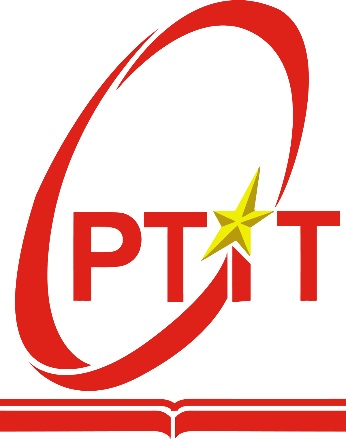
**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**KHOA CÔNG NGHỆ THÔNG TIN**

****

**BÁO CÁO THỰC TẬP CƠ SỞ**

**Chủ đề: Nghiên cứu và ứng dụng WEBRTC**

**Xây dựng ứng dụng Web Video Conferencing**

|  |  |
| --- | --- |
| Giảng Viên hướng dẫn | : TS.Nguyễn Trọng Khánh |
| Sinh Viên | : Phan Công Dũng |
| Mã Sinh Viên | : B19DCCN132 |

**Hà Nội - 2022**

MỤC LỤC

**DANH MỤC CÁC TỪ VIẾT TẮT**

**CHƯƠNG 1. GIỚI THIỆU TỔNG QUAN**

* 1. Đặt vấn đề
  2. Mục tiêu nghiên cứu

**CHƯƠNG 2. TÌM HIỂU VỀ WEBRTC**

2.1. Tổng quan về WebRTC

2.2. Cách thức hoạt động của WebRTC

2.3. Các APIs trong WebRTC

2.3.1. MediaStream

2.3.2. RTCPeerConnection

2.3.3. RTCDataChannel

2.4. Các tầng giao thức trong WebRTC

2.5. Servers trong WebRTC

2.5.1. Signaling Server

2.5.2. NAT Traversal Server

2.5.3. Media Server

2.5.4. Application Server

2.6. Kiến trúc của hệ thống WebRTC

**CHƯƠNG 3. ỨNG DỤNG WEBRTC CHO ỨNG DỤNG WEB VIDEO**

**CONFERENCING**

3.1. Tổng quan về ứng dụng

3.1.1. Mục đích

3.1.2. Công nghệ sử dụng

3.2. Cấu trúc chi tiết

3.2.1. Server

3.2.2. Store

3.2.3. Components

3.3 Hướng phát triển

**CHƯƠNG 4. KẾT LUẬN CHUNG**

**TÀI LIỆU THAM KHẢO**

**DANH MỤC CÁC TỪ VIẾT TẮT**

|  |  |  |
| --- | --- | --- |
| **STT** | **Từ viết tắt** | **Cụm từ** |
| 1 | ICE | Interactive Connectivity Establishment |
| 2 | STUN | Session Traversal Utilities for NAT |
| 3 | NAT | Network Address Translation |
| 4 | TURN | Traversal Using Relays around NAT |
| 5 | SDP | Session Description Protocol |
| 6 | TCP | Transmission Control Protocol |
| 7 | UDP | User Datagram Protocol |
| 8 | WebRTC | Web Real-Time Communication |
| 9 | API | Application Programming Interface |
| 10 | P2P | Peer to Peer |

**CHƯƠNG 1: GIỚI THIỆU TỔNG QUAN**

**1.1 Đặt vấn đề**

Nhu cầu giao tiếp ngày càng lớn đặc biệt trong thời kỳ dịch bệnh việc giao tiếp trực tiếp rất khó khăn. Nhờ nhu cầu đó mà các hình thức chia sẻ thông tin ngày càng phát triền từ việc truyền dữ liệu qua Email, hoặc các ứng dụng dịch vụ như Dropbox, OneDrive, Google Driver…Các ứng dụng chat như Skype, Viberm Whatsapp….Tuy nhiên vì nhiều lý do về tốc độ, bảo mật và đặc biệt là sự tiện dụng. WebRTC ra đời giải quy quyết được khá tốt các vấn đề trên qua đó đánh giấu một bước tiến mới trong việc giao tiếp thông qua intenet

**1.2 Mục tiêu nghiên cứu**

Bài nghiên cứu nhằm mục đích tập trung tìm hiểu về công nghệ WebRTC các giao thức được sử dụng để có thể chia sẻ và truyền dữ liệu trực tiếp thời gian thực, cách thức hoạt động và các chức năng chính của công nghệ WEBRTC. Tìm hiểu cách WEBRTC giải quyết các vấn đề NAT, Firewall để thiết lập kế nối Peer to Peer để trao đổi media cho các peer

Qua việc nghiên cứu WEBRTC kết hợp với các công nghệ khác ReactJS, Redux, Firebase… Để xây dựng ứng dụng demo cho việc truyền tải dữ liệu peer to peer cụ thể là ứng dụng Web Video Conferencing

**CHƯƠNG 2. TÌM HIỂU VỀ WEBRTC**

2.1 Tổng quan về WebRTC

2.2 Kiến trúc WebRTC

2.3 Các APIs trong WebRTC

2.4 Các tầng giao thức trong WebRTC

**2.1 Tổng quan về WebRTC**

WebRTC (Web Real-Time Communication) là một công nghệ cho phép các ứng dụng và trang web quản lý và tùy chọn phát trực tiếp âm thanh và video, cũng như trao đổi dữ liệu tùy ý giữa các trình duyệt mà không yêu cầu trung gian. Về bản chất, WebRTC là tập hợp các tiêu chuẩn và giao thức cho phép các trình duyệt Web thực hiện trực tiếp các tính năng truyền thông đa phương tiện thời gian thực như gọi điện, truyền hình, truyền dữ liệu, gửi tin nhắn bằng các APIs JavaScripts. WebRTC không yêu cầu plugin vì nó là một HTML5 Specification (Là một thứ cụ thể được định nghĩa trong HTML5 bởi các chuyên gia hàng đầu rồi tích hợp vào trình duyệt)

Ưu điểm giúp WebRTC thay thế các phương pháp củ:

1. Độ trể thấp
2. Hổ trợ gọi trực tiếp trên trình duyệt mà không cần thông qua ứng dụng thứ 3
3. WebRTC là một công nghệ có mã nguồn mỡ nên tiềm năng phát triển, hoàn thiện cũng như cộng đồng hổ trợ lớn
4. Kết nối P2P giúp tăng độ bảo mật trong việc truyền tải thông tin cũng như giảm bớt được độ trể.
5. WebRTC quy định mọi dữ liệu truyền P2P đều được bảo mật và mã hóa
6. Dễ dàng tích hợp trong các dịch vụ web/trang web bằng cách sử dụng JavaScript APIs và những Farmeword

Nhược điểm của WebRTC:

1. WebRTC bị cản bởi NAT và tường lửa khi cố gắng thực hiện kết nối P2P.
2. Không có cơ chế báo hiệu khi WebRTC tạo kết nối P2P giữa các trình duyệt
3. WebRTC vẫn chưa chính thức hoàn thiện, một số trình duyệt như IE, Safari chưa thực sự được hỗ trợ tốt nhất.
4. Các hãng trình duyệt chưa thống nhất được chuẩn video sử dụng cho WebRTC.
5. Số lượng hàm API WebRTC hỗ trợ cho mỗi trình duyệt là khác nhau, tăng rủi ro phát sinh lỗi khi sử dụng trên các trình duyệt khác nhau.

**2.2 Cách thức hoạt động của WebRTC**

Trong mô hình hoạt động truyền thống của web, mô hình client -

server được sử dụng khá là phổ biến khi mà server sẽ đảm nhiệm tất cả

bao gồm: điều khiển, truyền dữ liệu, security.... Tuy nhiên, nếu áp dụng

mô hình này vào trong các ứng dụng thời gian thực sẽ ảnh hướng tới

hiệu năng hoạt động của ứng dụng vì các lý do sau:

* Khối lượng dữ liệu trao đổi giữa các client là cực kỳ lớn.
* Chất lượng dịch vụ (Quality of service - QoS)

Do đó, các ứng dụng thời gian thực đòi hỏi một mô hình khác có thể đáp ứng được hai yêu cầu trên, đó là mô hình Peer-to-Peer (P2P).

Diagram

Description automatically generated

Tuy nhiên, trên thực tế, các peer đều sử dụng các địa chỉ IP private và sử dụng cơ chế NAT để kết nối internet, do đó việc kết nối trực tiếp giữa các peer sẽ gặp khó khăn nếu sử dụng dải mạng nội bộ. Do đó, chúng ta phải có cơ chế hỗ trợ phân giải sang địa chỉ IP public để Signaling Server có thể nhận biết các peer với nhau đó là cơ chế NAT.

(Cơ chế NAT được trình bày ở phần 2.4)

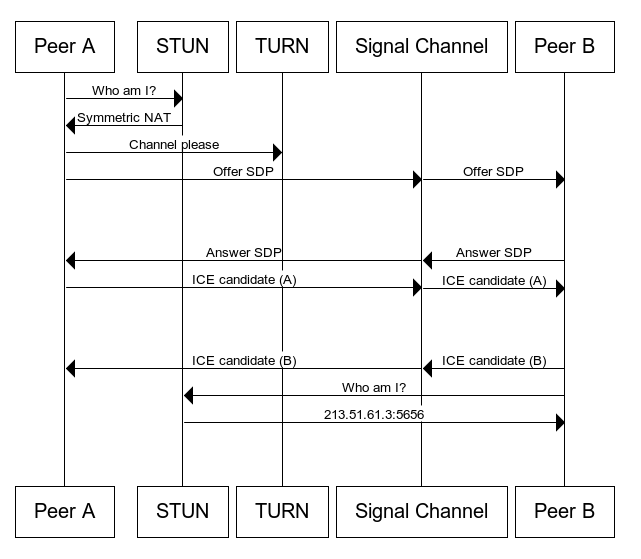
Thường các nhà mạng sẻ sử dụng cơ chế NAT để chuyển đổi từ địa chỉ IP private địa chỉ IP public.

Về cơ bản địa chỉ IP sẽ được chuyển đổi như sau : IP private <-> IP public: port.

Trong đó port ở đây là các cổng ứng dụng nằm trên Firewall hay router. Điều này giúp cho nhiều địa chỉ IP private có thể sử dụng chung IP public. STUN server sẽ đóng vai trò trung gian giúp cho các peer có thể lấy được địa chỉ của peer khác. Hiện nay, google đã xây dựng một số STUN server hỗ trợ hoàn toàn miễn phí giúp quá trình kết nối này hiệu quả hơn

Vậy cụ thể làm sao để tạo một kết nối P2P hoàn chỉnh ?

Toàn bộ quá trình khởi tạo một kết nối P2P được mô tả thành sơ đồ như sau:



Như đã nói WebRTC được tích hợp sẵn trong các brower chính là việc đóng vai trò như Peer ở hình trên. Vậy kiến trúc cụ thể như thế nào?

Diagram, schematic

Description automatically generated

Trong kiến trúc có 2 lớp riêng biệt:

* The Web: Là các kết nối với Brower thông quan Web API
* Browser: Là lớp chứa các thành phần core sâu hơn như Voice Engine (Xử lỹ âm thanh), Video Engine (Xữ lý hình ảnh), Transport (Xử lý kết nối) bên cạnh đó các nhà phát triển sẻ sử dụng WebRTC Native C++ API

Cụ thể hơn về 3 lớp API ta có:

* APIs cho nhà lập trình web: Lớp này chưa tất cả các APIs mà nhà lập trình web cần bào gốm các đối tượng chính là RTCPeerConnection, RTCDataChannel, MediaStream (Chi tiết ở phần 2.3)
* APIs cho nhà phát triển trình duyệt sử dụng
* Overridable API: nhà phát triển trình duyệt có thể thay đổi, phát triển APIs của riêng mình

Cụ thể về phần core ta có:

* Voice Engine xử lý chuổi âm thanh từ card âm thanh tới mạng và từ mạng ra loa
  + iSAC/iLBC codec: bộ mã hóa, giải mã tín hiệu âm thanh
  + NetEQ for voice: Bộ đếm jutter động mục đích tối ưu hóa âm thanh
  + Echo Canceler/ Noise Reduction: Acoustic Echo Canceler loại bỏ âm vọng trong thời gian thực. Noise Reduction một phần mềm dựa trên các thành phần xử lý tín hiệu, nhằm loại bỏ các loại tiếng ồn kết hợp với VoIP. (Hiss, fan noise, etc…)

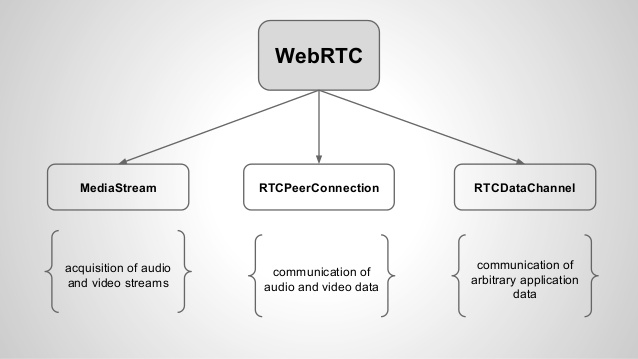
[VoIP truyền giọng nói của con người qua mạng máy tính sử dụng bộ giao thức TCP/IP]

* Video Engine xử lý chuổi hình ảnh từ camera tới mạng và từ mạng ra màn hình
  + Vp8 Codec một thiết kế cho độ trễ thấp (low latency)
  + Video jitter Buffer giúp che giấu ảnh hưởng của jitter và packet bị mất trong toàn bộ chất lượng video.
  + Image enhancements xóa tiếng ồn từ ảnh qua bởi webcam
* Transport giao thức vận chuyển phiên cho phép thiết lập và quản lý P2P qua các loại khác nhau (Chi tiết các giao thức ở mục 2.4)

**2.3. Các APIs trong WebRTC**

WebRTC cung cấp 3 APIs chính:

* getUserMedia: cho phép trình duyệt web truy cập vào camera và/hoặc microphone để lấy dữ liệu hình ảnh âm thanh cho việc truyền tải.
* RTCPeerConnection: dùng để cài đặt videocall/voicecall dùng cho việc truyền tải.
* RTCDataChannel: cho phép trình duyệt chia sẻ dữ liệu peer-to-peer.



Qua việc tìm hiểu sơ qua các các API hoạt động có thể suy được các bước cho 1 ứng dụng web RTC như sau:

1. Sử dụng getUserMedia API để truy cập vào camera và microphone
2. Lấy thông tin network như địa chỉ IP, ports và trao đổi thông tin đó với các peer khác (những peer mà mình muốn connect tới) để tạo connection (kết nối) dù cho có bị ngăn cản bởi NATs hay firewalls
3. Điều phối giao tiếp báo hiệu để báo cáo lỗi, khởi tạo hoặc đóng phiên kết nối
4. Trao đổi thông tin về khả năng hổ trợ kết media của từng Peer như độ phân giải codecs
5. Sau đó thì dùng RTCPeerConnection và RTCDataChannel để voice call/ video call hoặc chia sẻ dữ liệu sau khi đã có kết nối peer-to-peer

**2.3.1 MediaStream**

MediaStream là interface đại diện cho một luồng nội dung phương tiện. Luồng có thể bao gồm nhiều track như video, âm thanh…Mỗi track được chỉ định như một bản sao của MediaStreamTrack. MediaStream có thể chứa nhiều channel (Kênh) khác nhau đây là đơn vị nhỏ nhất mà MediaStream quản lý

Ngoài việc cung cấp các APIs yêu cầu luồng MediaStream còn cung cấp các APIs để thao tác, xử lý các luồng media đó. Các xử lý được thực hiện trên các track riêng biệt để xác định đầu ra

Đầu vào của MediaStream là thiết bị vật lý như microphone, webcam hoặc các file từ máy người dùng hoặc từ mạng.

Đầu ra là một hoặc nhiều đích thiết bị vật lý như microphone, webcam hoặc các file tại local hoặc peer từ xa

A picture containing timeline

Description automatically generated

**2.3.2. RTCPeerConnection**

Sau khi chúng ta đã có được session, giờ đây chúng ta có thể gửi session đó đến bất kỳ trình duyệt nào khác, nhưng chúng ta làm điều đó như thế nào? Chúng ta sẽ sử dụng RTCPeerConnection cho việc đó

RTCPeerConnection là một interface đại diện cho một kết nối WebRTC giữa máy tính cục bộ và một máy ngang hàng từ xa. Nó cung cấp các phương pháp để kết nối với một máy tính từ xa, duy trì và giám sát kết nối, đóng kết nối khi không còn cần thiết nữa

Chi tiết về nhiệm vụ của RTCPeerConnection gồm:

* Điều khiển toàn bộ quá trình ICE để vượt NAT
* Gửi tự động bản tin STUN giữ kết nối giữa các peers
* Tự động kiểm tra và kết nối lại MediaStream theo yêu cầu
* Cung cấp các APIs để khởi tạo những thông tin Offer/Anwser cần để trao đổi trong quá trình thiết lập kết nối, hoặc truy vấn trạng thái kết nối hiện tại

Diagram

Description automatically generated

**2.3.3. RTCDataChannel**

Cũng như hình ảnh và âm thanh, WebRTC cũng hỗ trợ giao tiếp real-time cho các loại dữ liệu khác.

RTCDataChannel là interface đại diện cho một kênh mạng có thể được sử dụng để truyền dữ liệu tùy ý ngang hàng hai chiều. Mọi kênh dữ liệu đều được liên kết với RTCPeerConnection và mỗi kết nối ngang hàng có thể có tối đa lý thuyết là 65.534 kênh dữ liệu (giới hạn thực tế có thể khác nhau giữa các trình duyệt).

DataChannel sử dụng SCTP, giao thức cho phép cấu hình việc gửi tin cậy tương tự TCP hay không tin cậy như UDP

Việc gửi data channel bằng cách sử dụng SCTP trên DTLS trên ICE/UDP giúp cho việc vượt NAT an toàn hơn vì có thể xác thực nguồn gốc và toàn vẹn dữ liệu. Dịch vụ vận chuyển dữ liệu này hoạt động song song với vận chuyển media bằng giao thức SRTP làm tăng tính linh hoạt của ứng dụng

Nhờ khả năng trao đổi với độ trể thấp và thông lượng cao RTCDataChannel thường được sử dụng trong ứng dụng game, ứng dụng remote destop, chat text thời gian thực truyền file…

**2.4. Các tầng giao thức trong WebRTC**

Do các đặc điểm cần thời gian thực cao hơn tính tin cậy, giao thức UDP được sử dụng trong WebRTC là giao thức vận chuyển.Nhưng để thỏa mãn yêu cầu của trình duyệt phải hỗ trợ giao thức và dịch vụ ở lớp khác nữa.Về cơ bản các giao thức chính sử dụng trong WebRTC được thể hiện ở hình dưới:

Chart

Description automatically generated

* **RTP**

RTP là một giao thức mạng mô tả cách truyền các media khác nhau (âm thanh, video) từ điểm cuối này sang điểm cuối khác theo thời gian thực. RTP thích hợp cho ứng dụng truyền video, điện thoại qua IP như Skype và các công nghệ hội nghị.

* **SRTP**

SRTP là một phiên bản bảo mật của RTP chính vì thế nó kế thừa các đặc điểm của RTP cộng thêm việc sữ dụng mã hóa và xác thực đẻ giảm thiết nguy cơ bị tấn công từ chối dịch vụ và vi phạm bảo mật

Với WebRTC sau khi thiết lập Peer Connection kết nối SRTP sẻ được thiết lập giữa các trình duyệt hoặc trình duyệt với máy chủ

* **SCTP**

SCTP là một giao thức theo tiêu chuẩn của IETF chạy trên DTLS trên UDP cho phép truyền thông điệp theo thứ tự, đáng tin cậy đồng thời cung cấp khả năng kiểm soát tắc nghẽn, đa giao thức và các tính năng khác để cải thiện độ tin cậy và ổn định của kết nối.Có thể truyền nhiều luồng dữ liệu cùng lúc vận chuyển không xác thực giúp nó trở nên nhanh hơn

Với WebRTC, SCTP được sử dụng để truyền tải DataChannel do có những tính năng tốt nhất của TCP (an toàn) và UDP (nhanh)

* **TLS**

TLS là một giao thức được các ứng dụng sử dụng để giao tiếp an toàn qua mạng, ngăn chặn giả mạo và nghe trộm email, duyệt web, nhắn tin và các giao thức khác. TLS là giao thức client/server đảm bảo quyền riêng tư trong giao tiếp bằng cách sử dụng các giao thức mật mã để cung cấp bảo mật qua mạng. Khi máy chủ và máy khách giao tiếp bằng TLS, nó đảm bảo rằng không có bên thứ ba nào có thể nghe trộm hoặc giả mạo.

* **DTLS**

DTLS dựa trên TLS qua đó đảm bảo tính an toàn. Hơn thế việc DTLS chạy trên UDP thích hợp cho việc vượt NAT trong các ứng dụng P2P

Cụ thể DTLS được sử dụng trong việc thống nhất khóa bảo mật mã hóa dữ liệu media và trong việc bảo mật sự vận chuyển

* **NAT**

NAT là một kỹ thuật cho phép nhiều máy tính chia sẻ một địa chỉ IP. NAT gán các địa chỉ duy nhất cho mỗi máy tính trong mạng cục bộ và điều chỉnh lưu lượng mạng đến / đi để gửi dữ liệu đến đúng nơi.

Vì việc chia sẻ địa chỉ IP nên có thể hiểu là “NAT sử dụng IP của chính nó làm IP công cộng cho mỗi máy con (client) với IP riêng. Khi một máy con thực hiện kết nối hoặc gửi dữ liệu tới một máy tính nào đó trên internet, dữ liệu sẽ được gởi tới NAT, sau đó NAT sẽ thay thế địa chỉ IP gốc của máy con đó rồi gửi gói dữ liệu đi với địa chỉ IP của NAT. Sau đó máy tính từ xa hoặc máy tính nào đó trên internet khi nhận được tín hiệu sẽ gởi gói tin trở về cho NAT computer bởi vì chúng nghĩ rằng NAT computer là máy đã gởi những gói dữ liệu đi. NAT ghi lại bảng thông tin của những máy tính đã gởi những gói tin đi ra ngoài trên mỗi cổng dịch vụ và gởi những gói tin nhận được về đúng máy tính đó (client).

Diagram

Description automatically generated

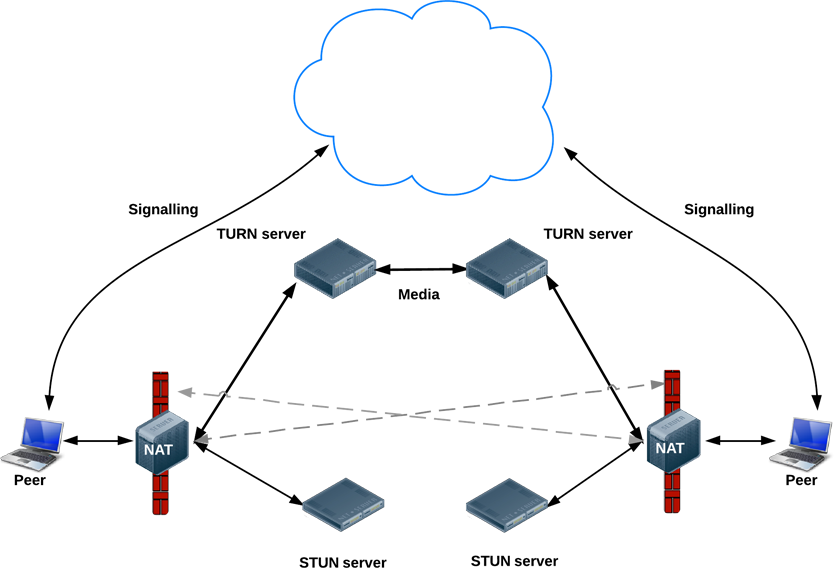
* **STUN**

**A picture containing text, businesscard, screenshot

Description automatically generated**

STUN là giao thức giúp cho việc vượt NAT. khi một máy chủ nào sử dụng 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 (IP nào, cổng mấy, NAT loại gì) để nhận biết . Nhưng STUN có một nhược điểm là nó không support Symmetric NAT khi đó phải sử dụng TURN. Bởi vì STUN (máy chủ bên ngoài), không thể biết cổng nào sẽ được cấp cho máy khách bởi thiết bị NAT. Trong Symmetric NAT, máy khách nhận được cổng duy nhất (ip: cổng trong tình huống chung) trên mỗi kết nối [Vì IP chung nên mỗi kết nối phải có cổng khác nhau để nhận biết]

STUN server sẽ đóng vai trò trung gian giúp cho các peer có thể lấy được địa chỉ của peer khác. Hiện nay, google đã xây dựng một số STUN server hỗ trợ hoàn toàn miễn phí như stun:stun.l.google.com:19302

* **TURN**

TURN là một mở rộng của STUN tuy nhiên 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. Dữ liệu thay vì được gửi trực tiếp tới các peer thì các peer sẽ gửi dữ liệu tới các TURN server và TURN server sẽ đóng vai trò trung gian vận chuyển gói tin. Điều này nâng cao giúp chất lượng dịch vụ của ứng dụng mà còn đảm bảo an toàn thông tin khi truyền dẫn.

Bất lợi của TURN là chi phí sử dụng lớn vì các xử lý được thực hiện ở TURN nên cần có băng thông và khả năng xử lý lớn nhất là còn phải xử lý video hình ảnh ở chất lượng cao

* **ICE**

ICE 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. Đầu tiên nó sẽ 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 nếu không thành công nữa thì nó sẽ chuyển lưu lượng mạng qua một máy chủ chuyển tiếp là TURN.

2.5. Servers trong WebRTC

2.5.1. Signaling Server

2.5.2. NAT Traversal Server

2.5.3. Media Server

2.5.4. Application Server

**2.5. Servers trong WebRTC**

2.5.1 Signaling Server

Diagram

Description automatically generated

Trước khi kết nối P2P được tạo ra giữa 2 thiết bị chúng cần nhận biết được nhau để khởi tạo kệnh truyền. Signaling Server là nơi xử lý vấn đề này

Một quy trình tạo kênh truyền được xử lý theo thứ tự như trên hình ảnh theo cơ chế offer-answer là một phần của WebRTC. Những tin nhắn có chứa SDP package (Các dạng connection mà 2 bên cho phép mở két nối). Đây là những đoạn tin nhắn mà ứng dụng cần báo hiệu cho thiết bị khác để kết nối một phiên. Nó thực hiện điều đó bằng cách sử dụng Signaling protocol và Signaling Server.

Bên cạnh việc khởi tạo kết nối Signaling server còn quản lý các kết nối giữa các thiết bị. Bản thân nó không xử lý lưu lượng truyền thông mà quan tâm đến… tín hiệu. Điều này bao gồm việc cho phép một người dùng tìm thấy một người khác trong mạng, tự thương lượng kết nối, đặt lại kết nối nếu cần và đóng nó lại

Signaling Server chỉ hỗ trợ chuyển các thông điệp xung quanh theo logic do ứng dụng của bạn ra lệnh.

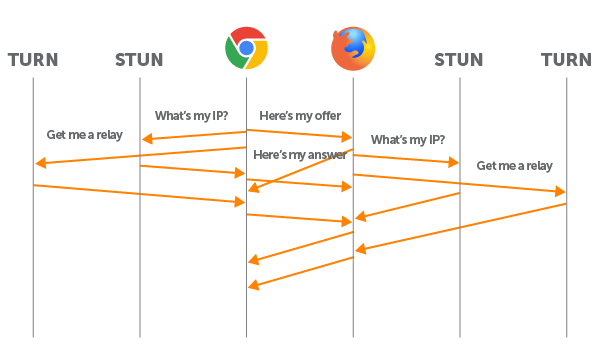
Một số công nghệ sử dụng cho Signaling Server:

+ Websocket/MQTT/XMPP

+ Realtime Database (Firebase, CouchDB)

+ Clond services (Pusher, Ably)

2.5.2. NAT Traversal Server



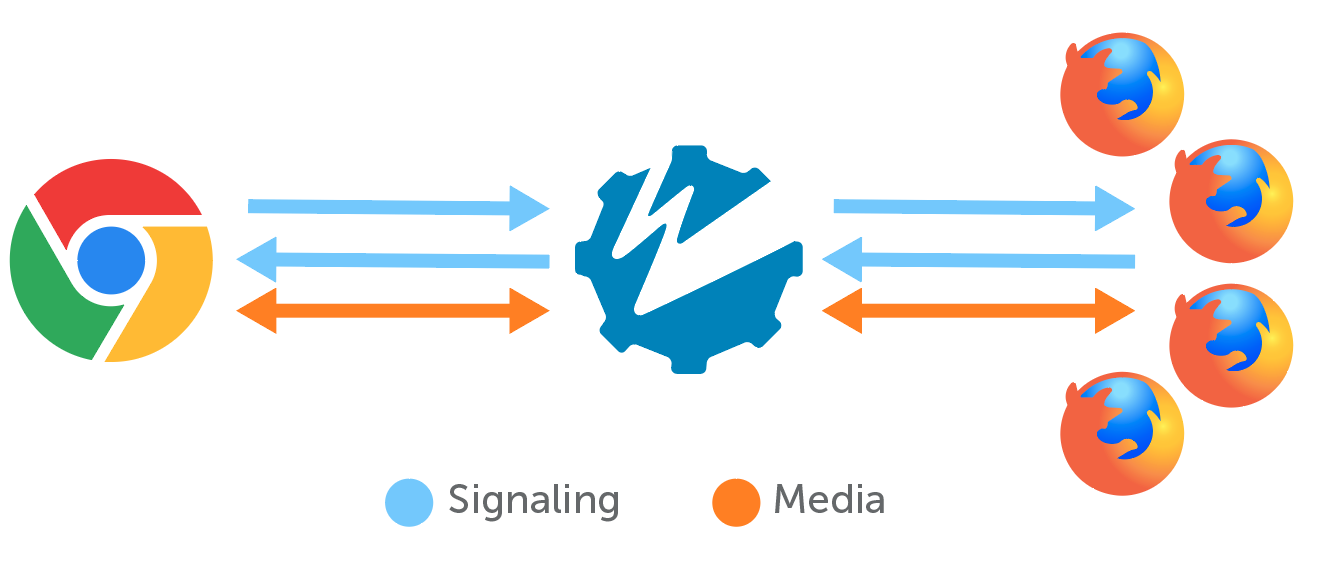
Một phần của những việc cần làm là cố gắng vượt qua tường lửa và thiết bị NAT. Điều này được thực hiện bằng cách sử dụng một giao thức gọi là ICE, giao thức này thu thập, trao đổi và sau đó cố gắng kết nối một phiên bằng cách sử dụng các ứng viên ICE. Các ứng cử viên ICE là các cặp địa chỉ tiềm năng có thể giúp các thiết bị kết nối với nhau 1) trực tiếp bằng cách sử dụng địa chỉ IP riêng hoặc công cộng có được thông qua máy chủ STUN hoặc 2) gián tiếp thông qua máy chủ TURN.

Tóm lại mục đích chính là vượt qua NAT và gửi địa chỉ IP đến cho Signaling Server để giúp nhận biết và tạo kết nối

2.5.3 Media Server

Giả sử rằng muốn tạo luồng video 1mbps bằng cách sử dụng WebRTC và phát trực tiếp đến 100 người xem. Nếu không có Media Server, thiết bị sử dụng sẽ cần sử dụng kết nối lên 100mbps, một điều gây lãng phí tài nguyên. Thêm vào đó là thách thức về việc một thiết bị phải chịu tải như vậy với một số lượng lớn các kết nối sẻ gây nguy hiểm. Giải pháp cho vấn đề này là sử dụng Media Server

Media Server là một phần cứng chuyên dụng hoặc phần mềm (máy chủ vật lý hoặc chương trình ứng dụng) trách nhiệm cung cấp đa phương tiện theo yêu cầu.



Có 2 cách thức vận chuyển dữ liệu chính:

Graphical user interface, application

Description automatically generated

Nhận stream từ 1 người dùng và chuyển đi cho tất cả người dùng khác

* Tốn băng thông vì phải gửi nhiều stream đi

Diagram

Description automatically generated

Gộp tất cả các stream đến thành 1 stream duy nhất rồi gửi đi

* Giảm Collection => Giảm bandwitch => Phù hợp với các thiết bị bandwitch nhỏ

Tuy nhiên việc xử lý gộp stream sẻ làm cho CPU server phải xử lý nhiều

Một số giải pháp Media Server:

Open-Source solutions: Janus-gateway, Mediasoup, Jitsi-videobridge, Kurento-Media-server

Commercial solution: wowza

2.5.4. Application Server

Diagram

Description automatically generated

Application Server là nơi để đẩy ứng dụng của mình lên. Giúp các peer biết phải connect với nhau ở đâu

Một số mày chủ như Apache, CDN….

**2.6 Kiến trúc của hệ thống WebRTC**

**Kiển trúc hệ thống web truyền thống**

**Diagram

Description automatically generated**

**Kiến trúc phổ biến của hệ thống WebRTC**

**Diagram

Description automatically generated**

**Kiến trúc hệ thống WebRTC khi có nhiều người dùng kết nối với nhau (Thường bé hơn 10 peer)**

**Diagram, engineering drawing

Description automatically generated**

Như ta thấy ở hình trên, bất cứ khi nào ta muốn kết nối tới một user khác ta cần tạo peer thêm peer để kết nối với hai bên. Theo hình trên, như ta thấy ở trên mỗi peer sẽ có 2 luồng kết nối. Nên khi số lượng peer lớn sẻ không thể đáp ứng được

**Kiến trúc hệ thống WebRTC khi có số lượng người dùng lớn**

**Diagram

Description automatically generated**

Với việc thêm vào Media Server ta thấy rõ sự giảm tải giữa các peer kết nối. Mỗi peer bây giờ chỉ cần giữ kết nối tới một Media Server . Vì thế với một server riêng biệt chúng ta có nhiều sức mạnh hơn trong việc thực hiện các tác vụ video nâng cao tăng tương tác với người dùng.Hiện tại thì có hai loại Media Server phổ biến được dùng là MCU và SFU.

**CHƯƠNG 3. ỨNG DỤNG WEBRTC CHO ỨNG DỤNG WEB VIDEO**

**CONFERENCING**

**3.1. Tổng quan về ứng dụng**

**3.1.1. Mục đích**

Tạo ứng dụng sử dụng công nghệ webrtc giúp người dùng giao tiếp với nhau một cách tiện lợi hơn. Giảm thiếu tối đa độ trể tín hiệu

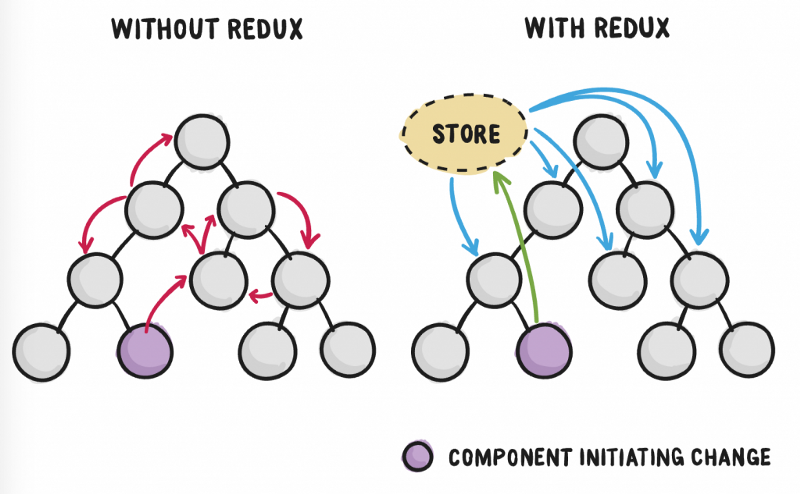
**3.1.2. Công nghệ sử dụng**

**React:**

Là một thư viện Javascript hổ trợ xây dựng ứng dụng với tư tưởng chia componets giúp dễ phát triển

**Redux:**

Là một tool hỗ trợ cho react trong việc lưu trữ và xữ lý dữ liệu nhất là là khi phải truyền từ component cha xuống component con



Diagram

Description automatically generated

**Firebase**

Là một dịch vụ cơ sở dữ liệu miễn phí được hoạt động ở trên nền tảng đám mây (Cloud). Với việc hở trợ realtime database cũng như deploy nó rất phù hợp cho việc triễn khai webrtc

**Webrtc**

**3.2. Cấu trúc chi tiết**

**Text

Description automatically generated**

**3.2.1. Server**

Lưu trữ thông tin server và thông tin kết nối peerConnection (Firebase, Webrtc)

Graphical user interface

Description automatically generated

**3.2.2. Store**

Lưu trữ store chung và xử lý của ứng dụng (Redux)

A picture containing text

Description automatically generated

**3.2.3. Components**

Các thành phần view của ứng dụng (React)

Graphical user interface, text, application

Description automatically generated

**Cấu trúc kết nối của các use**

Diagram

Description automatically generated

**Cấu trúc kết nối các thành phần**

Diagram

Description automatically generated

**3.3 Hướng phát triển**

- Áp dụng thêm RTCDataChannel giúp người dùng truyển tải dữ liệu: Text, File..

- Sử dụng Media Server để tăng khả năng chịu tải cho ứng dụng

**TÀI LIỆU THAM KHẢO**

1. <https://developer.mozilla.org/en-US/docs/Web/API/WebRTC_API>
2. <https://www.w3.org/TR/webrtc/>
3. <https://codelabs.developers.google.com/codelabs/webrtc-web/#0>
4. https://www.wowza.com/blog/webrtc-signaling-servers
5. <https://princiya777.wordpress.com/2017/08/19/webrtc-behind-the-browser/>
6. <https://github.com/w3c/mediacapture-extensions/issues/16>
7. <https://webrtc.org/>
8. <https://viblo.asia/p/webrtc-la-gi-hoat-dong-nhu-nao-yMnKM03N57P>
9. <https://viblo.asia/p/gioi-thieu-ve-webrtc-va-huong-tiep-can-media-server-maGK7k3MKj2#_mcu-12>
10. <https://viblo.asia/p/introduction-about-real-time-communication-in-web-webrtc-n6BkGyWeR5aV#_2-cau-truc-va-cach-thuc-hoat-dong-cua-webrtc-1>
11. <https://viblo.asia/p/webrtc-basic-phan-2-cach-thiet-lap-mot-cuoc-goi-thong-qua-webrtc-MgNvWWpRvYx>
12. <https://viblo.asia/p/webrtc-phan-2-kien-truc-webrtc-bWrZnvnrZxw#_vp8-13>
13. <https://viblo.asia/p/webrtc-phan-1-E375zEPdlGW>
14. <https://wiki.matbao.net/webrtc-la-gi-cach-viet-ung-dung-goi-video-bang-webrtc-va-firebase/#uu-va-nhuoc-diem-cua-webrtc-la-gi>
15. <https://filegi.com/tech-term/secure-real-time-protocol-secure-rtp-or-srtp-4891/>
16. <https://bizflycloud.vn/tin-tuc/vi-sao-giao-thuc-truyen-da-luong-sctp-van-chua-duoc-ung-dung-rong-rai-20190817111448263.htm#:~:text=SCTP%20c%C5%A9ng%20cho%20kh%E1%BA%A3%20n%C4%83ng,m%E1%BA%A1o%2C%20t%E1%BA%A5n%20c%C3%B4ng%20ngh%E1%BA%BDn%20m%E1%BA%A1ng>.