

WebRTC isn't just for (video) conference calls

Tim Panton
CTO |pipe|
@steely_glint

WebRTC

- Great for conf calls
- But also other things

Meetecho

The screenshot displays the Meetecho interface for an IETF RTCWEB session. At the top, the title bar reads "Meetecho IETF RTCWEB session" with a red mute button, a lock icon, and a refresh icon. Below the title bar, the user "Simon Romano" is identified as the "Teacher" with icons for mute, video, and microphone. The "Participants" section shows 32 attendees. The "Chat" window is active, displaying a conversation about API limits and storage. The main video feed shows a man in a striped shirt and cap speaking at a microphone. A smaller video feed in the top right shows another man standing in a room.

Meetecho IETF RTCWEB session

Simon Romano
Teacher

Participants 32

Chat

favor of send[plot] simply working unless you run out of storage.

hda

jesup: would you be happy if the limit was hidden from the API and just caused early failure on send? [10:28]

ted.h

ekr at the mic [10:29]

Hadriel Kaplan

Randell: still want me to say that? I'm at the mic ready to if so. [10:29]

jesup

Hadriel Kaplan: yes. And a Web Streams API would help once that's defined [10:30]

[10:30] - pm joined the room

[10:31] - tobcast joined the room

[10:32] - tobcast has left the room

Send Message to Chatroom

Zero install (for users)
Semantics of IETF meeting
Complex media priorities

Miku
Privacy
Nat traversal
Bandwidth costs



Podcast Recorder

github:
Pipe/PodCall

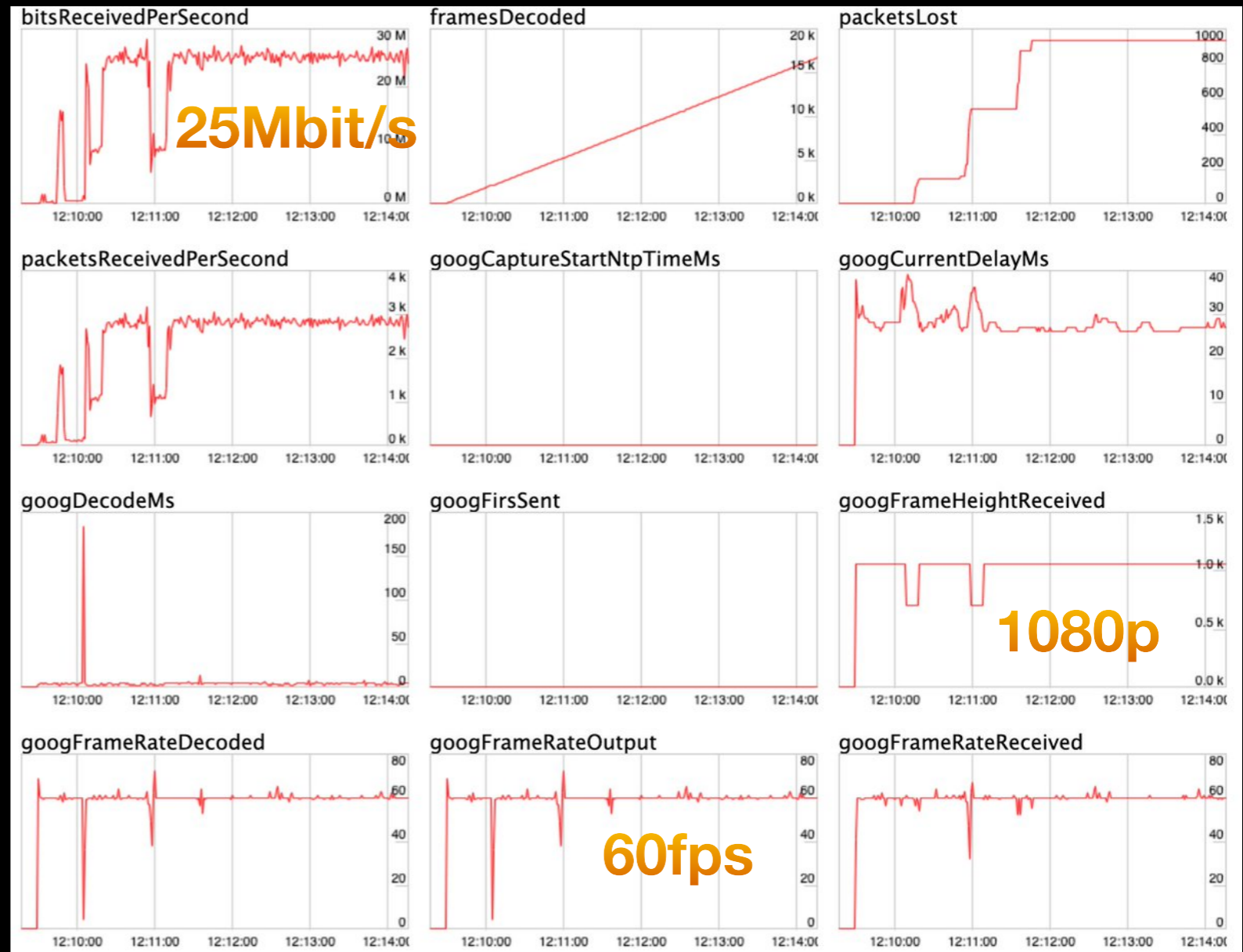
Audio only
High quality
Mobile first



<https://distributedfutu.re>

Stadia

Low latency
High bitrate
Playable games
In Chrome
On an old macbook



Remote access

Nat traversal
Zero install
Security

Demo

What have we learned?

- Not just video conferences
- Not all endpoints are laptop browsers
- W3C webRTC API (SDP) isn't ideal for all use cases
- Not all endpoints run libwebrtc

Available rtcweb impl

- Pion (GO)
- |pipe| java
- Meetecho C
- Aiortc - python
- Mediasoup javascript
- Frozen mountain C#
- Gstreamer

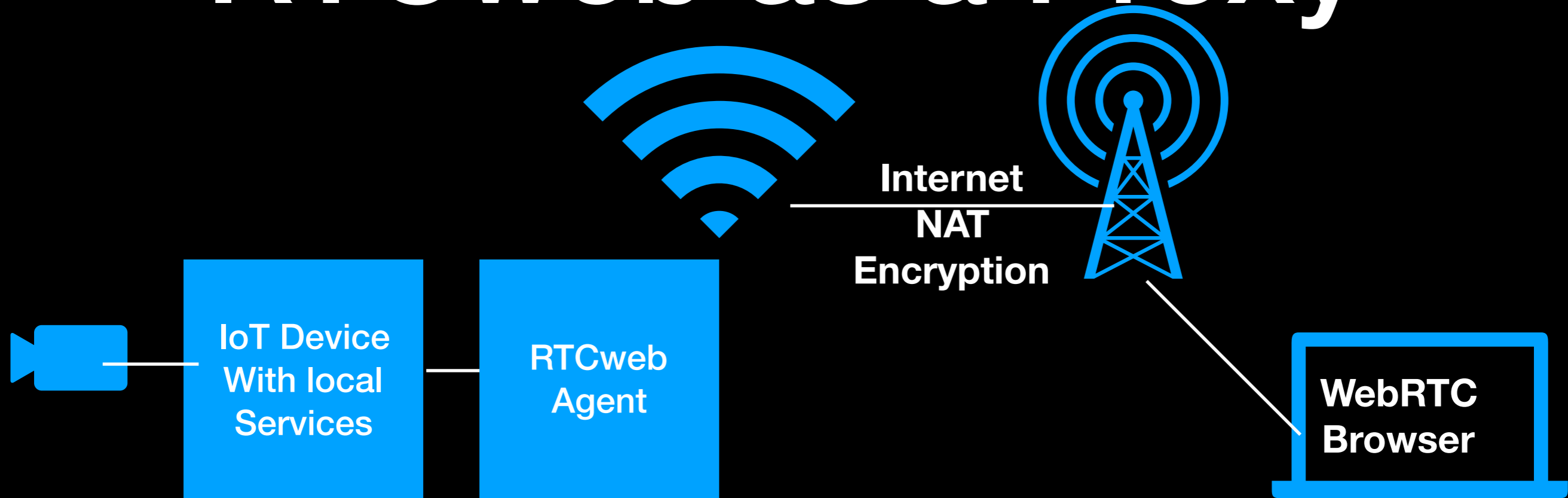
**Benefits of
open standards**

Interop Hackathon at IETF 107

What is a good API?

- I don't know.
- W3C doesn't know x 2
- Google doesn't know
- So... Let's try something else....

RTCweb as a Proxy



- Existing services
- Need webRTC magic dust (NAT traversal, zero install etc)
- |pipe| agent as a configurable proxy

Obvious one

- RTP (RTSP) -> DTLS SRTP - video/audio. Etc.
- (Only complexity is encoder control)

Easy one

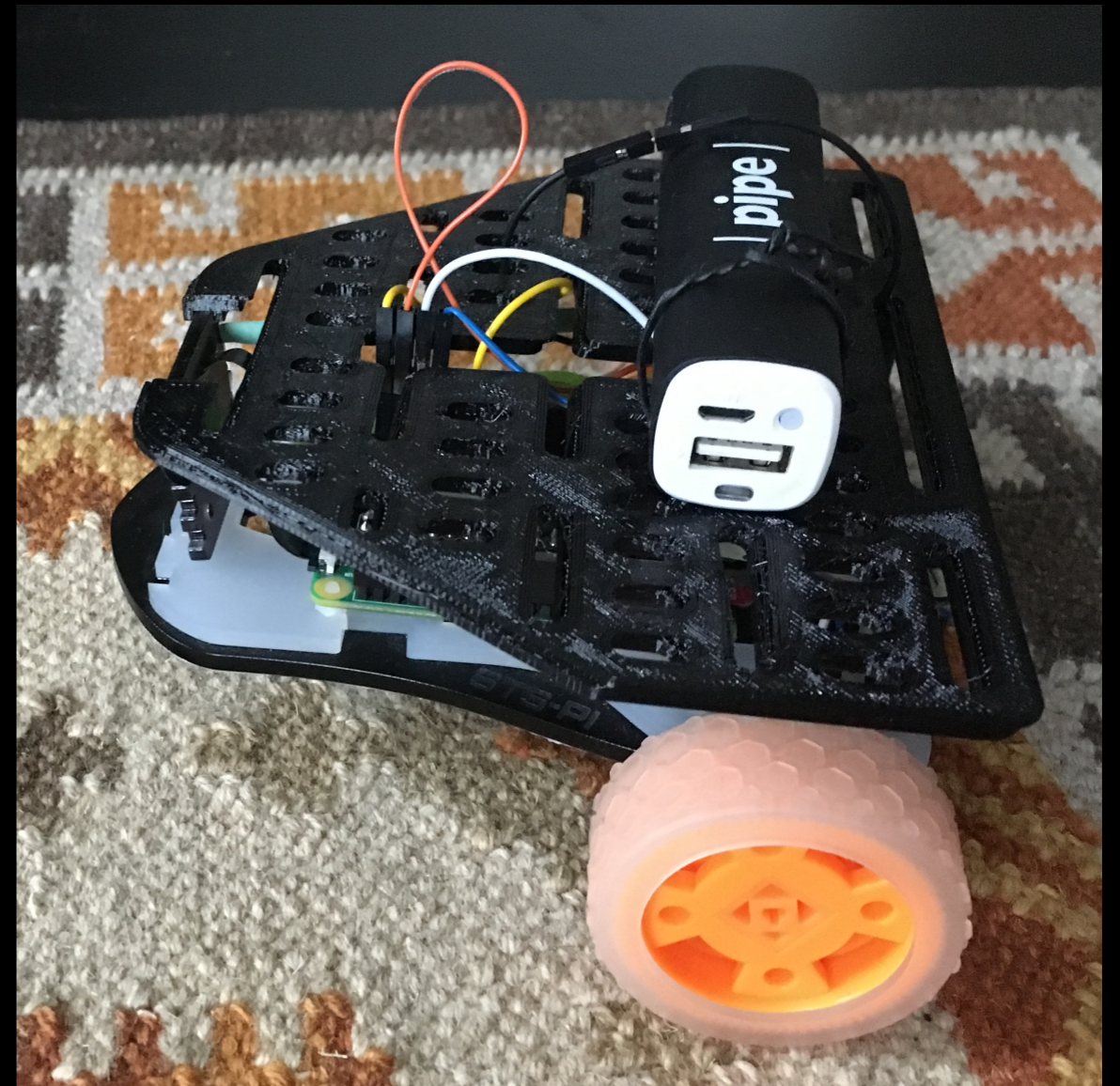
- Web sockets
- Data channel API in browser looks like websocket so, just return one and duct typing wins ;-)
- Proxy data over named rtcweb datachannel to agent, which opens actual local web socket.

The Magic one

- Http
- Abuse service worker
- Add iframe (because you cant have a peer connection in a service worker)
- Magic happens

What this lets us do:

- Droid pi zero
- Runs a local web service providing interface
- Websocket for motor control
- Gstreamer RTP for video
- Proxy http+ws+RTP to browser
- Drive the droid



How is that an API?

- Web page creates data channel
- Label contains a URI
- Proxy interprets the URI locally (eg ws://localhost/motor)
- Proxy verifies URI is on the permitted list
- Proxies the websocket traffic (and semantics) over the DC
- Web page hardly notices

Worth Standardising?

- Email me tim@pi.pe
- Tweet me [steely_glint](https://twitter.com/steely_glint)
- Catch me in person....

“ ... as simple as possible, but no simpler.”

–Albert Einstein