Deploying WebRTC: “Would you like it well-tested?..”

Vladimir Beloborodov

www.merasws.com

Major Areas of RTC Testing

	Signaling	Signaling + Media
Functional Testing / Interoperability Testing	• Singular calls are sufficient (typically) • Manual testing / Test automation with scripts	
Robustness Testing / Security Testing	• Singular calls are sufficient (typically) • Manual testing / Test automation with scripts • Randomized inputs (fuzz testing)	
Load Testing / Stress Testing	• Many concurrent calls • Usually driven by scripts or predefined flows • Monitoring and measuring	

Some WebRTC Testing Specifics

- Cross-browser interop. (*specs compliance, codecs*)
  - *Esp. before the official standard, MTI video codec(s), etc.*
- Testing signaling-server function
  - *Depends on specific signaling protocol(s) being used*
- Testing media-server function, *if there is one…*
  - *Do you do media transcoding? Conferencing?*
- TURN-server load testing, *if relying on TURN…*

Your mileage may vary…

- Using WebRTC-based 3rd-party services/frameworks
  – *Does your supplier offer help with testing or test tools?*
  – *Can you "skip" some test work thanks to SLA guarantees?*
- Testing with separate commercial products/services
- Opting for "DIY-testing"

(Web)RTC Test Automation Parts

Test-scenario logic (*e.g. with scripts*)

Call / Session signaling	Media signaling	Media payload(s)	Auxiliary functions
• SIP, XMPP *and alike* • JSON-based • REST-based • *"Project WONDER"*	• SDP (WebRTC 1.0) • *ORTC-based (post-WebRTC 1.0)*	• Audio / Video • Data, in needed format(s)	• Logging • Test Helpers • *etc.*

Basic RTC Mechanisms / Platform & Network Interactions

--use-fake-ui-for-media-stream
--use-fake-ui-for-media-stream

Browser-based Test Automation

- Use fake media devices in getUserMedia()
  - *Start Chrome with* **--use-fake-device-for-media-stream**
    - *You may also add* **--use-file-for-fake-video-capture=***sample.y4m*
  - *In Firefox:* **getUserMedia({** video: true, audio: true, **fake: true }**, …**);**
- Disable permission dialogs for camera/mic access
  - *Start Chrome with* **--use-fake-ui-for-media-stream**
  - *In Firefox preferences:* **media.navigator.permission.disabled:true** *(Apparently, FF 33 auto-disables the dialogs with "fake: true" ???)*

--use-fake-ui-for-media-stream
--use-fake-ui-for-media-stream

Browser-based Test Automation

- Scripting and testing UI interactions
  - *Selenium (http://www.seleniumhq.org/)*
- Running browsers in headless mode
  - *Linux and OS X: X virtual framebuffer (Xvfb / Xdummy)*
  - *As an example, check scripts from Otalk: https://github.com/otalk/webrtc-tester*

--use-fake-ui-for-media-stream

Node.js-based Test Automation

- node-webrtc
  - *http://js-platform.github.io/node-webrtc*
  - *WebRTC peer connections and data channels in Node.js!*
  - *Media stream objects are just "stubs", as of now…*
- "Missing pieces" (*work in progress, stay tuned!* ☺)
  - *Better tools for scripting signaling scenario tests*
  - *Media stream emulation (predefined or file-based)*

Signaling Traces with WebRTC

- Wireshark *is* your good friend!
  - *Limitations: Encrypted data, using pcap in 3rd-party apps*
- Collecting traces in signaling libs: *to be improved*
- Traces in browsers: *evolving, but still in early days*
- ***Proposing a common format for signaling archives***
  - ***"Simple Application-Level-Signaling Archive" (SALSA)***
  - *https://github.com/VladimirTechMan/salsa-format-spec/*

--use-fake-ui-for-media-stream

Thank you for your attention!

Questions and feedback are welcome!

Vladimir Beloborodov

VladimirTechMan@gmail.com

@VladimirTechMan

http://linkd.in/1q0CyPo