The Future of WebRTC

New APIs, WHIP, edge and web 2.5

Tim Panton - tim@pi.pe

Tim Panton

CTO @ pi.pe GmbH

- |pipe| licenses a lightweight cleanroom WebRTC implementation for IoT devices.
- cofounded a web-based telephony company, sold the IPR to Tropo Inc., which was then acquired by Cisco.
- technical cofounder of Westpoint, a web security company acquired by Capita.
- writes |pipe| software
- helps define WebRTC standards at the W3C and IETF
- gave a talk on History of WebRTC at fosdem 2021

photo by Neda Navaee

tim@pi.pe

Structure

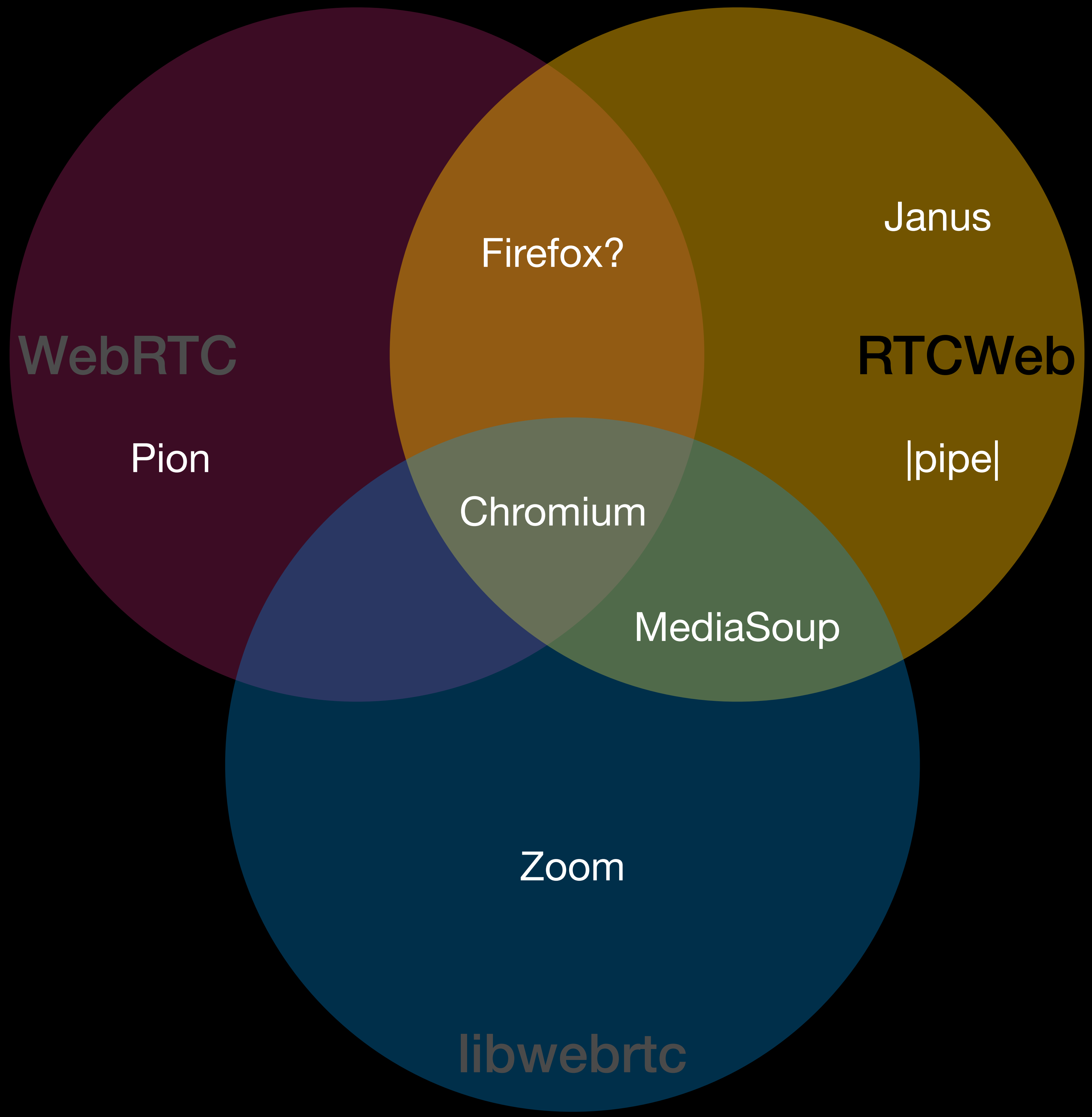
What I'm going to cover in this talk

- Define WebRTC
- Describe new protocols/APIs from the standards bodies
- Guess what they mean for existing WebRTC services
- Outline some of the market forces at play
- Talk about the tools available
- Describe some new usecases for WebRTC
- Describe how WebRTC fits into web3

What is WebRTC

My definition

- Things that implement or use W3C's WebRTC API (Typically in browsers)
- Things that implement or use IETF's RTCWeb wire protocols (Browsers, servers and devices)
- Things derived from Google's opensource libwebrtc (Native apps that have strayed from the first two)



Sources of info

What am I basing these guesses on?

- IETF Standards activity
- W3C Standards activity
- Code (open source libs etc)
- Market dynamics (What’s happening, who is recruiting for what)
- Hunch

IETF

NICER

A way to switch between 4g and wifi during a call

- Continues to check ICE paths
- Checks health of all working paths
- Switches to new path if it is ‘nicer’
- Repeat throughout the call

WISH

A way to upload live video to streaming sites

- Uses a limited subset of WebRTC
- No data channel
- Upload Only
- HTTP(S) based signalling standard
- Multiple implementations using multiple webRTC stacks in multiple languages
  - Janus
  - |pipe|
  - Millicast
  - Galene

Sframe

Keeping secrets from server based snoops

- Double encryption - one layer that the server can’t decode.
- Packet format is SRTP but payload is pre-encrypted
- Leaves enough info so SFU’s can make correct decisions on packets
- Key management/distribution
- Encryption method(s)

RTCWeb
Gone quiet

- Some minor tidy up, but no real changes.

W3C

What is happening at the W3C

(Why does it matter?)

- Reluctance of big browser vendor(s) to ship ‘non-standard’ interfaces
- Partly due to market dominance
- So they like to have a document with some support at W3C (or elsewhere)
- Current emphasis is very much on Video Conference usecases (no suprise)
- Unspoken goal is to ensure Google Meet (in the browser) can compete with native apps like Zoom.
- Not complete list - not official yet - not in final form - don’t use in prod.

Region Capture

Tweak on getDisplayMedia()

- You don’t always want/need to see a whole page in a screenshare.
- Often the page includes irrelevant or confusing controls
  - Speaker notes
  - Next slide button
  - Etc
- This API lets a page indicate which parts should be shared
- And allows the Video Conf app to cropTo() that area.

Capture Handle Action

Driving the page you screen shared

- When presenting an app, the user has to juggle between the slides and the video conference (often folks use 2 computers to make this easier)
- This API allows the video conference app to send Next/Prev actions to the captured app
- It captializes on the fact that the user has asked one to ‘share’ the other.
- The way these 2 web pages should communicate is under discussion
- But at minium there will be a way to identify each other and a way to send actions.

A word about workers.
You'll be seeing more of these…

- The Javascript model is single threaded.
- Increasingly web pages need to use the full multi-core capabilites of the computer
- Workers are separate javascript contexts that are used for cpu intensive or timing sensitive tasks. - Often in WASM.
- Workers you may have seen already:
  - AudioWorklet - manages realtime audio transforms
  - Service Worker - manages cache/offline/fetch data

Media Capture Transform
(AKA breakout box)

- A way to manipulate Video media streams before encoding/encrypting/sending them.
- Used to blur the background in a video call
- Wfh exposes more of our lives than is necessary
  (especially true of children's bedrooms etc)
- Video Stream is transfered to a worker, which can read/change the data in each frame.
- Altered Stream is then processed in the normal way

Encoded Media Transform

End to End Encryption, even if you have an SFU

- A way to access a media stream after it has been encoded but not yet encrypted/sent
- Allows adding a layer of encryption that the server can't read.
- Users don't want to have to trust the security of VMs that run SFUs with their sensitive boardmeeting discussions or virtual dates
  (Isolation between apps on the same metal isn't 100%)
- The encoded media is passed to a worker which encrypts it using keys that have been shared with the other participants, but not with the server.
- Pairs with IETF's SFrame

Transferable DataChannels

Moving heavy data processing off the main thread.

- This allows an app to create a DataChannel and a separate javascript worker to handle it.
  This worker can run on a different core from the rest of the browser

- In practice this means that you can postMessage() a newly opened datachannel to a worker - or indeed to an iframe or page.

WebCodecs

Not strictly webRTC….

- A way to access built in video and audio codecs from javascript
- Used when you want to compress video before sending it

WebTransport

Not strictly webRTC….

- A way to communicate with servers with low latency
- Support for unreliable and unordered communication
- Uses QUIC - which is the new http/3 hotness
- Same servers/firewalls/ports
- The UDP of websockets

A Future VideoConference Service

How you might build one….

What isn't here ?

No WebRTC

No WebRTC?

Future server based group video calls won't use WebRTC

- WebRTC has extra complexity to support
  - NAT/P2P (ICE)
  - E2E (DTLS)
  - Legacy telephony/video interop (SRTP)
- None of which is needed in a future Meet-like service
- WebTransport+WebCodecs simpler (in theory) and easier to customise
- But the transition will take a while

Tools

Browsers

Many browsers all alike.

- Chrome(ium) on Linux/Mac/windows/android/(iOS)
- Firefox on Linux/Mac/windows/android/(iOS)
- Edge on Linux/Mac/windows/android/(iOS)
- Safari on Mac/(iOS)
- Others….

Native apps

Pick a stack that suits you

- https://pi.pe
- https://pion.ly
- https://www.frozenmountain.com
- https://webrtc.googlesource.com/src/
- I wrote a minimal one for WHIP/WISH in a week, so doing your own is possible
- Or use WebRTC in an embedded webview

Servers

If you still need one

- https://freeswitch.org
- https://janus.conf.meetecho.com
- https://mediasoup.org
- But safer to start with one of the many CPAAS providers.

Market

Other Forces at Play

- Increasing legislative push against centralizing data
- Increasing capability of edge devices (smartphones/browsers etc)
- Embedding of AI capabilities on edge devices (at low cost)
  (an iPhone costs less than an nvidia GPU)
- Increasing demand for low-latency
  (games, cars, drones,videocalls, AR/VR, high quality audio)
- Increasing bandwidth available to edge devices
  (Median 5g 200Mbit/s down 29Mbit/s up)

  Why don't we use the capabilities of edge devices more?

NAT

Makes it hard to connect between 2 edge devices
(unless they happen to be behind the same router)

So is there a secure widely deployed way for two edge devices to connect through mulitple NAT layers?

WebRTC

WebRTC at the edge

Playing to it’s strengths

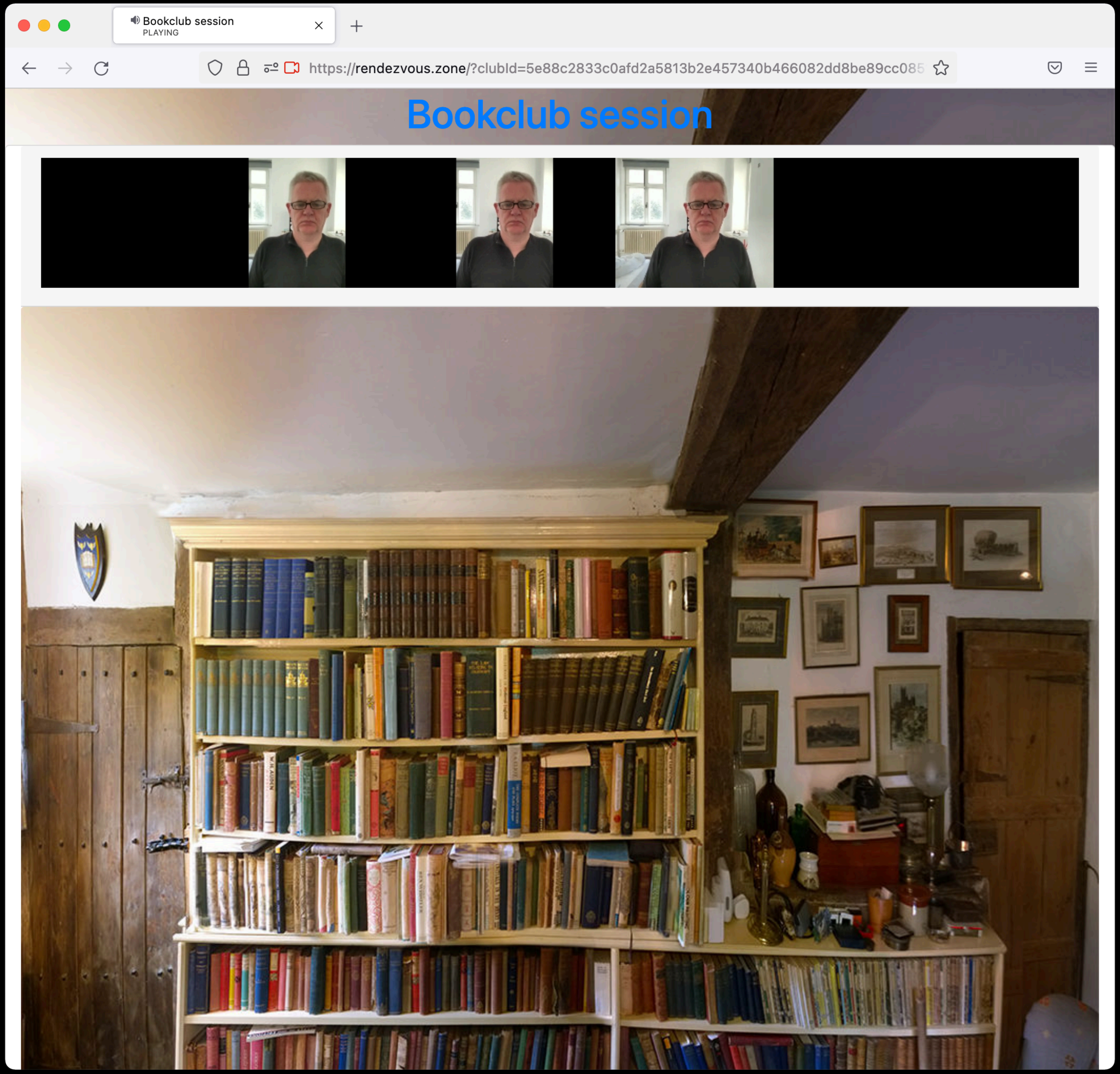
- WebRTC already has:
  - P2P NAT traversal
  - E2E Security
  - Bandwidth estimation
- We now have lightweight webRTC stacks for devices
- Endpoints have more bandwidth and capabilities than ever
- So let’s see what we can build!

New Example Apps on the edge

Testing the theory.

A Video Conference in the Browser

https://rendezvous.zone



- Provide a cosy space for bookclub meetings
- Support laugh/grunt/tuts without video flicker
- No server costs
- Own bandwidth (0.5Mbit/s/user)

(open source)

tim@pi.pe

A secure baby monitor

- Hardware WebRTC device
- Realtime
  - Audio/Video/Data
  - Stays local if possible (P2P)
  - Encrypted E2E
- Key features work off-line (e.g. cloud or ISP outage) because ICE finds local path and auth is local

tim@pi.pe

Modern Webcam

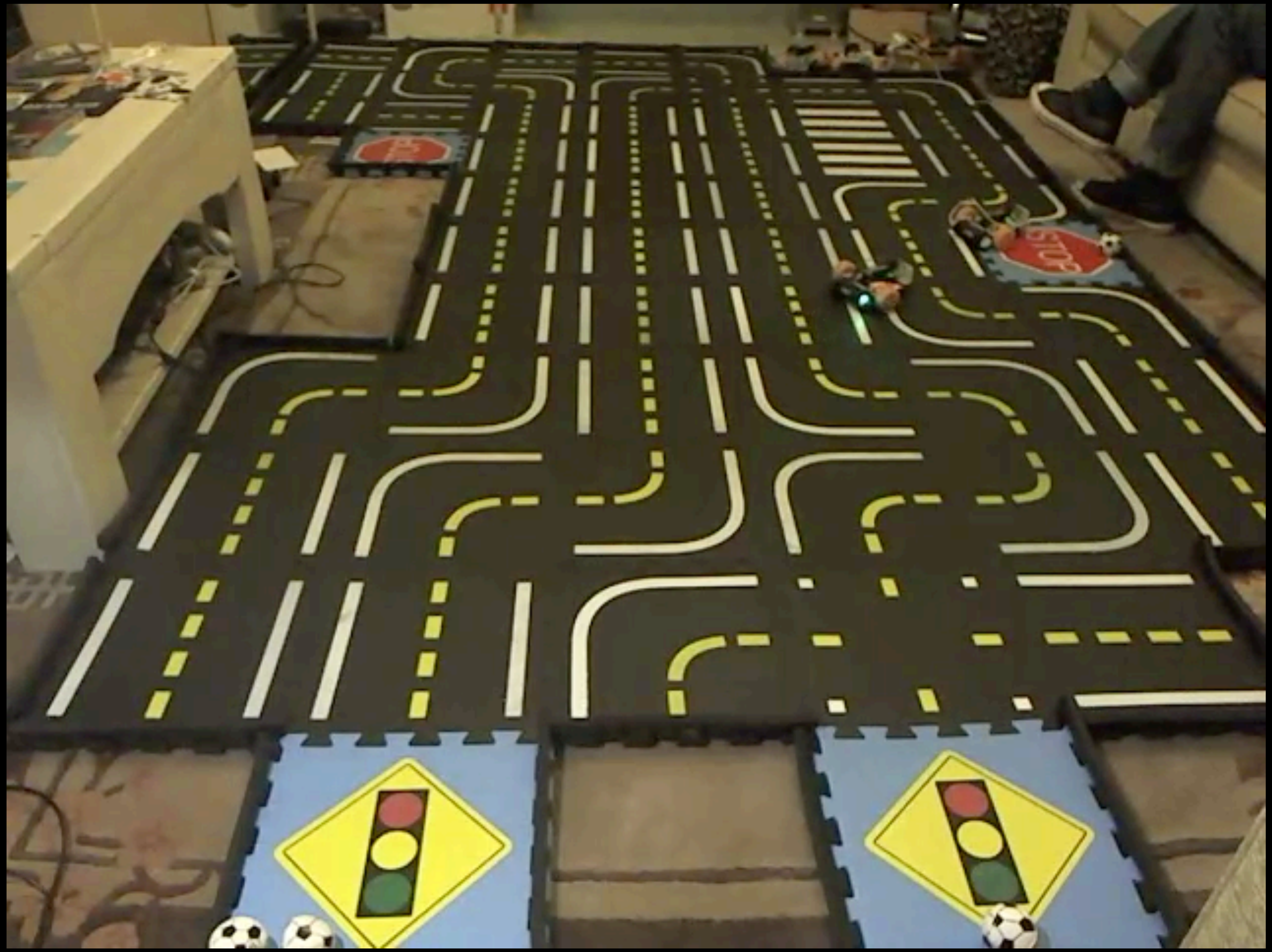
Multiple simultaneous viewers

- Live 1080p @30 H264 video
- To 20+ browser users
- From a modified IoT camera (not the stock firmware)
- Using vDSL
- No cloud processing (on device SFU)
- No central permissions (edge-to-edge security)
- Expect to see this in webcams and perhaps routers

Remote control

Fun with droids

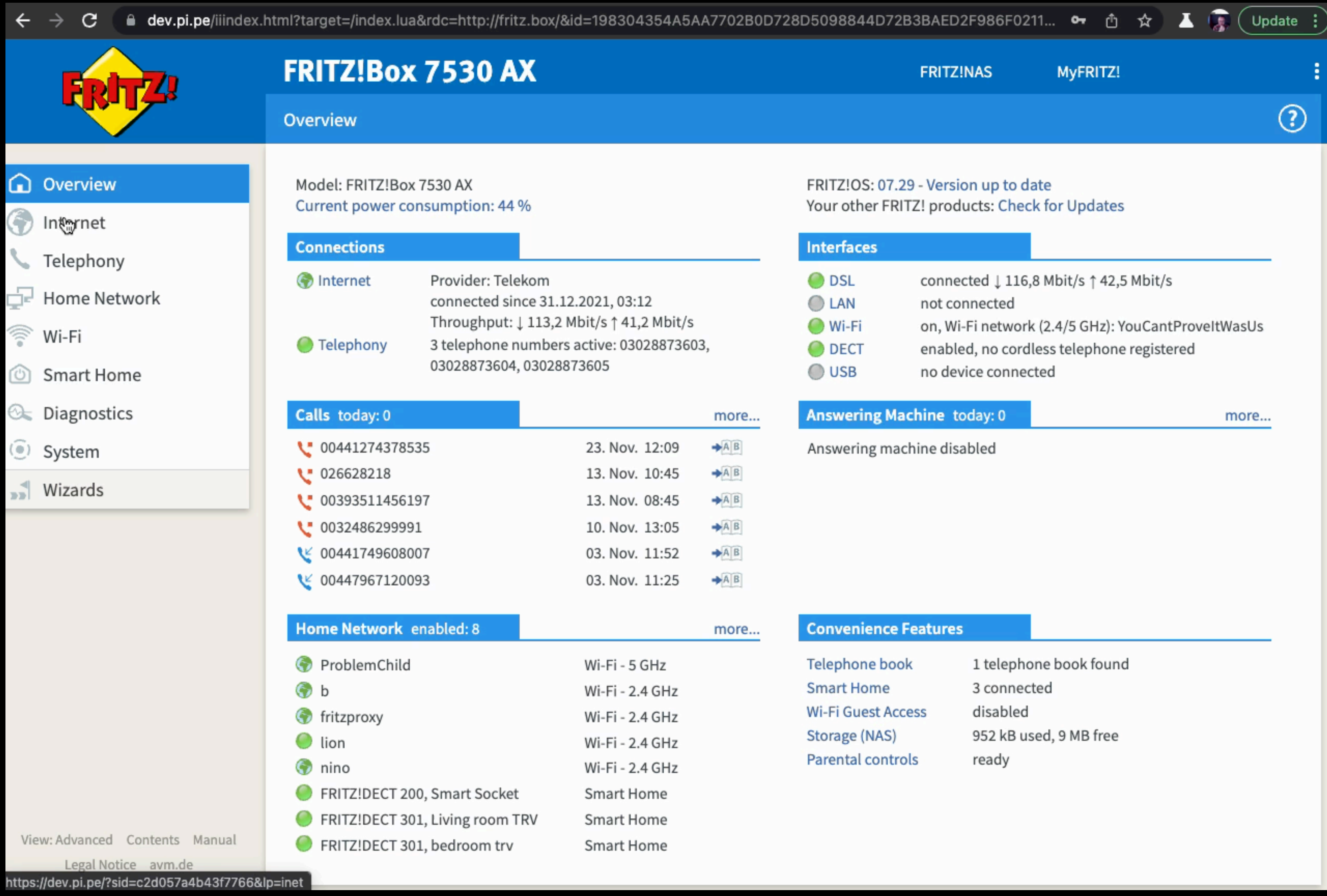
- This used |pipe|'s lightweight Arm friendly stack
- Remote users view and control up to 6 droids
- Shared audio space
- Runs on raspi
- Zero install
(web interface loaded over webRTC datachannels)



Remote web server

On Domestic DSL + NAT

- Note the domain is dev.pi.pe
- But the pages are proxied via
  - WebRTC datachannel
  - Service worker
  - Iframe
- Invisible to the user
- Invisible to unauthorized



tim@pi.pe

Web 3

Does that meet a lot of the web3 aims?

Some, but not all

- Web 3.0 is an open, ~~trustless~~ and permissionless network that enables a future where distributed users and machines are able to interact with data, ~~value~~ and other counterparties via a substrate of peer-to-peer network~~s~~ without the need for third parties.
- We can do most of that with WebRTC today
- We need trust on first use, it is a single network and we don't store value.
- Think of it as web2.5 ?

tim@pi.pe

What about the crypto?
WebRTC has that covered too

- PeerConnection.generateCertificate() creates a self-signed X509 cert
- IndexDB can store and retreive it
- PeerConnection can use it
- It is carried in the DTLS handshake
- Can be checked on the remote side to see if it is ‘known’
- Used as a locally stored identity

BONUS ! — Eco friendly

WebRTC (aka web2.5) ’s big win is energy effeicency

- WebRTC doesn’t need proof of work
- Doesn’t burn electricity like it was a country
- Doesn’t need forced air cooling
- Runs on your android phone, not your mining rig
- Is already in 2+bn browsers

Summary

The best is yet to come…

- Short term
    - there will be improvements in slide sharing
    - Workers will allow sites to take advantage of multicore devices
- Long Term
    - Large video call services will move to webTransport
    - Smaller more niche services will move to the edge
    - WebRTC will run on lots of IoT hardware (especially cameras)
    - WebRTC will keep adding features
- WebRTC may play a large part in the final shape of whatever web3 becomes

Questions ?

Or contact me:

- tim@pi.pe
- https://rendezvous.berlin
- https://pi.pe

  I'll put these slides up, with links to the relevant standards docs etc.

- Twitter: @pipe_iot
- GitHub: https://github.com/pipe

Links

To explainers and docs

- https://github.com/alvestrand/nicer-spec
- https://github.com/w3c/webtransport/blob/main/explainer.md
- https://github.com/w3c/webcodecs/blob/main/explainer.md
- https://datatracker.ietf.org/doc/html/draft-ietf-wish-whip
- https://tools.ietf.org/id/draft-omara-sframe-03.txt
- https://github.com/w3c/mediacapture-transform/blob/main/explainer.md
- https://github.com/w3c/webrtc-encoded-transform/blob/main/explainer.md
- https://github.com/w3c/mediacapture-region/
- https://github.com/wicg/capture-handle/