ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

NGUYỄN VIẾT THẮNG

ỨNG DỤNG CÔNG NGHỆ WEBRTC CHO GIẢI PHÁP CỘNG TÁC VÀ CHIA SẺ DỮ LIỆU ĐA PHƯƠNG TIỆN TẠI TRUNG TÂM MVAS-TCT VIỄN THÔNG MOBIFONE

Ngành	:	Công nghệ thông tin
Chuyên ngành	:	Truyền dữ liệu & mạng máy tính
Mã số	:	

TÓM TẮT LUẬN VĂN THẠC SĨ CÔNG NGHỆ THÔNG TIN

Hà nội - 2016

MỤC LỤC

CHƯƠNG 1: MỞ ĐẦU

Từ ý tưởng ban đầu của Google với dự án mã nguồn mở browser-based real-time communication, mục đích chính là tạo khả năng giao tiếp thời gian thực giữa trình duyệt, WebRTC ra đời và đang tiếp tục phát triển. Với sự phối hợp của các tổ chức tiêu chuẩn thế giới W3C, IETF trong việc chuẩn hóa các protocols, APIs; các vendor lớn trong việc hỗ trợ phát triển, trình duyệt, WebRTC thực sự đã mang Web đến với thế giới viễn thông.

Luận văn tập trung tìm hiểu về công nghệ WebRTC, các APIs trình duyệt, các giao thức được WebRTC sử dụng để có thể chia sẻ và truyền dữ liệu trực tiếp thời gian thực giữa các trình duyệt trong môi trường mạng. Luận văn cũng phân tích yêu cầu tính chất "thời gian thực" khi truyền dữ liệu media và cách thức WebRTC đang được xây dựng để giải quyết, cũng như vấn đề vượt NAT, Firewall để thiết lập kết nối Peer to Peer. Luận văn được chia thành ba chương với nội dung sau:

Chương 1 – Lời mở đầu

Chương 2 – Tổng quan về WebRTC, giới thiệu chung về lịch sử, sự tiện lợi, các APIs và giao thức được sử dụng trong WebRTC

Chương 3 – Nghiên cứu về báo hiệu, thiết lập phiên trong WebRTC.

Chương 4 – Nghiên cứu các cách tiếp cận sử dụng WebRTC trong xây dựng ứng dụng, giới thiệu framework EasyRTC và sử dụng EasyRTC demo ứng dụng cộng tác tại Trung tâm MVAS – TCT viễn thông Mobifone.

CHƯƠNG 2: TỔNG QUAN VỀ WEBRTC

2.1. Quá trình phát triển

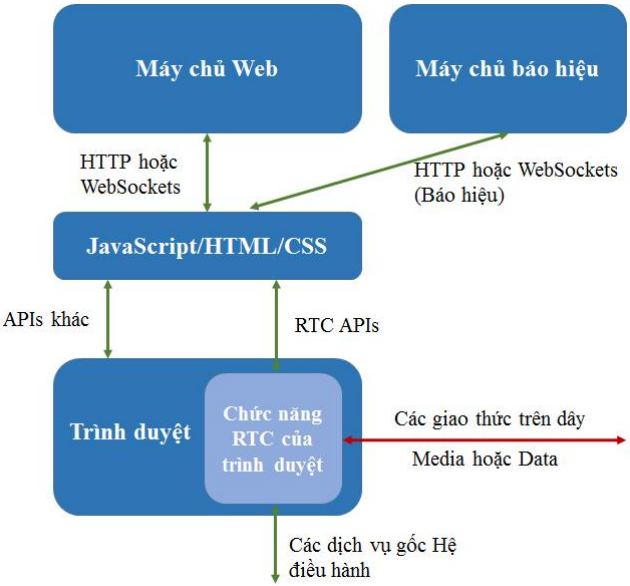
WebRTC (Web Real-Time Communication) là tập hợp các tiêu chuẩn và giao thức cho phép các trình duyệt Web thực hiện trực tiếp các tính năng truyền thông đa phương tiện thời gian thực như gọi điện, truyền hình, truyền dữ liệu, gửi tin nhắn bằng các APIs JavaScripts.

- 27/10/2011: Bản dự thảo WebRTC đầu tiên được W3C công bố.
- 10/02/2015: WebRTC 1.0 working draft chính thức được công bố. Đến nay đã được hỗ trợ bởi các trình duyệt Chrome (version 23 trở lên), Firefox (version 22 trở lên), Opera (version 18 trở lên) và được hỗ trợ trình duyệt trên nền tảng Android (Chrome 29 trở lên, Firefox 24 trở lên, Opera Mobile 12 trở lên, Google Chrome OS).

Những lợi ích của WebRTC: giảm giá thành, không Plugins, truyền thông P2P, dễ sử dụng, một giải pháp cho mọi nền tảng, mã mở và miễn phí, Built-in security.

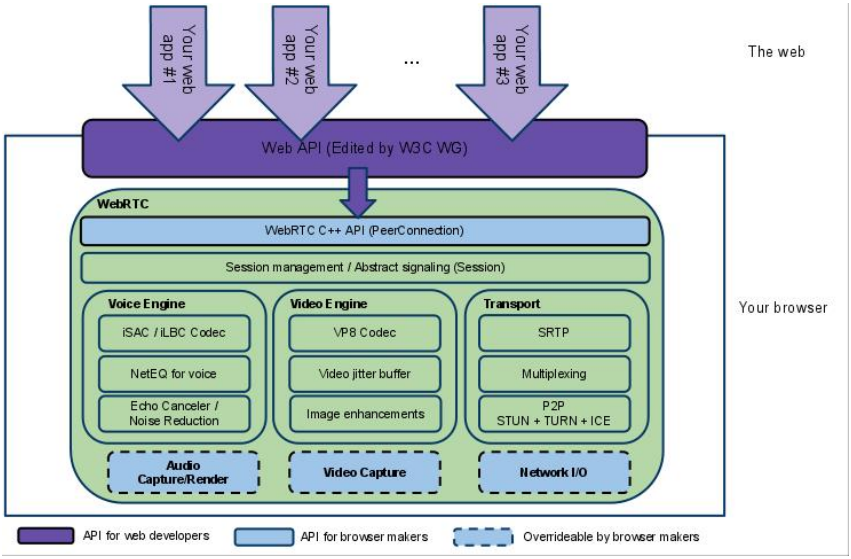
2.2. Kiến trúc WebRTC trong trình duyệt

Ứng dụng web với WebRTC (thường viết bằng HTML5 và JavaScript) tương tác với trình duyệt qua những WebRTC APIs đang được chuẩn hóa, cho phép nó khai thác hợp lý và điều khiển chức năng thời gian thực của trình duyệt.



Hình 2.1: Truyền thông thời gian thực trong trình duyệt (nguồn *[1]*)

Hình 2.1 cho thấy mô hình trình duyệt và vai trò của các chức năng truyền thông thời gian thực. Khối màu sáng là chức năng truyền thông thời gian thực (Real Time Communication – RTC) của trình duyệt. Do tính chất riêng và yêu cầu của truyền thông thời gian thực nên việc chuẩn hóa khối này là không đơn giản, hiện tại vẫn đang trong quá trình bàn thảo. Các chức năng RTC tương tác với các ứng dụng web sử dụng các APIs chuẩn. Nó giao tiếp với các hệ điều hành bằng cách sử dụng trình duyệt



Hình 2.2: Kiến trúc tổng thể WebRTC [Nguồn 10]

Trong kiến trúc WebRTC có 3 lớp API:

- APIs cho nhà lập trình web: lớp này chứa tất cả các APIs mà nhà lập trình web cần, bao gồm các đối tượng chính là RTCPeerConnection, RTCDataChannel, MediaStream (chi tiết mô tả ở mục 2.3.Các APIs trong WebRTC).
- 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.

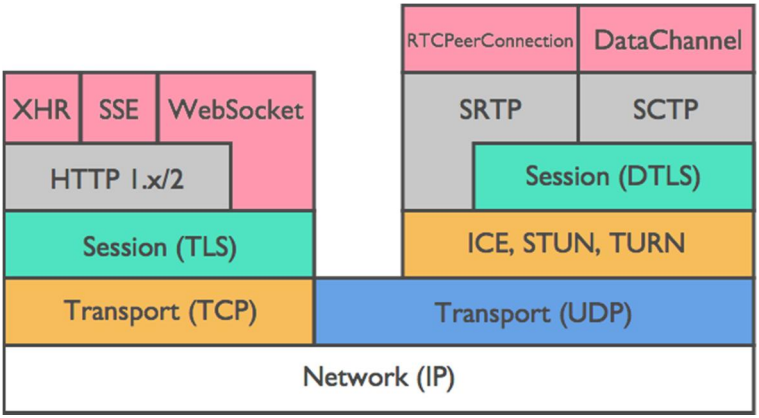
2.3. Các APIs trong WebRTC

WebRTC bao gồm các APIs, các giao thức liên quan và làm việc với nhau để hỗ trợ việc trao đổi dữ liệu đa phương tiện giữa các trình duyệt. WebRTC đang trong quá trình chuẩn hóa và sử dụng các APIs quanh ba khái niệm chính:

- RTCPeerConnection: thiết lập kết nối cho cuộc gọi audio/video/data, khả năng mã hóa và quản lý băng thông.
- MediaStream: truy cập vào dòng media, như camera hay microphone người dùng
- RTCDataChannel: giao tiếp peer-to-peer cho các dữ liệu non-media.

2.4. Các tầng giao thức trong WebRTC

Do đặc điểm yêu cầu thời gian thực cao hơn tính tin cậy, giao thức UDP được lựa chọn sử dụng trong WebRTC là giao thức vận chuyển. Tuy nhiên, để thỏa mãn tất cả yêu cầu của WebRTC, trình duyệt phải hỗ trợ các giao thức và dịch vụ khác ở lớp trên nữa. Về cơ bản, các giao thức chính sử dụng trong WebRTC thể hiện ở hình 2.3 dưới đây:



Hình 2.3: Protocol stack trong WebRTC [4]

✓ **SRTP**

Giao thức quan trọng nhất mà WebRTC sử dụng là Secure Real-time Transport Protocol, hay SRTP. SRTP được sử dụng để mã

hóa và chuyển các gói tin media giữa các WebRTC client. Sau khi thiết lập thành công PeerConnection, kết nối SRTP sẽ được thiết lập giữa các trình duyệt hoặc trình duyệt và máy chủ. Với dữ liệu non-audio hay video, SRTP không được sử dụng, thay vào đó là SCTP.

✓ **SCTP**

WebRTC sử dụng SCTP - Stream Control Transmission Protocol để truyền các dữ liệu non-media giữa các Peer. Giao thức SCTP là giao thức vận chuyển, tương tự như TCP và UDP, có thể chạy trực tiếp trên giao thức IP. Tuy nhiên trong WebRTC, SCTP chạy trên DTLS tên UDP. SCTP được lựa chọn do có những tính năng tốt nhất của TCP và UDP như: message-oriented transmission, khả năng cấu hình tùy biến tính tin cậy và thứ tự gói tin, có cơ chế quản lý lưu lượng và chống nghẽn.

✓ **SDP**

WebRTC sử dụng Session Description Protocol - SDP, được encode trong đối tượng RTCSessionDescription, để mô tả đặc tính media của hai đầu trong kết nối P2P như loại media đề truyền/nhận (audio, video, application data), network transports, loại codecs sử dụng và cấu hình, thông tin băng thông, và các thông tin metadata khác. Thông điệp SDP được trao đổi qua máy chủ báo hiệu hay còn gọi là được trao đổi qua kênh báo hiệu. Máy chủ báo hiệu có trách nhiệm gửi và nhận tất cả thông điệp đến tất cả các peers mà mong muốn kết nối đến peer khác. Mặc dù SDP là định dạng dữ liệu dùng để trao đổi, thống nhất thông số giữa kết nối Peer-to-Peer, nhưng do WebRTC không ràng buộc cho các SDP “offer” và “answer” giao tiếp như nào, nên nó không được thể hiện ở Hình 2.5 ở trên. Tuy nhiên, mô hình offer/answer được thiết kế tuân thủ theo RFC3264.

✓ **DTLS**

Datagram Transport Layer Security- DTLS dựa trên giao thức TLS, cung cấp tính bảo mật và toàn vẹn dữ liệu truyền giữa các ứng dụng tương tự TLS. Tuy nhiên, WebRTC sử dụng DTLS do nó chạy trên giao thức UDP thích hợp với việc vượt NAT cho các ứng dụng P2P. Tất cả các dữ liệu truyền P2P đều được bảo mật sử dụng DTLS. Cụ thể, DTLS được sử dụng trong việc thống nhất khóa bảo mật cho việc mã hóa dữ liệu media và cho việc bảo mật sự vận chuyển dữ liệu ứng dụng. Mặc dù cung cấp tính mã hóa, tính toàn vẹn, nhưng phần xác thực trong WebRTC được gán cho ứng dụng.

- ✓ **STUN**

Session Traversal Utilities for NAT - STUN, là giao thức giúp cho việc vượt NAT trong quá trình thiết lập kết nối. Trong WebRTC, một STUN client sẽ được xây dựng trong User Agent của trình duyệt để kết nối đến STUN server ngoài Internet. STUN server thực thi nhiệm vụ khá đơn giản, kiểm tra thông tin địa chỉ IP, port của request đến từ ứng dụng sau NAT, sau đó trả thông tin đó về dưới dạng response, nói cách khác là STUN giúp ứng dụng biết địa chỉ IP, cổng của nó sử dụng khi đi ra Internet. STUN có thể được vận chuyển trên UDP, TCP hoặc TLS

Trong đa số các trường hợp thì chỉ cần sử dụng STUN trong việc thiết lập kết nối P2P, trừ trường hợp một peer đứng sau symmetric NAT, một peer đứng sau Symmetric NAT hoặc port-restricted NAT. Trường hợp này quá trình hole punching sẽ không thành công, cần phải sử dụng đến TURN - *Traversal Using Relays around NAT*.

- ✓ **TURN**

Traversal Using Relays around NAT - TURN, là một mở rộng (extension) của giao thức STUN, cung cấp media relay cho tình huống thực hiện hole punching không thành công. Trong WebRTC, User Agent của trình duyệt sẽ bao gồm một TURN client. TURN server

được cung cấp trên Internet qua các nhà cung cấp dịch vụ, hoặc có thể cài đặt trong mạng doanh nghiệp. Giao thức UDP được sử dụng để giao tiếp giữa TURN client và TURN server qua NAT. Cổng UDP mặc định cho TURN là 3478. Trên thực tế TURN server thường là STUN server có bổ sung tính năng relay. TURN server thì có chức năng STUN, nhưng không phải mọi STUN server đều có chức năng TURN.

- ✓ **ICE**

Interactive Communication Establishment - ICE là giao thức quan trọng trong WebRTC, sử dụng kỹ thuật hole punching [RFC5128] để vượt NAT.

CHƯƠNG 3: BÁO HIỆU TRONG WEBRTC

3.1. Vai trò của báo hiệu

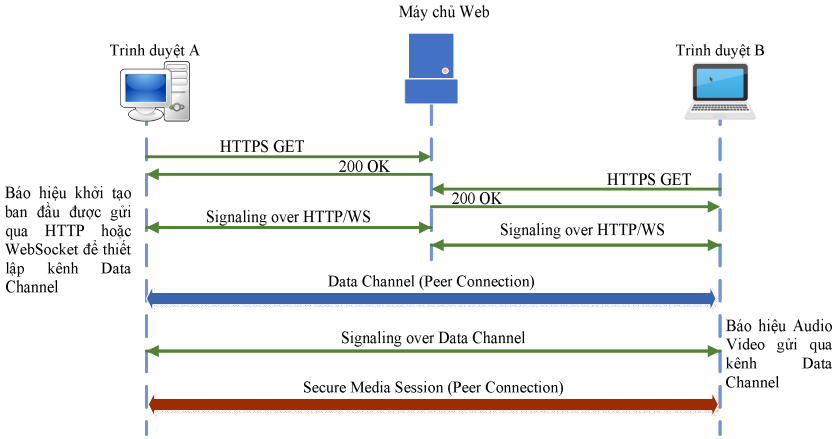
Quá trình chuyển các thông điệp từ trình duyệt này qua máy chủ trung gian đến trình duyệt khác được gọi là quá trình báo hiệu, hay gọi tắt là báo hiệu. Máy chủ trung gian là máy chủ báo hiệu. Báo hiệu không được chuẩn hóa trong WebRTC, cho phép nhà phát triển ứng dụng tự do lựa chọn phương án phù hợp. Trong truyền thông thời gian thực, báo hiệu có bốn vai trò chính:

- *Xác định địa chỉ vận chuyển (IP và port) của peer:* trao đổi địa chỉ ứng viên trong quá trình ICE hole punching
- *Thương lượng/thống nhất khả năng media và các thiết lập cấu hình*: đây là chức năng quan trọng nhất của báo hiệu, giúp trao đổi thông tin thường được chứa trong đối tượng SDP giữa các trình duyệt tham gia vào Peer Connection. SDP chứa tất cả các thông tin cần thiết cho RTP media stack trên trình duyệt để cấu hình media session, bao gồm loại media (voice, video, data), codecs sử dụng (Opus, G.711,..) hay bất kỳ tham số hay thiết lập nào cho codecs, thông tin về băng thông.
- *Định danh và xác thực các bên tham gia trong session*
- *Điều khiển media session*: khởi tạo, đóng, thay đổi session

Lý do báo hiệu không được chuẩn hóa trong WebRTC đơn giản vì nó không cần được chuẩn hóa để giúp trình duyệt có khả năng tương tác với nhau. Hiện nay, quá trình báo hiệu đang được xây dựng dựa trên Javascript Session Establishment Protocol (JSEP), là cơ chế cho phép ứng dụng JavaScript kiểm soát hoàn toàn phần báo hiệu của phiên đa phương tiện qua API RTCPeerConnection.

3.2. Giao thức vận chuyển báo hiệu

Các giao thức được sử dụng phổ biến cho vận chuyển báo hiệu WebRTC là HTTP, WebSocket và đặc biệt kênh Data Channel. Kênh dữ liệu, sau khi được thiết lập P2P giữa trình duyệt, cung cấp kết nối trực tiếp, độ trễ thấp nên cũng phù hợp với việc vận chuyển báo hiệu. Vì sự khởi tạo thiết lập kênh dữ liệu (Data Channel) đòi hỏi cơ chế báo hiệu riêng, kênh dữ liệu không thể sử cho tất cả báo hiệu WebRTC. Tuy nhiên, nó có thể sử dụng để handle các báo hiệu khác sau khi thiết lập thành công kênh trực tiếp, bao gồm báo hiệu cho audio, video media qua Peer Connection.



Hình 3.2. Vận chuyển báo hiệu trên Data Channel

3.3. Giao thức báo hiệu

Một phần quan trọng trong xây dựng ứng dụng WebRTC là lựa chọn giao thức báo hiệu, nó không cần thiết gắn với việc lựa chọn

giao thức vận chuyển báo hiệu. Ta có thể chọn giao thức báo hiệu chuẩn như SIP, Jingle hoặc một cách báo hiệu riêng tự định nghĩa..

✓*Sử dụng giao thức báo hiệu SIP*

SIP là giao thức báo hiệu thường sử dụng trong VoIP và hệ thống hội nghị truyền hình. SIP có thể sử dụng UDP, TCP, SCTP hay TLS cho việc vận chuyển, tuy nhiên thường thì sử dụng Websocket.

✓*Sử dụng giao thức báo hiệu Jingle over WebSockets*

Jingle là một mở rộng của XMPP (extensible Messaging and Presence Protocol), được biết đến là Jabber [RFC6120]), thêm khả năng báo hiệu media cho XMPP. Jingle cung cấp cách chuyển mô tả phiên SDP sang định dạng XML, sau đó có thể được vận chuyển qua TCP hoặc TLS đến máy chủ XMPP. Để triển khai báo hiệu sử dụng Jingle, client phải load đoạn mã JavaScrip từ máy chủ để thiết lập kết nối XMPP qua WebSockets với máy chủ XMPP. Client sau đó sẽ map thông tin SDP offer và SDP answer được tạo ra bởi trình duyệt thành thông điệp Jingle call setup và chuyển tiếp nó cho trình duyệt kia.

Sử dụng JSON over WebSockets

Hướng tiếp cận này được Google sử dụng và tương đối phổ biến hiện nay. JSON có thể coi là tập con cú pháp JavaScript nên có ưu điểm lớn khi thông dịch bởi trình duyệt web. Tất cả cấu trúc dữ liệu và thông tin trạng thái trong ứng dụng web được lưu trữ trong đối tượng đều được map rõ ràng, trực tiếp vào JSON nên không đòi hỏi nhiều nỗ lực cho việc mã hóa (encoding), phân tích (parsing) và xử lý (processing. Sử dụng JSON cùng với thư viện đảm nhận việc thiết lập và duy trì kênh hai chiều tin cậy với máy máy chủ báo hiệu là khá đơn giản và hiệu quả, dù phát sinh thêm chi phí dựng cổng ứng dụng tùy biến (custom gateway) nếu muốn kết nối ứng dụng web với dịch vụ thông tin liên lạc bên ngoài.

3.4. Các quá trình trong báo hiệu

Có ba quá trình "**bán**" bất đồng bộ chính trong thiết lập session WebRTC bao gồm:

- **WebRTC Javascript callback logic:** các logic này handle tất cả những xử lý mức trình duyệt của WebRTC
- **STUN/TURN Sever Session Description Protocol (SDP) messaging:** Đây là logic báo hiệu diễn ra ngoài kết nối WebRTC để cài đặt kết nối P2P giữa 2 trình duyệt theo yêu cầu.
- **ICE (Interactive Connectivity Establishment) messaging:** quá trình hỗ trợ vượt NAT, chuyển tiếp (relay) media/data trong trường hợp cần thiết.

Các quá trình tạm gọi là bán đồng bộ vì trong chuỗi kết nối, luồng logic call back và luồng logic báo hiệu được kích hoạt (gọi) lẫn nhau. Trong đó quá trình SDP, ICE nêu trên đều là một phần của báo hiệu, đều cần có máy chủ báo hiệu riêng để chuyển tiếp thông điệp SDP, hoặc để chuyển tiếp thông điệp ICE.

CHƯƠNG 4: ỨNG DỤNG WEBRTC CHO GIẢI PHÁP CỘNG TÁC VÀ CHIA SẺ DỮ LIỆU ĐA PHƯƠNG TIỆN TẠI TRUNG TÂM MVAS

4.1. Thư viện WebRTC và các hướng tiếp cận

4.1.1. Các thư viện WebRTC

Để đơn giản hóa việc triển khai ứng dụng, cộng đồng mã nguồn mở/các vendor đã phát triển rất nhiều các thư viện, framework WebRTC JavaScript, một số tên phổ biến: EasyRTC, PeerJS, Pubnub, Simple WebRTC, Xirsys…Có rất nhiều sự lựa chọn thư viện trên nền WebRTC để phát triển ứng dụng WebRTC. Điểm chung của các thư viện, framework này là cung cấp tập các chức năng qua các APIs của nó, giúp các nhà phát triển JavaScript tích hợp WebRTC vào trang web một cách đơn giản nhất. Ẩn trong các thư viện là các cơ chế gần giống nhau, giúp các nhà phát triển không phải quan tâm đến phần vận chuyển báo hiệu lớp dưới (HTTP, WebSocket, XMPP…), giao thức báo hiệu (SIP, XMPP, JSON,..) mà chỉ cần có kỹ năng lập trình Web với JavaScript. .

Ngoài những hàm API theo cơ chế chung, nhiều thư viện cung cấp các hàm "nâng cao" hơn, thường đặc thù cho các dịch vụ mà thư viện hướng tới. Dựa trên những thư viện đã và đang tiếp tục được hoàn thiện, đã có những ứng dụng hoàn chỉnh trên Internet cung cấp dịch vụ và giới thiệu WebRTC đến với người dùng, đặc biệt là các Web chuyên về chia sẻ file https:// www.sharefest.me, https://reep.io, https://www.sharedrop.io hay những Web hỗ chợ text/voice/video chat như https://appear.in, https://opentokrtc.com, https://helo.firefox.com, https://vline.com, https://talky.io, https://hubl.in/...

4.1.2. Các hướng tiếp cận sử dụng WebRTC

Có nhiều cách tiếp cận để xây dựng ứng dụng WebRTC phục vụ nhu cầu của doanh nghiệp/cá nhân, trong đó nhóm lại thành bốn hướng như sau:

- Chủ động tự xây dựng ứng dụng sử dụng WebRTC từ đầu.
- Sử dụng dịch vụ của những nhà cung cấp dịch vụ đám mây SaaS.
- Lựa chọn nền tảng WebRTC API và phát triển ứng dụng
- Hướng tiếp cận lai: Hướng tiếp cận này đề cập đến nhiều thành phần khác nhau, được chia theo hai nhóm: "Wrappers phía client + Máy chủ báo hiệu" và "Thành phần chức năng phía server"

4.2. Ứng dụng WebRTC thử nghiệm cho việc cộng tác, chia sẻ dữ liệu giữa các máy khách tại Mobifone

4.2.1. Hiện trạng

Tổng công ty viễn thông Mobifone được thành lập từ năm 1993, là doanh nghiệp đầu tiên của Việt Nam khai thác dịch vụ thông tin di động GSM 900/1800. Đánh giá chung về hạ tầng CNTT của Mobifone là khá hiện đại, tuy nhiên, vẫn có một phần nhỏ chưa được Mobifone quan tâm đúng mực, mà để các đơn vị (hơn 40 đơn vị) tự chủ động thực hiện: đó là việc cộng tác, chia sẻ dữ liệu giữa cá nhân, nhóm. Vì vậy tất cả các đơn vị vẫn sử dụng các hình thức chia sẻ file kiểu cũ nhiều nhược điểm : Sử dụng chức năng chia sẻ network file sharing của Windows trên SMB Protocol, Sử dụng FTP server trong mạng nội bộ, Sử dụng hệ thống Email, Sử dụng các công cụ đồng bộ trên private cloud (tương tự như Dropbox, OneDrive… nhưng sử dụng trong nội bộ), sử dụng các ứng dụng OTT, P2P. Về hoạt động cộng tác, trao đổi thông tin chủ yếu là sử dụng các ứng dụng chat nội bộ hoặc dùng những ứng dụng phổ biến như Skype, Viber, Facebook

Messenger, Zalo… không đảm bảo an toàn bảo mật buộc phải tin tưởng vào uy tín của các công ty phát triển phần mềm.

4.2.2. Yêu cầu hệ thống cộng tác thử nghiệm WebRTC tại Trung tâm MVAS Mobifone

Hệ thống cộng tác tại Trung tâm MVAS Mobifone cần đảm bảo một số yêu cầu chính như sau:

- Hệ thống cho phép xác thực qua hệ thống Active Directory của Mobifone
- Người dùng không cần cài đặt thêm phần mềm thứ 3, plugin
- Việc chat, chia sẻ file được thực hiện Peer to peer, không qua máy chủ trung gian. Hỗ trợ chia sẻ những file kích thước lớn.
- Hỗ trợ Voice chat P2P thời gian thực
- Có khả tăng tạo nhóm cộng tác, chat và chia sẻ file trong nhóm.
- Ngoài các yêu cầu trên, ứng dụng cung cần đảm bảo thông tin trao đổi cần được mã hóa, bảo mật, tránh thất thoát thông tin cho bên thứ ba.

4.2.3. Lựa chọn thư viện

Sử dụng framework EasyRTC do Priologic xây dựng. Lý do lựa chọn hướng tiếp này để đảm bảo cán bộ IT Trung tâm MVAS có thể quản lý toàn bộ giải pháp, không phải viết lại code từ đầu mà tận dụng thư viện có sẵn EasyRTC, không tiêu tốn chi phí thuê API Service ngoài do nhu cầu tương tác nội bộ.

Browser EasyRTC API

Easyrtc.js: là đối tượng quan trọng nhất, chứa các thuộc tính, phương thức chính cho việc viết ứng dụng bao gồm phương thức kết nối đến máy chủ báo hiệu, Phương thức khởi động cuộc gọi đến người khác, phương thức ngắt kết nối đến máy chủ báo hiệu, đặt listener cho

thông điệp nhận từ máy chủ, Phương thức gửi tin nhắn P2P, Phương thức gửi tin nhắn đến ứng dụng trên máy chủ, các phần liên quan đến xử lý lỗi (error handling)

Easyrtc_ft.js : Chứa các phương thức để làm việc với file (truyền nhận file) giữa các peer.

Server API

Chứa các đối tượng quản lý ứng dụng, kết nối, room, session..

4.2.4. Phân tích thiết kế hệ thống

Ứng dụng được xây dựng dựa trên những module chính sau:

- Module xác thực:
- Module gửi file:
- Module quản lý, hỗ trợ gửi message P2P giữa các client.
- Module quản lý, hỗ trợ gọi và nhận cuộc gọi audio streaming audio P2P giữa client.
- Module quản lý nhóm hay room cộng tác.

4.2.5. Phát triển ứng dụng

Môi trường phát triển

- Hệ điều hành: Windows, linux, OSX, Ubuntu…
- Thiết lập môi trường phát triển NodeJS, tích hợp các module passport-ldap hỗ trợ xác thực qua tài khoản LDAP, module express-session phục vụ quản lý phiên từ passport.
- Máy chủ báo hiệu được xây dựng sử dụng socket.io trên NodeJS, phục vụ gửi/nhận thông tin giữa server và client trình duyệt.
- Sử dụng các API của EasyRTC trong các file: easyrtc.js, easyrtc_int.js, easyrtc_ft.js, easy_app.js, adapter.js;
- LDAP server.

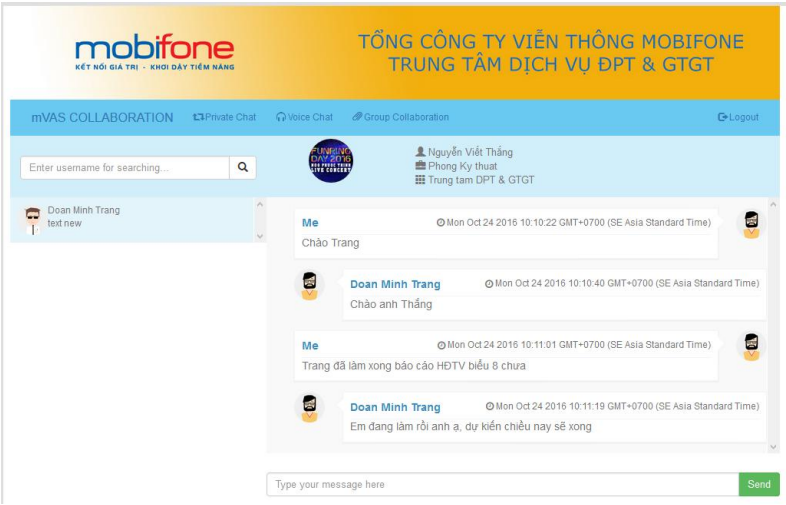
Giao diện:

- Thiết kế giao diện theo chuẩn "Responsive Design" giúp tự động dàn trang, hiển thị trên các kích cỡ màn hình khác nhau (PC, smartphone, tablet..). Thiết kế sử dụng Bootstrap framework, là framework phổ biến nhất cho việc thiết kế html, css, JavaScrip cho các web tương tác.
- Sử dụng EJS, ngôn ngữ bản mẫu phía client, cho phép kết hợp dữ liệu và mẫu template để tạo ra HTML. Về logic thiết kế, với ứng dụng cộng tác, phần danh sách người dùng được thể hiện danh sách ở bên trái, nội dung trao đổi ở phía tay phải. Thanh điều hướng cho các tính năng chat riêng, chat nhóm, voice chat được đặt ngay phía trên cho trực quan.

4.2.6. Kết quả thử nghiệm và đánh giá

Hệ thống được cài đặt và thử nghiệm trên máy nội bộ theo địa http://mChat.mobifone.vn:8080

Người dùng khi truy cập vào trang nếu chưa được xác thực sẽ được redirect sang trang xác thực tại địa chỉ http://mChat.mobifone.vn:8080/login. Trang login cho phép xác thực khi người dùng nhập username và mật khẩu của hệ thống email nội bộ Mobifone, thực chất là xác thực sử dụng giao thức LDAP với hệ thống Active Directory trên nền tảng Windows Server 2012 của Mobifone. Mã xác thực qua ldap được triển khai dựa trên bộ thư viện Google Passport. Thư viện này hỗ trợ cả xác thực qua Facebook, Google+.



Hình 4.6: Giao diện Private Chat

Người dùng sau khi truy cập vào web http://mchat.mobifone.vn đều được thể hiện ở cột bên trái. Khi người dùng cần trao đổi cộng tác với cán bộ khác, có thể sử dụng bộ filter theo username ở phía trên, sau đó click chuột chọn user cần trao đổi, chat và chia sẻ file đơn giản như giao diện ở dưới. Việc gửi text message sử dụng phương thức sendDataP2P trong module EasyRTC.js như nêu ở trên. Việc gửi file sử dụng các phương thức trong module Easyrtc_ft.js.

Người dùng tương tác nhóm có thể click vào link Group Collaboration trên thanh điều hướng giao diện. Trong phần cộng tác nhóm, người dùng có thể tạo ra các nhóm với tên và mật khẩu để đảm bảo chỉ người dùng phù hợp mới tham gia nhóm. Thông tin tên và mật khẩu người dùng có thể gửi cho người dùng phù hợp để tham gia nhóm. Khi muốn tham gia nhóm, người dùng có thể click chọn tab Join Group, chọn nhóm muốn tham gia với điều kiện biết tên và mật khẩu chủa nhóm.Thông tin cột bên trái thể hiện các nhóm mà người

dùng hiện tại đang tham gia. Thành viên của nhóm dược thể hiện ở cột giữa, xuất hiện khi người dùng click chọn nhóm ở cột trái. Cột ngoài cùng là thông tin trao đổi giữa những thành viên trong nhóm. Do thời gian hạn chế, đề tài chỉ demo phần cộng tác text tất cả cá nhóm trên một phần tử ul HTML5, và tính năng chia sẻ file chỉ thể hiện ở phần private chat.

Ứng dụng demo cũng hỗ trợ các cán bộ thực hiện call và voice chat với cán bộ khác. Để sử dụng chức năng này, người dùng truy cập vào menu Voice chat trên thanh điều hướng, click vào nút Connect để sẵn sàng kết nối với cán bộ khác.

Đánh giá ứng dụng

Với ứng dụng web đơn giản, cán bộ công nhân viên của Trung tâm MVAS đã có thể dễ dàng trao đổi thông tin, tài liệu với nhau trực tiếp, theo nhóm mà tương đối tiện lợi, đảm bảo an toàn bảo mật. Chất lượng voice tương đối tốt, hai đầu nghe rõ. Như vậy, ứng dụng đã đáp ứng được các yêu cầu chính đã đặt ra. Tuy nhiên, ứng dụng còn một số hạn chế chưa thể giải quyết ở thời điểm hiện tại như chưa hỗ trợ trình duyệt IE, Safari, là hai trình duyệt khá phổ biến với người dùng trong Mobifone. Ứng dụng cũng mới chỉ demo các tính năng riêng lẻ, chưa hoàn thiện phương án gộp lại trong một giao diện màn hình để tiện cho việc sử dụng hơn giống những ứng dụng OTT.

So với nhiều những dự án mã nguồn mở hỗ trợ text/voice/video chat miễn phí trên nền WebRTC đã public trên mạng như https://Sharefest.vn, https://hubl.in, *https://talky.io...,* ứng dụng tương tác trong Trung tâm MVAS điểm khác biệt sau:

- ✓ Tích hợp xác thực với hệ thống quản trị tài nguyên mạng tập trung Microsoft Active Directory của Mobifone. Phần xác thực chính là phần WebRTC

không định ra tiêu chuẩn, mà chuyển cho ứng dụng quản lý.

- ✓ Hỗ trợ tạo nhóm cộng tác và quản lý mật khẩu đảm bảo truy nhập an toàn cho nhóm.

Tuy nhiên, nếu so với những ứng dụng OTT phổ biến, ứng dụng demo chỉ có ưu điểm về việc đơn giản trong sử dụng, không cần cài đặt plugin, yên tâm về bảo mật mà không đòi hỏi đầu tư hạ tầng mạnh. Để có thể triển khai và áp dụng rộng rãi tại Mobifone, ứng dụng cần bổ sung thêm những tính năng nâng cao.

CHƯƠNG 5: KẾT LUẬN CHUNG

5.1. Các đóng góp của luận văn

Với yêu cầu của đề tài luận văn nghiên cứu ứng dụng WebRTC trong việc xây dựng giải pháp cộng tác và chia sẻ dữ liệu đa phương tiện tại Trung tâm MVAS – Mobifone, luận văn đã thu được một số kết quả sau:

- Tìm hiểu được những nội dung cơ bản của WebbRTC như: các chuẩn giao thức, APIs trong WebRTC, cách thức vượt NAT trong WebRTC
- Nghiên cứu sâu về báo hiệu, vai trò của báo hiệu và các quá trình trong báo hiệu WebRTC.
- Khảo sát và đánh giá các thư viện WebRTC, các hướng tiếp cận sử dụng thư viện WebRTC
- Nghiên cứu và sử dụng thư viện EasyRTC trong việc xây dựng ứng dụng Peer-to-Peer tương tác thời gian thực
- Phân tích yêu cầu, thiết kế ứng dụng, cài đặt thử nghiệm hệ thống cộng tác tại mạng nội bộ Trung tâm dịch vụ Đa phương tiện và giá trị gia tăng Mobifone - Tổng công ty Viễn thông Mobifone và đạt được những kết quả đáng khích lệ.

5.2. Một số hướng phát triển

Qua nghiên cứu WebRTC và thử nghiệm ứng dụng, tôi nhận thất WebRTC là công nghệ rất tiềm năng, đặc biệt hiệu quả khi triển khai nhanh những ứng dụng đòi hỏi tương tác thời gian thực giữa các tình duyệt với tính đơn giản khi cài đặt, dễ sử dụng với người dùng. Các hạn chế của WebRTC như chưa được hỗ trợ bởi IE của Microsoft, Safari của Apple hiện tại đã có giải pháp cài đặt thêm các WebRTC plugin, dù như vậy nó không đúng theo tiêu chí là không plugin mà

WebRTC hướng đến. Vì vậy, WebRTC vẫn đang tiếp tục được nghiên cứu chuẩn hóa (bản update mới nhất là vào tháng 9/2016), tiếp tục phát triển, tương lai ứng dụng WebRTC sẽ có tác động không nhỏ đến ngành công nghiệp web, thậm chí thay thế các ứng dụng cộng tác hiện tại.

Với phân tích đánh giá kết quả thử nghiệm ứng dụng, hướng phát triển tiếp theo của đề tài có thể nghiên cứu, phát triển tiếp những công việc sau:

- Hoàn thiện các tính năng tương tự như các OTT như hỗ trợ lưu thông tin chat office, cảnh báo notification.
- Bổ sung khả năng resumable cho việc gửi/nhận file
- Nghiên cứu khả năng phát triển tính năng chia sẻ màn hình – screen sharing. Đến thời điểm hiện tại mới chỉ có trình duyệt Chrome hỗ trợ để ứng dụng có thể phát triển tính năng này, về bản chất là chụp ảnh màn hình liên tục và gửi cho trình duyệt đầu xa.
- Nghiên cứu thêm về cách cài đặt tối ưu hiệu năng từng chức năng như máy chủ EasyRTC, máy chủ báo hiệu, lựa chọn phương án tối ưu trong trường hợp số lượng người dùng lên đến hơn 5000 cán bộ công nhân viên.

Hoàn thành các việc nghiên cứu này thì ứng dụng cộng tác chia sẻ dữ liệu đa Phương tiện tại Trung tâm MVAS có thể ứng dụng cho không chỉ TCT Viễn thông Mobifone nói riêng mà cho tất cả các doanh nghiệp, tổ chức nói chung, đảm bảo được người dùng đón nhận.

TÀI LIỆU THAM KHẢO

TIẾNG ANH

1. Alan B.Johnson, Daniel C.Burnett (2014), APIs and RTCWEB Protocols of the HTML5 Real-Time Web, Digital Codex LLC
2. Salvatore Loreto, Simon Pietro Romano (2014), Real-time Communication with WebRTC, O'Reilly, USA
3. Andrii Sergiienko (2014), WebRTC Blueprints, Packt Publishing Ltd, UK
4. Ilya Grigorik (2015), High Performance Browser Networking, O'Reilly Media.
5. Altanai (2014), WebRTC Intergrator's Guide, Packt Publishing Ltd, UK
6. WebRTC for Enterprises
7. Tsahi Levent-Levi (2013), WebRTC for Business People: Unraveling the challenges and opportunities of the WebRTC ecosystem
8. Dan Ristic (2015), Learning WebRTC, Packt Publishing Ltd, UK
9. Rob Manson (2013), Getting Started with WebRTC, Packt Publishing Ltd, UK
10. WebRTC Architecture, https://webrtc.org/architecture, Thời gian truy cập: 11-09-2016
11. RFC 1631 - The IP Network Address Translator (NAT), 1994, https://tools.ietf.org/html/rfc1631
12. RFC 6716 - Definition of the Opus Audio Codec, 2012 https://tools.ietf.org/html/rfc6716

13. RFC 5245, Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols, 2012, https://tools.ietf.org/html/rfc5245
14. RFC 5389, Session Traversal Utilities for NAT (STUN), 2008, https://tools.ietf.org/html/rfc5389
15. RFC 4960, Stream Control Tranmission Protocol (SCTP), 2007, https://tools.ietf.org/html/rfc4960
16. RFC 4347, Datagram Transport Layer Security (DTLS), 2006 https://tools.ietf.org/html/rfc4347
17. RFC 3711, The Secure Real-time Transport Protocol (SRTP), 2004, https://www.ietf.org/rfc/rfc3711.txt
18. RFC 4566, SDP: Session Description Protocol, 2006, https://tools.ietf.org/html/rfc4566
19. RFC 5246, The Transport Layer Security (TLS) Protocol Version 1.2, 2008, https://tools.ietf.org/html/rfc5246
20. RFC 5128, State of Peer-to-Peer (P2P) Communication across Network Address Translators (NATs), 2008, https://tools.ietf.org/html/rfc5128
21. RFC 5766, Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN), 2010, https://tools.ietf.org/html/rfc5766
22. EasyRTC website, https://easyrtc.com/docs/browser/easyrtc.php, Thời gian truy cập: 11-09-2016
23. https://www.pkcsecurity.com/blog. Thời gian truy cập 11-10-2016
24. RFC 3264, An Offer/Answer Model with the Session Description Protocol (SDP), 2002, https://tools.ietf.org/html/rfc3264

25. Javascript Session Establishment Protocol draft-ietf-rtcweb-jsep version 16, 20-09-2016, https://tools.ietf.org/html/draft-ietf-rtcweb-jsep-16