Mục lục

[1. Mô tả bài toán 2](#_Toc61216594)

[2. Giải pháp và kiến trúc hệ thống 2](#_Toc61216595)

[*2.1* *Cơ sở giải pháp bảo mật* 2](#_Toc61216596)

[*2.2* *Kiến trúc hệ thống và các luồng xử lý* 2](#_Toc61216597)

[3. Cài đặt 4](#_Toc61216598)

[*3.1* *Server* 4](#_Toc61216599)

[*3.2* *Client* 5](#_Toc61216600)

1. ***Mô tả bài toán***

Xây dựng hệ thống chat peer to peer bao gồm hệ thống server và phần mềm trên máy pc cho người dùng. Yêu cầu bảo mật dữ liệu của người dùng được lưu trữ trên hệ thống và bảo mật dữ liệu truyền nhận trong hệ thống tránh bị nghe lén hay thay đổi; nội dung trò chuyện giữa người dùng là riêng tư với tất cả các bên khác.

Các chức năng của hệ thống server:

* Cho phép đăng ký tài khoản trên hệ thống và đăng nhập.
* Quản lý các user đang online trên hệ thống
* Chứng thực tài khoản khi các user muốn kết nối để chat với nhau.

Các chức năng của phần mềm pc:

* Giao diện để người dùng đăng ký và đăng nhập hệ thống.
* Hiển thị danh sách người dùng trên hệ thống.
* Kết nối đến người dùng khác và giao diện chat.

1. ***Giải pháp và kiến trúc hệ thống***
2. ***Cơ sở giải pháp bảo mật***

Bảo mật trong quá trình lưu trữ: bảo vệ mật khẩu của người dùng bằng cách băm mật khẩu bằng hàm băm ***PBKDF2*** trước khi ghi xuống cơ sở dữ liệu để tránh lộ thông tin kể cả khi cơ sở dữ liệu bị rò rỉ.

Bảo mật trong quá trình truyền nhận dữ liệu:

* Bảo vệ dữ liệu giao tiếp giữa client-server và client-client bằng thuật toán mã hóa khối ***AES*** bảo đảm mức độ bảo mật và vẫn giữ tốc độ xử lý của hệ thống.
* Khóa bí mật AES của phiên giao tiếp được mã hóa bằng ***RSA*** trước khi client gửi cho server để tránh bị nghe lén và mạo danh khi bắt đầu phiên.

Chứng thực: khóa công khai RSA của các client được chứng thực bằng chữ ký của server dùng thuật toán ***SHA256withRSA*** nhằm bảo đảm client kết nối đến đúng tài khoản mong muốn.

1. ***Kiến trúc hệ thống và các luồng xử lý***

Diagram

Description automatically generated with medium confidence

***Luồng kết nối client-server:***

* Bước 1: server mở socket chờ kết nối từ client.
* Bước 2: client connect tới server.
* Bước 3: server gửi RSA public key cho client.
* Bước 4: client sinh AES secret key ngẫu nhiên.
* Bước 5: client gửi AES secret key cho server (dữ liệu gởi được mã hóa bằng RSA public key nhận được ở bước 3).
* Bước 6: client và server dùng AES secret key đễ mã hóa và giải mã dữ liệu truyền nhận.

***Luồng tạo certificate cho user:***

* Bước 1: user gởi RSA public key của mình cho server.
* Bước 2: server tạo certificate cho user nội dung là thông tin id của user, RSA public key của user và chữ ký được ký bằng RSA private key của server.
* Bước 3: server gửi certificate cho client lưu trữ.

***Luồng kết nối và chứng thực giữa các user (giả sử user A muốn kết nối và chat với user B):***

* Bước 1: user A kết nối tới socket server của user B.
* Bước 2: user B gửi certificate của mình cho user A.
* Bước 3: user A dùng RSA public key của server để xác thực certificate nhận được từ user B, nếu certificate hợp lệ userA chắc chắn mình đã nhận được RSA public key đúng từ user B (không bị giả mạo, sửa chữa).
* Bước 4: user A sinh AES secret key ngẫu nhiên.
* Bước 5: user A gửi AES secret key cho user B (dữ liệu được mã hóa bằng RSA public key của user B nhận được trong chứng chỉ ở bước 2).
* Bước 6: user A và B dùng AES secret key để mã hóa và giải mã dữ liệu truyền nhận trong phiên chat.

***Luồng hash password của user:***

* Bước 1: client gửi thông tin đăng ký đến server.
* Bước 2: sau khi xác nhận thông tin hợp lệ, server tao salt ngẫu nhiên và hash password user với salt bằng hàm băm ***PBKDF2***.
* Bước 3: server lưu thông tin user mới vào database gồm salt được tạo ngẫu nhiên cho user ở bước 2 và password đã hash.
* Bước 4: khi user đăng nhập sẽ gửi name và password bản rõ lên server.
* Bước 5: server tìm user trong database theo name và lấy ra salt cùng với password đã hash.
* Bước 6: server hash passworrd bản rõ do user gởi với salt đã query được ở bước 5 bằng hàm băm ***PBKDF2***. Nếu kết quả hash giốn với password hash lưu trong database thì thông tin đăng nhập hợp lệ.

***Các lệnh và thao tác yêu cầu xử lý giữa client và server:***

* Đăng ký: client gọi để tạo tài khoản mới trên hệ thống.
* Đăng nhập: client đã có tài khoản, gọi đăng nhập khi kết nối lại vào hệ thống.
* Cập nhật public key: khi client tạo lại cặp khóa RSA, client gửi thông tin public key cho server để tạo certificate.
* Lấy danh sách user trên hệ thống: user đã đăng nhập hệ thống gọi server để lấy về danh sách những user khác trên hệ thống.

1. ***Cài đặt***

Ngôn ngữ: Java

Database: SqLite

UI: Java Swing

Mã hóa và bảo mật: java.security

1. ***Server***

Project server gồm 3 package chính:

* Sercurity: các lớp hỗ trợ thực hiện các thao tác liên quan tới mã hóa, hash, chữ ký.
* Connection: thực hiện việc kết nối và gửi nhận dữ liệu với client.
* Cms: thực hiện các thao tác xử lý nghiệp vụ liên quan tới lưu trữ.

**Package *sercurity:***

* Class *RSA*: các hàm hỗ trợ tạo, lưu trữ cặp khóa RSA, các hàm encode và decode message bằng thuật toán RSA.
* Class *AES*: các hàm hỗ trợ tạo secret key, encode và decode message bằng thuật toán AES.
* Class *Signing*: các hàm hỗ trợ việc ký vào một message và verify chữ ký.
* Class *Hash*: hàm hash một chuỗi input với salt.

**Package *connection:***

* Class *Server*: tạo socket để chờ kết nối từ client. Tạo thread để quản lý kết nối từ nhiều client. Tiếp nhận dữ liệu từ client và gửi dữ liệu về client.

**Package *cms*:**

* Class *dao*: các hàm hỗ trợ việc kết nối database và đọc ghi dữ liệu.
* Class *service*: xác định hàm xử lý cho các yêu cầu tương ứng từ client. Các hàm xử lý logic nghiệp vụ cho các yêu cầu, gọi dao để thao tác với database.

1. ***Client***