WEBRTC OVERVIEW

GENERAL INTRODUCTION

2023-04-18

WEBRTC

WEBRTC OUTLINE

Client-side:

Standard JS API

Server-side:

Signal channel
ICE Servers (STUN/TURN)
Media handling (SFU/MCU) (optional)

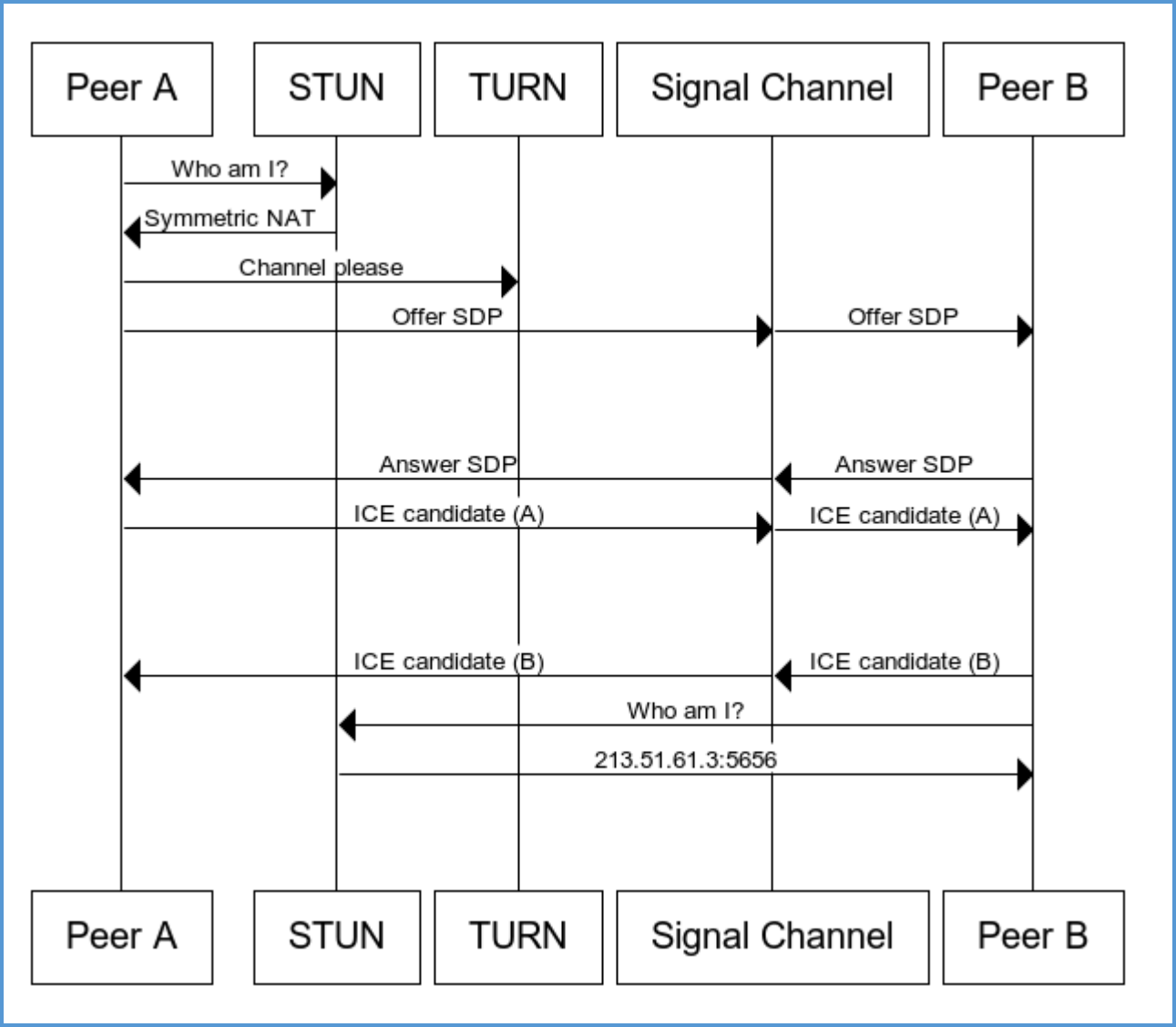
WEBRTC ARCHITECTURE OVERVIEW

WEBRTC DEEPER DIVE

WEBRTC ICE CANDIDATES

1. **Host** Candidate: Direct interface
2. **Server-Reflexive** Candidate: NAT binding (STUN)
3. **Peer-Reflexive** Candidate: NAT binding (trickle)
4. **Relayed** Candidate: Relay server (TURN)

WEBRTC SEQUENCE DIAGRAM



WEBRTC JS API

MediaStream

- const stream = getUserMedia(constraints);

RTCPeerConnection

- const pc = new RTCPeerConnection(ICEServers);

A: pc.createOffer(); pc.setLocalDescription(offer); **=signal=>** pc.setRemoteDescription(offer);
B: pc.createAnswer(); pc.setLocalDescription(answer); **=signal=>** pc.setRemoteDescription(answer);
pc.addEventListener('icecandidate'); **=signal=>** pc.addIceCandidate(remoteCandidate);

stream.getTracks(); pc.addTrack(track); || pc.addEventListener('track');

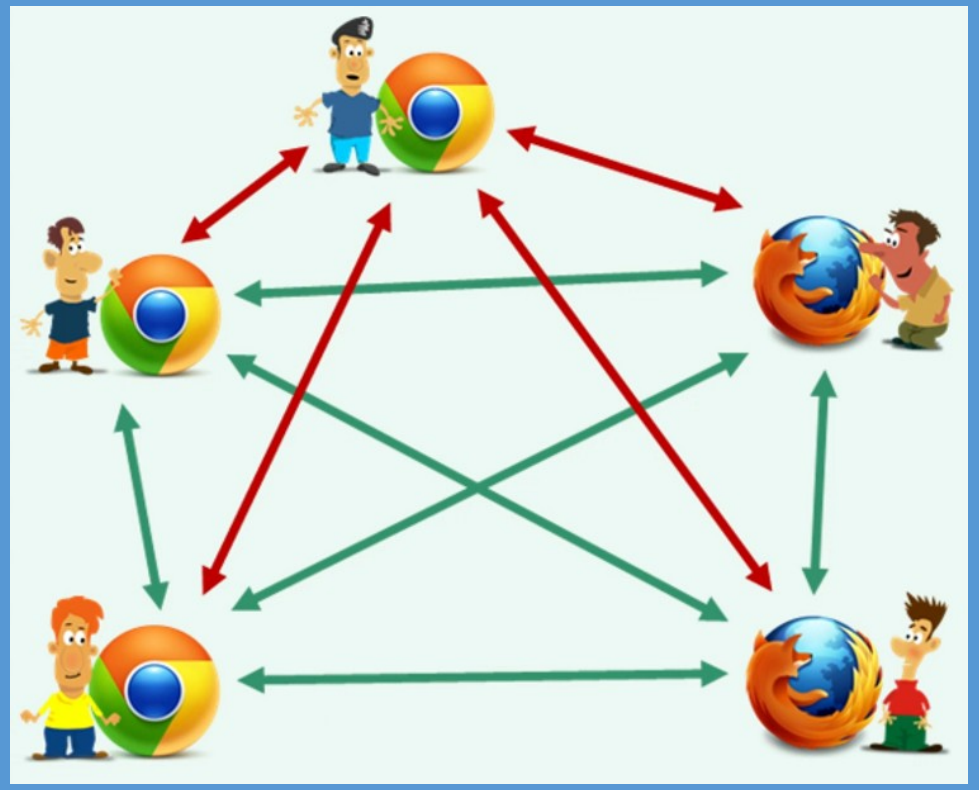
RTCDataChannel

- const channel = pc.createDataChannel(); || pc.ondatachannel

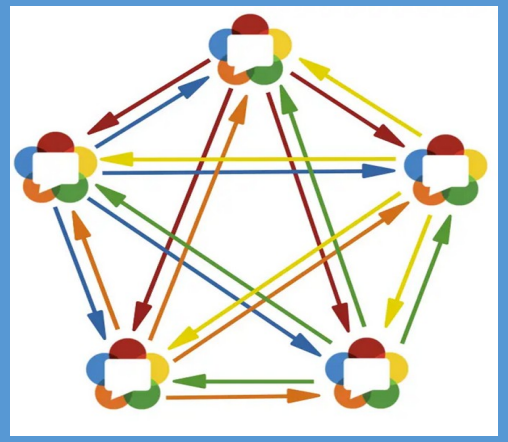
channel.onmessage

WEBRTC TOPOLOGIES

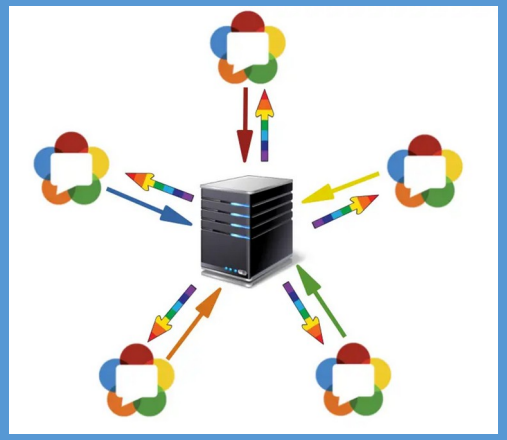
WEBRTC SCALABILITY?



WEBRTC MEDIA SERVERS



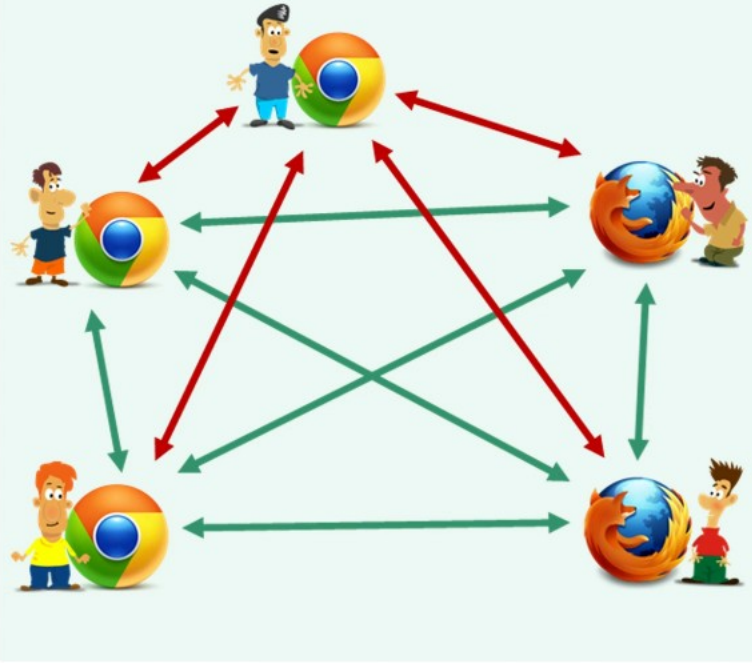
P2P MESH



MCU

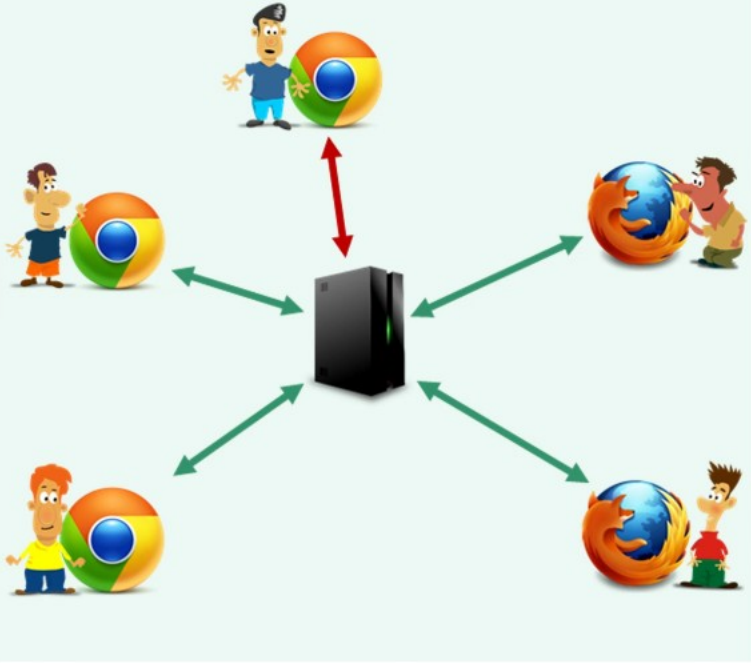
SFU

Connections:	**4 | 10**
Uplink:	**4** mbps
Downlink:	**4** mbps
Total:	**20** mbps



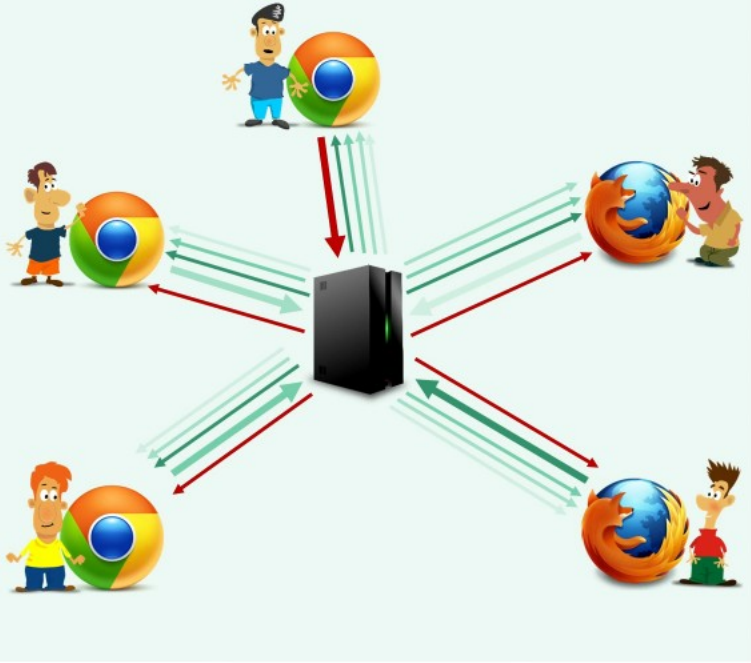
P2P MESH

Connections:	**1 | 5**
Uplink:	**1** mbps
Downlink:	**1** mbps
Total:	**10** mbps



MCU

Connections:	**5 | 25**
Uplink:	**1** mbps
Downlink:	**4** mbps
Total:	**25** mbps



SFU