RTCWeb Considerations for Mobile Devices
draft-isomaki-rtcweb-mobile

markus.isomaki@nokia.com
Scope

- RTCWeb in mobile endpoints such as smartphones or tablets
  - Cellular and Wi-Fi interfaces

- Introduce the specific challenges and suggest recommendations

- Topics
  - Connectivity to the Calling Site
  - Media and Data Channels
  - Recovery from interface switching (Cellular <-> Wi-Fi)
  - Congestion avoidance

- Some of the issues maybe addressed by IETF, some by W3C, some may have to be left for browser or application implementation
Assumptions

- Battery-operated devices

- Cellular network characteristics mainly based on WCDMA/HSPA
  - CDMA or LTE not so different
  - Networks as they are deployed today (wrt. radio timers, NAT/FW timers, buffering, link mode, QoS)

- Interface/network selection and switching according to how it is done in today’s smartphones
Power consumption not based on traffic amount but pattern
- Power-save (Idle) mode entered based on inactivity timers (~5-30 s.)
- Getting active again takes a little while (~0.5-2s.) – seen on the “first” RTT!

Firewalls and/or NATs in regular use
- UDP timeouts often ~30s, TCP timeout anywhere between 1 and 60 min.

Link layer operates in acknowledged (retransmission) mode
- Single L2 loss delays also the subsequent packets

Buffers are notoriously large
Connectivity to the Calling Site

- WebSocket protocol or HTTP long polling over TCP
- Keepalives needed to keep the NAT/FW happy
  - Major source of power consumption

Recommendations
- Keep the keepalives as infrequent (and small) as possible
  - WebSocket PING-PONG better than HTTP with full headers
  - Synch keepalives from different sources to have just a single cycle

Help from the standards
- Common messaging/notification/push services for Web/Browser apps
- Better app and/or browser visibility to NAT/FW timers
Media and Data Channels

- Keeping **inactive** channels up is expensive
  - NAT/FW keepalive, liveliness, consent refresh
- Data Channels run on UDP
  - 30s NAT/FW timeout makes them infeasible to keep up for a long time

Recommendations

- Tear down inactive channels if possible

Help from the standards

- Channel suspend/resume?
- TCP-binding for Data Channel?
- Adaptive consent refresh timers
  - draft-muthu-behave-consent-refresh
Interface Switching

- Most devices (OS’s, Connection Managers) do hard handovers between cellular and Wi-Fi
  - The old interface and IP address are gone as soon (or before) the new one is up
- Recovering calling site connectivity
  - Re-initiate WebSocket or HTTP long polling state
- Recovering media
  - Update the media addresses to the peer

- Possibilities
  - ICE restart: takes time to setup signaling, send new Offer
  - draft-wing-mmusic-ice-mobility: a faster method
  - Recreate the whole PeerConnection
- Suggest to address this explicitly in RTCWeb “1.0” or “2.0”
Congestion Avoidance

- Bufferbloat in cellular networks
- Separate users are segregated to some extent, but user’s own TCP traffic will easily ruin real-time media

Recommendations
- Less-than-best effort congestion control for Data Channels for large transfers
- Browser should be careful about its HTTP/TCP use while media is on
- DSCP marking: draft-jennings-rtcweb-qos
- RMCAT BOF

- Would be nice to get RTP/UDP on unacknowledged link mode