WebRTC terminology

Harald Alvestrand
IETF 91

Why terminology?

- Different entities have different jobs
- Different jobs lead to different requirements
- One size fits all? No.

The Browser / Device Split

- The model has a browser trusted by the user.
- Both above (Application) and below (Network), untrusted entities reside
- If there are no untrusted applications, security requirements are different

Anywhere in the Net?

- ICE, STUN, TURN requirements are to deal with NATs and firewalls
- If at least one end doesn’t have these issues, can we make things simpler?

Endpoints?

- The core model is endpoint to endpoint.
- Gateways are a reality - but not core.
- Need language to talk about them.
- Gateway to gateway is out of scope - Someone Else’s Problem

Proposed terminology

- WebRTC browser - all requirements
- WebRTC device - no JS API (controlled, thus trusted, “upper” surface)
- WebRTC endpoint - browser or device
- WebRTC compatible endpoint - relax net requirements
- WebRTC gateway - what it says

Subsets

- All browsers fulfil all requirements on devices.
- It’s confusing to call them devices too, so let’s use the term “endpoint” to cover both.
- On the net side, they are identical.
- Gateways are WebRTC-compatible endpoints.

The concept of “compatible”

- If we can talk successfully - we’re compatible.
- Not all apps will talk to all compatible endpoints
    - Missing functionality (video, datachannels)
    - Incompatible signalling (not standardized)
- Only one working app is necessary to be compatible.

Open Questions

- Browsers need full functionality. (are there any audio browsers?)
- Do devices support everything? Datachannels? Audio? Video?
- Does it make sense to talk about “WebRTC libraries”?
- Exactly what do we require?