National College of Ireland

Scaling WebRTC video broadcasting using partial mesh model with location based signalling

MSc Research Project
Cloud Computing

Adesh Rohan D'Silva
Student ID: x18176097

School of Computing
National College of Ireland

Supervisor: Manuel Tova-Izquierdo

National College of Ireland
Project Submission Sheet
School of Computing

National College of Ireland

Student Name:	Adesh Rohan D'Silva
Student ID:	x18176097
Programme:	Cloud Computing
Year:	2020
Module:	MSc Research Project
Supervisor:	Manuel Tova-Izquierdo
Submission Due Date:	17/08/2020
Project Title:	Scaling WebRTC video broadcasting using partial mesh model with location based signalling
Word Count:	879
Page Count:	6

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

I agree to an electronic copy of my thesis being made publicly available on TRAP the National College of Ireland's Institutional Repository for consultation.

Signature:	
Date:	16th August 2020

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Scaling WebRTC video broadcasting using partial mesh model with location based signalling

Adesh Rohan D'Silva
x18176097

1 Introduction

This User Manual gives detailed instructions on installation, configuration and execution/testing of the partial-mesh based WebRTC application artefact which is part of the research project for this thesis. The instructions outlined here for testing the application are provided for Amazon Web Services (AWS) but the same instructions can be followed in any other cloud provider on any Virtual Machine (VM) but with the required software installed. This guide assumes that the reader has knowledge in launching VMs on cloud service providers like AWS or Microsoft Azure and is familiar with SSH and Linux operating systems.

2 System Specification

2.1 Hardware requirements

The hardware requirements given here can be used to run at most 20 parallel sessions (users) so if you want to run more than this number you may need more powerful hardware or use multiple VMs with the same hardware.

Equivalent EC2 instance type	t2.medium
No. of vCPU	2
Memory	4GB

2.2 Software requirements

The below software requirements are for both running WebRTC web application and the test sessions. Apart from below requirements, to see the actual application, the user can use any browser (Chrome preferred) on their computer.

- Ubuntu 18.04 LTS (latest recommended)
- Node.js
- Git

3 AWS EC2 environment setup

3.1 Launching EC2 instance

We will be launching a t2.medium instance with Ubuntu Server 18.04 LTS or the Amazon Machine Image (AMI) ami-0bc556e0c71e1b467 using AWS EC2 launch instance wizard.

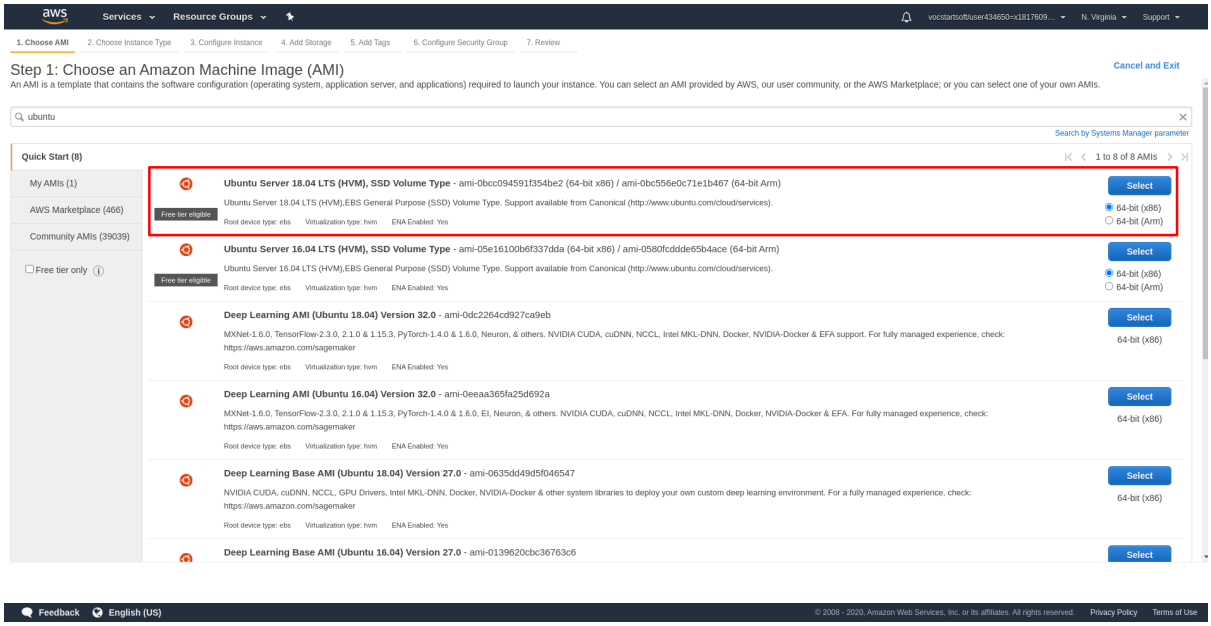

Figure 1: Choosing AMI image

Select the highlighted instance as shown in Figure 1, then select **t2.medium** as the instance type and then continue clicking "Next" keeping all options as default until you reach **Configure Security Group**

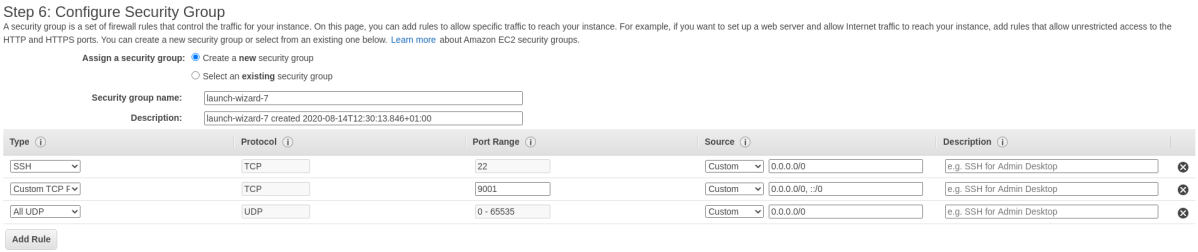

Figure 2: Security rules

You need to add security rules as shown in Figure 2 so that the node application can run and communicate in the network. After configuring the security group, you can launch the instance.

3.2 Configuring EC2 instance

Connect to the launched EC2 instance with SSH using your private key file for the launched instance. You can run below commands to configure and install all the requirements for running the test application.

Installing Node.js(Rahul; 2020) and test app:

```
$ curl -sL https://deb.nodesource.com/setup_14.x | sudo -E bash -
$ sudo apt-get install -y nodejs
$ git clone https://github.com/adeshrd/webrtc-test
$ cd webrtc-test
$ npm install
```

4 Launching WebRTC Application

4.1 Installing pre-requisites

You will be running these commands from your computer with Ubuntu installed. This will install Heroku CLI which is required for deploying the application

```
$ sudo snap install --classic heroku
```

4.2 Running the application

We will be launching the application to Heroku [1] platform as it offers SSL support byfefault which is required for WebRTC [2]. You can optionally launch the app in a AWS VM but you need to ensure that the application is being served over https protocol.

Run application [3]:

```
$ git clone https://github.com/adeshrd/webrtc-scalable-broadcast
$ cd webrtc-scalable-broadcast
$ heroku login
$ heroku create
$ git push heroku master
```

These commands will first clone the code from the Github repository and initialize the Node.js application in the Heroku platform.

[1] http://heroku.com/

[2] https://groups.google.com/g/discuss-webrtc/c/sq5CVmY69sc?pli=1

[3] https://devcenter.heroku.com/articles/getting-started-with-nodejs?singlepage=true

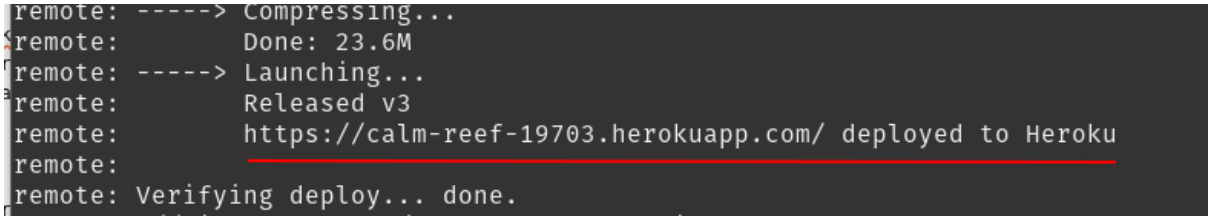

Figure 3: Application deployment output

After running above commands, you will see that the application is deployed to Heroku to an url as shown in Figure 3. You need to save this url somewhere as it will be required later.

4.3 Verify application

We can verify if the deployment was successfull by opening the url which you saved previously in any web browser by going to "/scale.html".
For example in this case you will visit: *https://calm-reef-19703.herokuapp.com/scale.html*

You should see the below page (Figure 4):

Figure 4: Application page

4.4 Launching broadcaster

We can now create a room and start the session as the broadcaster by clicking the button "Open or Join Broadcast". The Figure 5 shows the page when you create a room and start broadcasting your video.

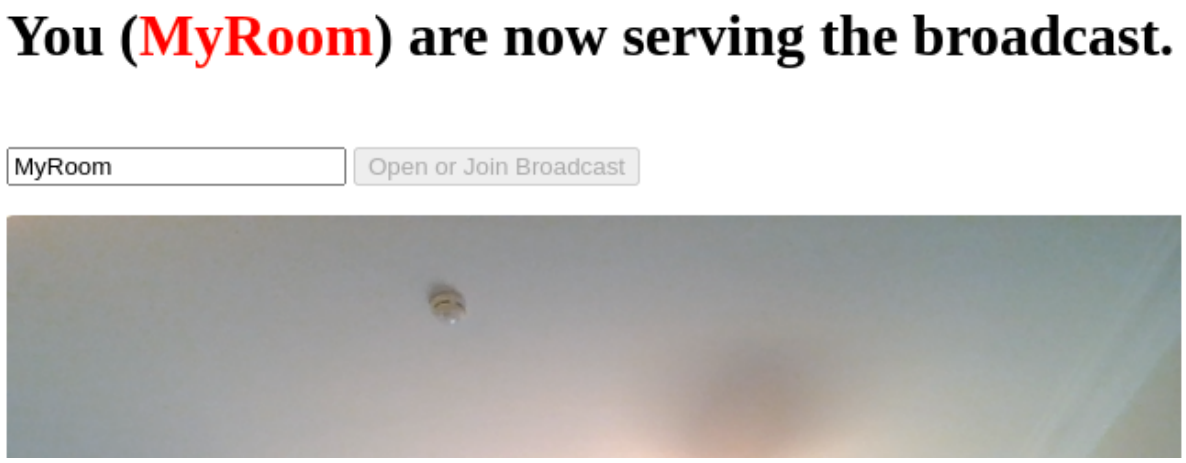

Figure 5: Broadcasting output

The video in image has been cropped so that only the relevant part of the application is visible.

5 Test sessions and output

5.1 Launch test sessions

Connect to the launched EC2 instance with SSH and run the following commands to start the test sessions using Pupeteer [4] to launch headless chrome browsers in parallel.

Launch parallel headless browsers:

```
$ cd webrtc-test
$ ./scale-par.sh N URL
```

In above command, replace N with the number of sessions you want to run in parallel and replace URL with the deployment url that you saved previously in Section 4.2

[4] https://github.com/puppeteer/puppeteer

5.2 Verify statistics/output

The WebRTC application page keeps collecting statistics during an ongoing session and will display updated statistics every 10 seconds. Once you have launched the test sessions, if you go back to the Chrome browser where the broadcasting page was opened, you will see the statistics as shown in the below Figure 6 after a few seconds.

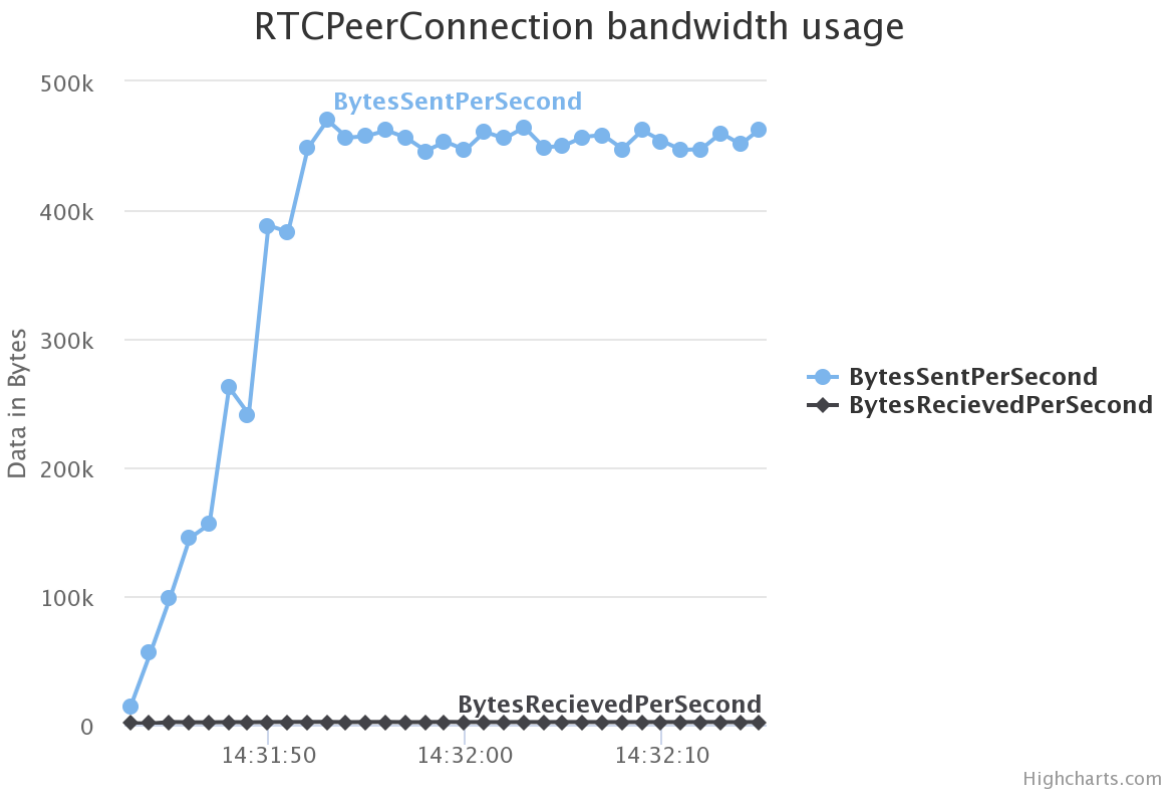

Figure 6: Broadcasting output

References

Rahul, W. b. (2020). How to install node.js on ubuntu 18.04 / 16.04 lts.
URL: *https://tecadmin.net/install-latest-nodejs-npm-on-ubuntu/*