

# NADA Implementation in Mozilla Browser

draft-ietf-rmcat-nada-09

Xiaoqing Zhu, Rong Pan, Michael A. Ramalho, Sergio Mena,

Paul E. Jones, Jiantao Fu, and Stefano D'Aronco

Nov 2018 | IETF 103 | Bangkok, Thailand

# Code Changes in Mozilla Repo

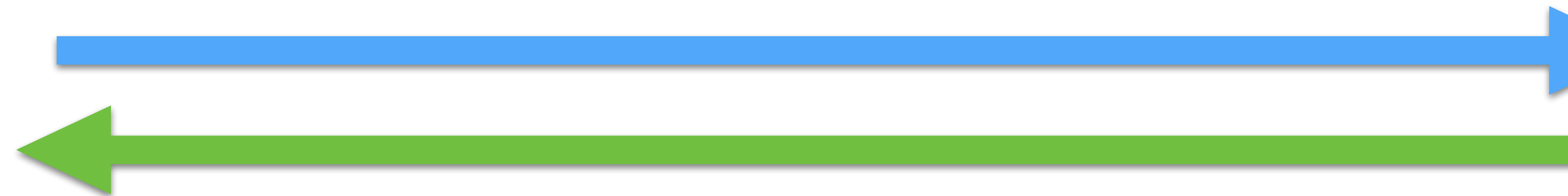
- Modified sender behavior:
  - Changes in [~/media/webrtc/trunk/webrtc/modules/bitrate\\_controller](#)
  - Replaces default [send\\_bandwidth\\_estimation.cc/h](#) modules with [nada\\_bandwidth\\_estimation.cc/h](#)
  - Using relative RTT as the congestion signal, ignores packet loss, updates interval ~1 sec
- Unmodified receiver behavior:
- Added logging via existing WebRTC logging framework

# Real-World Test: Setup

Client A



Bi-directional audiovisual calls via appr.tc



Client B



Firefox Nightly w/ modifications

- NADA bandwidth estimation
- maximum sending rate at 4Mbps
- Default video height: 720p
- Logging stats of outgoing flow in the NADA module

Unmodified  
Chrome browser

# Real-Life Test: Local Session

Client A:

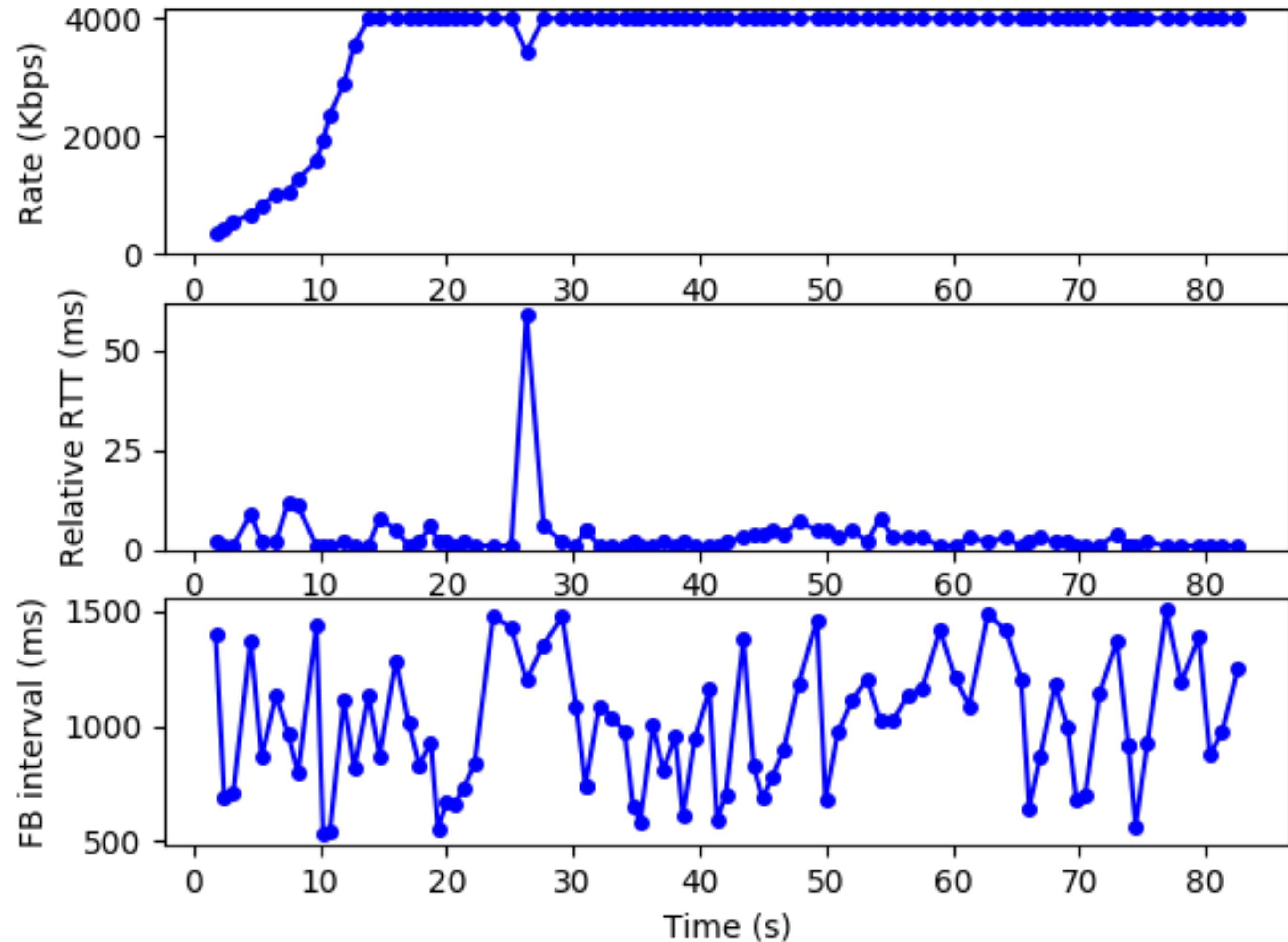
- Location: Austin, Texas
- Internet connection: home WiFi

Client B:

- Location: Austin, Texas
- Internet connection: home WiFi

\* Both clients connect to the Internet via the same home WiFi AP

\* Base RTT: ~1ms



# Real-Life Test: Remote Session within US

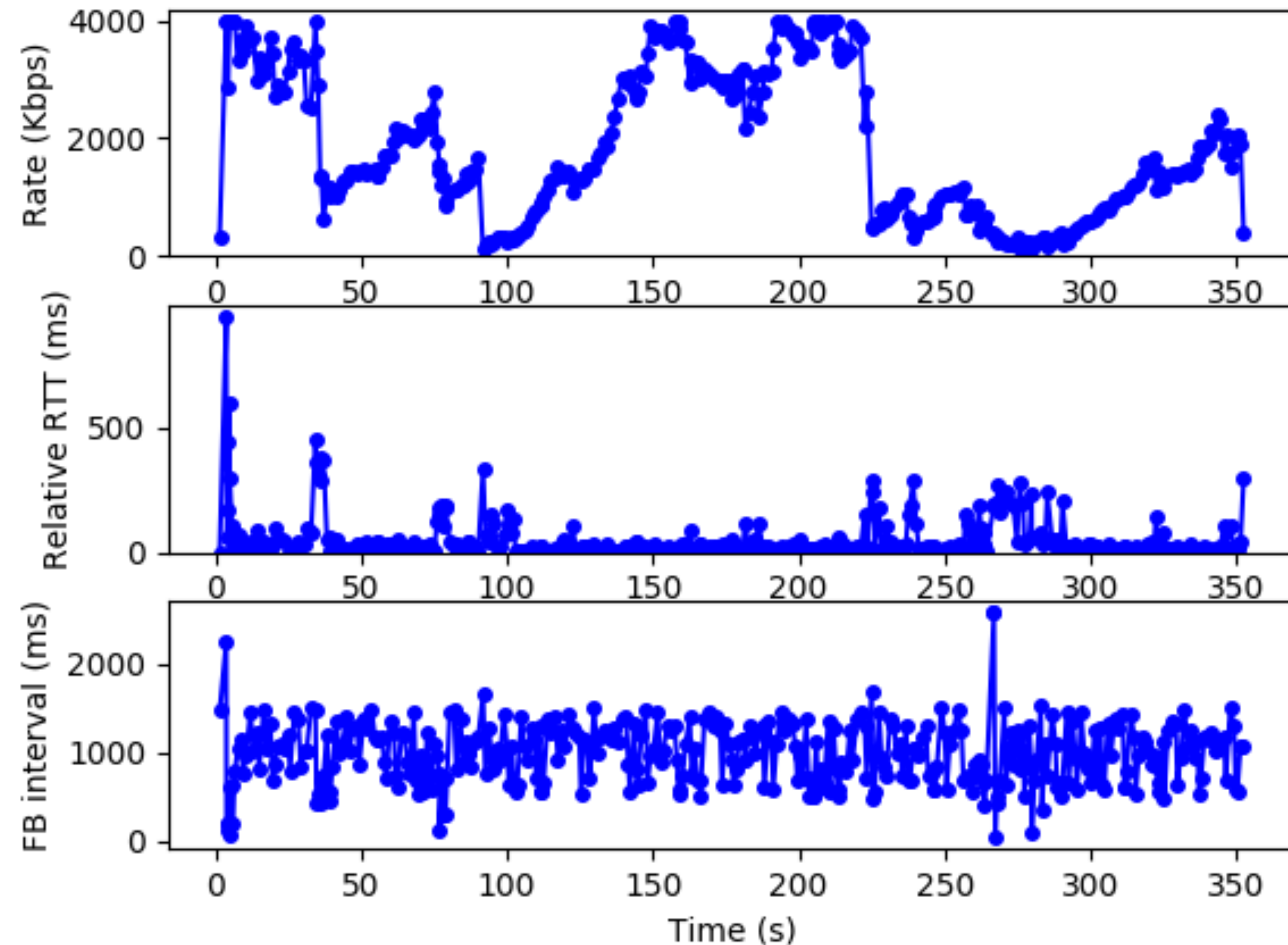
Client A:

- Location: Austin, Texas
- Internet connection: enterprise office

Client B:

- Location: San Jose, California
- Internet connection: home WiFi

Base RTT: ~160ms



# Real-Life Test: Remote Session across Atlantic

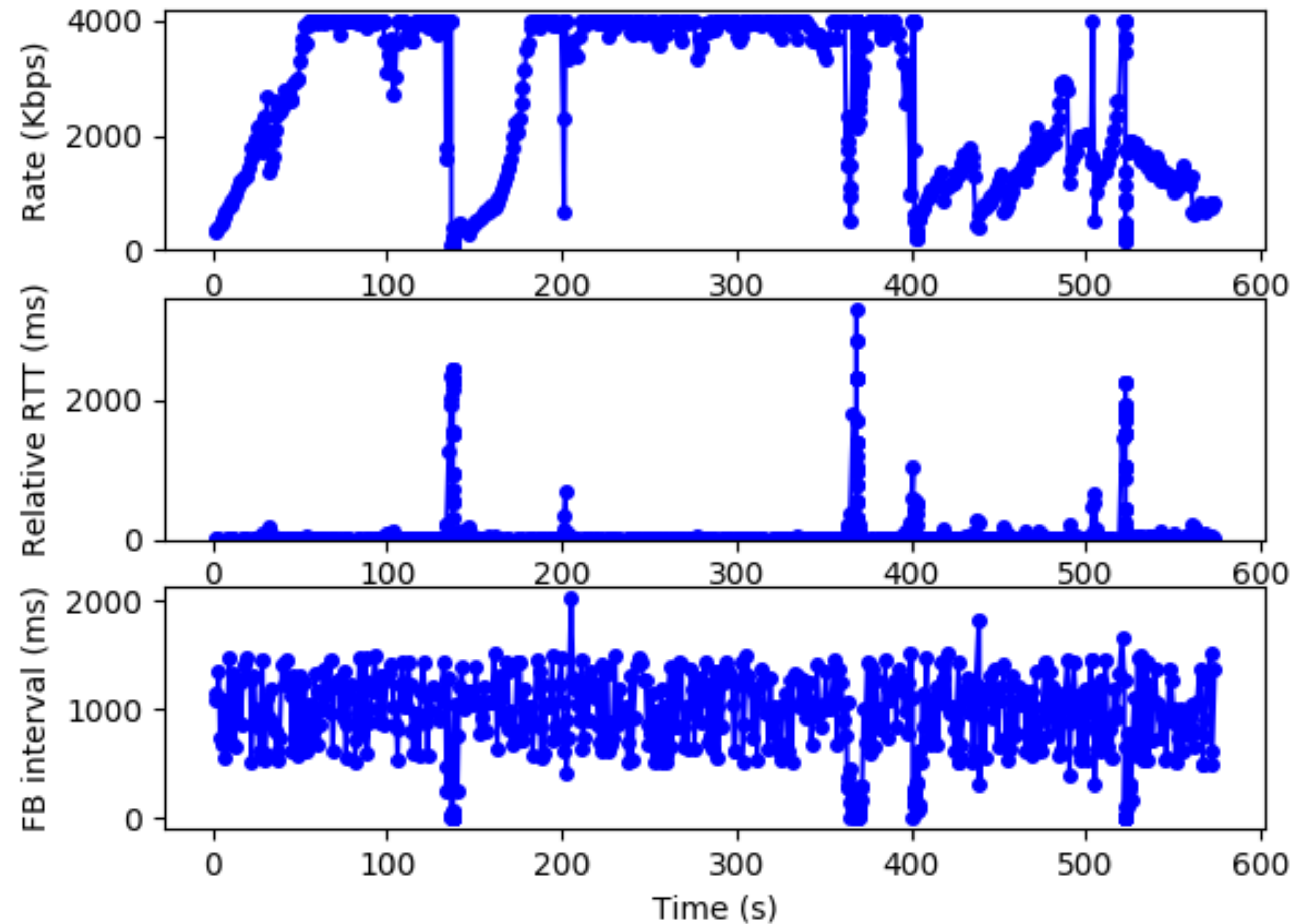
Client A:

- Location: Austin, Texas, USA
- Internet connection: enterprise office

Client B:

- Location: Lausanne, Switzerland
- Internet connection: home WiFi

Base RTT: ~235ms



# Summary of Observations

- Over ideal uncontested scenarios:
  - Ramp up to maximum rate within 15ms
  - Sending rate dips briefly due to occasional RTT spikes (~50 ms) but recovers quickly
- Over remote connections:
  - Reacts to RTT spikes over 500ms by dropping rate to minimum and recovers more slowly (~100 s)
  - Otherwise sustains maximum streaming rate with random fluctuations
- Overall, fairly robust to high RTTs and high FB intervals

# Next Steps

- Integrate with new FB format so that congestion control signal is based on relative one-way delay values, as described in draft
- Experiment with the impact of different FB intervals
- Add logging on loss statistics and experiment over lossy links