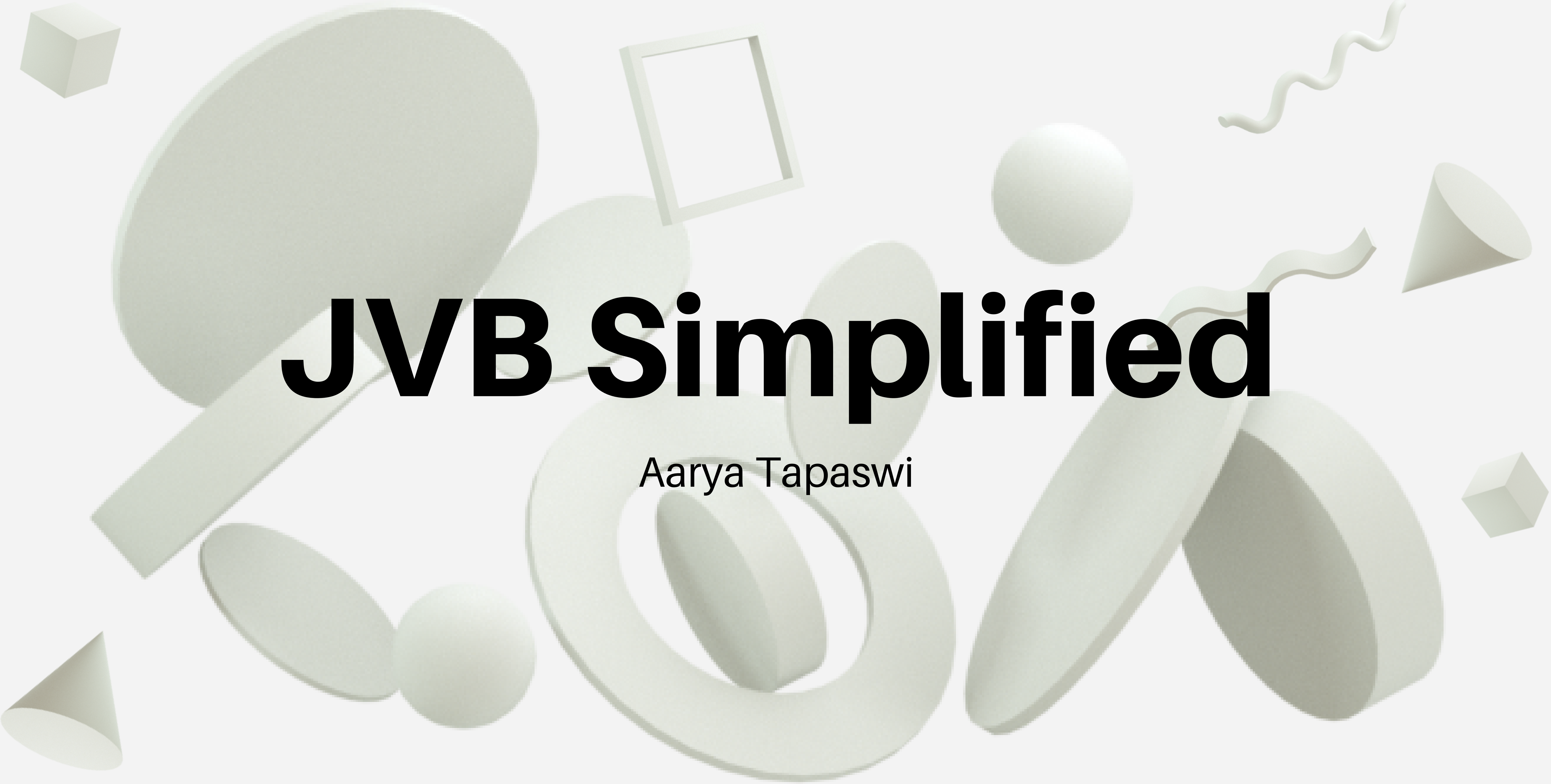
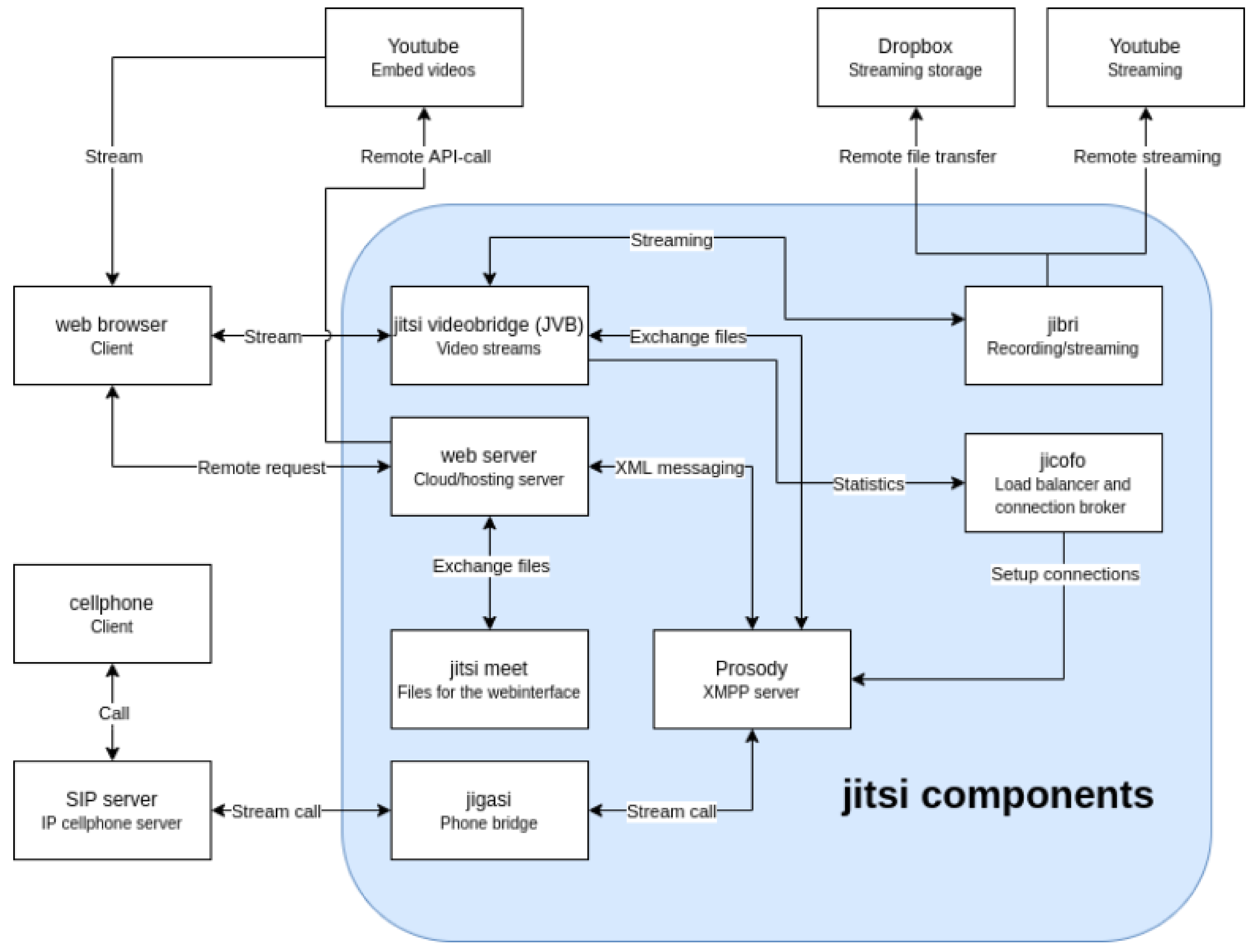


JVB Simplified

Aarya Tapaswi

What is JVB?

- Jitsi Videobridge is the media server of Jitsi Meet.
- It's an XMPP Server component
- WebRTC compatible
- and open source.

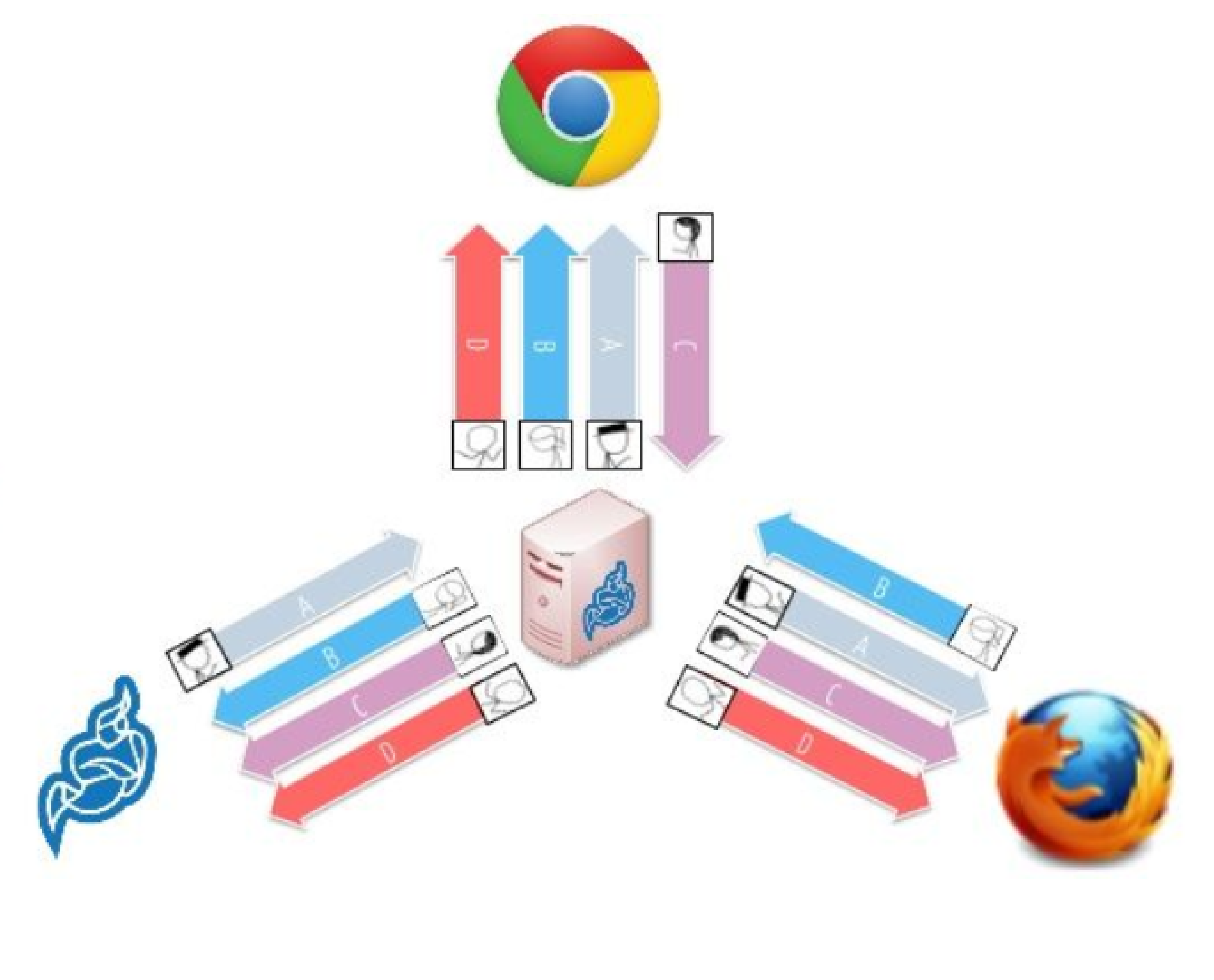


Jitsi Architecture

Jitsi Meet Architechture

How JVB works?

When building conferences with WebRTC which are not peer to peer most solutions adapt some sort of media server which act as the
the central server which handles all the streams of clients either by forwarding the incoming streams to all clients (SFU) or mixing the incoming streams and forwarding a single stream to everyone(MCU).
Jitsi Videobridge follows the first approach as it forwards the incoming streams to all clients who are connected to the videobridge.



The Pros and Cons of the JVB

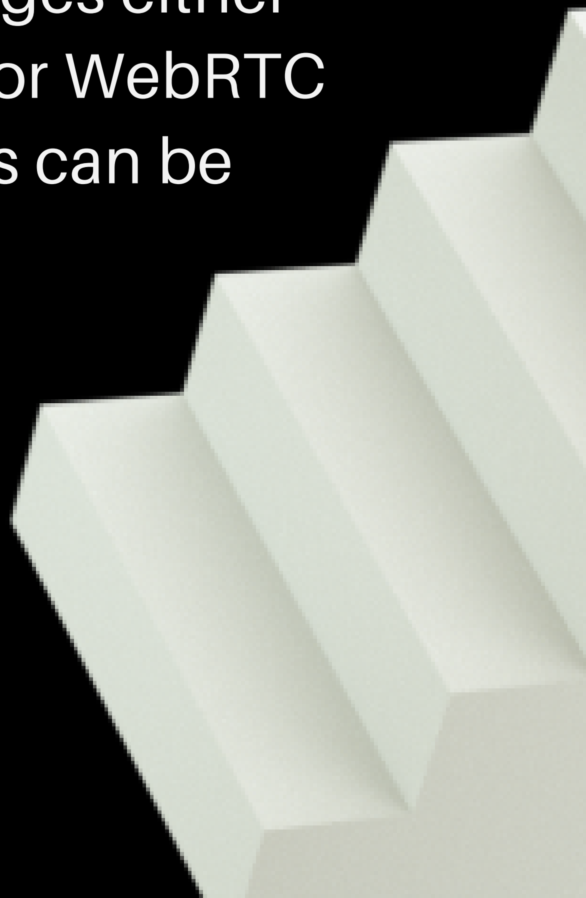
a single videobridge can handle 1000 video streams at 550 Megabits 20% CPU which is really good. Another good side of this is that the quality of the videos is expected to be great as the videobridge does not do any mixing and simply relays the streams.

The downside of this would be on the client-side as they would experience higher CPU and bandwidth usage as the number of participants grows.

Components

Colibri is a protocol and feature set within the Jitsi video conferencing platform that facilitates real-time monitoring, statistics collection, interactive features, and administrative controls.
Details like ongoing conferences, details of the individual conferences, and other statistics like
RTP loss, bitrate download, audio channels, etc.

- WebRTC interface - Jitsi Videobridge supports WebRTC through both UDP and TCP
- Java Base - The core of Jitsi Videobridge is in Java. It handles all the media relays, TURN functionality, Image processing, WebRTC implementation
- XMPP Modules and Colibri - Jitsi Videobridge is controlled by XMPP and its extension COLIBRI. Signaling, Communication with the Client is handled by XMPP. For client-to-bridge messages either WebSockets or WebRTC DataChannels can be used.



Steps for JVB communication

Step 1

Signaling

- When a participant joins a Jitsi Meet conference, the web browser or mobile app they are using communicates with the Jitsi Meet frontend, which, in turn, communicates with the Prosody XMPP server.
- The Prosody server manages user authentication and conference room creation/joining.
- To establish media channels, a process called signaling is used. The signaling process involves the exchange of information between the participant's client and the Jitsi VideoBridge through the Prosody XMPP server.

Step 2:

SDP Offer and Answer

- SDP (Session Description Protocol) is used for describing media communication sessions, including codecs, media types, and network information.
- When a participant joins a conference, their client creates an SDP offer, which contains details about the participant's audio and video capabilities, such as supported codecs, resolutions, and network information.
- The client sends this SDP offer to the Jitsi VideoBridge through the Prosody server.

Step 3:

Jitsi VideoBridge Processing:

- Jitsi VideoBridge receives the SDP offer from the participant's client. It processes this offer to understand the participant's capabilities and preferences for audio and video.
- The VideoBridge then generates an SDP answer, which includes information about the media channels it will use for the participant.
- The SDP answer may include details such as the selected codec, IP addresses, and ports for media transmission.

Step 4

Response to the Participant

- Jitsi VideoBridge sends the SDP answer back to the participant's client through the Prosody server.
- The participant's client processes the SDP answer and prepares to establish media channels accordingly.

Step 5

ICE Candidate Exchange:

- In addition to SDP exchange, Jitsi VideoBridge and the participant's client engage in ICE (Interactive Connectivity Establishment) candidate exchange.
- ICE helps in establishing direct peer-to-peer connections when possible, even if participants are behind NAT (Network Address Translation) or firewalls.
- Candidates are network addresses and ports where media can be sent. These candidates are exchanged between participants to help establish direct connections.

Step 6

Establishing Media Channels

- Based on the SDP offer and answer exchange and ICE candidate information, the participant's client and Jitsi VideoBridge establish media channels for audio and video transmission.
- These media channels use the negotiated codecs and network configurations to ensure real-time communication.

Step 7

Secure Communication

Jitsi VideoBridge supports secure communication through encryption. It can encrypt media streams using protocols like Secure Real-time Transport Protocol (SRTP) and transport encryption using Datagram Transport Layer Security (DTLS).

Ports

- Web Ports (HTTP/HTTPS):
  - TCP Port 80: This is the default HTTP port used for unencrypted web traffic. It can be used for initial HTTP communication when participants access the Jitsi Meet web interface.
  - TCP Port 443: This is the default HTTPS port used for secure web traffic. It is used for encrypted web communication when HTTPS is enabled.
- WebSocket Ports:
  - TCP Port 443: WebSocket traffic is often tunneled through the HTTPS port (443) for secure communication. WebSocket is used for signaling and real-time updates in Jitsi Meet.
- Media Ports:
  - UDP and TCP Ports 10000-20000: Jitsi VideoBridge uses a range of UDP and TCP ports for media transmission. These ports are primarily used for sending audio and video streams between participants' clients and the Jitsi VideoBridge server.
  - The specific ports within this range can be dynamically allocated as needed, depending on the number of participants and the configuration.

- ICE Ports (Interactive Connectivity Establishment):
- UDP Ports 3478 and 5349: These ports are used for the ICE protocol, which helps establish peer-to-peer connections between participants. Port 3478 is often used for non-encrypted traffic, while port 5349 is used for encrypted traffic.
- DTLS Ports (Datagram Transport Layer Security):
- UDP Ports 10000-20000: These are the same ports used for media transmission. DTLS is used to secure the transport of media streams.
- TURN Server Ports (Traversal Using Relays around NAT):
- UDP and TCP Ports 3478 and 5349: If a TURN server is used for relaying media streams (in cases where direct peer-to-peer connections cannot be established), these ports are used for TURN traffic.

How to connect to prosody server?

Choose an XMPP Client: You need an XMPP client to connect to the Prosody server. There are various XMPP clients available for different platforms (Windows, macOS, Linux, Android, iOS). Some popular XMPP clients include Jitsi Meet

Install and Configure the XMPP Client: Download and install the XMPP client of your choice on your device. Once installed, you'll need to configure it to connect to the Prosody server.

Provide Server Information:

- Server Address: You'll need to specify the domain or IP address of the Prosody server you want to connect to. This is typically provided by your organization or the service you are using.
- Username: Enter your XMPP username, which is usually in the format of "username@domain.com" or simply "username" depending on the server configuration.
- Password: Provide the password associated with your XMPP account on the Prosody server.

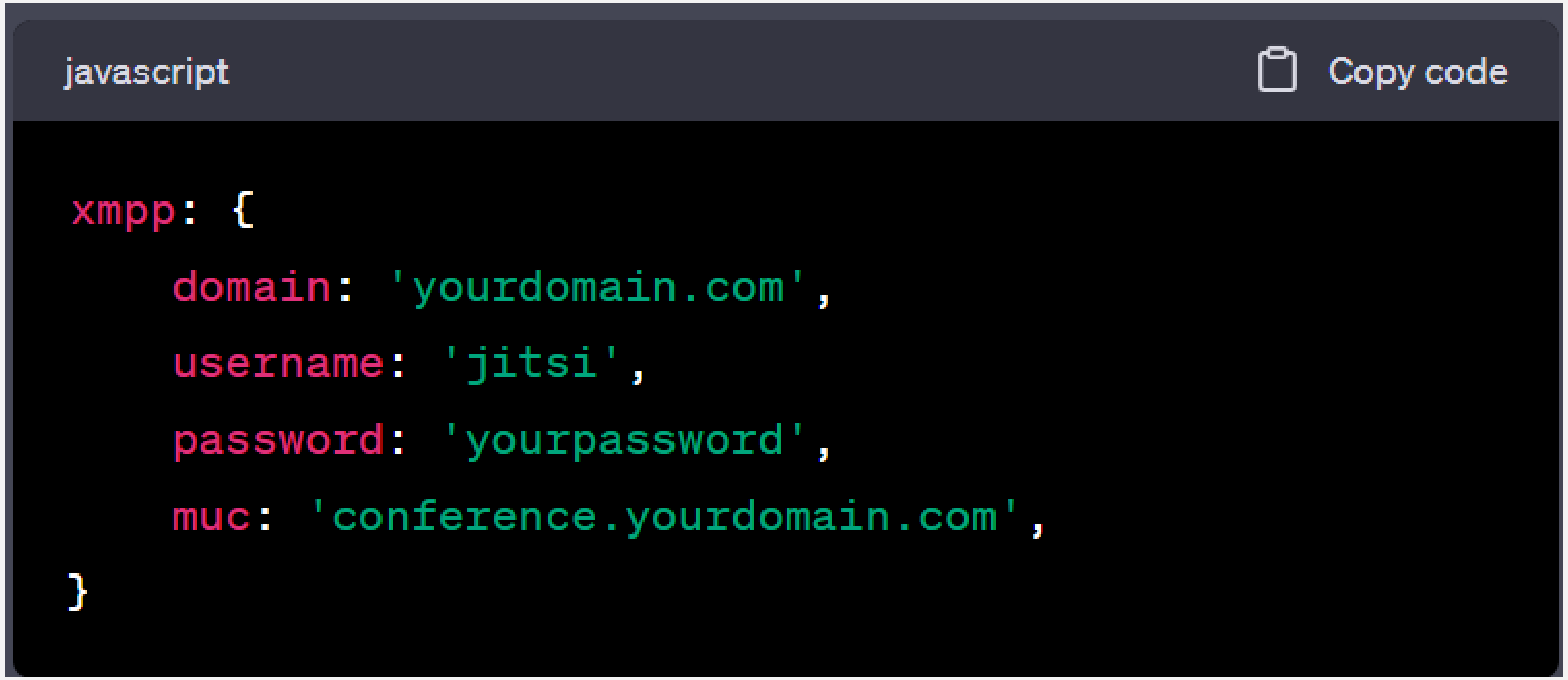
Advanced Settings: Depending on your client, you might need to configure advanced settings such as the port, security settings (SSL/TLS), and other preferences. In most cases, the client will have default settings that work for common XMPP servers like Prosody.

1. Standard Unencrypted Connection (Insecure):
   - Client-to-Server (C2S): Port 5222
   - Server-to-Server (S2S): Port 5269
2. Secure Connection (Encrypted with TLS/SSL):
   - Client-to-Server (C2S): Port 5223
   - Server-to-Server (S2S): Port 5270

How to connect to prosody server?

Connect: Once you have entered the necessary information, click the "Connect" or "Sign In" button in your XMPP client. The client will attempt to establish a connection with the Prosody server using the provided credentials.

Chat and Communicate: After a successful connection, you can start using the XMPP client to send and receive messages, join chat rooms (MUC - Multi-User Chat), and perform other XMPP-based communication activities.



```javascript
xmpp: {
    domain: 'yourdomain.com',
    username: 'jitsi',
    password: 'yourpassword',
    muc: 'conference.yourdomain.com',
}
```

Features

Selective Forwarding:

- JVB intelligently selects which media streams to forward to each participant based on their network conditions and preferences.
- It forwards only the necessary media streams to each participant, reducing bandwidth consumption. For example, if Participant A is only interested in seeing Participant B's video, JVB will forward Participant B's video stream to Participant A.

Dynamic Adaptation:

- JVB can dynamically adapt the quality and resolution of the forwarded media streams based on network conditions. For instance, if a participant's network experiences degradation, JVB can reduce the resolution or quality of the forwarded video to ensure a smoother experience.

Simulcast Support:

- JVB supports simulcast, which means it can receive multiple versions of the same video stream at different resolutions and qualities from a participant. It then forwards the most appropriate version to each recipient based on their network and device capabilities.

Thank You!