International Journal for Modern Trends in Science and Technology, 6(12): 407-410, 2020
Copyright © 2020 International Journal for Modern Trends in Science and Technology
ISSN: 2455-3778 online
DOI: https://doi.org/10.46501/IJMTST061277
Available online at: http://www.ijmtst.com/vol6issue12.html

Open Source Software, Low Cost Video Conferencing Solution

Prasanna Bisen

Department of Information Technology, Delhi, India

To Cite this Article
Prasanna Bisen, "Open Source Software, Low Cost Video Conferencing Solution", *International Journal for Modern Trends in Science and Technology*, 6(12): 407-410, 2020.

Article Info
Received on 16-November-2020, Revised on 09-December-2020, Accepted on 12-December-2020, Published on 18-December-2020.

ABSTRACT

Video conferencing may be a technology that permits users in several locations to carry face-to-face meetings without having to maneuver to one location together. Uses for video conferencing include holding routine meetings, negotiating business deals, and interviewing job candidates

INTRODUCTION

The pandemic brings us a replacement reality where people can't be near one another and social relationships take a replacement breath. Staying at home is not easy for anyone, and that is why video calls are gaining popularity. Online conference apps help to maintain business and family connections when you can't all appear in the same room. There is a wealth of video conferencing and video chat apps to settle on from. However, if you're talking about personal business or discussing the small print of a business contract, you would like to understand the service you're using will protect your privacy. Maybe it is the privacy issues, the safety issues, or simply the entire misrepresenting its encryption thing. Regardless of the precise reason, you recognize that there has got to be a far better video-conferencing tool out there, and you're determined to seek out it.

Though such a system already exists and used for a long time now, now the adoption has increased multifold. The usage in classrooms, for example now almost video-conferencing is almost universal among schools, universities, and offices across the globe and forced us to do Work-from-Home for their employee, necessitating the need for virtual meetings and conferences to give references.

This project will overcome the problem and effects of data security and privacy violation done by the video service provider. We are proposing an alternative way to make a more secure video call, improved by an open-source community that will reduce the data breaching and increases the video call security on the internet not least in terms of scale. A variety of solutions have become popular -Zoom, Microsoft Teams, Cisco WebEx, to name a few. However, all of these tools are proprietary platforms, thereby not just increasing the cost of adoption but also raising critical privacy concerns. It is in this context that the need for a privacy-preserving / privacy-respecting video conferencing system becomes the need of the hour. Many governments and government establishments across the world have either issued warnings about their usage or directly banned these applications. Such a VC system must also be

- Scalable - able to cater to an increased number of users, quickly

- Privacy-preserving - should give the choice to self-host and therefore the ASCII text file should be freely available for public scrutiny
- Secure - communications must be encrypted and prevent from any any external influence
- Robust - able to withstand and network failures, system errors create

It is an open-source JavaScript WebRTC application and may be used for videoconferencing. One can share desktop and presentations and with just a link can invite new members for the videoconference. It is often employed by downloading the app or directly during a browser and it's compatible with any recent browser. Every user can use Jitsi.org servers or can download and install the server software on a Linux-based machine.

BACKGROUND

Jitsi may be a set of open-source projects that permits you to simply build and deploy secure videoconferencing solutions. At the guts of Jitsi are Jitsi Videobridge and Jitsi Meet, which allow you to have conferences on the web, while other projects within the community enable other features such as audio, dial-in, recording, and simulcasting [1].

Aws (Amazon web service) provide infrastructure service to business in the form of web service now commonly known as cloud computing.

Specific jitsi-meet Project

Jitsi Meet – Secure, Simple, open-source, and Scalable Video Conferences that you host on your server and use as an embed in your web application

Jitsi Videobridge – the media server engine that powers all of Jitsi's multi-party video conferences

Jigasi – a gateway service that connects SIP telephony to a Jitsi Video bridge conference

Jibri – help us to broadcast and recorder used for saving video call recordings and streaming to YouTube Live

Jidesha – it is a Chrome and Firefox extension that helps us to share screen

Specific Aws tools

Ec2 Service

SYSTEM UNDER TEST AND ENVIRONMENT

A) Cloud and Network Settings

Cloud and Network Settings All tests were done using Amazon Web Services (AWS)Elastic Compute Cloud (EC2). Each SFU and every of its connecting web client apps were run on separate Virtual Machines (VMs) within the same AWS Virtual Private Cloud(VPC) to avoid network fluctuations and interference. The instance types for the VMs used are described in Table I.

B) Web Client Applications

To test our server with the parameter to gather useful information, we made the subsequent modification to corresponding web client apps:

- increase the utmost number of participants per meeting room to 40
- support for displaying up to 9 videos with the precise same dimensions because the original test video (540×360 pixels)

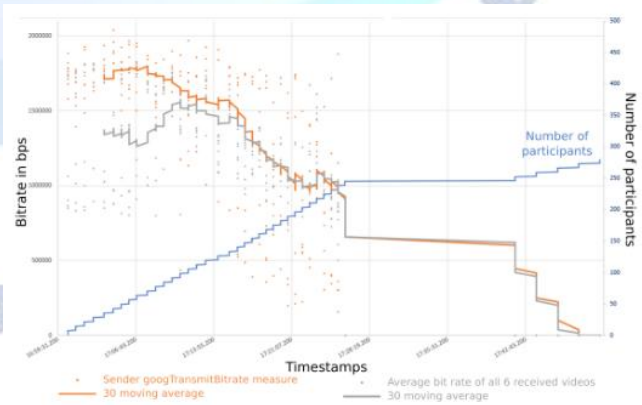
C) Metrics and Probing

1) Client-side: Video Verification

Once a <video>component has been loaded, we verify that it displays a video which it's neither blank, static nor a frozen image.

2) Client-Side: Video Quality Assessment
Chrome is launched with a fixed window size of 1980×1280 pixels, and the screenshot is taken using the jitsi function which will generate a PNG file of the same dimensions

RESULTS



I. Quantitative Results
The below figure represent the rate of success and failures that occur during the test

Fig 1 (success and failure rate)

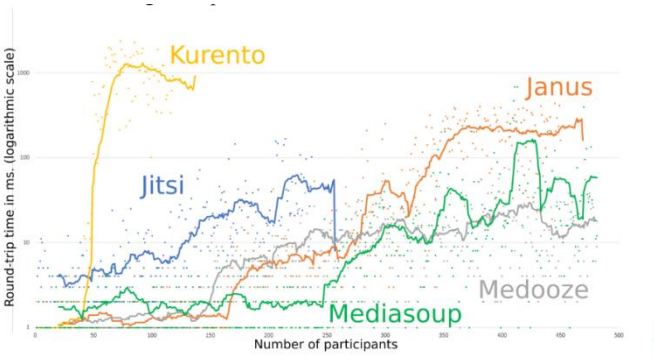


Fig 2 (Number of participants vs round trip)

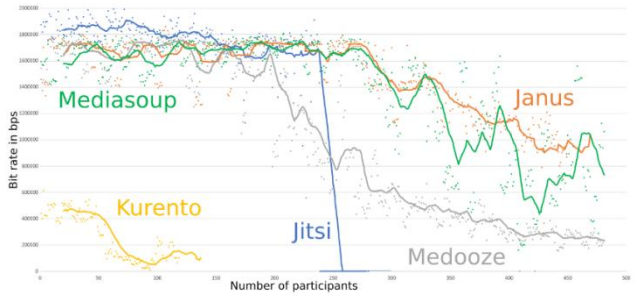


Fig 3(Number of participants vs Bit Rate)

Jitsi is able to keep about the same image quality score for the whole test. Even when 200 participants have joined the test, image quality remains stable,

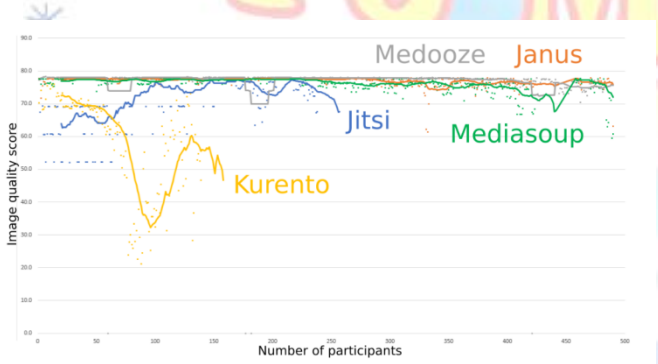


Fig 4 (number of participants vs video quality)

Video Quality Assessment

Estimation of video quality scores is presented in Fig. 4. One may expect video quality to deteriorate as the average bit rate measured falls down, but the graphs of video quality remain remarkably flat until the end of the test.

LIMITATION

210 participants can join a room at a time. After 210 participant room, automatically disconnected all the user from the room

We cant create two-room of the same name. If we try to create two-room with the same name then it will treat as a single room

ANALYSIS

This study exhibits interesting behavior of aws and jitsi meet that are evaluated once they need to handle a rise within the number of room and peers within the jitsi meet

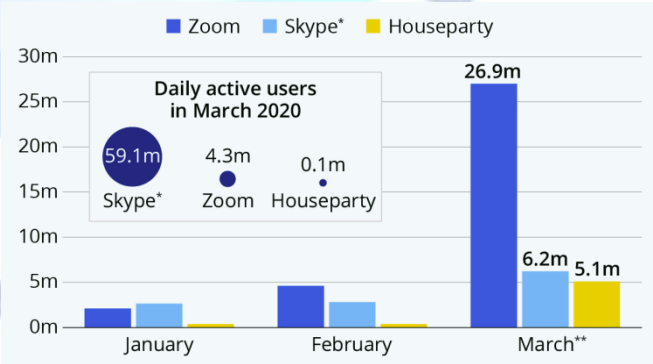
Jitsi has some internal problem that creates it suddenly stop transmitting videos when there are quite 245 peers within the test.

CONCLUSION AND FUTURE SCOPE

We have shown that it's now possible to make a videoconferencing solution hosted on our own server. Several bugs and oddities are found and reported to their respective team within the process

This work was focused on the testing system, and not on the tests themselves. within the future, we might wish to add ssh certificate to make sure our user is an end to finish encrypted, for instance using certbot we will ensure encryption.

We would wish to extend this work to variations by hosting this service on specified URLs and making changes in UI consistent with our needs



VI.A Figures and Tables

Fig 1 surge in Demand of video confessing.

	SFU	(A) Available send bandwidth (bps)	(B) Actual encoder bit rate sent (bps)	(C) Transmit bit rate sent (bps)	(D) Target encoder bit rate sent (bps)	(E) Average bit rate video sent (bps)	(F) Sender googRtt (ms)
MIN (value > 0 only)	Jitsi	113,118	153,952	156,744	113,118	130,881	1
	Janus	744,557	676,624	690,904	744,557	611,518	1
	Medooze	92,471	79,440	83,184	92,474	94,565	1
	Kurento	34,476	28,088	32,272	34,476	35,501	1
	Mediasoup	93,617	91,488	95,576	93,617	93,860	1
MAX	Jitsi	8,342,878	2,397,504	2,544,144	1,700,000	2,218,333	168
	Janus	5,000,000	2,081,072	2,108,224	1,700,000	1,963,089	435
	Medooze	9,130,857	2,782,136	4,660,656	1,700,000	4,052,553	103
	Kurento	600,000	893,912	1,003,728	600,000	676,504	2,371
	Mediasoup	7,184,000	2,049,872	2,088,776	1,700,000	2,131,814	688
Average	Jitsi	2,830,383	1,360,703	1,392,607	1,362,722	1,388,670	18
	Janus	4,210,276	1,647,705	1,677,538	1,641,714	1,682,062	61
	Medooze	2,309,558	1,000,979	1,045,173	1,009,288	1,044,573	10
	Kurento	369,775	335,393	359,979	356,457	359,540	576
	Mediasoup	2,326,640	1,385,142	1,416,096	1,401,947	1,414,368	18
% bit rate > 1 Mbps (columns (A) to (E)) % RTT > 50 ms (column (F))	Jitsi	65.4%	65.4%	65.7%	65.4%	65.7%	6.4%
	Janus	98.9%	98.4%	98.4%	98.4%	98.9%	27.8%
	Medooze	47.4%	47.4%	47.8%	47.8%	48.6%	1.9%
	Kurento	0.0%	0.0%	0.8%	0.0%	0.0%	59.3%
	Mediasoup	77.6%	76.1%	76.9%	77.6%	76.3%	6.5%

Table 1(Overview of the sender's video statistics collected on each web client)

Acknowledgment

We would like to thanks Boris Grozev (Jitsi), dr Neha Agrawal and other media server experts who provided live feed back as early as possible

References

[1][1]https://jitsi.org/what-is-jitsi/

[2][2]https://indiarxiv.org/ e94u3/

[3]https:/ieeexplore.ieee.org/abstract/document/8567642

International Journal for Modern Trends in Science and Technology

IJMTST