BỘ THÔNG TIN VÀ TRUYỀN THÔNG

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

-----□□□□□-----

BÁO CÁO

MÔN HỌC: CÁC HỆ THỐNG PHÂN TÁN

HỆ THỐNG CHAT NGANG HÀNG (P2P)

Giảng viên : **TS. Nguyễn Duy Phương**

Học viên : **Lê Đăng Khoa - B25CHHT031**

Đỗ Thị Ngọc Mai - B25CHHT039

Đặng Hoàng Đức - B25CHHT013

Lớp : **M25CQHT01-B**

HÀ NỘI – NĂM 2025

MỤC LỤC

I. Giới thiệu đề tài

1. Lý do chọn đề tài

Trong bối cảnh nhu cầu trao đổi thông tin ngày càng cao, các ứng dụng chat đóng vai trò quan trọng trong đời sống và công việc. Mô hình chat ngang hàng (P2P) cho phép các thiết bị giao tiếp trực tiếp với nhau mà không cần máy chủ trung tâm, giúp giảm chi phí hạ tầng, tăng tính riêng tư và giảm độ trễ.

Đề tài này đặt mục tiêu là tạo ra 1 ứng dụng cho một team nhỏ (< 20 người) có một ứng dụng để nói chuyện phiếm riêng với nhau trong mạng nội bộ của công ty

2. Mục tiêu đề tài

- Tìm hiểu mô hình mạng P2P
- Xây dựng ứng dụng chat cho phép các client giao tiếp trực tiếp
- Đảm bảo gửi/nhận tin nhắn theo thời gian thực
- Hiểu rõ cơ chế kết nối mạng, socket và truyền dữ liệu

II. Công nghệ sử dụng

- Ngôn ngữ: Python 3.10 (phù hợp với môi trường Conda).
- Giao thức truyền tải:
  - TCP: Giao tiếp P2P tin cậy.
  - UDP Broadcast: Cơ chế tự động tìm kiếm Discovery Server trong mạng LAN.
  - HTTP/WebSocket: Giao tiếp nội bộ giữa Backend (Python) và Frontend (Web UI).
- Framework: Flask (Python) để xây dựng Web UI cho từng nút mạng.
- Deployment: Docker & Docker Compose giúp đóng gói môi trường nhất quán trên nhiều máy tính.

III. Thiết kế hệ thống

1. Phân tích yêu cầu

1.1. Quản lý Danh tính & Truy cập

- Xác thực mật khẩu (Passphrase Authentication): Người dùng truy cập vào hệ thống thông qua một Passphrase cá nhân. Hệ thống tự động tạo mới tài khoản nếu tên người dùng chưa tồn tại.

- Bảo mật dữ liệu tĩnh (Data-at-rest Security): Toàn bộ dữ liệu chat và cấu hình tại máy người dùng phải được mã hóa AES-256 dựa trên Passphrase để chống truy cập trái phép từ file hệ thống.

1.2. Quản lý Mạng và Định danh

- Tự động phát hiện Server (UDP Broadcast): Các Node tự tìm kiếm Discovery Server trong LAN mà không cần nhập IP thủ công.
- Đăng ký trạng thái (Registration): Node tự động cập nhật IP/Port và Public Key lên danh bạ hệ thống khi Online.
- Đồng bộ danh sách bạn bè: Tự động cập nhật danh sách người dùng đang hoạt động (Active Peers) định kỳ (Polling).

1.3. Giao tiếp & Truyền tin

- Chat trực tiếp (Direct P2P Chat): Thiết lập kết nối TCP trực tiếp giữa hai máy để truyền tin mà không thông qua server trung gian.
- Lưu trữ lịch sử: Lưu lại toàn bộ nội dung hội thoại vào Local DB để truy xuất lại khi cần.
- Cơ chế Hàng đợi tin nhắn (Store-and-Forward): Lưu tin nhắn vào hàng đợi "Pending" khi người nhận offline.
- Tự động đẩy tin (Push) ngay khi phát hiện người nhận Online trở lại.
- Bảo mật truyền tin (E2EE): Mã hóa đầu cuối bằng RSA-2048, đảm bảo chỉ người nhận mới có thể giải mã nội dung tin nhắn.

1.4. Phân tích yêu cầu phi chức năng

Tiêu chí	Yêu cầu chi tiết
Tính sẵn sàng	Hệ thống vẫn hoạt động (chat 1-1) ngay cả khi Server danh bạ gặp sự cố (nếu đã biết IP của nhau).
Hiệu năng	Độ trễ tin nhắn trong mạng nội bộ < 100ms.
Khả năng mở rộng	Tối ưu hóa cho nhóm < 20 người, đảm bảo không bị nghẽn băng thông do cơ chế direct routing.
Tính tin cậy	Đảm bảo tin nhắn không bị mất (Lossless) thông qua cơ chế ACK (Acknowledge) trong giao thức TCP.
Tính toàn vẹn và bảo mật	Đảm bảo tin nhắn không bị đọc trộm bởi bên thứ ba và file lưu trữ không bị xâm nhập

Tính nhất quán	Đảm bảo thứ tự tin nhắn đồng nhất trên mọi thiết bị nhờ thuật toán Vector Clock

1.5. Mô hình luồng dữ liệu (Data Flow)

- Giai đoạn kết nối: Node A gửi yêu cầu lên Server để lấy IP của Node B.
- Giai đoạn gửi tin:
  - Nếu Node B Online: A thiết lập kết nối socket tới B và gửi tin.
  - Nếu Node B Offline: Tin nhắn vào hàng đợi tại Local DB của A, gắn cờ "Pending".
- Giai đoạn nhận tin: Node B online -> Server thông báo cho A -> A tự động đẩy tin nhắn "Pending" cho B.

1.6. Liệt kê các tính năng

Node - Node: Có thể chat trực tiếp, lưu trữ lịch sử chat, lưu dữ liệu chat chờ để gửi khi online, bảo mật thông tin, có thể tìm được nhau:

a. Có thể chat trực tiếp
b. Lưu trữ lịch sử chat
c. Lưu dữ liệu chat chờ để gửi khi online
d. Bảo mật thông tin
e. Có thể tìm được nhau

Các chức năng chính :

- UPDATE: Cập nhật danh sách người dùng mới nhất từ Server hoặc qua mạng LAN.
- PEERS: Xem danh sách những người đang online.
- CHAT \[username\] \[nội dung\]: Gửi tin nhắn mã hóa.
- HISTORY: Xem lịch sử trò chuyện (chỉ hiển thị các tin nhắn bạn có quyền đọc).
- EXIT: Thoát ứng dụng và lưu dữ liệu.

2. Thiết kế phần mềm

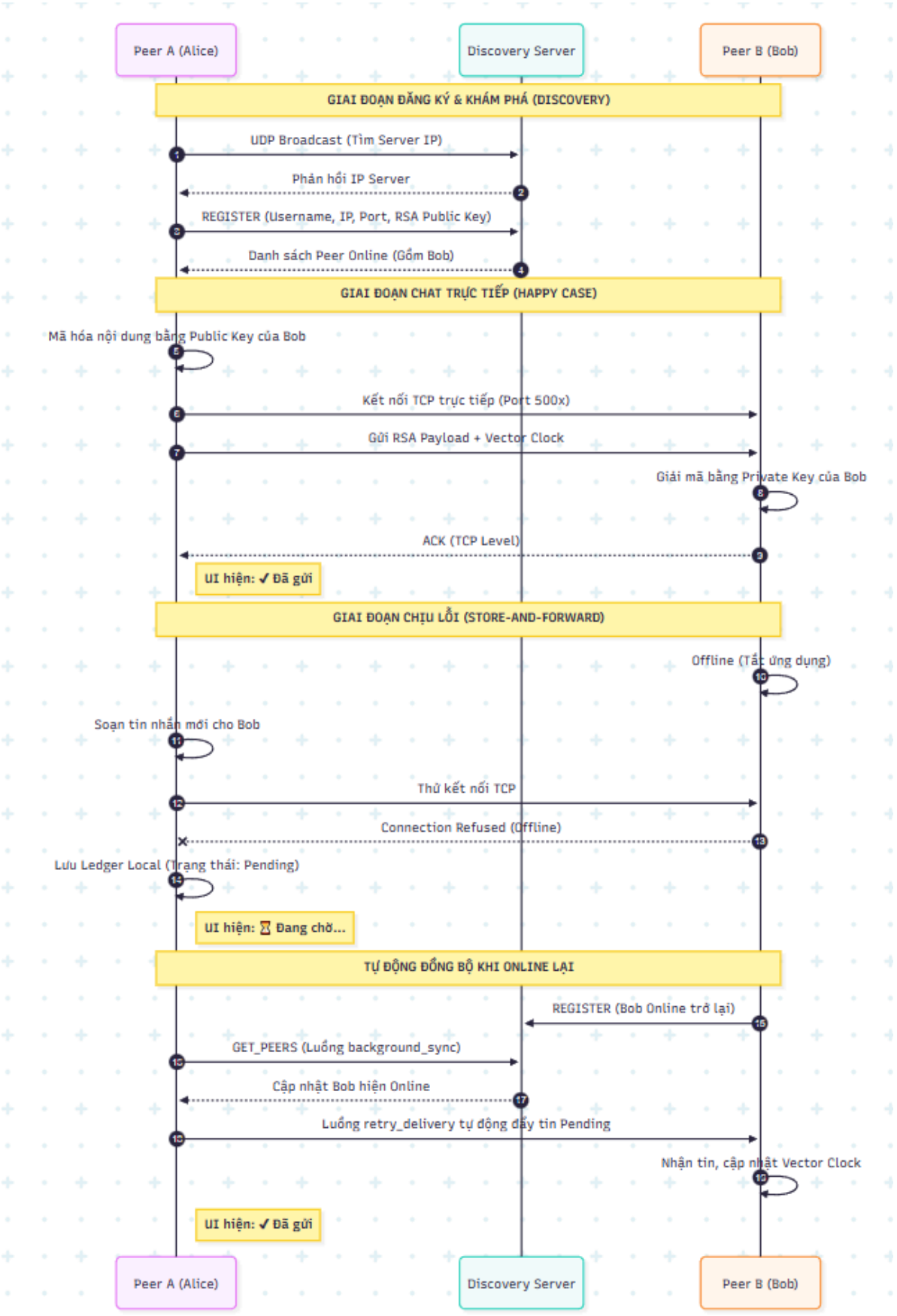
2.1. Tổng quan hệ thống

Hệ thống cho phép các máy trong cùng mạng LAN chat với nhau theo hướng P2P, trong đó mỗi peer vừa gửi vừa nhận tin. Thiết kế hiện tại tập trung vào:

- Discovery peer trong LAN bằng UDP Broadcast.
- Truyền tin P2P bằng TCP + JSON payload.
- Đồng bộ dữ liệu theo *eventual consistency* bằng Gossip + Vector Clock.
- Bảo mật: Mã hóa nội dung khi truyền bằng RSA (public key của peer đích).

Hệ thống có kèm Discovery Server để đăng ký danh bạ peer.

2.2. Sơ đồ tuần tự:



2.3. Kiến trúc Hệ thống (System Architecture):

- Mô hình Hybrid P2P: Kết hợp giữa Server trung tâm làm nhiệm vụ điều phối (Discovery Server) và các nút mạng giao tiếp trực tiếp (Peer-to-Peer).
- Discovery Service: Server chỉ đóng vai trò là "danh bạ" lưu trữ IP/Port và Public Key, không lưu trữ hay luân chuyển nội dung tin nhắn.
- Direct Communication (TCP Sockets): Tin nhắn được truyền trực tiếp giữa các Peer sau khi đã lấy được địa chỉ từ Server, đảm bảo tính phi tập trung.
- Local Storage (Shared-nothing): Mỗi node tự quản lý dữ liệu riêng, đảm bảo dữ liệu vẫn tồn tại cục bộ ngay cả khi Server bị sập.

2.4. Kiến trúc phân tầng:

Hệ thống được chia thành 3 tầng chính:

1. Tầng Giao diện (Frontend - Web UI): Sử dụng HTML/CSS/JS (Sidebar style) để người dùng thao tác chọn bạn và nhắn tin.
2. Tầng Điều khiển (Controller - Flask App)**:** Tiếp nhận yêu cầu từ UI, điều phối việc gửi nhận tin nhắn và quản lý luồng đồng bộ Server.
3. Tầng Logic & Dữ liệu (Engine - P2P Logic): Xử lý mã hóa (RSA/AES), quản lý Vector Clock để đảm bảo thứ tự tin nhắn và thực hiện các thao tác đọc/ghi file JSON.

2.5. Thuật toán sử dụng

2.5.1 Thuật toán Phân tán (Distributed Algorithms)

- Vector Clock (Đồng hồ Vector):
  - Sử dụng để xác định Thứ tự nhân quả (Causal Ordering) giữa các tin nhắn.
  - Giúp hệ thống sắp xếp lịch sử chat chính xác mà không cần đồng bộ thời gian thực (Physical Clock) giữa các máy.
- Eventual Consistency (Nhất quán cuối cùng): Đảm bảo sau khi tin nhắn được lan truyền, tất cả các node tham gia sẽ có nội dung lịch sử giống nhau.
- Gossip-like Propagation: Hệ thống sử dụng cơ chế Polling định kỳ từ Discovery Server, nhưng dữ liệu tin nhắn được lan truyền trực tiếp giữa các Peer (Gossip-style) nhằm đảm bảo dữ liệu được đồng nhất mà không cần qua trung gian.

2.5.2 Cơ chế Bảo mật (Security Mechanisms)

- Mã hóa bất đối xứng (RSA-2048):
    - Dùng để mã hóa đầu cuối (E2EE) tin nhắn.
    - Public Key dùng để mã hóa khi gửi, Private Key dùng để giải mã khi nhận.
- Mã hóa đối xứng (AES-256):
    - Sử dụng thư viện Fernet để mã hóa file lưu trữ cục bộ (.json).
    - Bảo vệ dữ liệu "nghỉ" (Data-at-rest), yêu cầu Passphrase chính xác mới có thể truy cập lịch sử chat.
- PBKDF2 (Password-Based Key Derivation Function 2): Thuật toán băm (hashing) để tạo ra khóa AES từ Passphrase của người dùng, chống tấn công dò mật khẩu.

2.5.3 Xử lý đồng thời & Chịu lỗi (Concurrency & Fault Tolerance)

- Multi-threading: Mỗi kết nối P2P đến được xử lý trên một luồng riêng, không chặn (Non-blocking) luồng chính của giao diện.
- No Single Point of Failure (SPOF): Nếu Server Discovery sập, các Peer đã biết IP của nhau vẫn có thể tiếp tục chat trực tiếp và truy cập dữ liệu cũ lưu tại local.
- Mutex/Lock Control: Sử dụng threading.Lock để bảo vệ tài nguyên dùng chung (Ledger, Peer List), tránh lỗi Race Condition khi nhiều tin nhắn đến cùng lúc.