WebRTC Lip-Sync Debugging Checklist

Fora Soft Learn

Two timestamps, one SFU problem, three engines - how to find a group-call sync bug.

The mechanism: two timestamps + a CNAME

RTP timestamp	On every packet. 48 kHz for Opus audio, 90 kHz for video. Measures intervals in one stream only.
RTCP Sender Report	Every few seconds. Pairs an NTP wall-clock time with the RTP timestamp - the cross-stream anchor.
CNAME	Same value for one speaker's audio and video, so the receiver knows they belong together.

Why the SFU breaks it

1:1 call	Sender's own Sender Reports reach the receiver untouched. Sync is solved for free.
SFU group call	SFU terminates RTCP and regenerates Sender Reports on its own clock. Danger point.
Absolute Capture Time	Header extension stamping original capture NTP time on packets - survives the SFU if forwarded.

mediasoup vs Pion vs LiveKit

mediasoup	Rewrites RTP timestamps from the SR NTP relationship on stream switch. App must wire A/V identically.
Pion	Correct SR primitives + report interceptor. You own all the sync logic; nothing is automatic.
LiveKit	Server-side sync from track SR NTP timestamps out of the box. Edge cases (RED, egress) drop an SR.

Five checks when sync drifts

- [ ] Split media paths: audio on one server, video on another = no shared anchor. Go all-in SFU or all-in MCU.
- [ ] Missing SR for one stream: check chrome://webrtc-internals shows Sender Reports for BOTH audio and video.
- [ ] Capture-time extension dropped: verify Absolute Capture Time is negotiated AND forwarded on every hop.
- [ ] Blaming the network: steady one-directional drift = clock or timestamp problem, never jitter.
- [ ] Tell-tale: fine 1:1, broken in group calls = the SFU's timing, not the codec or the connection.