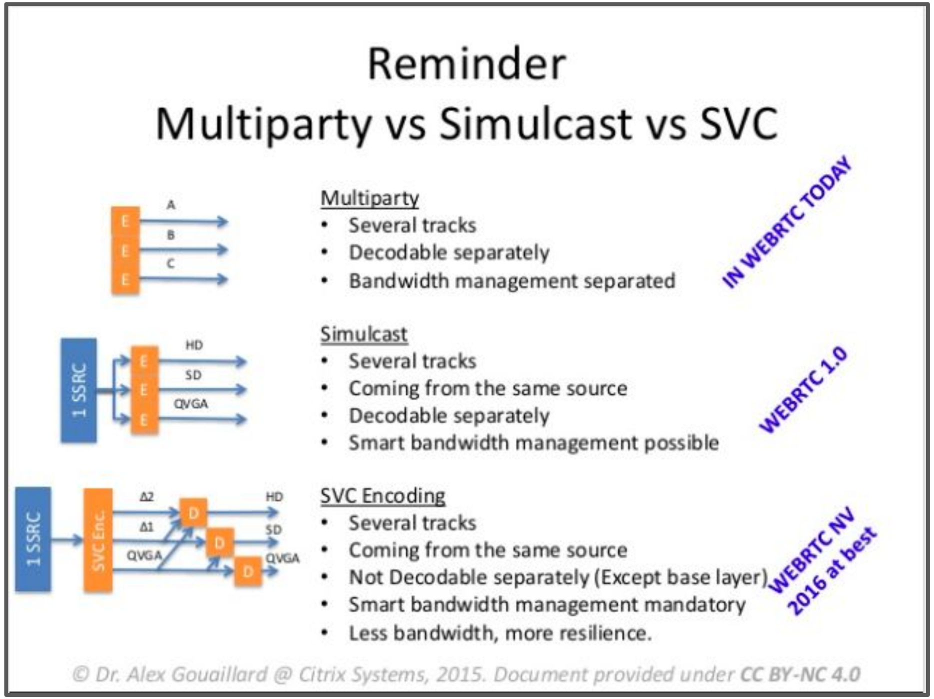
WebRTC 1.0 Simulcast

Facilitators: Dr. Alex G., Alvestrand H.

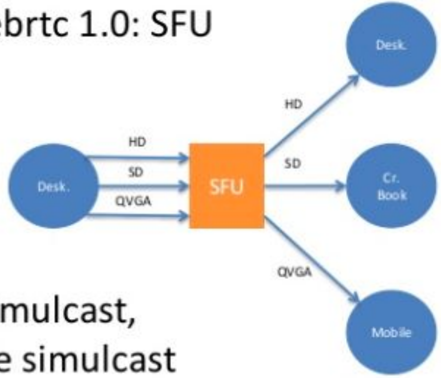
Simulcast in WebRTC 1.0



Simulcast in WebRTC 1.0

Simulcast: Use case for webRTC 1.0

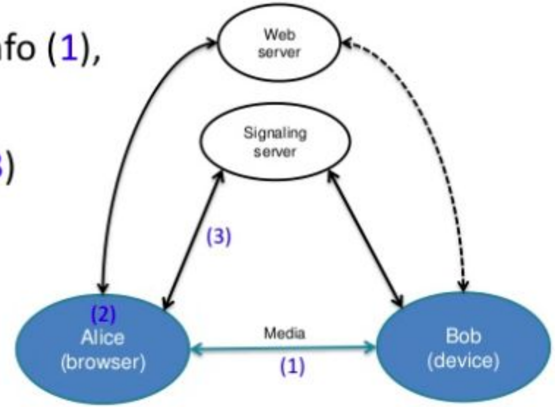
- Use Case for webrtc 1.0: SFU



- Browser send simulcast, does not receive simulcast (in webrtc 1.0)

Simulcast (and SVC later):
3 layers, 3 working groups, 2 standard bodies

- Separate the
- "on-the-wire" info (1),
- JS API (2),
- and signaling (3)



Generic Goal

Iron out the latest bits of WebRTC 1.0 spec / RTCWEB and dependencies

More specifically: Simulcast.

It is special in the sense that it requires interaction between a web app and an SFU. Hence the set up of a hack session with both sides around the table to test interoperability.

Open Source KITE used to help testing against many browsers, to automate the tests, and keep them running after this week-end.

Expected Outcome and Success Metrics

#bugs reported to Browser Vendors, resp SFU devs.

#bugs fixed in browsers, resp SFU

#KITE tests written

#WPT test written

Biggest WebRTC Hack session ever

19 registered

(13 listing ONLY WebRTC)

All main Browser vendors: MS, Google, Mozilla, Apple

Many SFU Tech Lead on-site: Meetecho, Medooze, …

Many SFU Tech Lead prepared tests: MediaSoup, ….

Specific Individual goals

- **Meetecho - Lorenzo**: make Janus better, understand FlexFEC in chrome
- **Medooze - Sergio**: make medooze better
- **Apple - Youenn**: WPT tests (DC PR review), KITE Tests, investigate medooze crash
- **Microsoft - Nikita**, learn KITE to be able to use it internally,
- **Google - Marina**: ICETransport tests in WPT,
- **Google - Henrik**: getSats evaluation and fixing in the scope of Simulcast. Stupid Tests.
- **CoSMo - Manu**: Help everybody to set up KITE and write new tests.
- **Google - Harald**: Get feedback from SFU devs in general, and about SSRC specifically. Write some WPT test for the new RTCSctpTransport JS object.
- **W3C - Dom and Carine**: Investigate WPT Coverage computation.
- **CoSMo - Dr Alex**: Glorified secretary. Prepare docs, facilitate tests, report.

Status Report - 10 bugs fixed, 3 new bugs filed

- **Harald:** Day 1: [Chromium: webrtc:10468] - SCTP transport close does not cause event - **FIXED**
- **Dom:** wpt bugs and coverage
- **Lorenzo:**
  - Day 1: Bug in Janus SFU happening with Firefox. **FIXED**
  - Day 2: Bug H.264 simulcast layer switching logic was wrong leading to artefact on receiving side. **FIXED.**
- **Sergio:**
  - Day 1: [webrtc-10470] - simulcast H.264 in chrome 75, is broken when using addTransceiver API. **FILED**
  - Day 1: [webrtc-10469] +lorenzo +nikita - chrome bandwidth allocation for simulcast layers depends on order (and should not). **FILED**
  - Day 2: [w3c/webrtc-pc #2141] - Missing spec on how to assign bandwidth between simulcast layers. **FILED**
- **Henrik:**
  - Day 1: [cr-803014] - updated webrtc-internals to report more compliant data - **FIXED**
  - Day 2: [webrtc-9547] - Have better visibility on possible bug (fps reported wrong in the scope of simulcast).
- **Marina:**
  - Day 1: [cr-907849] - DTLS transport related: **FIXED**
  - Day 2: [cr-944105] - and other things related to ICE.
- **Youenn:**
  - Day 1: Launch Janus test in STP+chrome using KITE on local computer. DONE.
    - subset of lennart DC tests (modified) pushed to WPT - PR #16038 - **MERGED.**
  - Day 2: write a test (any SFU), to test screen sharing, and gather corresponding stats.
    - WPT: write generateCertificate() test. **PR**
    - W3C webrtc-pc bug #2142. **PR**
    - WPT: Replace generateOffer by generateAudioReceiveOnlyOffer (#16042) **PR**
    - WPT: WebRTC doSignalingHandshake helper routine should be split in two methods (#14633) **PR**

Status Report - Browser support card

			chrome 75 (canary)	chrome stable	Safari TP	Safari	firefox
Media Simulcast / ABR		h264 simulcast	yes - but bug pending	only via SDP mungling	yes	yes	no
		vp8 simulcast	yes	only via SDP mungling	yes	yes	yes
W3C Browsers APIs	RTCTransceiver	Have transceivers	yes - with unified plan	yes - unified plan	yes	yes	yes
	Stats API	Compliant Stats	yes - but bug pending	no	no	no	no
		Per layer Stats	no	no	no	no	no
	Simulcast enabling	Standard API + createOffer()	yes	no	no	no	yes - but old setParameter()
		legacy SDP mangling	yes	yes	yes	yes	no
IETF Internet protocols	Signalling (JSEP, SDP O/A)	Standard Unified Plan	yes	yes	yes	opt-in	yes
		Legacy Plan B	opt-in	opt-in	opt-in	yes	no
	Media Transport (RTP) simulcast features	rid	yes - if using addTransceiver	no	no	no	yes
		repairedId (RTX)	yes	no	no	no	no (no RTX at all)
		legacy ssrc in SDP	no - if using addTransceiver	yes	yes	yes	yes
	Bandwidth evaluation and congestion control	transport-wide-cc	yes	yes	yes	yes	no
		REMB	yes	yes	yes	yes	yes
	not all standards, but some IETF doc exists		vetted by henrik and harald		vetted by Youenn		vetted by nils

Status Report
SFU support table

		Open Source Media Servers								Commercial PaaS		
tested at IETF 104		Yes	Yes	No	Yes	No	No	No	No	No		
team member present at IETF 104		Yes	Yes	No	No	No	No	No	No	No		
Point of Contact		lorenzo	sergio	emil / boris / saul	inaki	?	?	micael gallego	Voluntas	gustavo garcia	?	?
Name		janus (VideoRoom plugin)	medooze	jitsi	mediasoup	INTEL	licode	openvidu / KMS	sora shiguredo	houseparty	tokbox	twilio
SDP Plan semantics	Plan B	yes	yes	Yes	yes		yes	yes	yes	yes		
	Unified Plan	yes	yes	One way only through convertion	yes		no	no	yes	no		
SDP O/A signaling	direct SDP signalling	yes	yes	no	ORTC: RTCRtpParameters on the wire, SDP O/A locally		yes	yes	yes	no		
	other	no	JSON on the wire, SDP locally	Jingle / COLIBRI on the wire, SDP locally			no	no	no	JSON on the wire SDP locally		
simulcast enabled via	SDP munging	yes	yes	yes	yes		yes	simulcast not supported	yes	yes		
	setParameter	yes	yes	no	yes		yes		no	no		
	addTransceiver	yes	yes	no	yes		no		no	no		
PC and stream handling	separate publisher and Subscriber PC	yes	no	no	yes.		no	yes	no	no		
	multiple PC	"master" => multiple,	no	no	no		it's flexible, depends on scalability: M multistream x N PC	?	no	no		
	single multi-stream PC	"unified-plan" => single multistream PC	yes	yes	sending (MID and RID), receiving (SSRCs), both with BUNDLE				yes	yes		
video codecs	VP8	yes + simulcast	yes + simulcast	Depends on configuration, but mainly VP8	yes + simulcast		yes + simulcast	yes	yes + simulcast	yes		
	H.264	yes + simulcast	yes + simulcast		yes + simulcast		yes	yes	yes + simulcast	no		
	VP9	yes + SVC	yes + SVC		yes		yes + SVC	no	no	no		
	rids supported	yes	yes	no	yes		yes	no	no	no		
	repairid supported	yes	yes	no	yes		no	no	yes	no		
	ssrc-less supported	yes (simulcast only)	yes	no	yes		no	no	no	no		
bandwidth congestion control	transport-wide-cc	yes - only receiver side	yes	yes	no		no	no	yes - only receiver side	yes		
	remb	yes	yes	yes	yes		yes	yes	yes			
bandwidth limitation on senders		REMB + SDP AS	no	simulcast layers dropping	proprietary client API		proprietary client API	Proprietary client API or settings	no	REMB		
mid rewritting		no	yes	no	no		no	no	yes	no		

		Open Source Media Servers							Commercial PaaS
tested at IETF 104		Yes	Yes	No	Yes	No	No	No	No
team member present at IETF 104		Yes	Yes	No	No	No	No	No	No
Point of Contact		lorenzo	sergio	emil / boris / saul	inaki	?	micael gallego	Voluntas	gustavo garcia
Name		janus (VideoRoom plugin)	medooze	jitsi	mediasoup	licode	openvidu / KMS	sora shiguredo	houseparty
SDP Plan semantics	Unified Plan	yes	yes	One way only through convertion	yes	no	no	yes	no
simulcast enabled via	addTransceiver	yes	yes	no	yes	no	simulcast not supported	no	no
PC and stream handling	single multi-stream PC	yes in "unified-plan" branch	yes	yes	sending (MID and RID), receiving (SSRCs), both with BUNDLE	it's flexible, depends on scalability: M multistream x N PC	no	yes	yes
video codecs	VP8	yes + simulcast	yes + simulcast	Depends on configuration, but mainly VP8	yes + simulcast	yes + simulcast	yes	yes + simulcast	yes
	H.264	yes + simulcast	yes + simulcast		yes + simulcast	yes	yes	yes + simulcast	no
	rids supported	yes	yes	no	yes	yes	no	yes	no
	repairid supported	yes	yes	no	yes	no	no	yes	no
	ssrc-less supported	yes (simulcast only)	yes	no	yes	no	no	no	no

Annex A: Simulcast Testing with KITE

Specifically for this event, CoSMo created a gitHub repository with two automated kite interoperability tests. https://github.com/ManuCosmo/KITE-Hackathon

One test is a "typical" SFU test: KITE-Janus-Test is provided, which can be easily adapted to test any SFU, and should be the starting point for SFU developers wanting to automatically test against all the browser configuration CoSMo will provide for testing that week end:

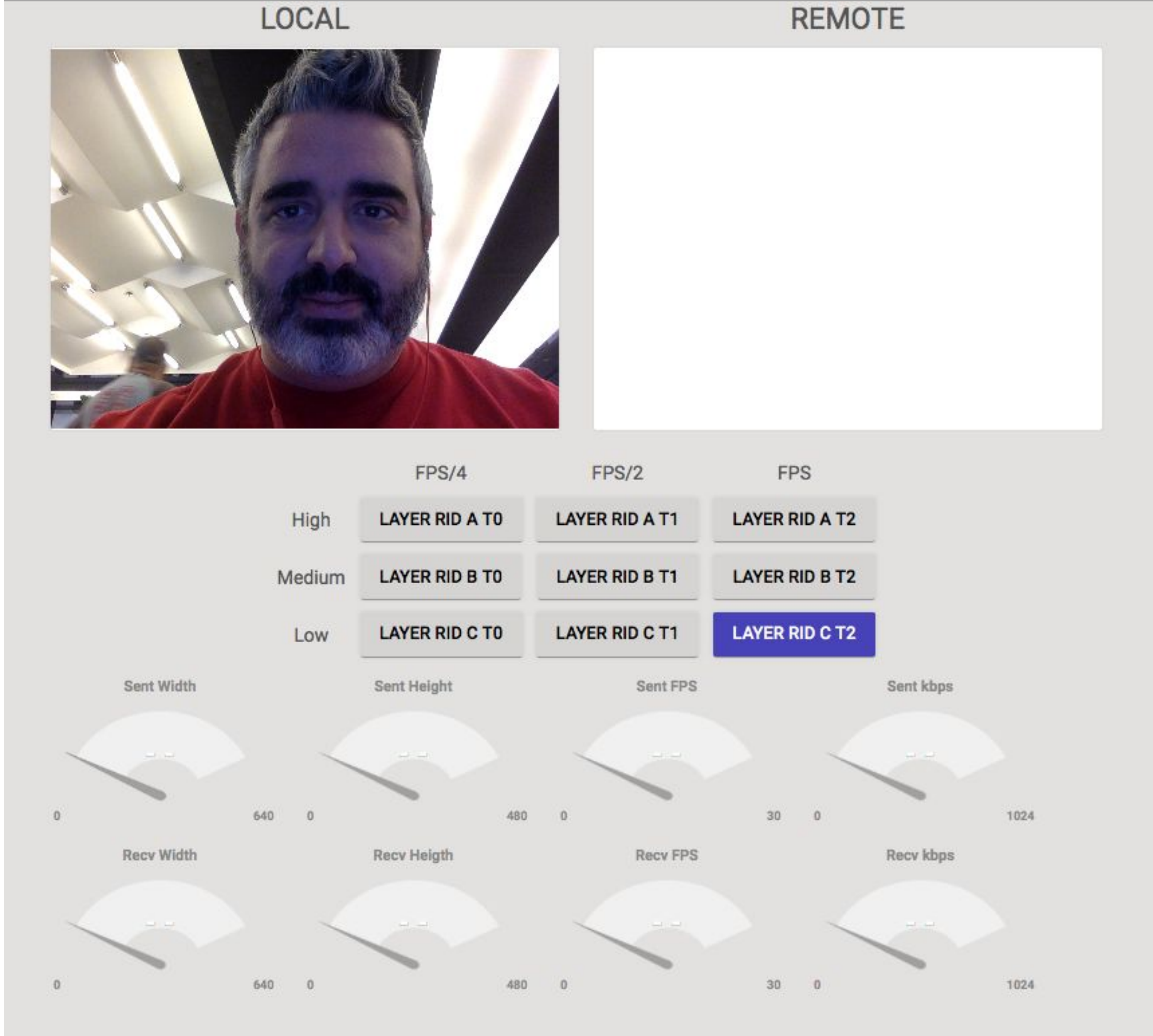
- Loopback setting to simplify testing configuration.
- SFU vendors to set-up and host SFus and loopback web-app
- KITE test to launch web browsers, connect to the web-app, run a scenario and report.
- At least one test written per bug found, to protect for future regression.

Annex B: Medooze test

KITE repo

Meedoze Playground

Choose a simulcast layer (High, Medium and Low) and a Temporal layer (not always present), and you can visually compare the sent and received Width, Height, FPS and kbps.



Annex C: Janus (VideoRoom/Echo) test

https://github.com/ManuCosmo/KITE-Hackathon

To test Janus, a test server is available:

- https://d10.conf.meetecho.com/ietf104/ (deployed locally in the IETF NOC)

The easier way to test simulcasting is to use the EchoTest plugin, which will allow you to choose which layers to send back. A couple of query strings are available to enable simulcast and force a specific codec:

- simulcast=true will enable old-style simulcasting (SDP munging for Chrome and Safari, rid-based for Firefox),
- simulcast2=true will enable the new rid-based simulcasting on Chrome M74 and M75;
- vcodec=X forces a specific codec (e.g., vcodec=h264).

Here's a couple more examples:

- https://d10.conf.meetecho.com/ietf104/echotest.html?simulcast=true
- https://d10.conf.meetecho.com/ietf104/echotest.html?simulcast2=true
- https://d10.conf.meetecho.com/ietf104/echotest.html?simulcast2=true&vcodec=h264

See annex E for a use case.

Annex D: Media Soup Test

https://github.com/ManuCosmo/KITE-Hackathon

To test Mediasoup, one can use https://v3demo.mediasoup.org

Some global variables in the browser console for debugging:

- PC1: the PeerConnection? that sends mic/webcam.
- PC2: the PeerConnection? that receives remote audio/video tracks using BUNDLE.
- CLIENT._micProducer and CLIENT._webcamProducer: mediasoup Producers, useful to check their rtpParameters that have been signaled to the SFU.
- sendSdps(): prints the local and remote SDP of the sending PeerConnection? (PC1).
- recvSdps(): prints the local and remote SDP of the sending PeerConnection? (PC2).

Annex E: The Bandwidth allocation bug - case study

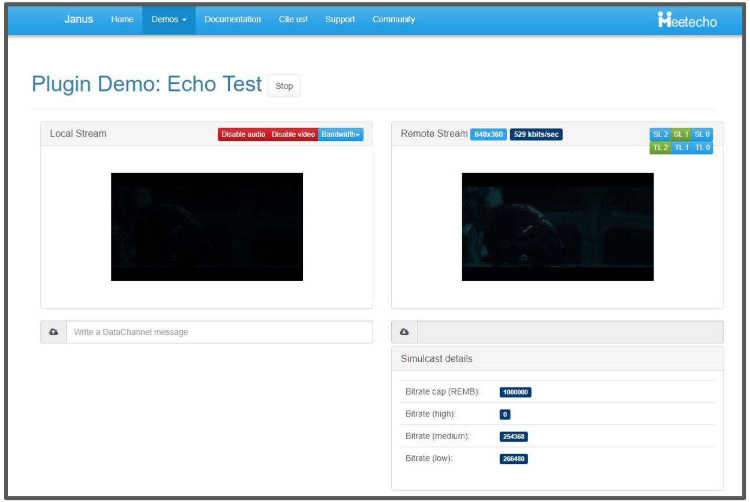
Set-up

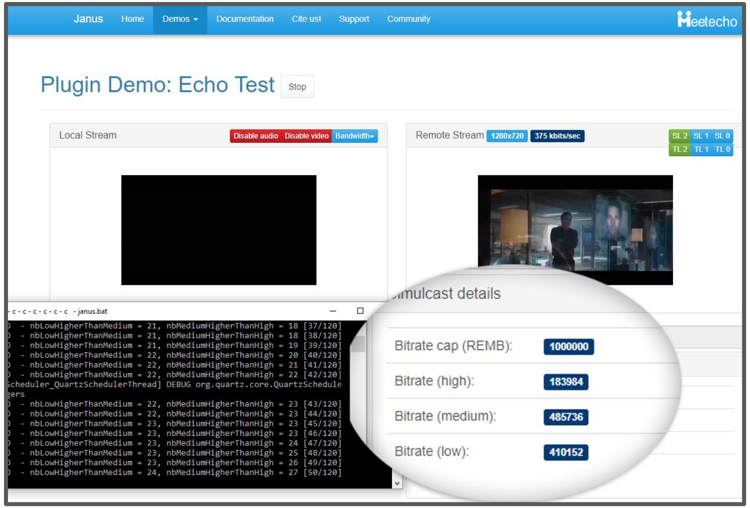
- Repo: https://github.com/ManuCosmo/KITE-Hackathon
- Config: https://github.com/ManuCosmo/KITE-Hackathon/blob/master/KITE-Simulcast-Test/configs/janus.simulcast.config.json
- App: Simulcast loop back page from Janus with VP8
  - https://d10.conf.meetecho.com/ietf104/echotest-cap.html?simulcast2=true&vcodec=vp8
- Browser(s) config(s):
  - Chrome m75 (canari) on Windows 10

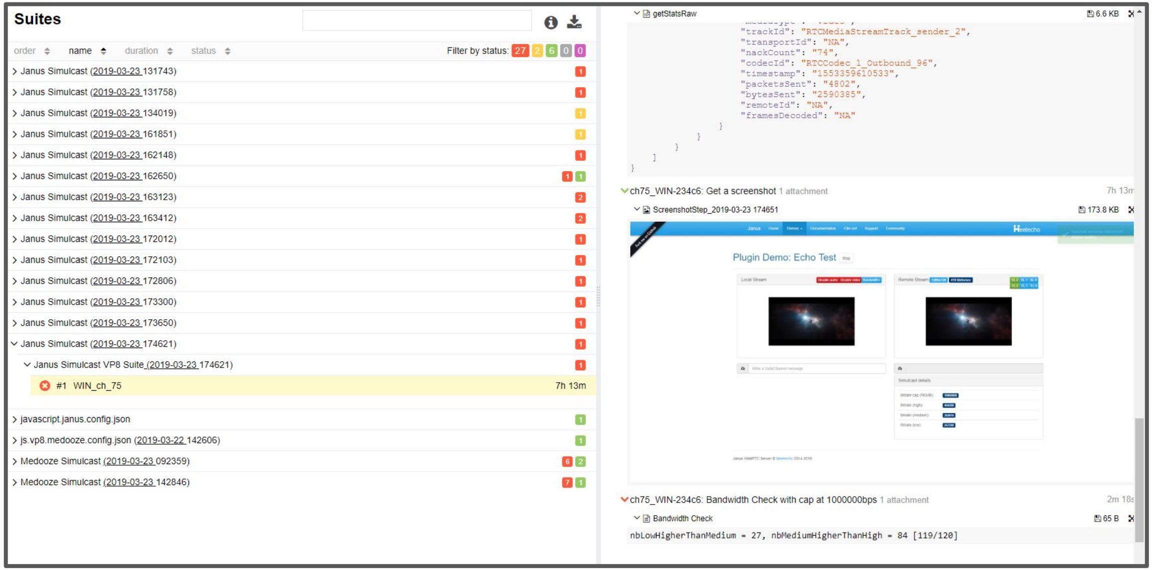
KITE Test: Steps & Checks

- open the page and checks that the call is established (video element display media)
- call getStats
- set the cap (REMB) to 1000000 bps
- every 1s for 120s, check the bitrates for low, medium and high simulcast profiles and:
- increment the `nbLowHigherThanMedium`if the low bitrate is higher than the medium bitrate
- increment the `nbMediumHigherThanHigh`if the low bitrate is higher than the medium bitrate

=> Fail the test if `nbLowHigherThanMedium`or `nbMediumHigherThanHigh` are higher than 0.







- Top left hand: the Echo Test demo page modified to illustrate simulcast and temporal layers selections, as well as bandwidth per simulcast layer.

- Bottom left hand: the app running in an instrumented browser thanks to KITE, you can see that the bandwidth is being allocated to the layers. You can also see it is not being consistent with what it should be (higher bandwidth allocation for higher resolution).

- Top right hand, the Dashboard view of things: list of tests, JSON output of the test, screenshots, and much more