# CacheCatalyst

*Enhancing Web Caching for the Latency-Constrained Internet*

*Presented by*
**Hannaneh B. Pasandi** (UC Berkeley)

**NSDI 2026**

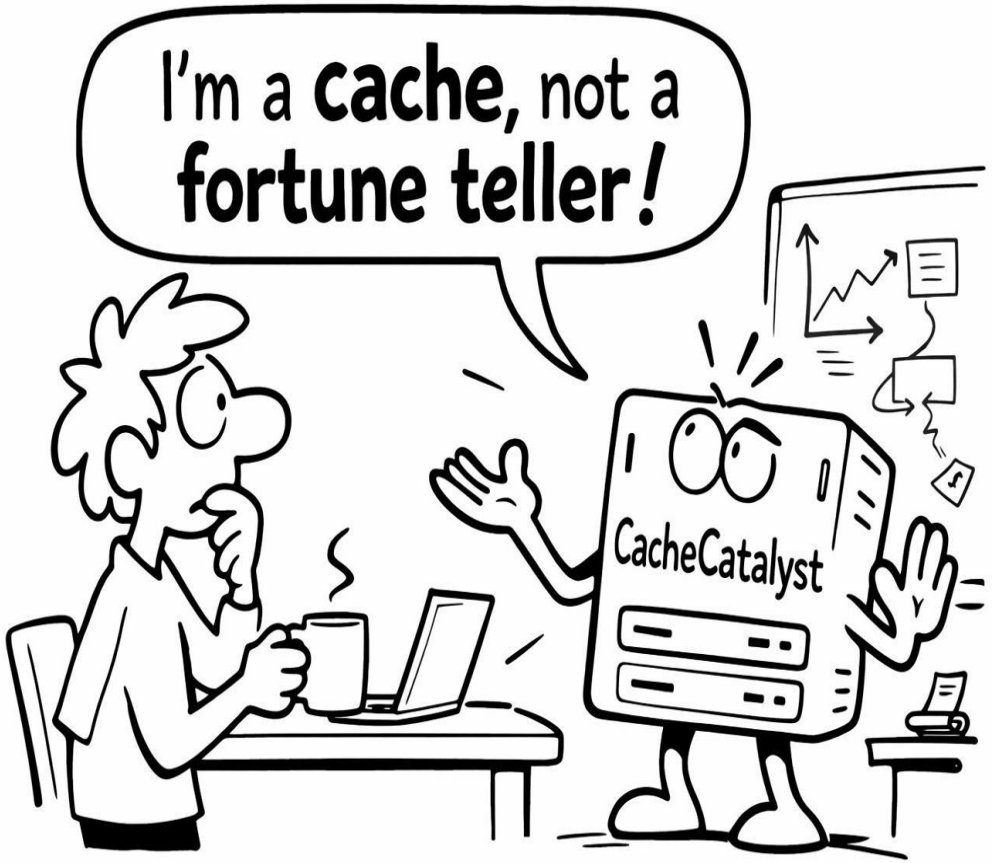
**Mohammad Hosseini** (Shahid Beheshti) **Sina Darabi** (USI) **Patrick Eugster** (USI) **Mahmood Choopani** (Shahid Beheshti)



# Caching is what makes the web feel fast

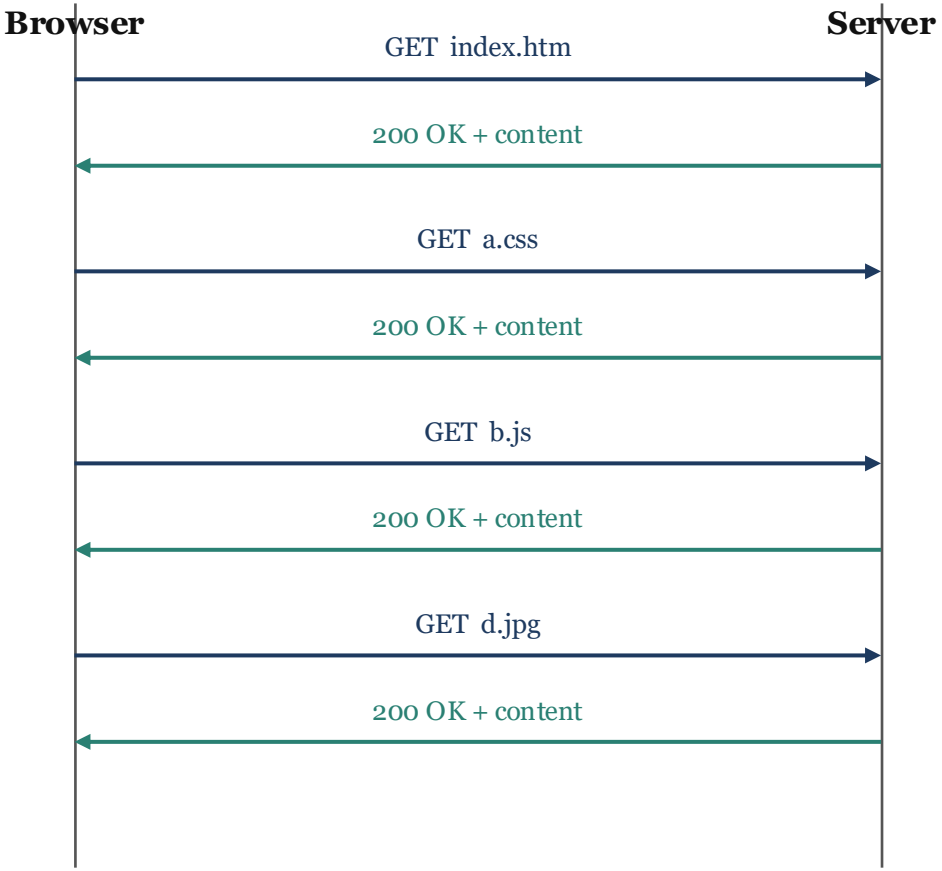
When you click a link to a page you've visited before, your browser reuses dozens of resources from a local cache.

***So if everything is already cached, why does a revisit still take so long?***



# First visit versus revisit

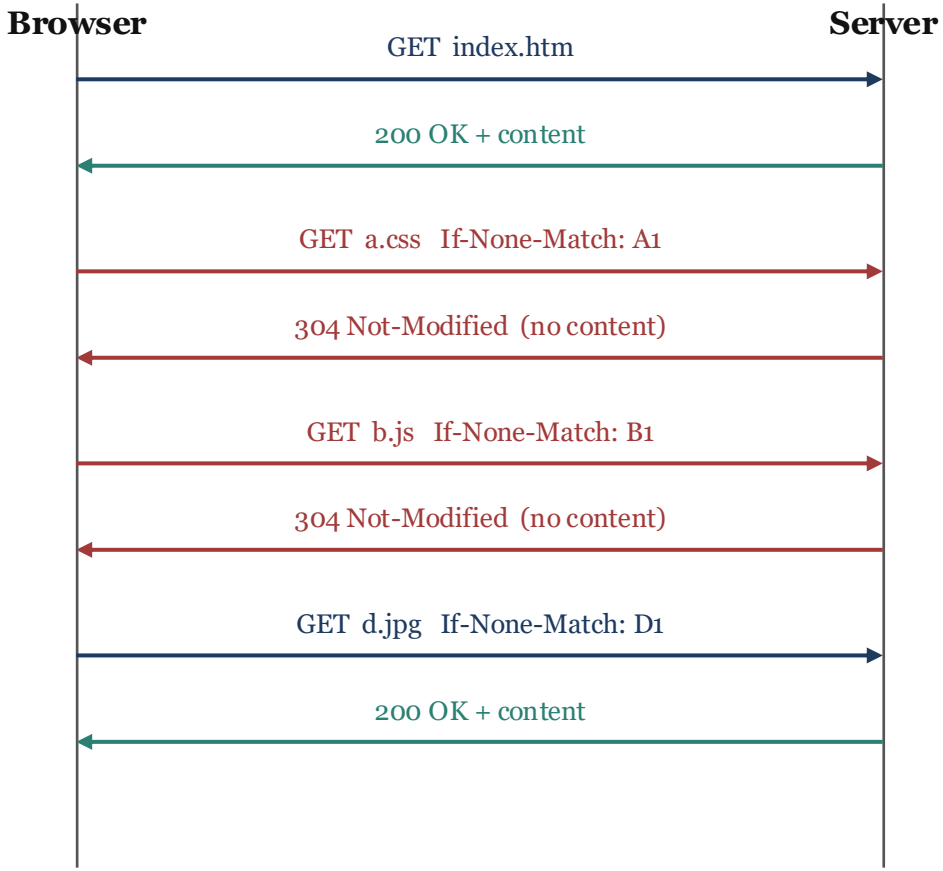
## First visit




*Every response carries useful content.*

*RTT + download*

## Revisit




***Most responses are empty. The browser already had the content.***

*RTT + no content download*

# What a good solution needs

## 1. Cut the round trips on revisits

The browser should not have to ask permission resource by resource.

## 2. Do not waste bandwidth

Do not send what the client already has.

## 3. Preserve end-to-end Security

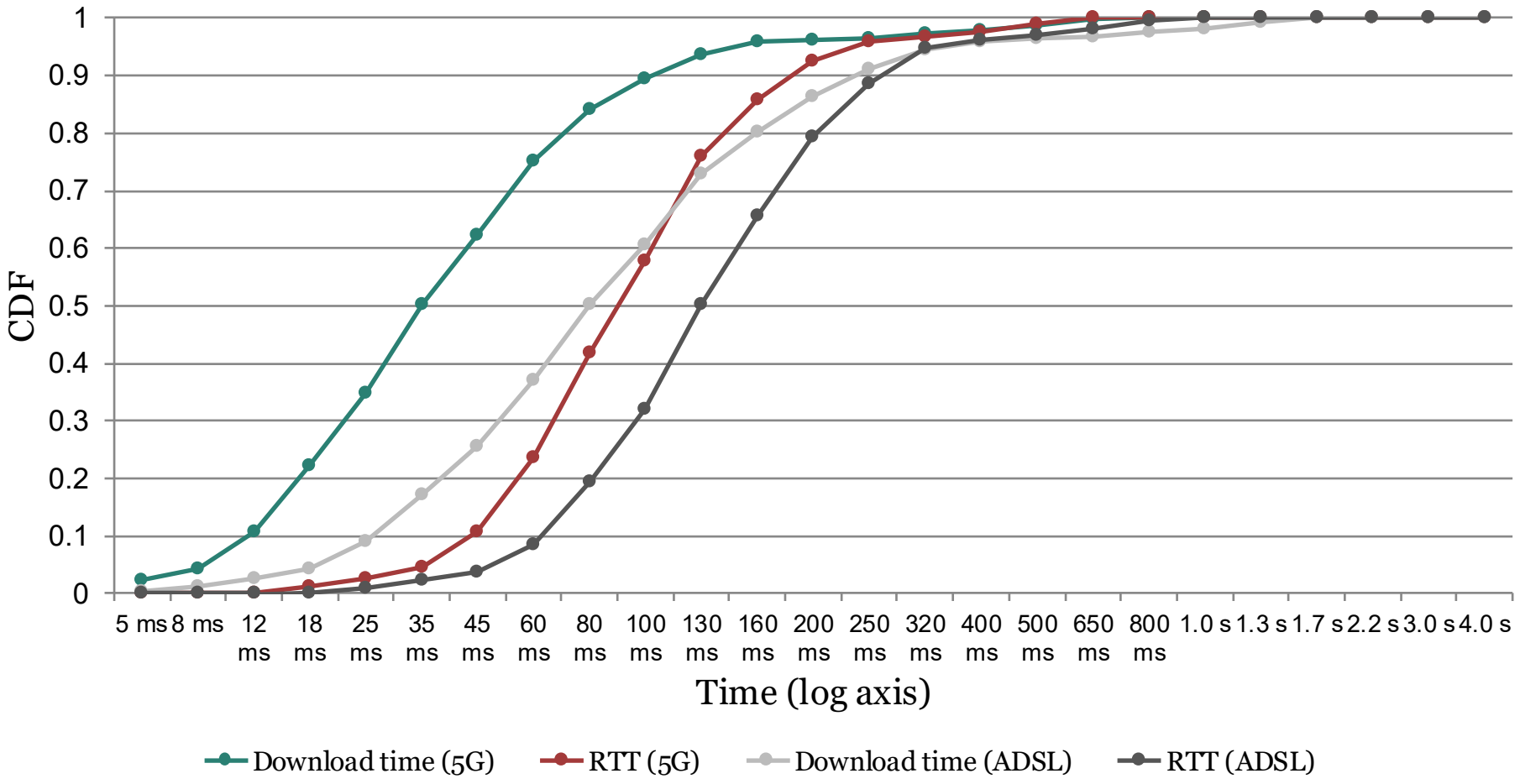
The encrypted channel should still run from the user's browser to the origin server, with no third party reading the traffic in the middle.

# None of the existing approaches solve all three

| Approach | Cuts revisit RTTs | No bandwidth waste | End-to-end TLS preserved |
|---|---|---|---|
| Standard HTTP cache | no | yes | yes |
| Conservative TTLs | no | no | yes |
| HTTP/2 Server Push | yes | no | yes |
| **CacheCatalyst** | yes | yes | yes |

*CacheCatalyst is the first to clear all three columns*

# Latency, not bandwidth, is the bottleneck

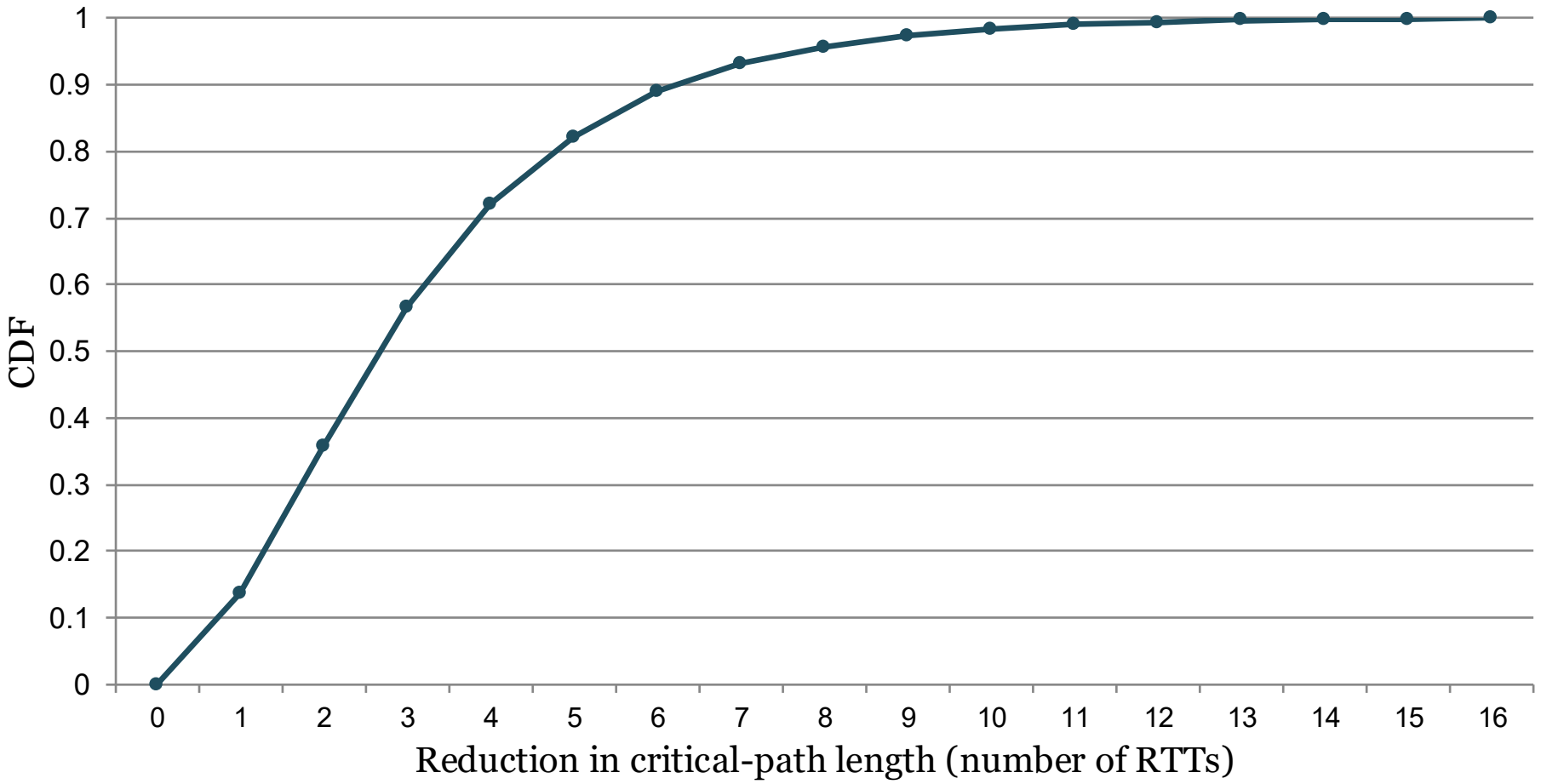



**On 5G,**
median RTT is

**2.6x**

the median download time

# Critical paths shrink if those round trips vanish




**Average critical path**

**6.7 RTTs**

**Projected PLT reduction**

**56%**

*if Not-Modified RTTs disappear*

**If most resources have not changed,**

**why does browser keep asking?**

---

**Validate everything once,**

**at the start of the page load.**

***CacheCatalyst key idea***

# CacheCatalyst piggybacks the validation step


*One request returns the page and the validation status of every cached resource*

# CacheCatalyst in 3 steps

1

## Pre-validate

Browser piggybacks ETags of cached resources on the HTML request. Server replies with a fresh-or-stale flag for each.

2

## Decide locally

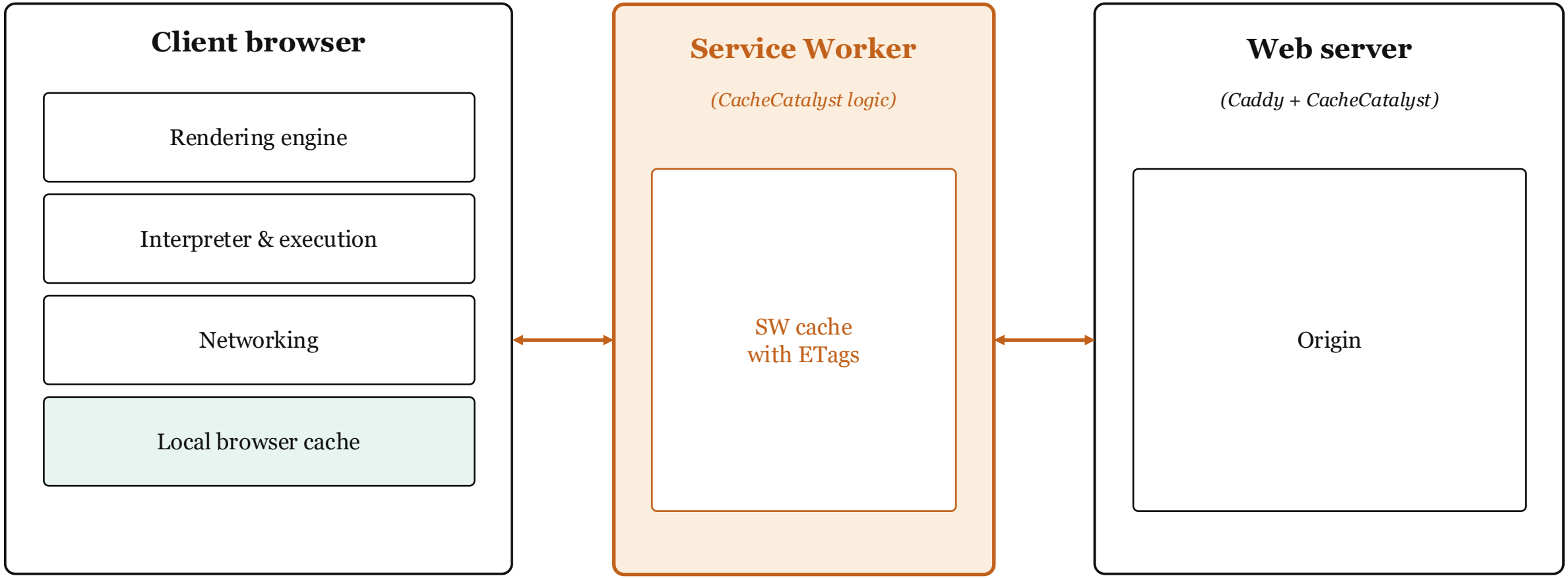
Browser uses fresh entries straight from cache. No extra revalidation requests, no 304 round trips.

3

## Push deltas

Cache-aware push sends only resources the client truly lacks. Avoids the bandwidth waste that killed Server Push.

# We deploy CacheCatalyst with a Service Worker




*Browser-side compatible. Server-side runs on a modified Caddy. No browser changes required.*

# Evaluation setup

**Corpus** 10,000 pages from the top 100K websites **Server** modified Caddy **Client** Chrome via Selenium and Lighthouse

## Metrics we measured

| Acronym | What it measures | Why it matters |
|---|---|---|
| **PLT** | Page Load Time | Click to last byte |
| **FCP** | First Contentful Paint | First text or image visible |
| **LCP** | Largest Contentful Paint | Page looks essentially done |
| **CLS** | Cumulative Layout Shift | Visual stability while loading |
| **SI** | Speed Index | How quickly content fills in |
| **TTI** | Time to Interactive | User can actually click |

## Network conditions (12 settings)

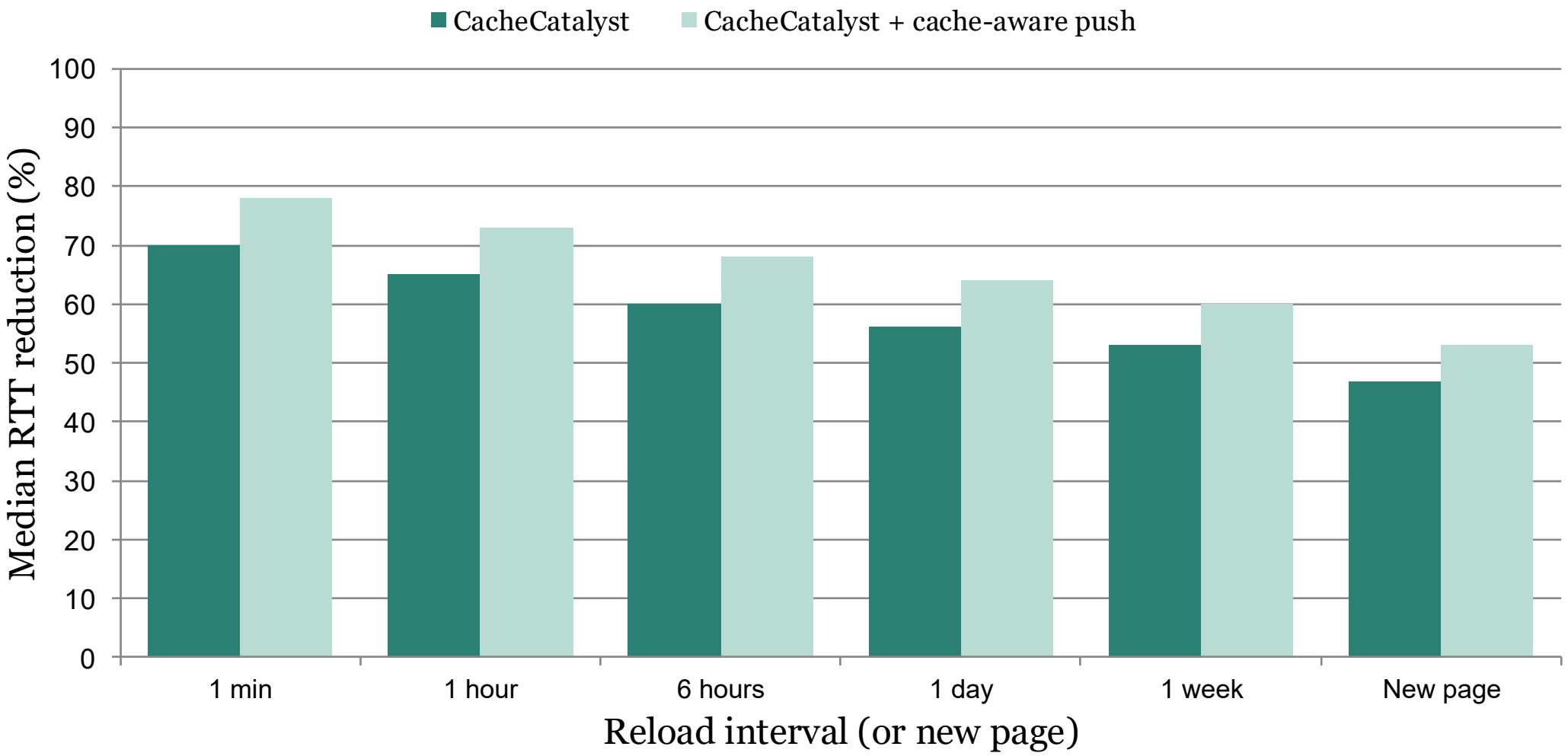
| | 40 ms | 100 ms | 200 ms |
|---|---|---|---|
| **0.5 Mbps** | ● | ● | ● |
| **8 Mbps** | ● | ● | ● |
| **20 Mbps** | ● | ● | ● |
| **60 Mbps** | ● | ● | ● |

*Bandwidth from dial-up to fast 5G*

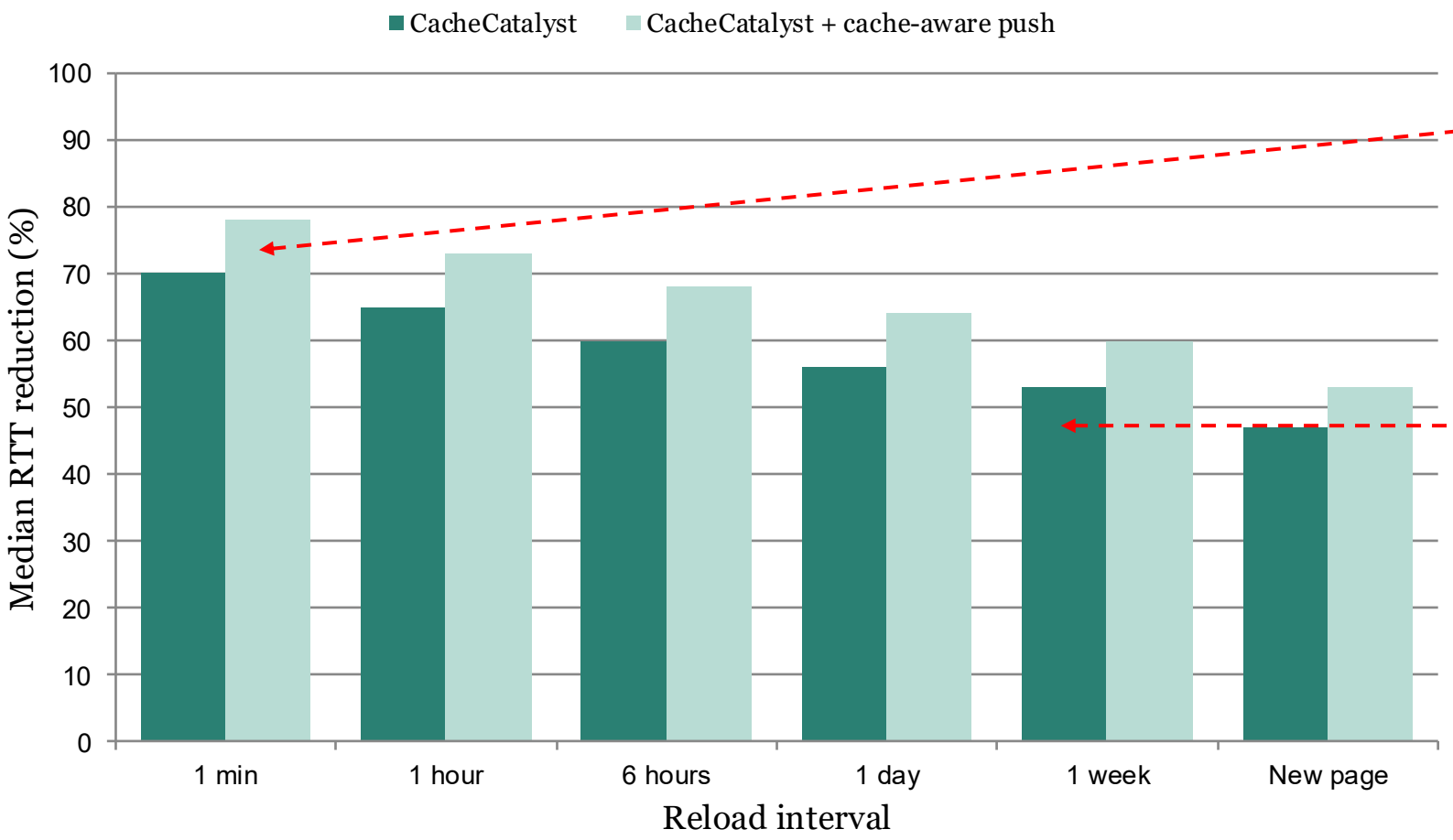
*Latency from fast home to slow mobile*

**Reload delays** 1 minute, 1 hour, 6 hours, 1 day, 1 week, plus a new page on the same site

# CacheCatalyst eliminates most revisit RTTs



# Three numbers to take away




**78%**

*best case*
at 1 minute, the most common revisit

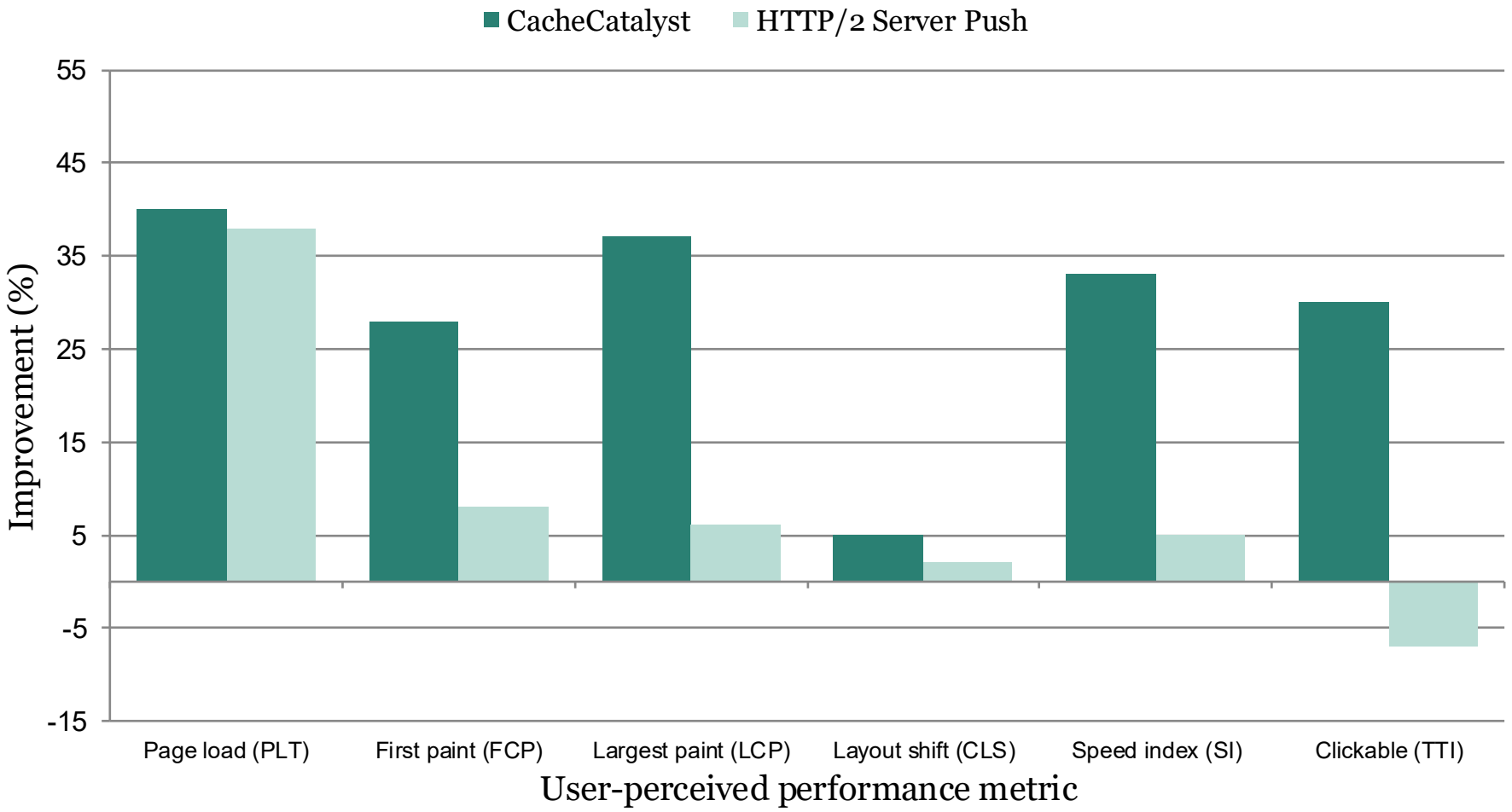
**53%**

*worst case*
at 1 week, the least-median across intervals

**4.3**

*average*
RTTs eliminated per page

# CacheCatalyst vs. Server Push




**Median improvement**

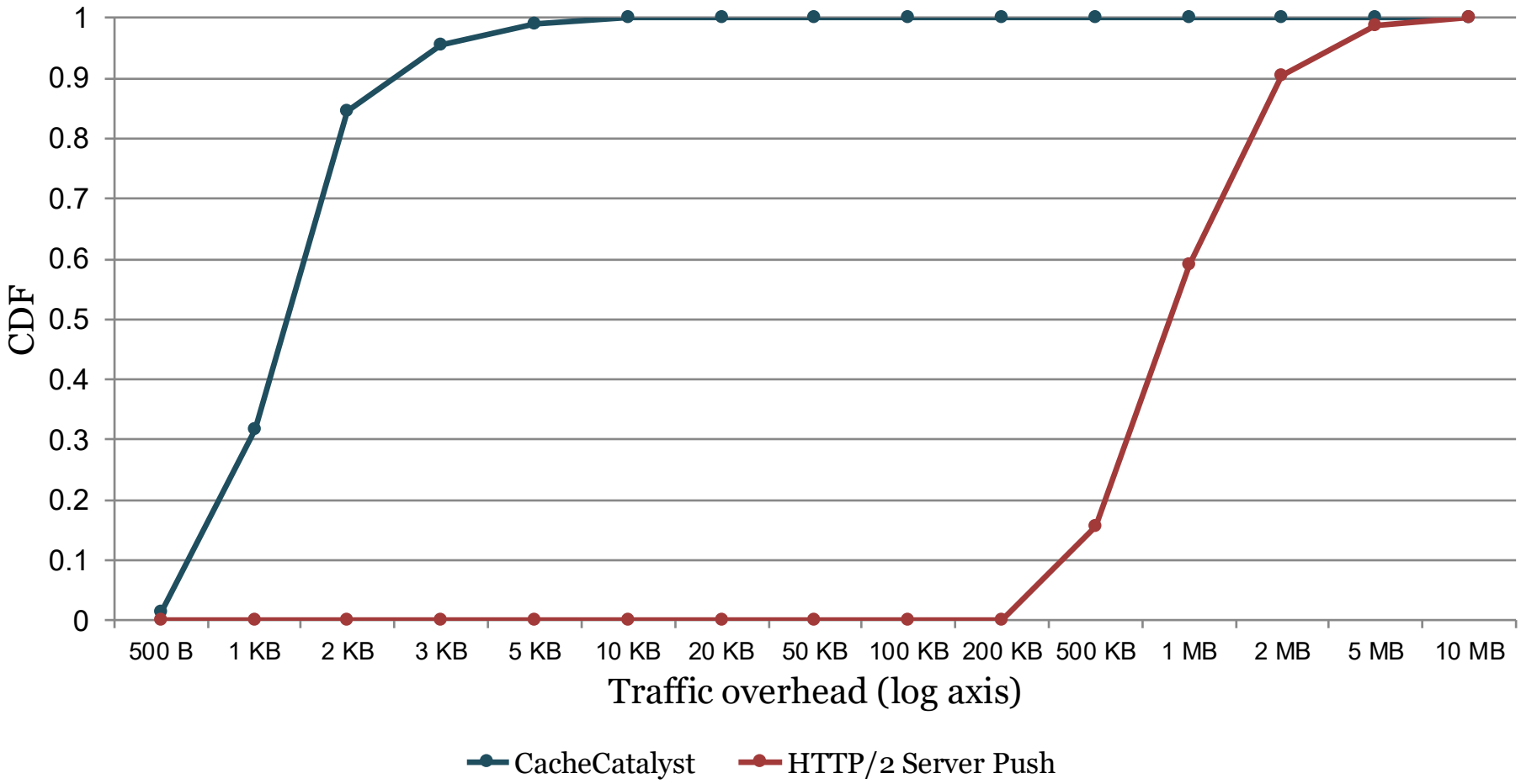
**PLT** 40%

**FCP** 28%

**LCP** 37%

*Server Push hurts FCP, LCP, and TTI in many cases*

# Overhead is three orders of magnitude lower




**Median overhead**

**CacheCatalyst**

**1.5 KB**

**HTTP/2 Server Push**

**1 MB**

# CacheCatalyst, the takeaway

1. Latency, not bandwidth, is what slows revisits. The 304 round trip is the worst offender.
2. Validate the whole cache once, on the HTML request. Decide locally for everything else.
3. 40% PLT, 37% LCP, 28% FCP. Three orders of magnitude less overhead than Server Push.

**Open source** github.com/toorin-lab/CacheCatalyst

*includes the modified Caddy and the Service Worker reference implementation*

# Limitations

1. **New page navigation, partial coverage**
2. **CORS on script-fetched resources**
3. **Service Worker integration on existing sites**

*CacheCatalyst still wins substantially in every experimental condition we tested.*

# Thank you

*Questions?*