Scalable media delivery on the Web with HTTP Server Push

Presentation to W3C Media and Entertainment Interest Group

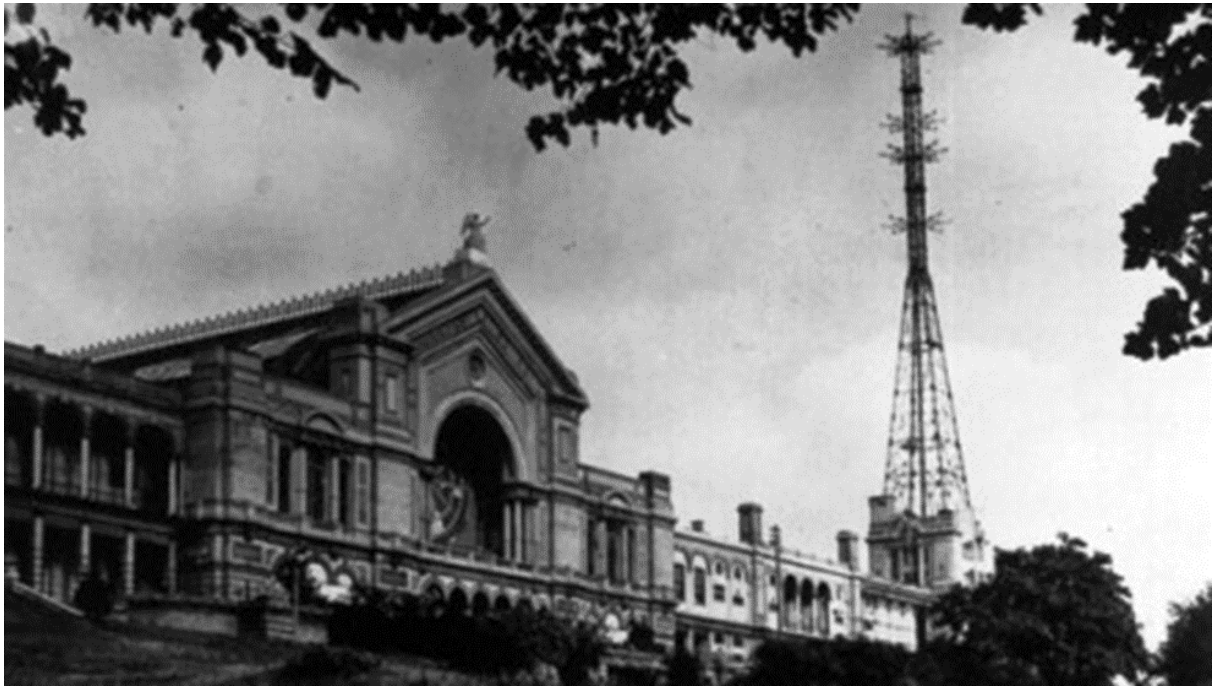
07th August 2018

Lucas Pardue

BBC | Research & Development

Motivation

Life for broadcasters used to be simple…

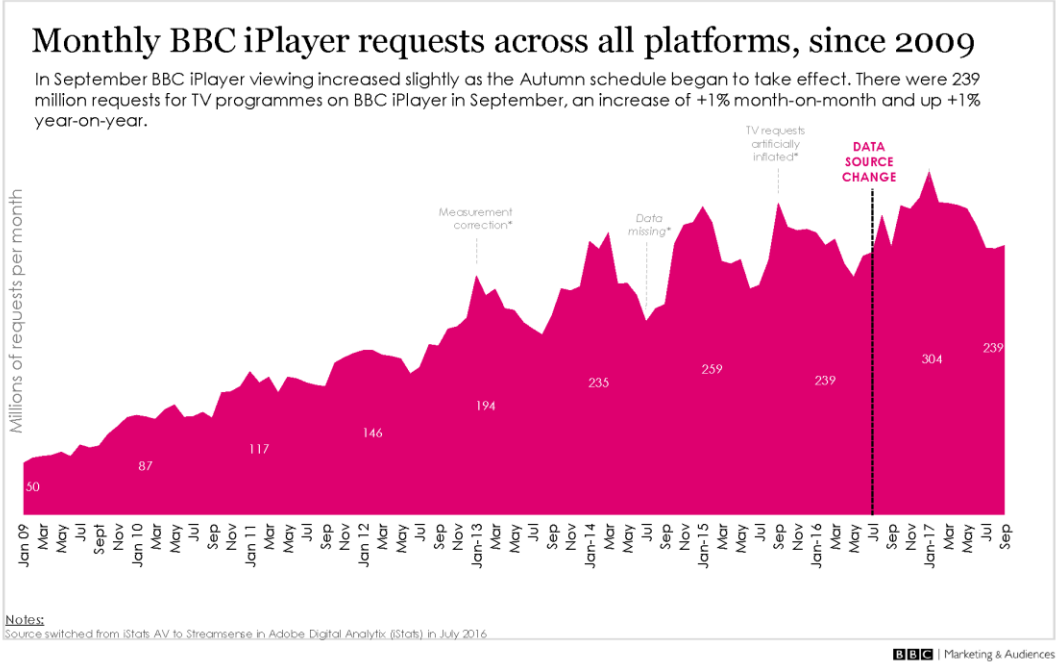


Source: BBC

- Broadcasting from a terrestrial transmitter has a fixed cost.
- The cost doesn't depend on how many people tune in.

The Internet doesn't work like this!

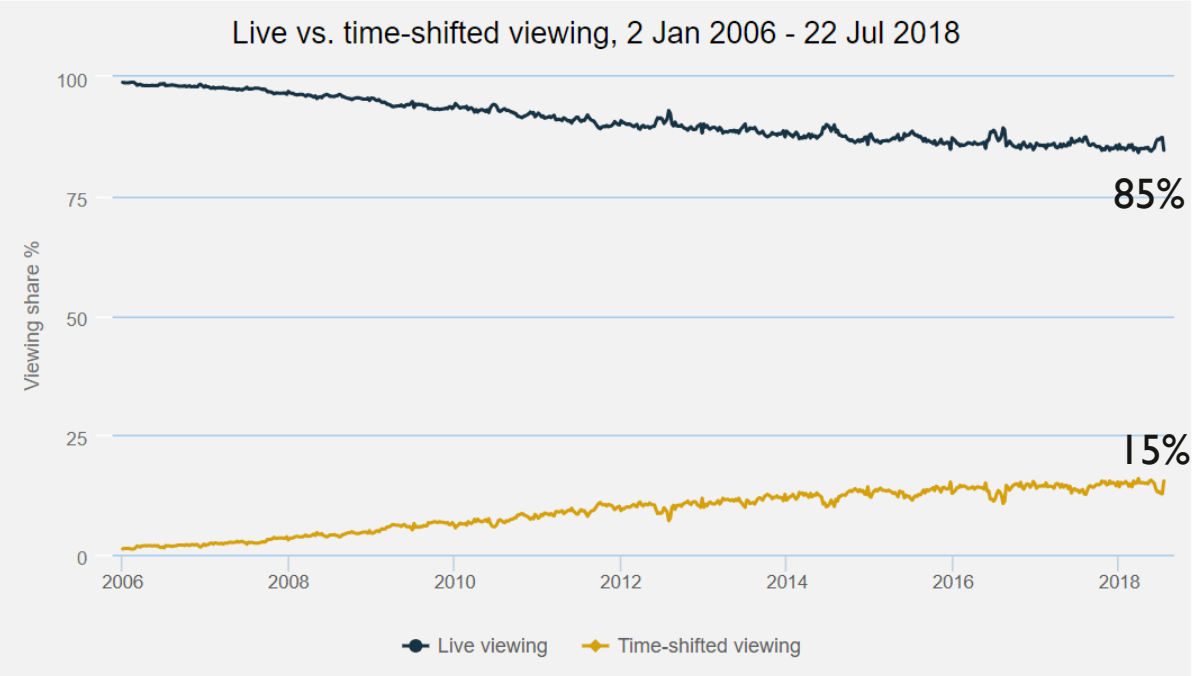
The iPlayer service keeps getting more popular



- Usage of the BBC iPlayer service is climbing steadily.
  - **272M** on-demand programmes requested per month in 2017.
  - First episode of Blue Planet II was requested **4.8M** times.
- CDNs charge per byte delivered by the edge cache.

The cost of providing the service rises in proportion to its popularity.

On-demand viewing is gaining ground…
…but linear viewing still dominates

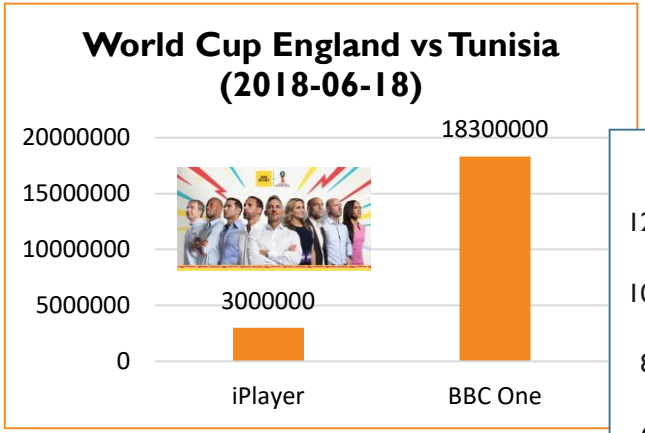


Source: BARB

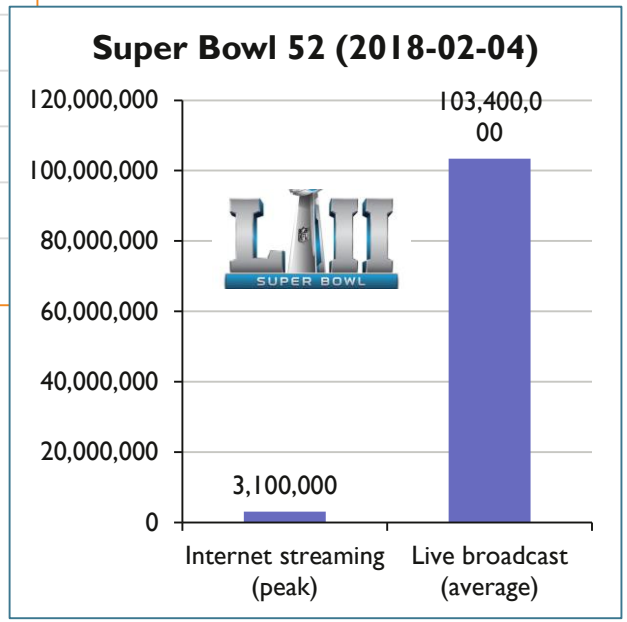
- For the main TV channels in the UK, live viewing represents only **85%** of consumption.
- The remaining **15%** is **DVR time-shifting**, **downloading** and **on-demand streaming**.
  - Time-shifting works best for genres like drama, comedy, entertainment and documentary.

But linear viewing still plays a major role for news, sport and big events.

Linear television is still popular for big events



Source: BBC



Source: AdWeek; USA Today

- One of the largest BBC streaming event to date was England vs. Tunisia in the World Cup 2018.
  - ~3M simultaneous users.
  - About 14% of total peak.
- Super Bowl 52 was watched by nearly fifty times as many viewers.
  - Streaming represented only 3% of total audience.

The potential audience for linear streaming is huge and scary.

There are more and more bits to shift!

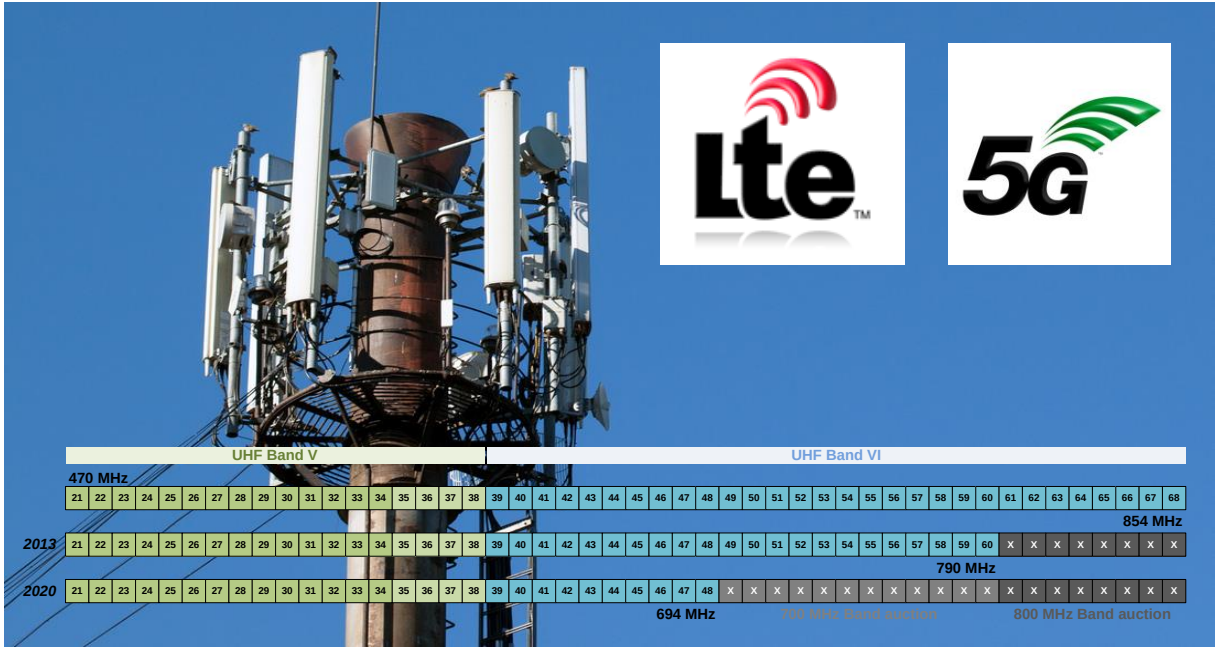
- Higher spatial resolution (SD, HD, UHD).
- Improved colour fidelity (High Dynamic Range).
- Better motion depiction (Higher Frame Rate).

Not to mention:

- New content experiences (3D, 360° Video, AR, VR).
- Next Generation Audio.

All this keeps driving up our CDN distribution costs.

Cellular is biting at our heels



- MNOs are hungry and have deep pockets.
  - 800 MHz band auctioned in 2013 for 4G.
  - 700 MHz band due to be auctioned soon for 5G.
- 3GPP has developed a technology stack called MBMS for media streaming over cellular radio networks.

Our ability to innovate in the broadcast space is now hampered by lack of available spectrum.

The challenge

10 × audience

5 × encoded bit rate

=

50 × load

- How to reach 98% of the UK population without any terrestrial spectrum.

To put that into perspective:

- The last Royal Wedding (April 2011) attracted a total UK live audience of more than 25M across all distribution modes.

Even if CDNs could deliver to that size of audience, we probably couldn't afford to pay for it.

What about IP multicast?

Media streaming over IP multicast

- Layer 3 packet replication reduces redundant transfer.
  - Particularly well suited for large scale audiences.
  - A great way to address the scalability challenge.
- IPTV is deployed around the world today and is supporting pay TV services.
  - Specified by DVB and published as ETSI **TS 102 034**.
  - Works well for managed networks / vertical deployments.
- 3GPP also makes use of IP multicast for its Multicast/Broadcast Multimedia Service (MBMS).

Media streaming over HTTP

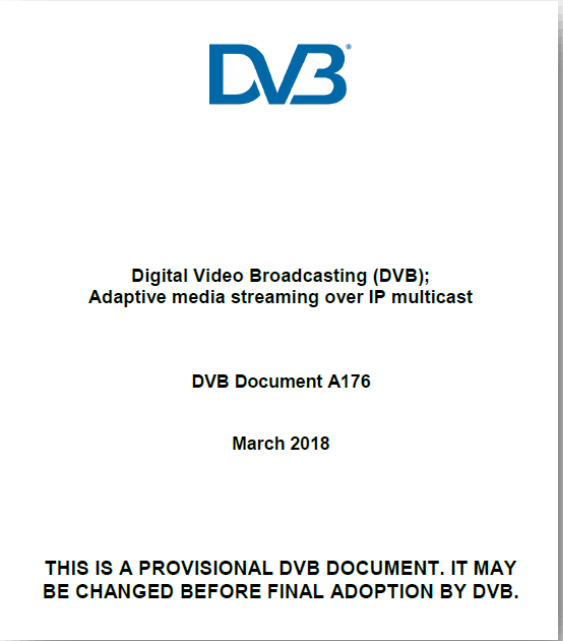
- Industry has embraced unicast HTTP-based streaming.
  - Enables delivery over many types of **access network** (managed and unmanaged), to many types of **end-user equipment**.
  - This is the basis of myriad over-the-top (OTT) Internet streaming services including the BBC iPlayer.
  - Success driven by use of **existing network technologies** and **Content Delivery Networks** (CDN).

Adaptive media streaming over unicast HTTP

- Divide media stream into short duration **segments.**
- Each segment is encoded at several bit rates.
- Client-side adaptation between these encodings, to suit local network conditions.
  - Graceful changes with respect to network degradation or improvement.
- Several proprietary streaming technologies exist:
  - Microsoft Smooth Streaming.
  - Adobe HDS.
  - Apple HLS.
- Industry aligning toward **MPEG-DASH**.

Standardisation of adaptive media streaming over IP multicast

- CableLabs specification for ABR Multicast on CATV networks.
- 3GPP MBMS specification for enhanced television (enTV) on cellular mobile networks.
- DVB specification on adaptive media streaming over IP multicast.
  - Commercial requirements driven by use cases.
  - Technical work undertaken by TM-IPI ABR Multicast task force led by Richard Bradbury, BBC R&D.
  - **DVB BlueBook A176**, released March 2018.



DVB

Digital Video Broadcasting (DVB);
Adaptive media streaming over IP multicast

DVB Document A176

March 2018

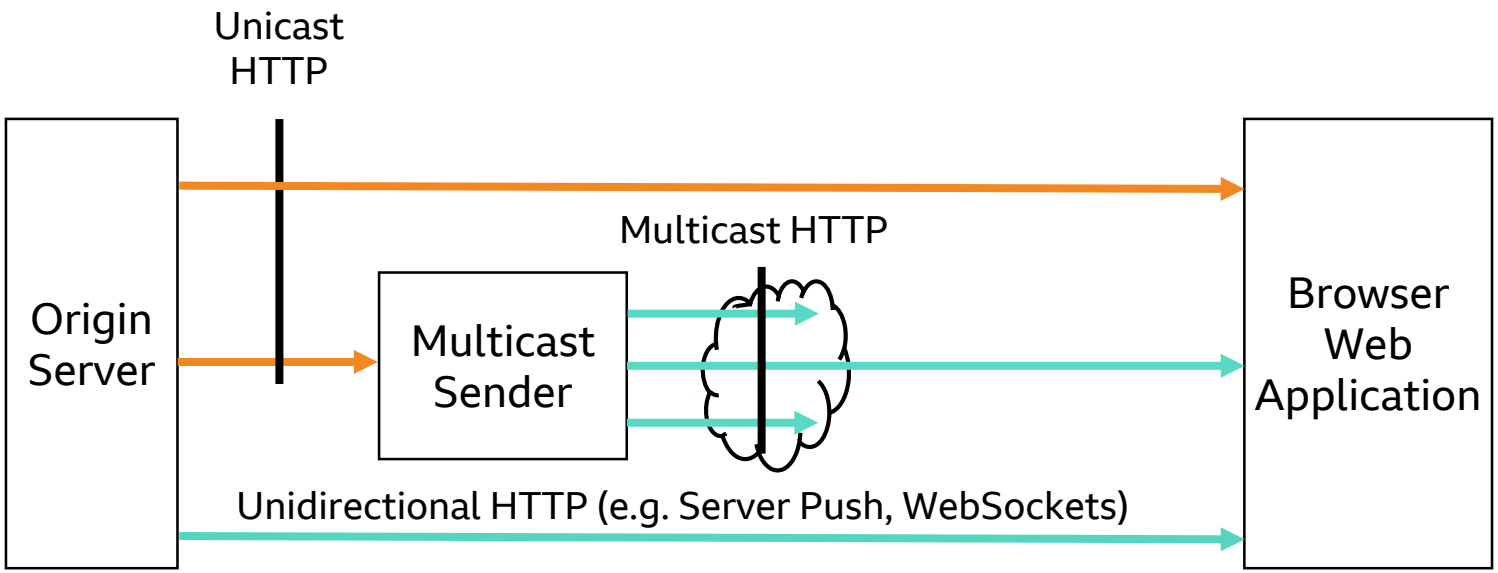
THIS IS A PROVISIONAL DVB DOCUMENT. IT MAY BE CHANGED BEFORE FINAL ADOPTION BY DVB.

Solution space

- Several candidate multicast transports:
  - NORM, FLUTE, ROUTE, Multicast HTTP/QUIC, etc
- Unidirectional delivery of HTTP resources.
  - **Push mode**.
- Clients reconstruct resources from received packets.
  - Repair and validation if required.
- Clients are typically software components external to the User Agent.
  - What would it look like to have unidirectional HTTP support **integrated** in the Web Platform?

Unidirectional HTTP flows

Push mode distribution



Pull mode

Push mode

Push mode Use Cases

- Scalable media distribution
- Low-latency media distribution
- Notifications/updates (including non-textual objects)

Push mode API high-level goals

- Deliver HTTP resources to userland JavaScript:
  - Resources identified by web-friendly **URLs**.
  - Metadata in a standard, well understood format.
  - Align with same-origin and secure context requirements.
- User Agent does the heavy lifting:
  - Avoid the need to implement parse, decode or decapsulation in userland.
- Allow use in several contexts.
- Use familiar paradigms.
- Provide methods and policies for opting in to push mode delivery.

A comparison of push mode APIs

	URL visibility	Directionality	Distribution model	Supported payload types
HTTP Server Push*	Native	Unidirectional	Centralised	Any permitted type
Server Sent Events	Mapped/ encapsulated			Text only
Push API				Any permitted type
WebSockets		Bidirectional		
RTCDataChannel			Peer-to-peer	

* *No standard API exists today*

A collection of HTTP Server Push API discussions

- HTTP/2 server push support, Martin Thomson - Fetch Issue #51
- Observer API for fetch groups, Illya Grigorik – Fetch Issue #65
- Interacting with HTTP/2 push, Jake Archibald – Service Worker Issue #215
- Delivering H/2 Push Payloads To Userland, Alex Russell – GitHub Gist
- API for detecting and controlling server push with HTTP/2, Brendan Long – W3C Bug 28083

Fetch Observer

- Continued enhancement of the Fetch Observer API seems to be the most viable option.
- Work is required to close this out.
- How closely coupled is a Push mode API to HTTP Server Push?
  - Server Push delivers mixed results for web performance
  - Internet community is discussing its continued relevance in HTTP
  - Google is asking "*If we destroyed push, would anyone really notice?*"

https://datatracker.ietf.org/meeting/102/materials/slides-102-httpbis-chromes-view-on-push-01

Resource Push API proposal

- Simple API, based on public discussion.

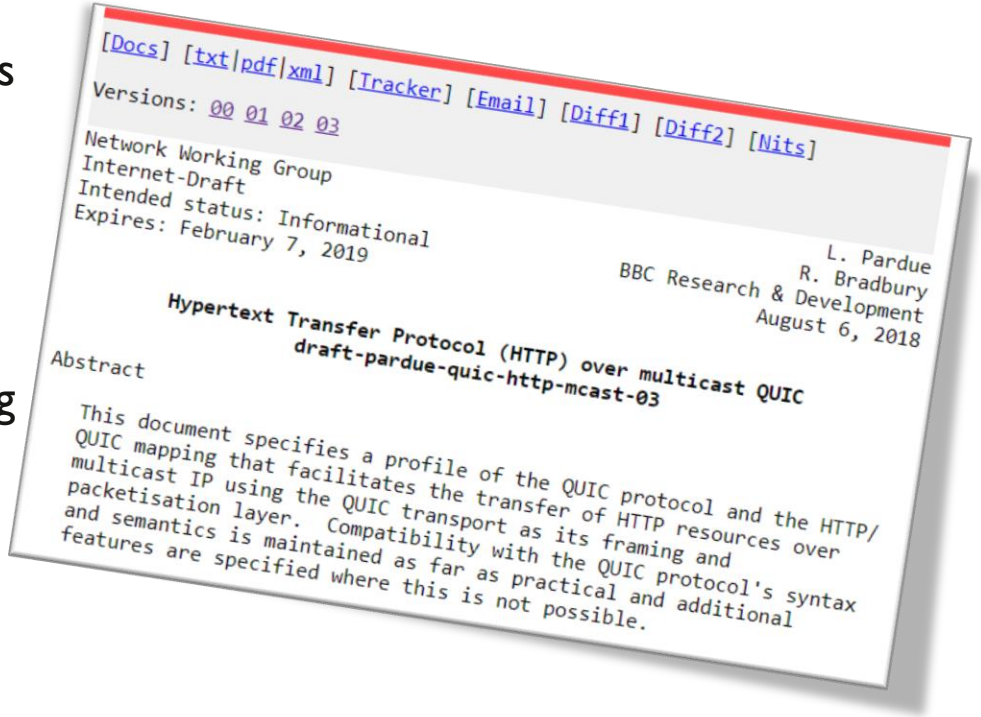
```
interface ResourcePushEvent : Event {
  readonly attribute Request request;
  readonly attribute Promise<Response> response;
}

dictionary ResourcePushEventInit : EventInit {
  required Request request;
  required Promise<Response> response;
}
```

Multicast HTTP/QUIC to the Browser

HTTP over multicast QUIC

- An independent **Internet Draft** that fills some gaps between IP unicast and multicast.
  - Based on a new transport protocol called **QUIC**.
- Unidirectional delivery of resources using HTTP **Server Push**.
- Describes a means of service discovery using **HTTP Alternative Services** [RFC 7838].



[Docs] [txt|pdf|xml] [Tracker] [Email] [Diff1] [Diff2] [Nits]

Versions: 00 01 02 03

Network Working Group
Internet-Draft
Intended status: Informational
Expires: February 7, 2019

L. Pardue
R. Bradbury
BBC Research & Development
August 6, 2018

Hypertext Transfer Protocol (HTTP) over multicast QUIC
draft-pardue-quic-http-mcast-03

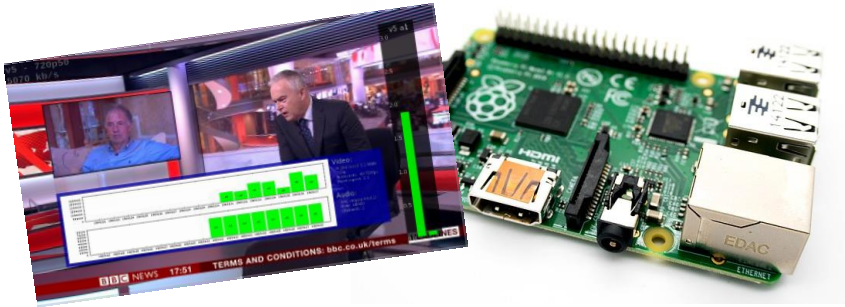
Abstract

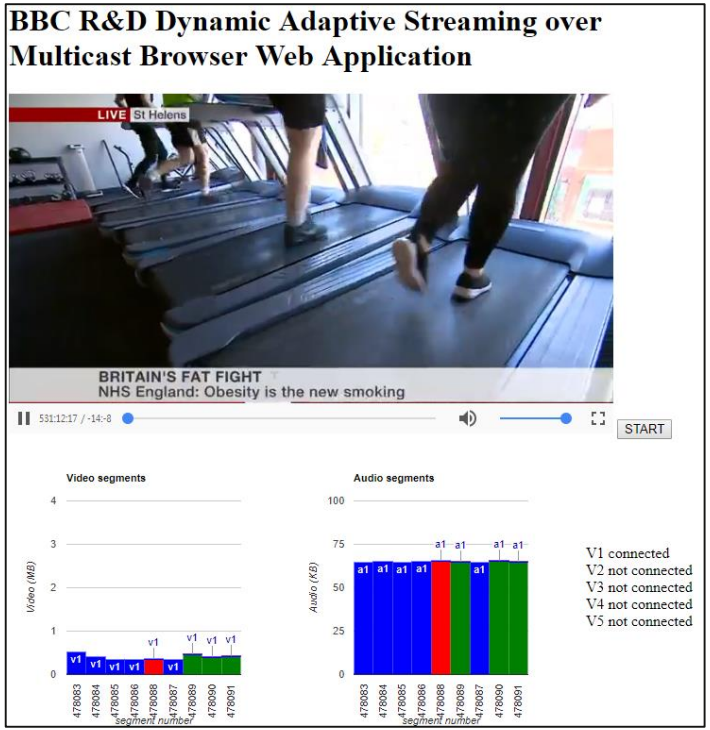
This document specifies a profile of the QUIC protocol and the HTTP/QUIC mapping that facilitates the transfer of HTTP resources over multicast IP using the QUIC transport as its framing and packetisation layer. Compatibility with the QUIC protocol's syntax and semantics is maintained as far as practical and additional features are specified where this is not possible.

https://datatracker.ietf.org/doc/draft-pardue-quic-http-mcast/

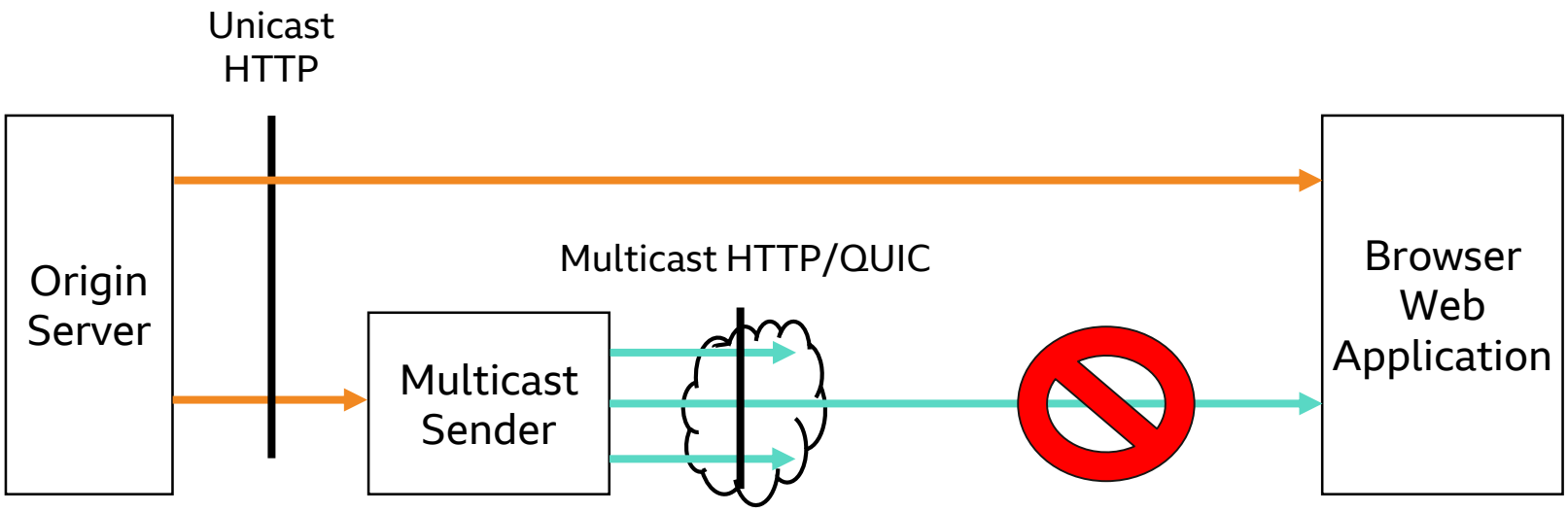
Prototyping

- We have an **end-to-end video streaming prototype** of a multicast HTTP/QUIC sender and a receiver.
  - The multicast receiver runs on embedded devices such as the Raspberry Pi.
- We have recently completed an experimental proof of concept in **Web browsers**.
  - We will be demonstrating this in the IBC **FutureZone, Stand 8.F08**.





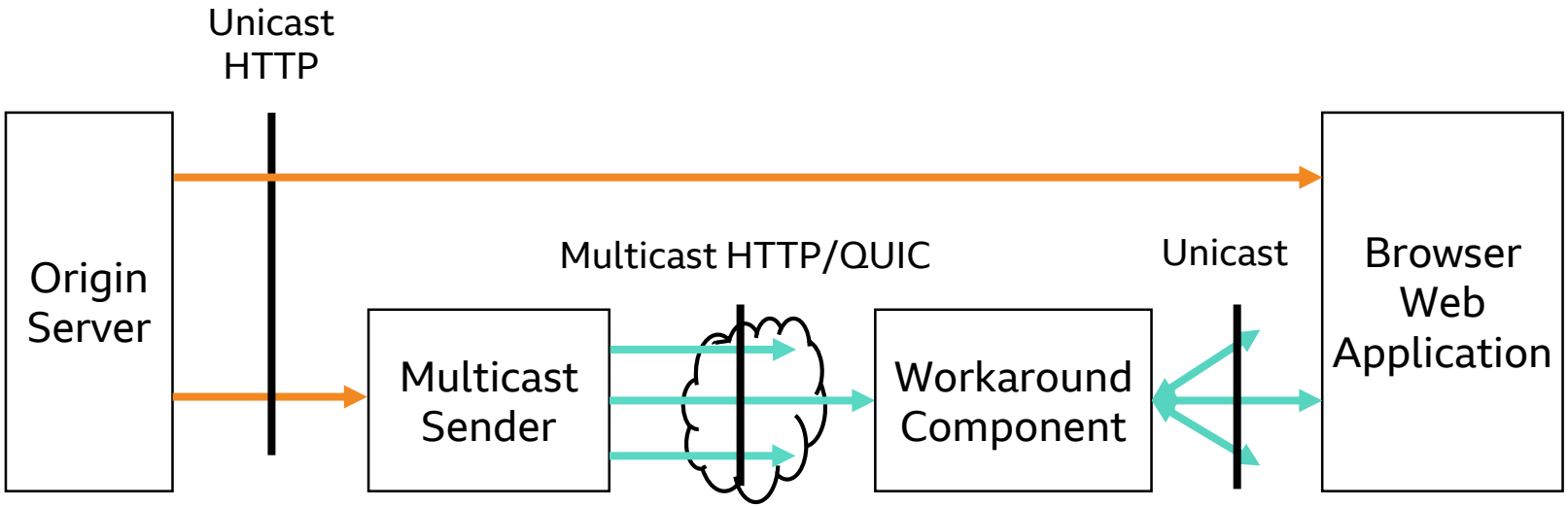
Dynamic adaptive streaming over multicast HTTP/QUIC



Pull mode

Push mode

Dynamic adaptive streaming over multicast HTTP/QUIC



Pull mode

Push mode

Browser web application overview

- A page with a `<video>` element, Dash.js media client, and a Service Worker.
- The Service Worker intercepts client MPEG-DASH manifest request.
  - Performs multicast HTTP/QUIC discovery.
- If multicast is available, take control of streaming session, managing subscription to different quality bitrates.
- Service Worker receives QUIC packets and passes them to a deserialisation library.
  - The library generates ResourcePushEvent events containing HTTP Request and Response objects.

Browser support

Advertising multicast HTTP/QUIC with HTTP Alternative Services (Alt-Svc)

- Alt-Svc provides a means to advertise alternative protocols or endpoints that a client may wish to switch to.
- We use it to decorate unicast HTTP responses with the details of our multicast HTTP/QUIC streams.
- As part of this advertisement we need to be able to signal QUIC session parameters that would normally be negotiated between a QUIC client and a QUIC server at connection establishment.

Unicast request:

```
GET /video/segmentN.m4s HTTP/1.1
Host: representation1.media.example.org
```

Unicast response:

```
HTTP/1.1 200 OK
Content-Type: text/html
Alt-Svc: hqm="[ff3e::1234]:2000"; ma=7200;
source-address="2001:db8::1"; quic=1; session-id=10;
session-idle-timeout=60; max-concurrent-resources=10;
peak-flow-rate=10000; cipher-suite=1301;
key=4adf1eab9c2a37fd
```

RFC 7838 key fields

- ALPN protocol ID
- Alternative host
- Port number
- Maximum age parameter

Scalable media delivery on the Web with HTTP Server Push

- Public White Paper on these topics
- https://www.bbc.co.uk/rd/publications/whitepaper336

Scalable media delivery on the Web with HTTP Server Push

Lucas Pardue

2018 Apr

White Paper WHP 336 Apr 2018

Download

Abstract

Version 2 of the Hypertext Transfer Protocol (HTTP/2), published in 2015, introduced a Server Push protocol primitive, offering the possibility of improving user-perceived web page performance through the unsolicited delivery of additional dependent resources by a web server in response to an initial request from a client. Analysis of deployments in the wild has shown actual performance gains to vary greatly, and even reduced performance in some cases.

This paper explores Server Push from the perspective of Web Application logic implemented using JavaScript and the Web Platform. Current web browsers hold pushed resources in a so-called "Push Cache", where they exist in effective purgatory until explicitly requested. Server Push is, in essence, hidden from the Web Application.

We contend that the failure to expose Server Push events to the Web Application layer has impeded the realisation of promised performance improvements. Furthermore, hiding Server Push restricts a new set of use cases that would benefit from a reactive approach to web-oriented HTTP delivery of resources, in particular, unidirectional flows such as long-lived bulk data delivery and low-latency delivery.

Wrap up

- We think there are compelling Use Cases for push mode interactions.
  - A standard reactive API in the Web Platform, available to userland, can accelerate realisation of these.
- Non-browser clients have strong capability for push mode interactions.
  - Feature parity provides producers and vendors with flexibility, which can enhance innovation.
- We think that Server Push is particularly well suited to our high-level goals, it avoids the efforts of defining and implementing custom sub-protocols.
  - However, an API that surface mode events is not coupled to Server Push, it could abstract the transport delivery technology.
- We are also interested in understanding how to enable true multicast reception in browsers, in a way that is compatible with security considerations.

Thank you

bbc.co.uk/rd

Email:

lucas.pardue@bbc.co.uk

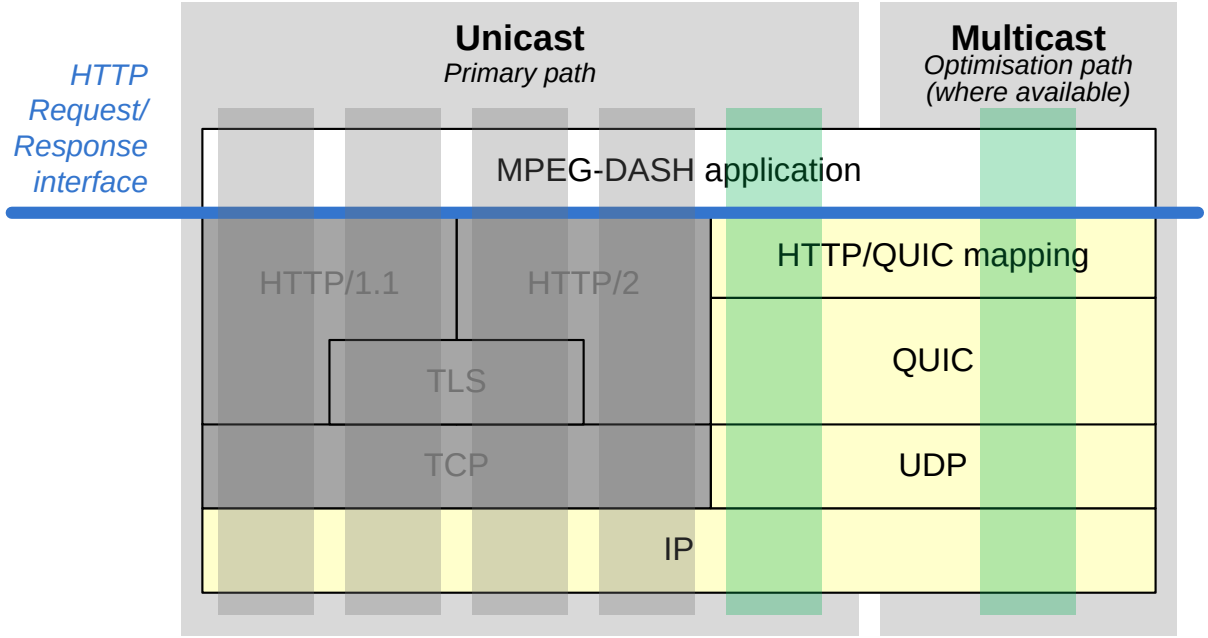
Twitter

@SimmerVigor

BBC | Research & Development

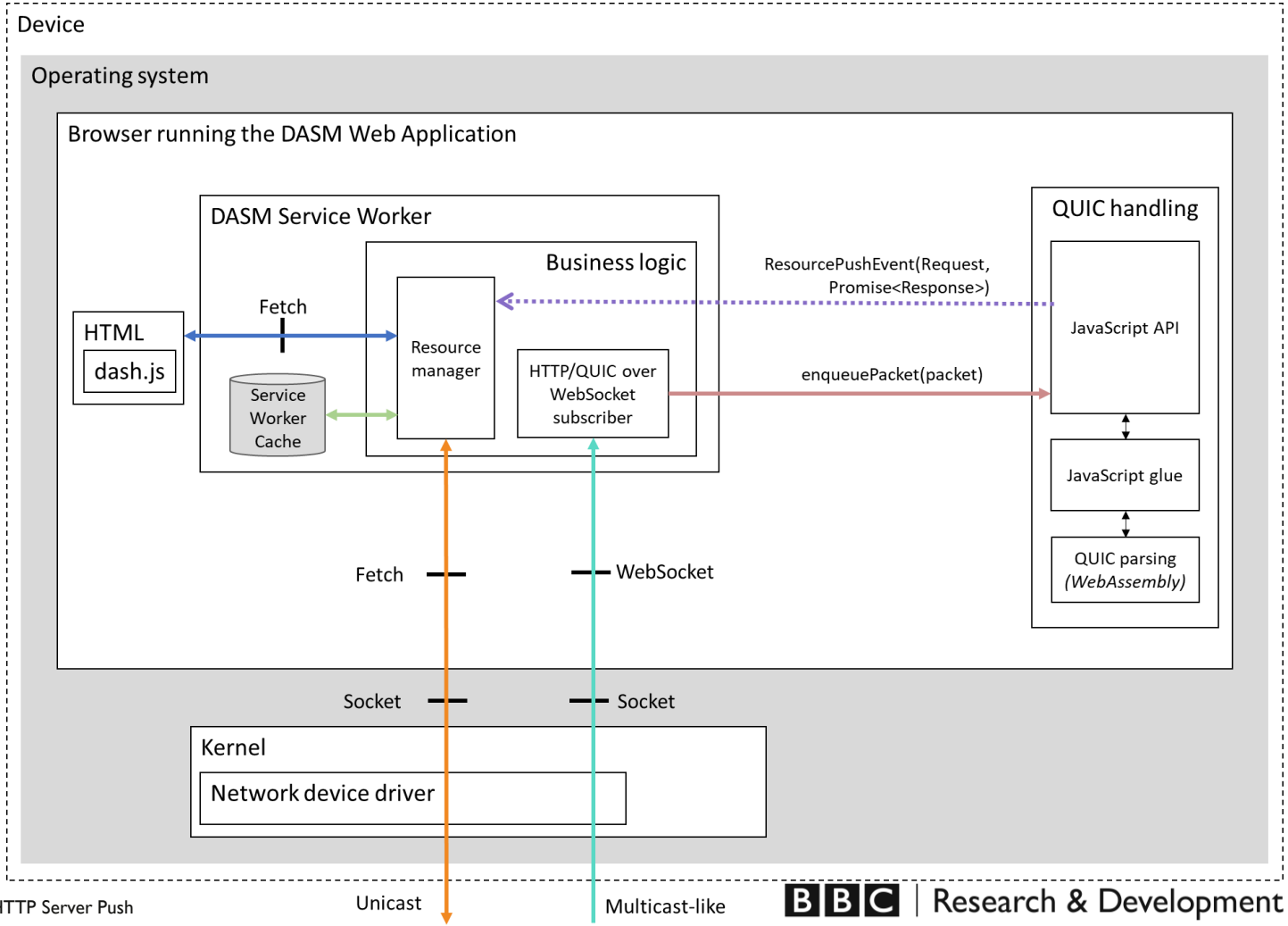
Backup slides

Protocol stack: Common layer 7 and layer 4



- HTTP provides a well-understood request/ response abstraction.
- Conventional (unicast) QUIC supports this abstraction by means of an HTTP/QUIC mapping layer.
- We have adapted this to multicast usage, so we end up with a common Layer 7 semantic and the same URL.
- At Layer 4, the QUIC datagrams also share a common packet format.

Browser web application detailed view



HTTP/QUIC deserialisation

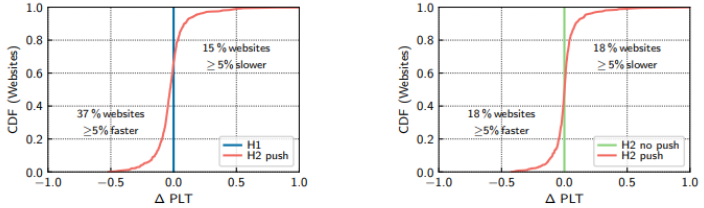
- Delivering QUIC packets into the Web Application.
- We need to deserialise them back into HTTP messages.
  - No support from the browser core to do this.
- Port our existing C/C++ implementation to the browser environment using **emscripten.**
  - Generates performant code in the form of **WebAssembly.**
- JavaScript "glue" layer to allow interaction with web applications.
  - Server Push API (`ResourcePushEvent`).
  - Expose an `enqueuePacket()` function that lets us inject QUIC packets.

Server Push web page performance impact I

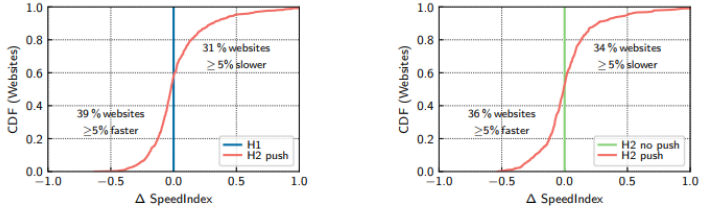
- Large-scale public results are hard to come by.
- Torsten Zimmermann presented some data at IETF 101.

Analysis of performance impact of Server Push

- **January 2017, automate Chrome to repeatedly visit websites**
  - 16 Mbit/s Down, 1 Mbit/s Up, 50 ms symmetric delay
  - Page Load Time: time between `connectEnd` and `loadEventStart`



  - SpeedIndex: measures how quickly page contents are visually populated

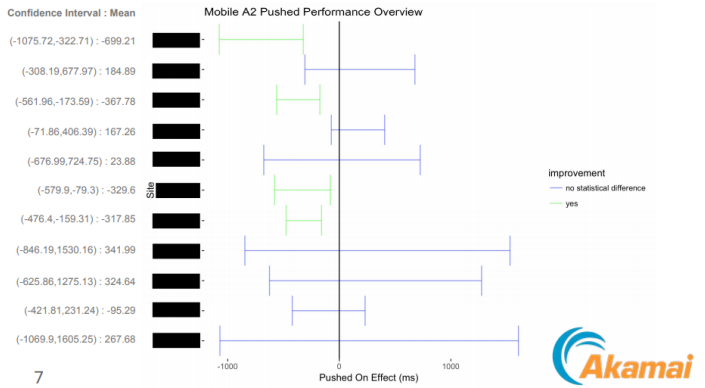


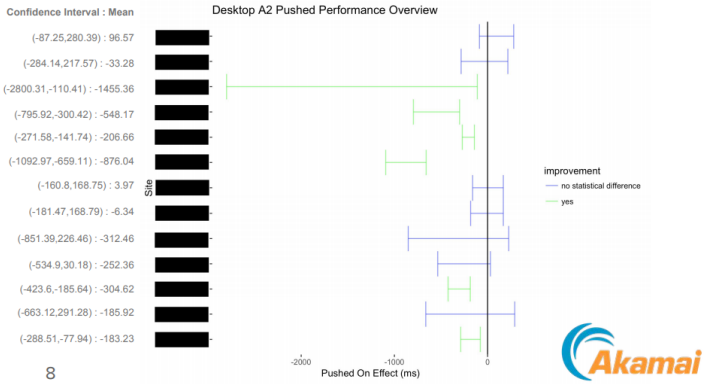
12 Torsten Zimmermann https://netray.io COM SYS RWTH AACHEN UNIVERSITY

https://datatracker.ietf.org/meeting/101/materials/slides-101-maprg-adoption-human-perception-and-performance-of-http2-server-push-00

Server Push web page performance impact 2

- Akamai presented some data at IETF 102:
  - *"Using RUM data to find critical resources … Push those critical resources during the HTML generation "think time""*
  - *"Of the results where there is a statistical difference, we are seeing positive improvement with H2 Server Pushes applied"*
  - *"Mobile performance data has more variation and noise, resulting in higher uncertainty (longer bars)"*

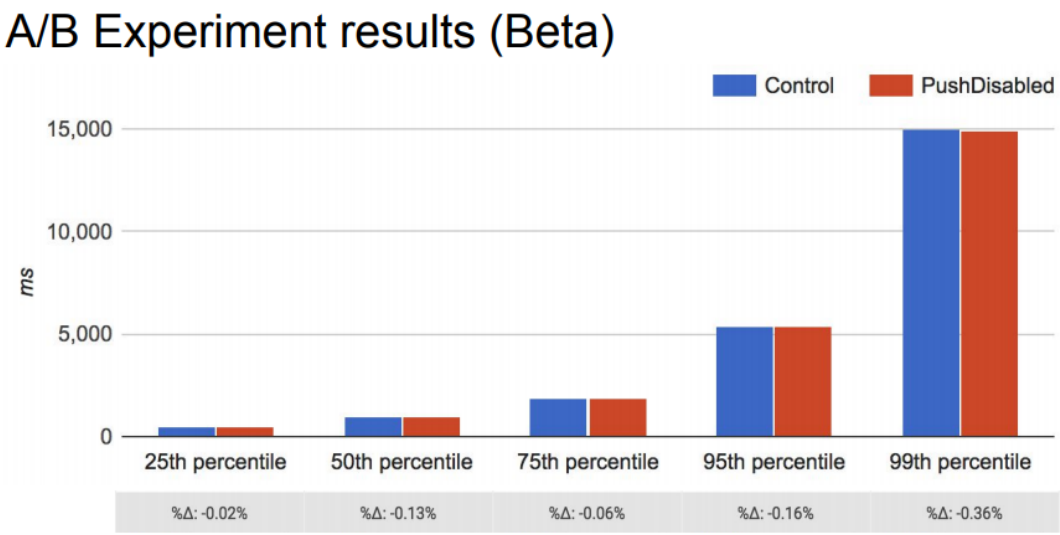




https://datatracker.ietf.org/meeting/102/materials/slides-102-httpbis-akamai-h2-server-push-performance-00

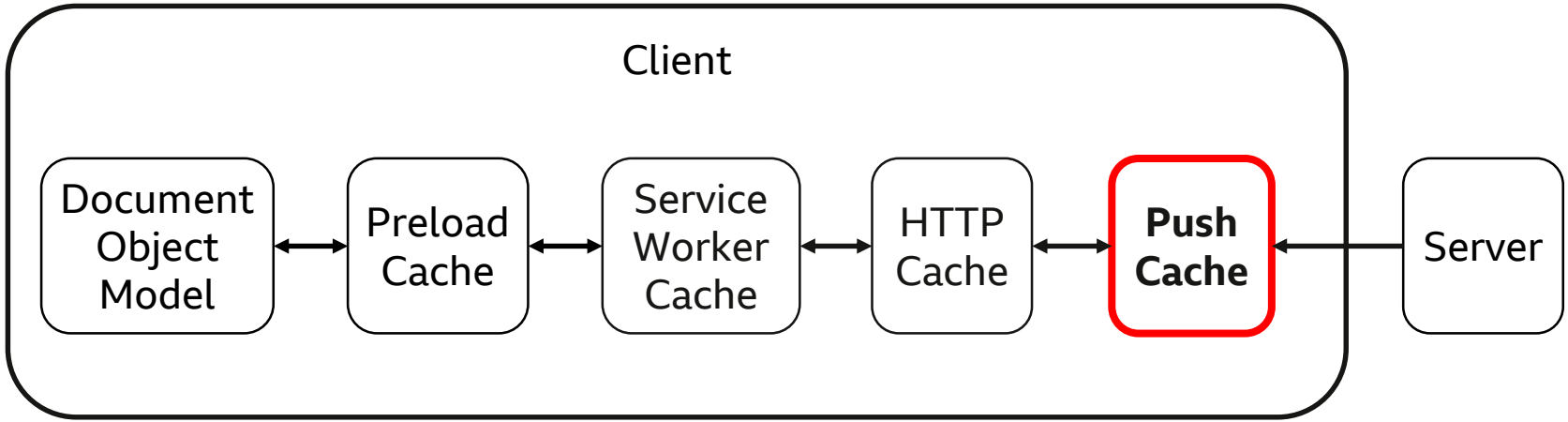
Server Push web page performance impact 3

- Google presented some more data at IETF 102:
  - *"0.04% of HTTP/2 sessions have a push frame"*
  - *"The average amount of pushed data in a session is 32kb"*
  - *"63.51% of pushed streams are accepted"*
  - *"22.35% time out"*
  - *"13.39% are duplicate URLS"*



https://datatracker.ietf.org/meeting/102/materials/slides-102-httpbis-chromes-view-on-push-01

Opaque Server Push



Service Worker cache