TRƯỜNG ĐẠI HỌC BÁCH KHOA TP.HỒ CHÍ MINH

BÁO CÁO BÀI TẬP LỚN
MẠNG MÁY TÍNH

GVHD: Lê Bảo Thịnh

Sinh viên: Lê Hoàng Dương 1710900

Nguyễn Vũ Hoàng Phúc 1712691

Nguyễn Bảo Phúc 1712674

Mục lục

1. Các chức năng của ứng dụng chat

1.1 Đăng nhập

Người dùng có thể đăng nhập vào úng dụng chat của nhóm bằng cách nhập.

1.2 Đăng ký

Người dùng muốn sử dụng ứng dụng chat phải đăng ký thông tin người dùng trên ứng dụng web bằng cách điền đầy đủ vào form sau:

- Tên đăng nhập
- Mật khẩu
- Nhập lại mật khẩu

Các thông tin trên được lưu trong database

1.3 Thêm liên hệ

- Thêm liên hệ: Người dùng tìm một liên hệ cần chat và thêm vào danh sách liên hệ.

1.3 Chat với nhiều người

Tại cùng một thời điểm, một người có thể nhắn tin với nhiều người khác nhau.

1.4 Truyền tải file

Trong quá trình chat giữa hai người, họ có thể gửi file qua lại cho nhau.

1.5 Xem danh sách online

Người dùng có thể xem trạng thái online của tất cả user trong danh sách liên hệ

Khi không có kết nối giữa client với sever thì xem như người dùng đang offline. Ví dụ: thoát trình duyệt, logout

2. Thiết kế chi tiết ứng dụng

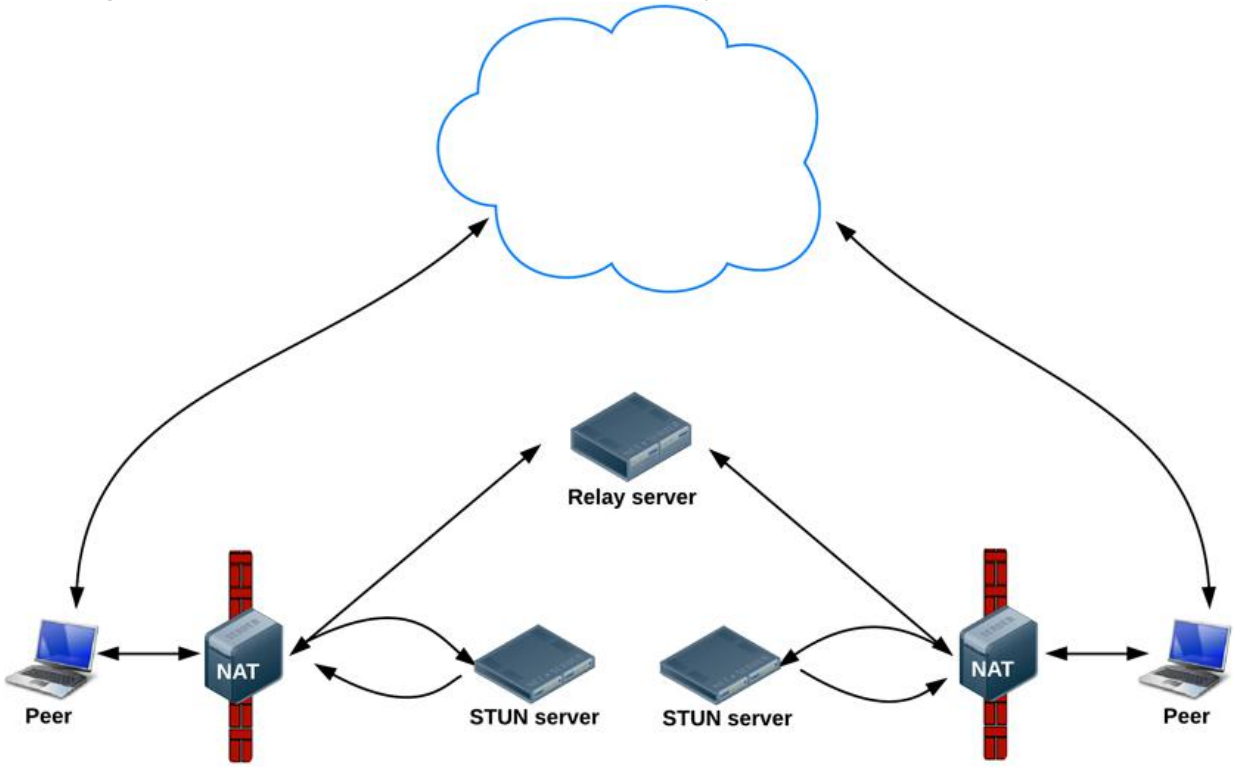
2.1 Mô hình WebRTC

WebRTC là Web Real-Time Communication và là một web API được phát triển bởi World Wide Web Consortium (W3C), khả năng hỗ trợ trình duyệt (browser) giao tiếp với nhau thông qua VideoCall, VoiceCall hay transfer data "Peer-to-Peer" (P2P) mà không cần browser phải cài thêm plugins hay phần mềm hỗ trợ nào từ bên ngoài.

WebRTC truyền dữ liệu "Peer-to-Peer", vì vậy việc đầu tiên cần làm là phải tạo kết nối "Peer-to-Peer"

Đối với một số máy, việc tạo kết nối "Peer-to-Peer" giữa A và B sẽ bị cản trở. Nên giống như các hệ thống VoIP, WebRTC cũng bị cản trở khi tạo kết nối peer-to-peer bởi tường lửa và NAT.

- NAT: có 2 loại IP là IP public và IP private, các máy trong mạng LAN được đặt IP Private vì IP Private không tồn tại ngoài Internet (Vd: 192.168.1.1, 192.168.1.2 etc) .
  - Vậy khi A gửi cho B một gói tin với thông tin tóm lược như : IP nguồn (IP của A) : 192.168.1.1 , IP đích: (113.xxx.y.z)
  - IP private không tồn tại ngoài Internet nên chúng ta cần NAT để thay thế IP private của A thành IP public.
- STUN: Khi một máy chủ nào xài NAT (behind NAT) thì STUN server sẽ giúp cho client đó biết được địa chỉ IP và Port mà thiết bị NAT sử dụng. Và từ đó giúp cho các peer có thể lấy được địa chỉ của peer khác. Nhược điểm: không support Symmetric NAT (NAT có nhiều loại)
- TURN: hỗi trợ cả giao thức TCP làm giao thức truyền tải. TURN bổ xung cho hạn chế của STUN là hỗ trợ Symmetric NAT.



Mô hình WebRTC gồm cả STUN/TURN server

- ICE(Interactive Communication Establishment): là một giao thức được cùng để thiết lập phiên media dựa trên UDP đi qua NAT một cách nhanh nhất

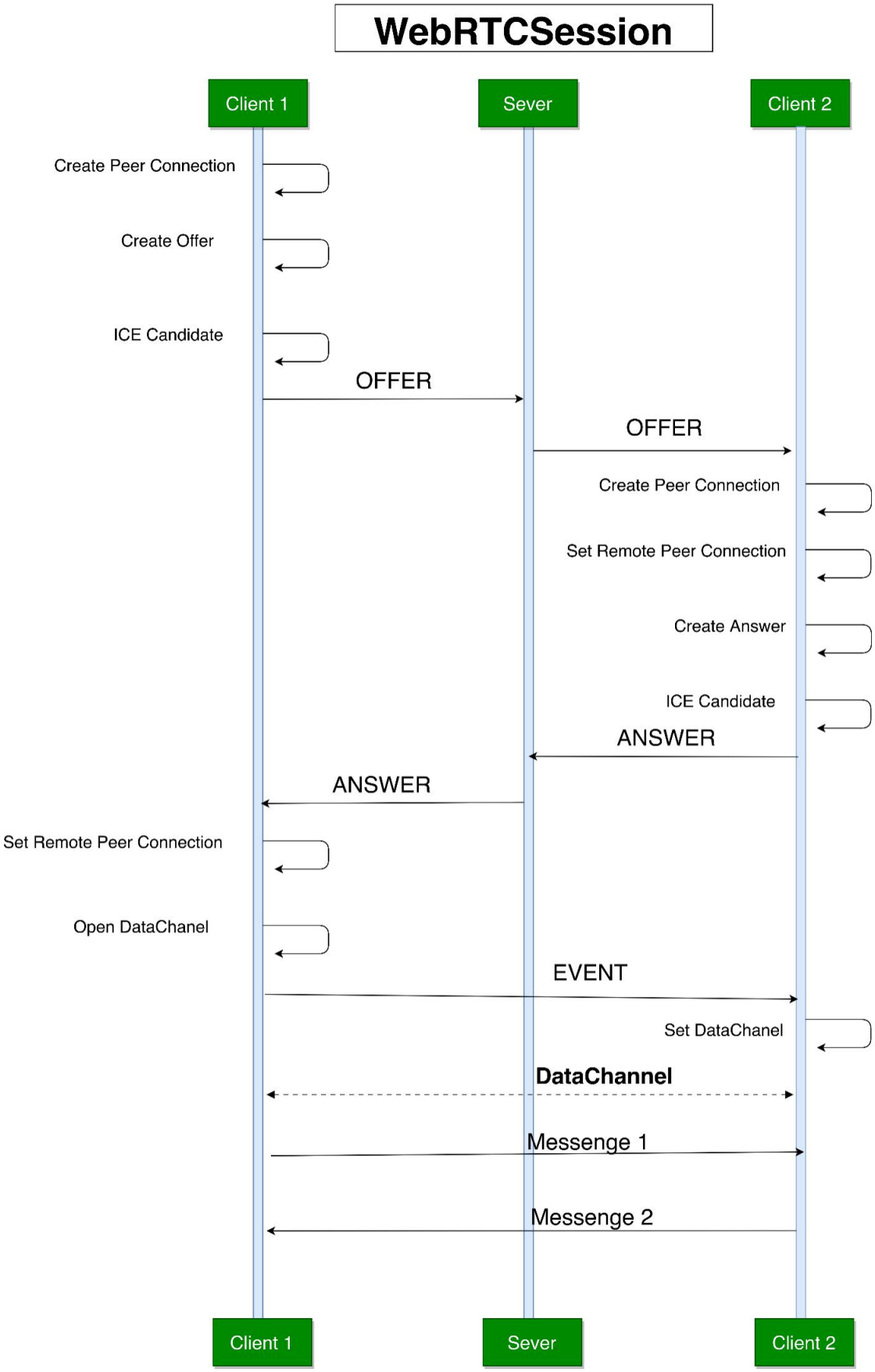
- ICE sẽ tìm đường tốt nhất để kết nối giữa các peer, nó thử tất cả khả năng có thể kết nối một cách song song và lựa chọn con đường hiệu quả nhất
- ICE cố gắng tạo ra một kết nối bằng cách sử dụng địa chỉ thu được từ hệ điều hành và card mạng của thiết bị, nếu không thành công (có thể thiết bị đằng sau NAT) thì ICE sẽ lấy địa chỉ bên ngoài của thiết bị bằng cách sử dụng máy chủ STUN với UDP, nếu không thành công nữa thì nó sẽ chuyển lưu lượng mạng qua một máy chủ chuyển tiếp là TURN với TCP.

2.2 WebRTC Session (thiết lập kết nối P2P)

WebRTC sử dụng RTCPeerConnection để giao tiếp và truyền dữ liệu giữa hai peers với nhau.

Các bước thực hiện:

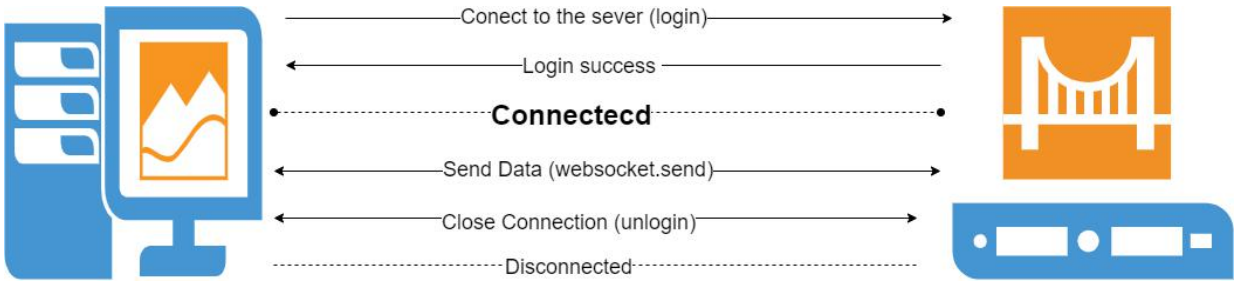
- Client 1 khởi tạo yêu cầu kết nối
- Client 1 sẽ tạo một offer, các ICE candidate và setLocalDescription với argument là offer, sau đó gửi offer tới Client 2 thông qua signaling sever để hỏi Client 2.
- Client 2 nhận được yêu cầu từ Client 1
- Client 2 chấp nhận yêu cầu kết nối từ Client 1, setRemoteDescription với argument là offer của client 1
- Sau đó Client 2 tạo answer, các ICE candidate và setLocalDescription với argument là answer. Gửi cho Client 1 qua Signaling server.
- Từ đây thì Client 1 và Client 2 có thể tự do liên lạc với nhau mà không cần signaling server nữa.


WebRTCSession
Client 1
Sever
Client 2
Create Peer Connection
Create Offer
ICE Candidate
OFFER
OFFER
Create Peer Connection
Set Remote Peer Connection
Create Answer
ICE Candidate
ANSWER
ANSWER
Set Remote Peer Connection
Open DataChanel
EVENT
Set DataChanel
DataChannel
Messenge 1
Messenge 2
Client 1
Sever
Client 2

2.3 WebSocket (thiết lập kết nối Client - Sever)

Websocket là giao thức hỗ trợ giao tiếp hai chiều giữa client và server để tạo một kết nối trao đổi dữ liệu real time. Giao thức này không sử dụng HTTP mà thực hiện nó qua TCP. Được thiết kế để chuyên sử dụng cho các ứng dụng web

Khi Sever khởi động, sever nhận kết nối (login) từ phía client



Quá trình kết nối giữa client và sever:

Client gửi username về phía sever

Sever thêm username vào list USER để theo dõi các socket đang hoạt động

Trong suốt quá trình kết nối, sever luôn nhận message từ client

- Nếu message là trạng thái kết nối, server sẽ gửi cho toàn bộ user trong list USER
- Nếu message là nội dung, sever sẽ tìm User mà client muốn gửi tới và yêu cầu nhận

Client ngắt kết nối: Xoá User khỏi list USER online

2.4. Các giao thức trong quản lý user

2.4.1 Đăng ký, đăng nhập

User sẽ gửi yêu cầu bao gồm tên tài khoản và mật khẩu về cho server xử lý thông qua phương thức HTTP POST để đảm bảo bảo mật thông tin. Sau khi đăng ký(đăng nhập) tài khoản xong, server sẽ gửi 1 yêu cầu HTTP có status code là 302 để yêu cầu trình duyệt redirect sang trang chính của ứng dụng.

- Sau đó trình duyệt sẽ gửi một yêu cầu về lại server để kiểm tra xem có phải đúng người dùng đã đăng nhập yêu cầu vào ứng dụng hay không.

- Nếu thỏa tất cả các điều kiện server sẽ gửi về trình duyệt phản hồi theo giao thức HTTP có status code 200 chứa nội dung là trang web chính của ứng dụng

2.4.2 Đăng xuất

-User sẽ gửi yêu cầu đăng xuất thông qua giao thức HTTP GET chứa yêu cầu đăng xuất khỏi hệ thống

- Sau khi server tiến hành xóa dữ liệu của người dùng khỏi phiên đăng nhập sẽ phản hồi lại cho browser lại thông qua 1 giao thức HTTP có status code là 200 yêu cầu trình duyệt chuyển về lại màn hình đăng nhập.

2.4.3 Tìm kiếm người dùng khác và thêm vào danh sách liên lạc

- Khi người dùng nhập tên người dùng khác vào ô tìm kiếm. Trình duyệt sẽ gửi 1 giao thức HTTP GET có url là */search?search-input=<tên cần tìm kiếm>&myname=<tên người dùng hiện tại>* về cho server

- Sau khi server kiểm tra trong database danh sách người dùng chưa được thêm vào danh sách liên lạc của người dùng hiện tại, sẽ gửi về lại cho phía trình duyệt một json chứa thông tin các người dùng có thể thêm vào danh sách liên hệ thông qua giao thức HTTP có status code 200.

- Khi người dùng chọn thêm vào danh sách liên lạc, trình duyệt sẽ gửi yêu cầu thêm danh sách liên hệ về cho server thông qua 1 giao thức HTTP GET có url là */addContact?name=<tên người dùng hiện tại> & cname=<tên người dùng cần thêm vào liên hệ>*

- Nếu yêu cầu thêm vào danh sách liên hệ được server xử lý thành công thì server sẽ trả về 1 phản hồi HTTP có status code 200 cho trình duyệt

3. Hướng dẫn cài đặt server

Yêu cầu: đã cài python3 và pip

- Download hoặc clone repo `` `https://github.com/PhucNVH/P2PChat` `` về máy.
- Vào thư mục *P2PChat* mở terminal/powershell chạy lệnh và chạy lệnh `` `pip install -r requirements.txt` ``
- Vào thư mục *P2PChat/app* “”
- Trong thư mục *P2PChat/app/static/js/script.js* đổi: `` `addr=“10.23.65.1”;` `` thành `` `addr=“<địa chỉ ip của server>”;` ``
  lưu ý: có thể đổi port của server websocket trong file `` `websocketSV.py` `` và server flask trong file `` `main.py` `` nếu cần.

- Trong thư mục *P2PChat/app, mở* terminal/powershell chạy lệnh `python main.py`
- Giờ có thể mở browser lên nhập địa chỉ server và port vào thanh URL và sử dụng trong mạng LAN

4. References

Các nguồn tham khảo khi thực hiện assignment:

- https://webrtc.org/start
- https://www.html5rocks.com/en/tutorials/webrtc/infrastructure
- https://websockets.readthedocs.io/en/stable/intro.html
- https://websitebeaver.com/insanely-simple-webrtc-video-chat-using-firebase-with-codepen-demo
- https://kipalog.com/posts/WebRTC-basic---Phan-1--Tim-hieu-ve-NAT--STUN--TURN-vs-ICE