cả khi mạng lưới phát triển lên đến hàng ngàn hoặc hàng triệu nút.

- *Khả năng chịu lỗi (Fault Tolerance):* Giao thức cực kỳ mạnh mẽ và bền bỉ. Do có nhiều đường truyền tin nhắn song song và ngẫu nhiên, việc một số nút bị lỗi hoặc một số kết nối mạng bị gián đoạn không ngăn cản được sự lan truyền của thông tin. Tin nhắn sẽ tự động tìm các con đường khác để đi.

- *Tính phi tập trung:* Giao thức Gossip là hiện thân của sự phi tập trung. Nó hoạt động hoàn hảo mà không cần bất kỳ nút điều phối, lãnh đạo hay trung tâm nào.

Nhược điểm:

- *Tính nhất quán cuối cùng (Eventual Consistency):* Gossip không đảm bảo rằng tất cả các nút sẽ nhận được tin nhắn cùng một lúc. Sẽ có một khoảng thời gian trễ khi thông tin đang lan truyền, trong đó các nút khác nhau có thể có các phiên bản trạng thái khác nhau. Giao thức chỉ đảm bảo rằng "cuối cùng" (eventually) tất cả các nút sẽ hội tụ về cùng một trạng thái.

- *Thông điệp trùng lặp và dư thừa:* Một nút có thể nhận cùng một tin nhắn nhiều lần từ các hàng xóm khác nhau. Điều này gây ra sự dư thừa và đòi hỏi mỗi nút phải có một cơ chế để nhận diện và loại bỏ các tin nhắn đã xử lý (ví dụ, bằng cách lưu lại ID của các tin nhắn đã thấy).

- *Độ trễ (Latency):* So với việc gửi trực tiếp (multicast), độ trễ để một tin nhắn đến được một nút ở xa có thể cao hơn, vì nó phải đi qua nhiều "bước nhảy" (hops) trung gian.

3.3.3. Các biến thể và chiến lược tối ưu hóa

Để cân bằng các ưu và nhược điểm, có nhiều biến thể của giao thức Gossip:

- *Push Gossip:* Một nút có thông tin mới sẽ chủ động "đẩy" (push) thông tin đó cho các hàng xóm ngẫu nhiên. Cách này giúp lan truyền thông tin mới rất nhanh nhưng có thể tạo ra nhiều lưu lượng mạng dư thừa khi hầu hết các nút đã biết thông tin.

- *Pull Gossip:* Một nút định kỳ "kéo" (pull) thông tin từ một hàng xóm ngẫu nhiên để hỏi xem có thông tin gì mới không. Cách này hiệu quả hơn trong việc giảm dư thừa nhưng có thể làm chậm quá trình lan truyền ban đầu.

- *Push-Pull Gossip:* Một biến thể lai kết hợp cả hai phương pháp. Một nút vừa đẩy thông tin mới của mình, vừa kéo thông tin từ hàng xóm. Đây thường là cách tiếp cận cân bằng và hiệu quả nhất trong thực tế.

Việc lựa chọn giao thức Gossip có một hệ quả kiến trúc quan trọng: nó đòi hỏi hệ thống phải chấp nhận và được thiết kế xung quanh mô hình Tính nhất quán cuối cùng. Bản chất ngẫu nhiên và không đồng bộ của Gossip làm cho việc đạt được tính nhất quán mạnh (strong consistency) trở nên bất khả thi. Do đó, lớp giao tiếp (Gossip) và lớp quản lý trạng thái (sẽ được thảo luận ở phần tiếp theo) phải được đồng thiết kế với một sự hiểu biết chung về các đảm bảo nhất quán của hệ thống.

3.4. Module Đồng bộ hóa trạng thái: Đạt được tính nhất quán cuối cùng (Eventual Consistency)

Module cuối cùng giải quyết một trong những vấn đề phức tạp nhất trong hệ thống phân tán: làm thế nào để đảm bảo rằng tất cả các peer, dù tham gia vào các thời điểm khác nhau, cuối cùng đều có một bản sao nhất quán của lịch sử trò chuyện.

3.4.1. Thách thức của tính nhất quán trong hệ thống phân tán

Trong một hệ thống chat tập trung, máy chủ là nguồn chân lý duy nhất (single source of truth). Mọi client đều đồng bộ với máy chủ này. Trong hệ thống P2P, không có nguồn chân lý nào như vậy. Mỗi peer có bản sao lịch sử của riêng mình, và các bản sao này có thể tạm thời không đồng nhất do độ trễ mạng hoặc do các peer offline/online.

- Kiến trúc này chấp nhận một sự đánh đổi có chủ đích bằng cách áp dụng mô hình Tính nhất quán cuối cùng (Eventual Consistency). Đây là một đảm bảo rằng, nếu không có cập nhật mới nào được thực hiện, tất cả các bản sao dữ liệu trên các peer cuối cùng sẽ hội tụ về cùng một trạng thái. Đây không phải là một "lỗi" hay một dạng nhất quán "yếu" theo nghĩa tiêu cực; đó là một lựa chọn kiến trúc chiến lược để ưu tiên

- Tính sẵn sàng cao (High Availability) và khả năng chịu phân vùng (Partition Tolerance), phù hợp với định lý CAP. Đối với một ứng dụng chat toàn cầu, khả năng gửi và nhận tin nhắn ngay cả khi một phần mạng bị cô lập thường quan trọng hơn việc đảm bảo mọi người dùng thấy mọi tin nhắn theo cùng một thứ tự tại cùng một nano giây.

Thách thức chính là: khi một peer mới tham gia hoặc một peer đã offline trong một thời gian dài quay trở lại, làm thế nào để nó có thể cập nhật lịch sử trò chuyện một cách hiệu quả mà không cần phải tải xuống toàn bộ dữ liệu từ một peer khác?

3.4.2. Kỹ thuật đồng bộ hóa hiệu quả với Merkle Trees

Merkle Tree (Cây Merkle) là một cấu trúc dữ liệu dựa trên hàm băm, cung cấp một giải pháp cực kỳ hiệu quả cho vấn đề đồng bộ hóa dữ liệu.

- *Cấu trúc Cây Merkle:* Hãy tưởng tượng toàn bộ lịch sử chat được chia thành các khối dữ liệu nhỏ (ví dụ, mỗi tin nhắn là một khối).

+ Mỗi khối dữ liệu được băm (hashed) để tạo ra một "nút lá" (leaf node) của cây.

+ Các cặp nút lá kề nhau sau đó được kết hợp và băm lại để tạo ra một "nút cha" (parent node).

+ Quá trình này được lặp lại lên các cấp của cây cho đến khi chỉ còn lại một nút duy nhất ở trên cùng, gọi là nút gốc (root hash).

Nút gốc này là một đại diện mật mã nhỏ gọn cho toàn bộ trạng thái của lịch sử chat. Bất kỳ thay đổi nào, dù là nhỏ nhất, ở bất kỳ tin nhắn nào, cũng sẽ dẫn đến một giá trị root hash hoàn toàn khác.

- *Quy trình so sánh và đồng bộ hóa:*

+ Bước 1: Khi hai peer (A và B) muốn đồng bộ hóa, chúng chỉ cần trao đổi **root hash** của mình.

+ Bước 2: Nếu root hash giống hệt nhau, điều đó có nghĩa là lịch sử chat của chúng đã hoàn toàn đồng bộ. Quá trình kết thúc mà không cần truyền thêm dữ liệu.

+ Bước 3: Nếu root hash khác nhau, chúng biết rằng có sự khác biệt. Peer A sẽ gửi các hash của các nút con ngay dưới nút gốc cho Peer B. Peer B so sánh chúng với các hash con của mình.

+ Bước 4: Chúng sẽ chỉ cần đi sâu vào (traverse) nhánh cây nào có hash không khớp. Bằng cách lặp lại quá trình này một cách đệ quy, chúng có thể nhanh chóng xác định chính xác khối dữ liệu (tin nhắn) nào bị thiếu hoặc khác biệt.

+ Bước 5: Cuối cùng, chỉ những khối dữ liệu bị thiếu thực sự mới được truyền qua mạng.

Lợi ích của phương pháp này là rất lớn. Nó biến một bài toán đồng bộ hóa có độ phức tạp O(N) (với N là tổng số tin nhắn) thành một bài toán có độ phức tạp O(log(N)). Thay vì phải so sánh toàn bộ dữ liệu, lượng dữ liệu cần trao đổi chỉ tỷ lệ với độ sâu của cây, giúp tiết kiệm băng thông một cách đáng kể và làm cho việc đồng bộ hóa trở nên khả thi ngay cả với lịch sử chat rất lớn.

3.4.3. Mở rộng với CRDTs (Conflict-free Replicated Data Types)

Để giải quyết các xung đột có thể xảy ra khi nhiều người dùng chỉnh sửa cùng một dữ liệu đồng thời (ví dụ: trong một tài liệu cộng tác), một khái niệm nâng cao hơn có thể được áp dụng là CRDTs (Conflict-free Replicated Data Types). CRDTs là các cấu trúc dữ liệu được thiết kế đặc biệt để cho phép các bản

sao được cập nhật độc lập mà không cần phối hợp, và cung cấp một hàm hợp nhất (merge function) đảm bảo rằng tất cả các bản sao cuối cùng sẽ hội tụ về cùng một trạng thái một cách toán học.

Khi kết hợp với Cây Merkle, chúng ta có khái niệm Merkle-CRDT**s**. Cấu trúc này sử dụng Cây Merkle không chỉ để đồng bộ hóa hiệu quả mà còn để mã hóa lịch sử nhân quả (causal history) của các cập nhật, tạo ra một hệ thống đồng bộ hóa trạng thái tự xác minh, mạnh mẽ và cực kỳ phù hợp cho các ứng dụng P2P thế hệ tiếp theo.

CHƯƠNG 4: KẾT LUẬN VÀ ĐÁNH GIÁ TỔNG THỂ

4.1. Tổng hợp kiến trúc hệ thống tích hợp

Báo cáo này đã phân tích một kiến trúc toàn diện cho ứng dụng chat P2P phi tập trung, được xây dựng trên bốn module chính hoạt động phối hợp nhịp nhàng. Sự thành công của toàn bộ hệ thống không nằm ở một module đơn lẻ nào, mà ở sự tương tác và phụ thuộc lẫn nhau một cách chặt chẽ giữa chúng, tạo thành một thể thống nhất và bền vững.

- *Module định danh & xác thực người dùng* đóng vai trò là nền móng, sử dụng mật mã khóa công khai để tạo ra các "diễn viên" có danh tính tự chủ và có thể xác minh trong mạng lưới. Nó giải quyết vấn đề niềm tin cơ bản mà không cần đến một cơ quan trung ương.

- *Module khám phá peer* hoạt động như cơ chế xã hội, cho phép các "diễn viên" này tìm thấy nhau thông qua UDP Broadcast và duy trì kết nối bằng cơ chế Heartbeat. Nó biến các cá thể riêng lẻ thành một "sân khấu" nơi các tương tác có thể diễn ra.

- *Module giao tiếp & lan truyền tin nhắn,* với Giao thức Gossip làm cốt lõi, cung cấp "kịch bản" cho việc giao tiếp. Nó định ra cách các diễn viên trên sân khấu trao đổi thông tin một cách hiệu quả, chịu lỗi và có khả năng mở rộng, đảm bảo rằng thông điệp được lan truyền rộng rãi.

- *Module đồng bộ hóa trạng thái* sử dụng các cấu trúc dữ liệu hiệu quả như Cây Merkle để đảm bảo rằng tất cả các diễn viên đều có cùng một "bộ nhớ" chung về những gì đã diễn ra trên sân khấu. Nó mang lại tính nhất quán cuối cùng cho lịch sử của cuộc hội thoại.

Không có module nào có thể hoạt động độc lập. Không có danh tính, việc khám phá và giao tiếp sẽ vô nghĩa. Không có cơ chế khám phá, các peer sẽ không bao giờ tìm thấy nhau để giao tiếp. Không có giao thức lan truyền hiệu quả, mạng lưới sẽ không thể mở rộng. Và không có đồng bộ hóa trạng thái, các cuộc hội thoại sẽ trở nên rời rạc và không nhất quán.

4.2. Khẳng định về một môi trường giao tiếp bền vững và tự chủ

Kiến trúc được phân tích đã chứng minh thành công trong việc đáp ứng các mục tiêu cốt lõi của một ứng dụng giao tiếp phi tập trung. Nó tạo ra một môi trường có các đặc tính sau:

- *Bền vững (Resilient):* Bằng cách loại bỏ hoàn toàn các điểm lỗi đơn và phân tán mọi chức năng trên toàn mạng, hệ thống có khả năng tự phục hồi và tiếp tục hoạt động ngay cả khi đối mặt với sự cố của các nút riêng lẻ hoặc sự phân mảnh mạng tạm thời.

- *Riêng tư (Private):* Giao tiếp được thực hiện trực tiếp giữa các peer (device-to-device) mà không qua máy chủ trung gian, kết hợp với một hệ thống định danh do người dùng toàn quyền kiểm soát, giúp giảm thiểu đáng kể bề mặt tấn công và nguy cơ bị giám sát, tăng cường quyền riêng tư cho người dùng.

- *Chống kiểm duyệt (Censorship-Resistant):* Bản chất phi tập trung triệt để của kiến trúc khiến cho việc kiểm soát, lọc hay chặn nội dung từ một cơ quan trung ương trở nên cực kỳ khó khăn, nếu không muốn nói là bất khả thi. Điều này thúc đẩy một môi trường giao tiếp tự do và cởi mở.

Nhìn xa hơn, kiến trúc này không chỉ là một giải pháp kỹ thuật cho một ứng dụng chat. Nó đại diện cho một mô hình mẫu, một bản thiết kế cho thế hệ tiếp theo của các ứng dụng xã hội và giao tiếp trong kỷ nguyên Web3 và Internet phi tập trung. Bằng cách đặt người dùng vào trung tâm, trao cho họ quyền sở hữu dữ liệu và danh tính, mô hình này mở đường cho một tương lai kỹ thuật số công bằng, minh bạch và tự chủ hơn, nơi quyền lực thực sự được trả lại cho người dùng.

TÀI LIỆU THAM KHẢO

1. Difference between Client-Server and Peer-to-Peer Network - GeeksforGeeks, truy cập vào tháng 8, 2025, https://www.geeksforgeeks.org/computer-networks/difference-between-client-server-and-peer-to-peer-network/
2. Client Server là gì? Tìm hiểu về mô hình Client Server A-Z - Vietnix, truy cập vào tháng 8, 2025, https://vietnix.vn/mo-hinh-client-server/
3. What are Decentralized Applications (dApps)? Characteristics & Use Cases - Fireblocks, truy cập vào tháng 8, 2025, https://www.fireblocks.com/glossary/decentralized-applications/
4. Ultimate Guide to DApps(Decentralized Apps): Build, Monetize & Scale - Rapid Innovation, truy cập vào tháng 8, 2025, https://www.rapidinnovation.io/post/decentralized-applications-dapps-101-comprehensive-guide-blockchain-developers-entrepreneurs
5. Decentralized Applications (dApps): Definition, Uses, Pros and Cons - Investopedia, truy cập vào tháng 8, 2025, https://www.investopedia.com/terms/d/decentralized-applications-dapps.asp
6. Decentralized identity – Put your users in control - One Identity, truy cập vào tháng 8, 2025, https://www.oneidentity.com/learn/what-is-a-decentralized-identity.aspx
7. Giải thích về mạng ngang hàng - Binance Academy, truy cập vào tháng 8, 2025, https://academy.binance.com/vi/articles/peer-to-peer-networks-explained
8. Định nghĩa mạng P2P là gì? Phân loại và 5 lợi ích của mạng ngang hàng P2P - Mstar Corp, truy cập vào tháng 8, 2025, https://mstarcorp.vn/p2p-la-gi/

9. P2P là gì? Hoạt động của mạng ngang hàng peer to peer - Viettel IDC, truy cập vào tháng 8, 2025, https://viettelidc.com.vn/tin-tuc/mang-luoi-ngang-hang-peer-to-peer
10. Ứng Dụng Phi Tập Trung (DApp) Là Gì & Tại Sao Chúng Được Sử Dụng? - Bybit Learn, truy cập vào tháng 8, 2025, https://learn.bybit.com/vi/blockchain/decentralized-apps-dapps
11. Dapp là gì? Ưu - nhược điểm & 4 dạng lừa đảo thường gặp - BitcoinVN.io, truy cập vào tháng 8, 2025, https://bitcoinvn.io/news/dapp-la-gi-ung-dung-phi-tap-trung-co-loi-gi-trong-doi-song/
12. Ứng dụng phi tập trung (DApps) là gì? - TabTrader, truy cập vào tháng 8, 2025, https://tabtrader.com/vi/academy/articles/what-are-decentralized-apps-dapps
13. Ứng Dụng Phi Tập Trung (DApp) Là Gì? - Binance Academy, truy cập vào tháng 8, 2025, https://academy.binance.com/vi/articles/what-are-decentralized-applications-dapps
14. Decentralized application - Wikipedia, truy cập vào tháng 8, 2025, https://en.wikipedia.org/wiki/Decentralized_application
15. What is a DApp? A Comprehensive Guide to Decentralized Applications | AuditFirst, truy cập vào tháng 8, 2025, https://auditfirst.io/blog/what-is-a-dapp
16. Decentralized Authentication Mechanism and Platforms - Identity Management Institute®, truy cập vào tháng 8, 2025, https://identitymanagementinstitute.org/decentralized-authentication-mechanism-and-platforms/
17. GIÁO TRÌNH Mạng ngang hàng và định tuyến trong mạng ngang hàng (P2P) - SlideShare, truy cập vào tháng 8, 2025, https://www.slideshare.net/slideshow/gio-trnh-mng-ngang-hng-v-nh-tuyn-trong-mng-ngang-hng-p2p/249846072

18. Chapter 35. ProActive Peer-to-Peer Infrastructure - Inria, truy cập vào tháng 8, 2025, http://www-sop.inria.fr/oasis/proactive2/doc/release-doc/html/p2p.html
19. P2P là gì? Mạng ngang hàng P2P hoạt động như thế nào? - Bizfly Cloud, truy cập vào tháng 8, 2025, https://bizflycloud.vn/tin-tuc/p2p-la-gi-20211112141719259.htm
20. Peer to Peer là gì? Tổng quan về mạng ngang hàng P2P - Interdata.vn, truy cập vào tháng 8, 2025, https://interdata.vn/blog/peer-to-peer-la-gi/
21. P2P là gì? Kiến trúc và ưu điểm của mạng ngang hàng - BKHOST, truy cập vào tháng 8, 2025, https://bkhost.vn/blog/peer-to-peer-p2p-mang-ngang-hang/
22. Peer to peer là gì? Mạng ngang hàng P2P hoạt động như nào? - Vietnix, truy cập vào tháng 8, 2025, https://vietnix.vn/peer-to-peer-la-gi/
23. So sánh Mô hình mạng ngang hàng (Peer to Peer) và máy khách-máy chủ (Client-Server) |, truy cập vào tháng 8, 2025, https://thegioimang.vn/dien-dan/threads/so-s%C3%A1nh-m%C3%B4-h%C3%ACnh-m%E1%BA%A1ng-ngang-h%C3%A0ng-peer-to-peer-v%C3%A0-m%C3%A1y-kh%C3%A1ch-m%C3%A1y-ch%E1%BB%A7-client-server.919/
24. Công nghệ giúp người dân không cần dùng giấy tờ công chứng khi giao dịch - Báo Mới, truy cập vào tháng 8, 2025, https://baomoi.com/cong-nghe-giup-nguoi-dan-khong-can-dung-giay-to-cong-chung-khi-giao-dich-c52691996.epi
25. Mã hóa khóa công khai (PKC) là gì? Giao thức mật mã có hai khóa riêng biệt | Gate.io, truy cập vào tháng 8, 2025, https://www.gate.com/vi/learn/articles/what-is-public-key-cryptography-pkc/311

26. Nhận dạng phi tập trung (DiD): Trụ cột của Web3 - Plisio, truy cập vào tháng 8, 2025, https://plisio.net/vi/blog/decentralized-identity-did-the-pillar-of-web3
27. Hiểu vai trò của khóa công khai và khóa riêng trong bảo mật tiền điện tử. - Plisio, truy cập vào tháng 8, 2025, https://plisio.net/vi/blog/understanding-the-role-of-public-and-private-keys-in-crypto-security
28. Decentralized Identity: The Ultimate Guide 2025 - Dock Labs, truy cập vào tháng 8, 2025, https://www.dock.io/post/decentralized-identity
29. Decentralized Identity Management in Distributed Systems - GeeksforGeeks, truy cập vào tháng 8, 2025, https://www.geeksforgeeks.org/system-design/decentralized-identity-management-in-distributed-systems/
30. A Blockchain-Based Decentralized Public Key Infrastructure for Information-Centric Networks - MDPI, truy cập vào tháng 8, 2025, https://www.mdpi.com/2078-2489/13/5/264
31. Network Discovery via UDP - Remote Control, Monitoring and the Internet - LAVA, truy cập vào tháng 8, 2025, https://lavag.org/topic/16630-network-discovery-via-udp/
32. How to do Network discovery using UDP broadcast - Stack Overflow, truy cập vào tháng 8, 2025, https://stackoverflow.com/questions/22852781/how-to-do-network-discovery-using-udp-broadcast
33. Building a P2P Chat Application in Go: A Learning Journey | by Jimmy Sinjaradze | Medium, truy cập vào tháng 8, 2025, https://medium.com/@jimsinjaradze/building-a-p2p-chat-application-in-go-a-learning-journey-8d7122897bf3
34. Peer Discovery over UDP - teukka.tech, truy cập vào tháng 8, 2025, https://teukka.tech/peer-discovery.html

35. P2P Peer Discovery - Jordan Santell, truy cập vào tháng 8, 2025, https://jsantell.com/p2p-peer-discovery/
36. Network Heartbeat Traffic Characterization - University of Calgary, truy cập vào tháng 8, 2025, https://cspages.ucalgary.ca/~cwill/talks/Heartbeats-CLW.pdf
37. Gossip là gì? Đi với giới từ gì? Những tác động của Gossip - CellphoneS, truy cập vào tháng 8, 2025, https://cellphones.com.vn/sforum/gossip-la-gi
38. Introduction to gossip/epidemic protocol and memberlist - DEV Community, truy cập vào tháng 8, 2025, https://dev.to/satrobit/introduction-to-gossip-epidemic-protocol-and-memberlist-59f1
39. Understanding Gossip protocol - Codemia, truy cập vào tháng 8, 2025, https://codemia.io/knowledge-hub/path/understanding_gossip_protocol
40. Gossip Protocol in Disrtibuted Systems - GeeksforGeeks, truy cập vào tháng 8, 2025, https://www.geeksforgeeks.org/system-design/gossip-protocol-in-disrtibuted-systems/
41. Gossip Protocol - System Design, truy cập vào tháng 8, 2025, https://systemdesign.one/gossip-protocol/
42. Gossip Protocol in Decentralised Networks | by Kode with KT - Medium, truy cập vào tháng 8, 2025, https://medium.com/@kartikjain42/gossip-protocol-in-decentralised-networks-83eee5eaa0ca
43. Mastering Gossip Protocols in Distributed Systems - Number Analytics, truy cập vào tháng 8, 2025, https://www.numberanalytics.com/blog/ultimate-guide-gossip-protocols-distributed-algorithms
44. Gossip Protocol Explained - High Scalability, truy cập vào tháng 8, 2025, https://highscalability.com/gossip-protocol-explained/
45. aerospike.com, truy cập vào tháng 8, 2025, https://aerospike.com/glossary/eventual-

consistency/#:~:text=In%20distributed%20systems%2C%20eventual%20consistency,arise%20due%20to%20concurrent%20updates.

46. Eventual consistency - Wikipedia, truy cập vào tháng 8, 2025, https://en.wikipedia.org/wiki/Eventual_consistency
47. Architecting for Scale [Part 2] : Eventual Consistency and Handling Large-Scale Data | by Abhishek Kumar | Medium, truy cập vào tháng 8, 2025, https://medium.com/@kumarabhishek0388/architecting-for-scale-part-2-eventual-consistency-and-handling-large-scale-data-bbb9e72fedd9
48. What is eventual consistency? - Aerospike, truy cập vào tháng 8, 2025, https://aerospike.com/glossary/eventual-consistency/
49. What is Eventual Consistency? Definition & FAQs - ScyllaDB, truy cập vào tháng 8, 2025, https://www.scylladb.com/glossary/eventual-consistency/
50. Anti-entropy Through Merkle Trees - Design Gurus, truy cập vào tháng 8, 2025, https://www.designgurus.io/course-play/grokking-the-advanced-system-design-interview/doc/antientropy-through-merkle-trees
51. OWebSync: Seamless Synchronization of Distributed Web Clients - Kristof Jannes, truy cập vào tháng 8, 2025, https://kristofjannes.com/papers/tpds_owebsync.pdf
52. Merkle Search Trees: Efficient State-Based CRDTs in Open Networks - ResearchGate, truy cập vào tháng 8, 2025, https://www.researchgate.net/publication/340304230_Merkle_Search_Trees_Efficient_State-Based_CRDTs_in_Open_Networks
53. Merkle-CRDTs (DRAFT) - Protocol Labs Research, truy cập vào tháng 8, 2025, https://research.protocol.ai/blog/2019/a-new-lab-for-resilient-networks-research/PL-TechRep-merkleCRDT-v0.1-Dec30.pdf
54. Merkle-CRDTs Merkle-DAGs meet CRDTs - Protocol Labs Research, truy cập vào tháng 8, 2025,

https://research.protocol.ai/publications/merkle-crdts-merkle-dags-meet-crdts/psaras2020.pdf

55. (PDF) Merkle-CRDTs: Merkle-DAGs meet CRDTs - ResearchGate, truy cập vào tháng 8, 2025, https://www.researchgate.net/publication/340374488_Merkle-CRDTs_Merkle-DAGs_meet_CRDTs