RTCWEB Architecture

Harald Alvestrand
Overview editor

What I will talk about

- Goals for RTCWEB
- Architecture layers and their context
- Security in context
- Data transport, format, framing and securing
- Connection Management
- Presentation, Control, Local functions

What I will *not* talk about

- Details...

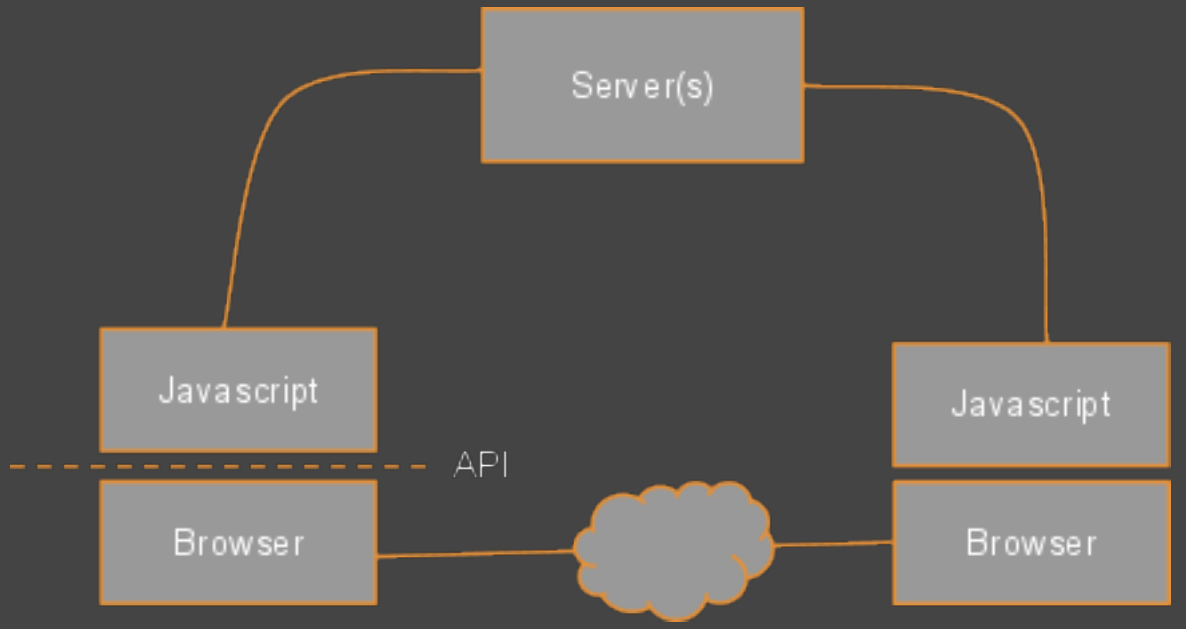
That's by design.

Goal for RTCWEB:

- Enable Realtime Communication between browsers.
  - No plugins. Intended to be in standard browser
  - No relays required (but relays possible)
  - Real time = 100ms timescale; "interactive"
  - Media = Audio, Video and "other stuff"
- Drive the design by use cases
  - We expect real world use to be innovative, different.
  - Use cases ambition is: "at least this should be possible"
  - Design general functions, not one-off solutions

Architecture in context

- At startup, browsers do not know each other
- Javascript mediates the setup process through server.
- Media flows through the shortest possible path for latency



Architecture layers

- Data transport
  - Data path establishment: NAT traversal using ICE
  - Transmission: UDP (TCP backup)
  - Congestion management
- Data encapsulation
  - RTP
  - Some non-RTP method for non-media data
- Data formats
  - Codec choices go here
- Connection management / signaling
- Presentation and control
- Local system support functions

Security in context

- All components (except the RTCWEB-implementing browser) must be assumed evil
- Browser that executes JS using RTCWEB is responsible for both its own security and that of victims it can reach (such as other tabs in the same browser, or other devices on the same LAN)
- Keep trust to a minimum

Data Transport

- Data path establishment: NAT traversal using ICE
  - Secures against "voice hammer" attacks
- Transmission: UDP (TCP backup)
  - Relays are sometimes needed
- Congestion management is necessary
  - Self-fair
  - Plays well with others
  - Would be nice not to invent one here

Data framing and securing

- RTP exists. We will reuse it.
- We have no need to carry unencrypted data.
  - SRTP for media
  - DTLS for non-media data
- DTLS-SRTP key negotiation
  - SDES key negotiation allows for retrospective decoding of wiretap data, reveals key to Web browser
  -
- Note: UI issues are important for security
  - Mostly not IETF specs, but IETF knowledge informs W3C discussions

Data formats

- Data formats must be negotiated
  - Any consenting adults can agree on a data format
- A mandatory to implement codec prevents interoperability failure
- Need to focus on requirements for the baseline case (where MTI would come into play)

Connection Management

- Needed for setup:
  - Negotiation of data formats, transport options and security parameters (incl keys)
- Needed while connected:
  - Reaction to changed connectivity and needs (ex: resolutions)
- Least controversial proposal: ROAP
  - Specify a format for JS-Browser exchange
  - Usable as browser-to-browser format
  - Possible to gateway to SIP and XMPP
- We expect innovation in what-connects-to-what
- Active area of discussion!

Other Local Functions

- User action needs to cause net communication
  - Local and remote media mute -> stop/start sending
  - Display window size change -> change resolution

- Functions are needed to achieve usable applications
  - Automatic Gain Control -> consistent audio levels
  - Acoustic Echo Cancellation -> no feedback loops
  - Dynamic jitter buffers -> consistent (low) playout times

Interfaces

- <Video> tag and friends
  - Needs to connect to a MediaStream
  - Also want to connect canvas, WebGL....
- Audio interfaces
  - Would like to NOT define those here
  - Advanced Audio APIs exist, not universally adopted

Summary

- Overview is a means of ensuring:
  - we can talk about things separately, while being aware where the interfaces are
  - we feel confident we have all the pieces covered

draft-ietf-webrtc-overview-00.txt