codavel

Fast Mobile Apps
Regardless of your user's network

The only CDN putting a stop to unstable
user experience caused by the wireless links instability.

The reality of typical mobile user experience is far from smooth

Buffering happens often

Half of the video sessions get buffering

From Mux

Experience is highly unstable

Almost half the users have to wait for more than 11s to start video, at least once

From Snapchat and our data

Uber is seeing high latency everywhere

From Uber

Legacy CDN

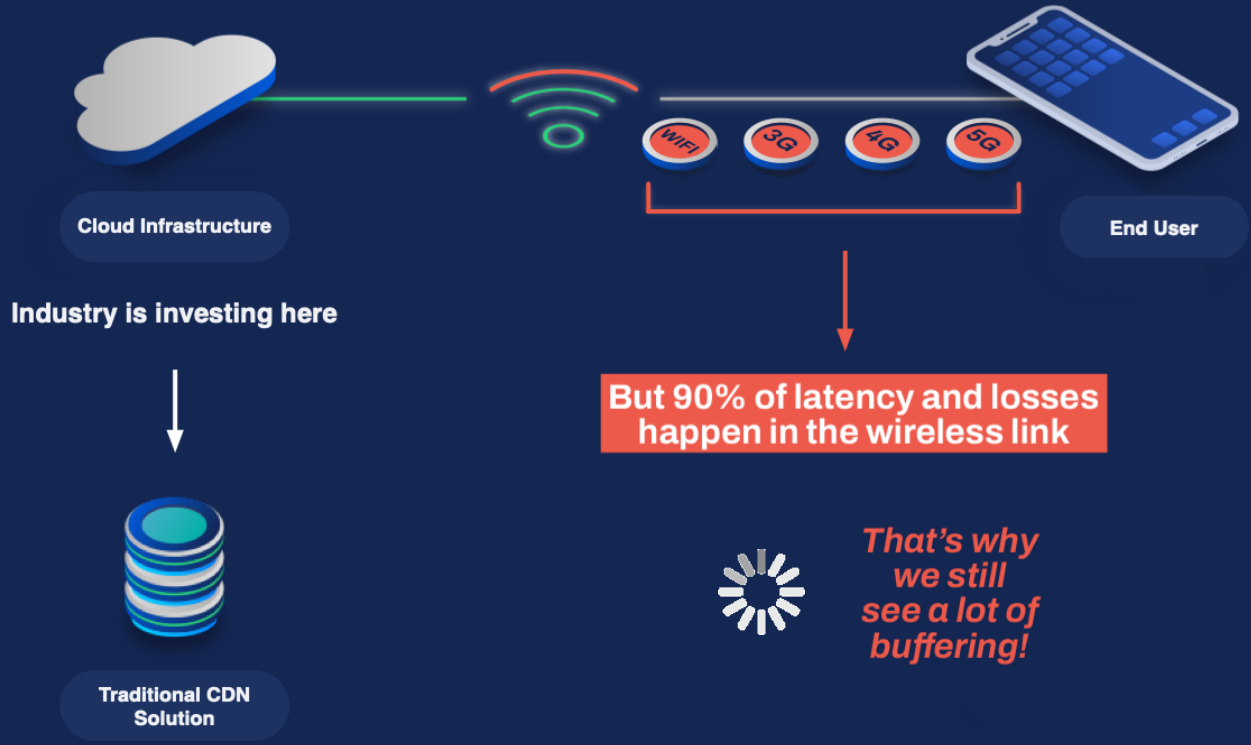
Lower user engagement

Less time spent in app

Less ads showed

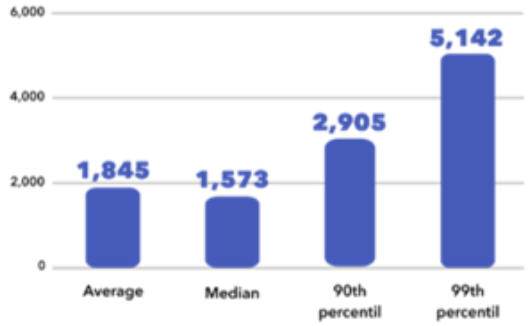
Less revenue

Industry is putting a lot of effort, but...

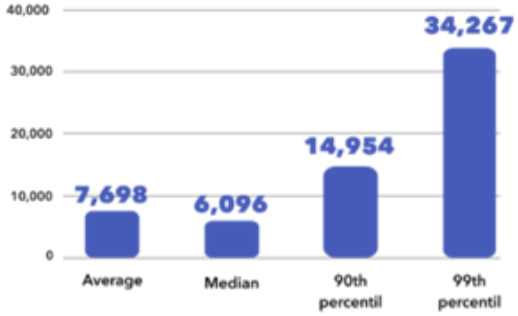


The devil hides in the tail

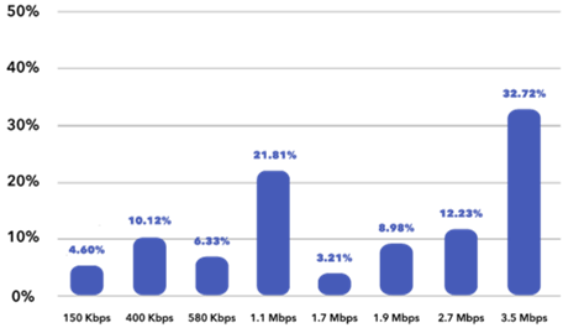
Video Startup Time (ms)



Rebuffering Time (ms)



Video Quality (Mbps)



Source 1, Source 2

The tail is way more common that one would expect:

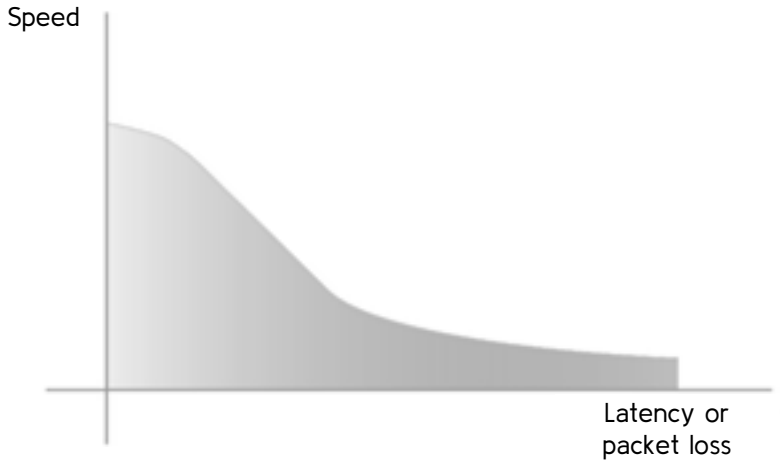
Snapchat: **more than 50% of the users get a percentile 99% experience at least once a week**

"Humans do not perceive the commonplace; they do not forgive you because your average is good... pain and bad experience is what you remember". Gil Tene

Wireless links are jeopardizing user engagement. Why?

Wireless is way different than wired

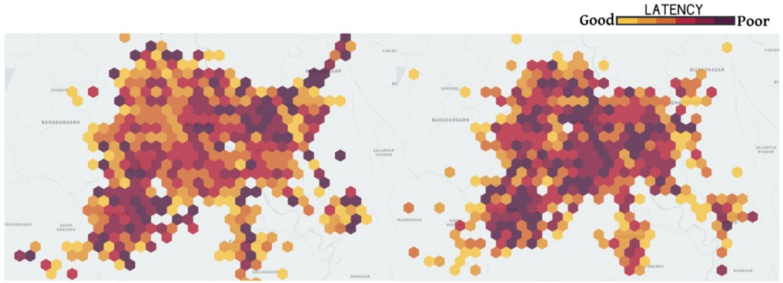
Last mile Tech	Latency	Packet Loss
Fiber	5ms	0.01%
WiFi	25ms	0.20%
4G	50ms	0.50%



0.50% of packet loss slows down TCP by 5 times

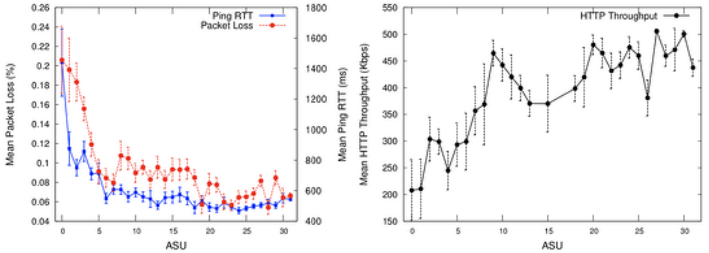
Wireless is highly unstable

Different carriers, different performance



Signal Strength

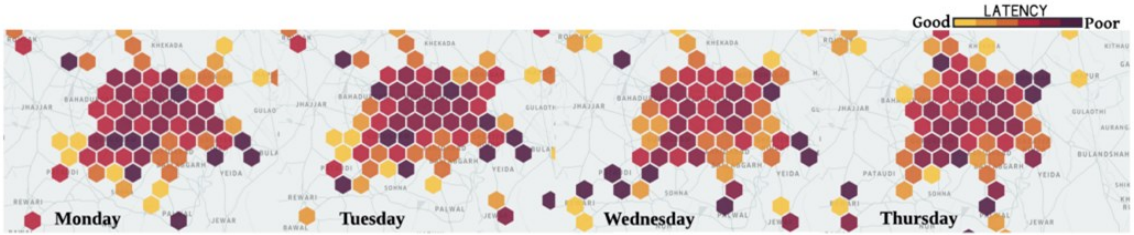
How much it can affect performance?



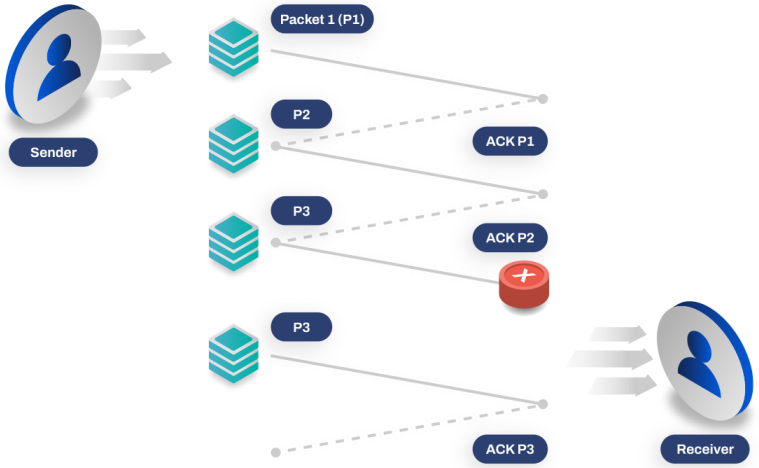
(a) Ping RTT and Packet Loss (b) HTTP Throughput

AT&T HSDPA (Seattle), Arbitrary Strength Units (ASU)

Same carrier, different day, different performance



Why HTTP/TCP sucks for wireless

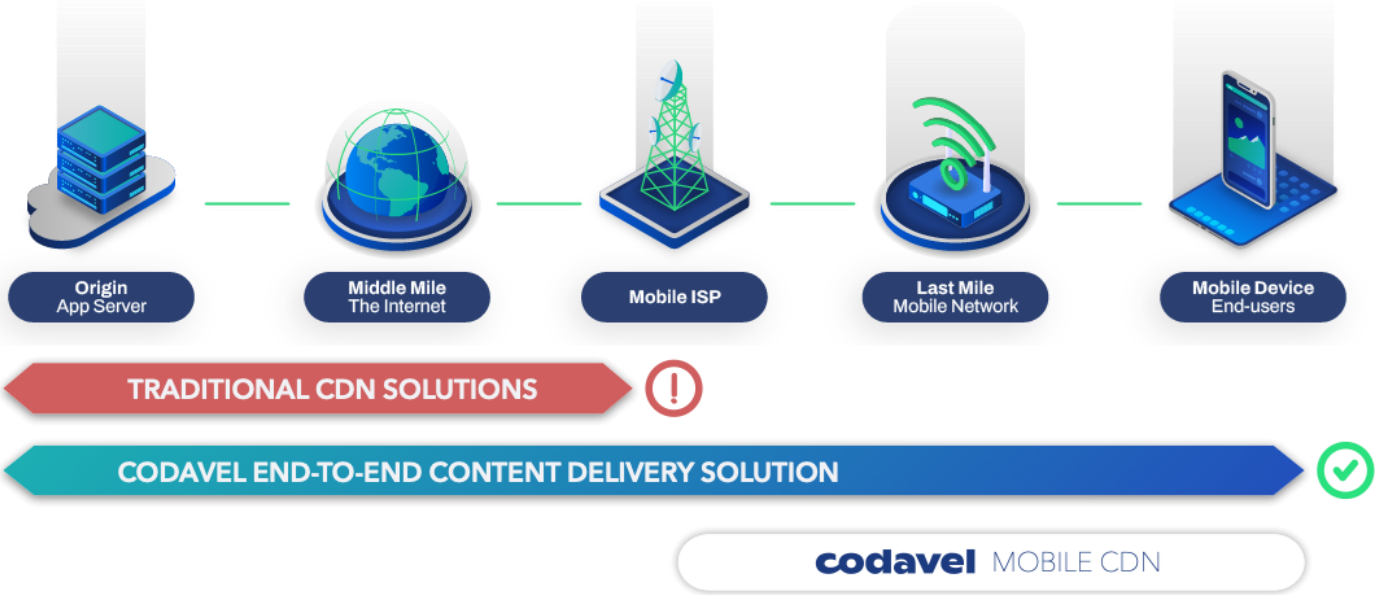


Packet retransmissions (**ARQ**) can be **highly inefficient in the presence of latency**:

Need to wait for ACKs before advancing the transmission window

PLUS: TCP uses **packet loss as a signal of network congestion**, which is not at all the case over wireless

Codavel is a unique end-to-end content delivery solution



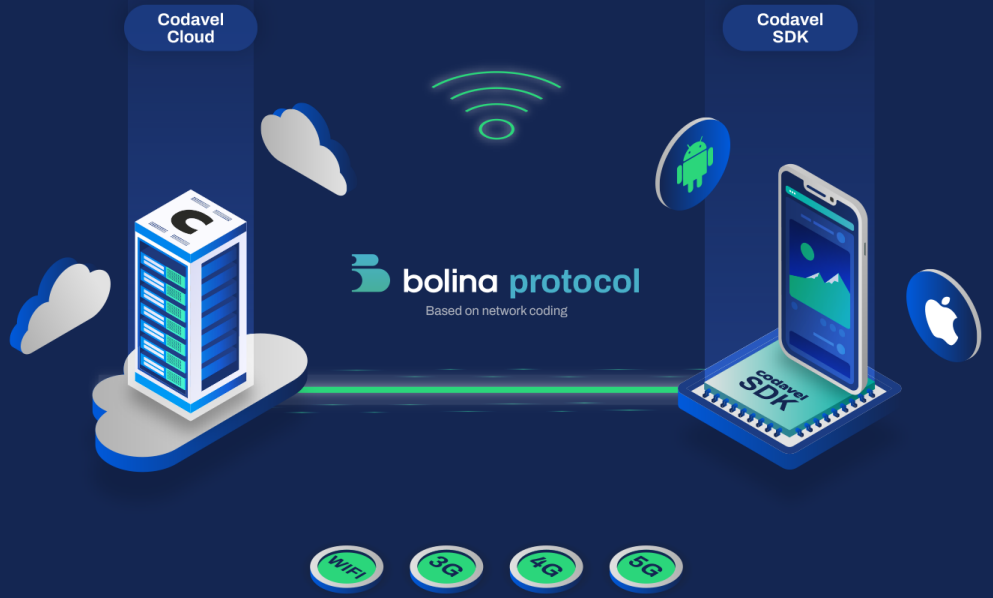
Codavel controls user experience end-to-end

Mobile-first CDN
End-to-end content delivery optimization

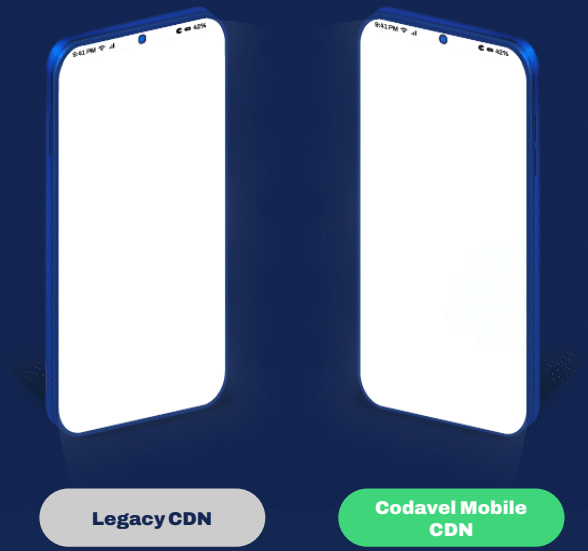
Resilient to latency
Uninterrupted user experience even under the worst network conditions

Easy to set up
All or selected traffic through Codavel with only a few lines of code to install Codavel SDK

Risk free
Automatic fallback included. Even if something goes wrong, your users won't be affected



What you get with Codavel



Impact on the average

Less rebuffering
- 55% rebuffering time

Faster startup
- 32% video startup time

Higher quality
+ 27% video quality

Improvements on the tail-end

+ 94% faster experience
for the 10% worst sessions

+ 116% faster experience
for the 5% worst sessions

+ 384% faster experience
for the 1% worst sessions

How Codavel achieves superior performance

What is Network Coding?

Network Coding: operating over data packets

Example: 3 packets to send:

- P1 = 0110
- P2 = 1110
- P3 = 1010

Traditionally, **we would transmit**:

1. P1
2. P2
3. P3

With Network Coding, we transmit:

- **P1 + P2** (= 1000)
- **P2 + P3** (= 0100)
- **P1 + P3** (= 1100)

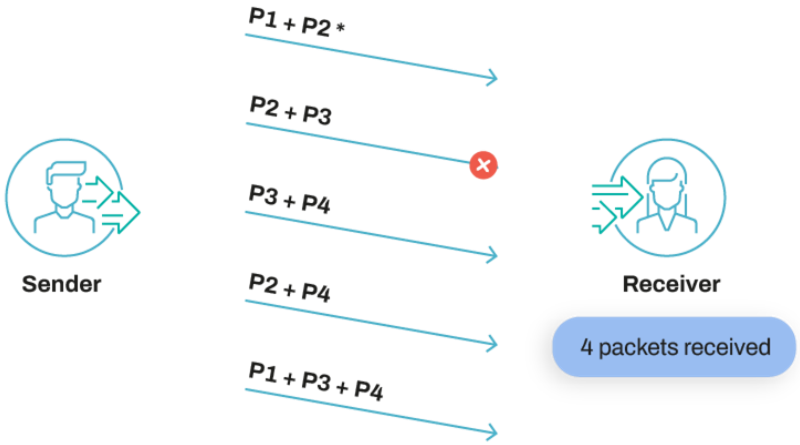
The **receiver gets linear system of equations to solve**, on P1, P2 and P3

1	1	0	1000
0	1	1	0100
1	0	1	1100

And is then **capable of computing the original packets**, P1, P2 and P3

1	0	0	0110
0	1	0	1110
0	0	1	1010

Why is Network Coding useful?



In general, the **receiver needs *n* successful transmissions to recover *n* original packets**, w.h.p., **irrespective of what transmissions were lost**

Receiver gets

- P1 + P2
- P3 + P4
- P2 + P4
- P1 + P3 + P4

And from that, **receiver decodes the 4 original packets in 4 successful transmissions**

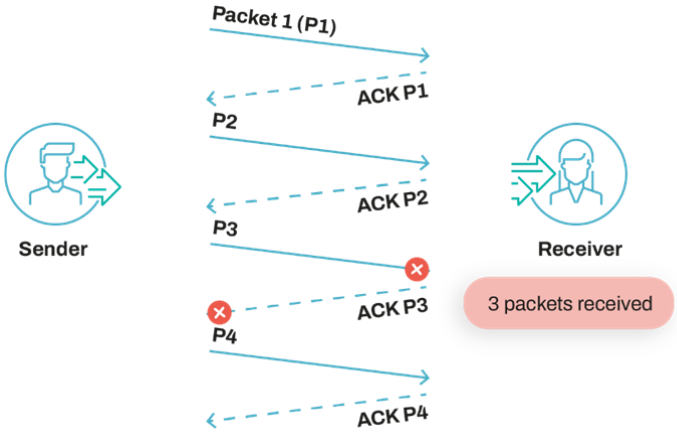
Network Coding achieves this with:

- **No need to know which packet was lost**
- **No need to wait for feedback** to overcome the loss

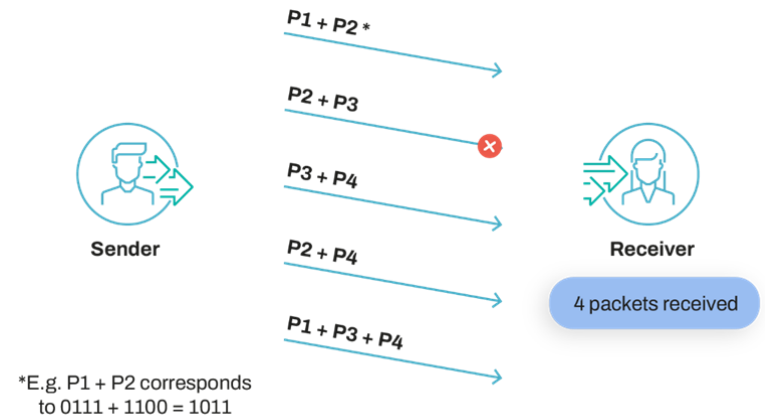
Network Coding for packet loss recovery

#1 Feedback is not necessary for packet loss recovery

Standard Protocols



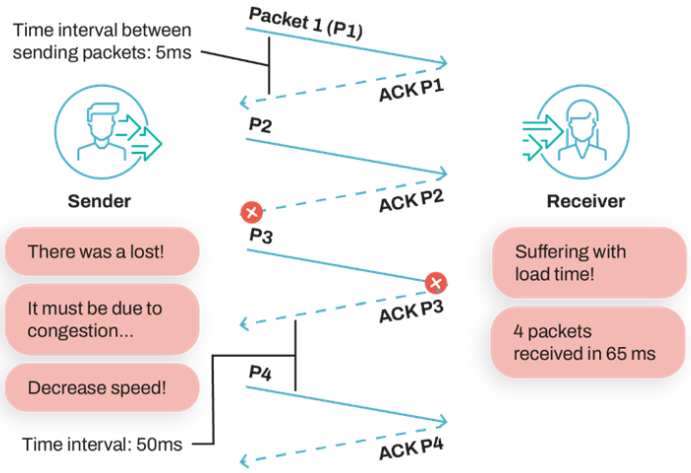
Network Coding Based Protocols



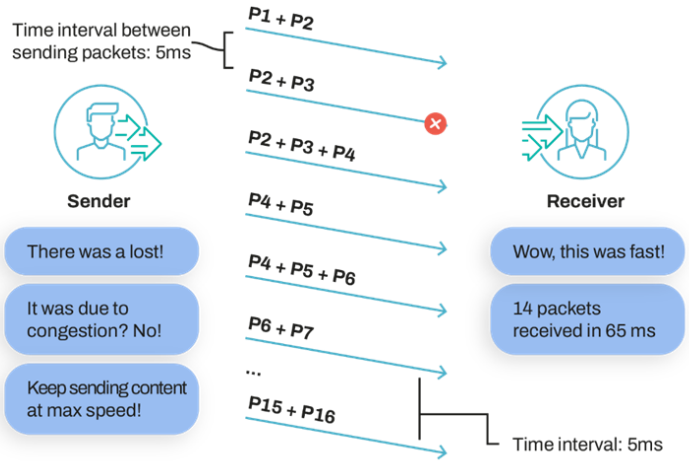
Network Coding for packet loss recovery

#2 Feedback is used to detect loss nature

Standard Protocols



Network Coding Based Protocols



Less Buffering

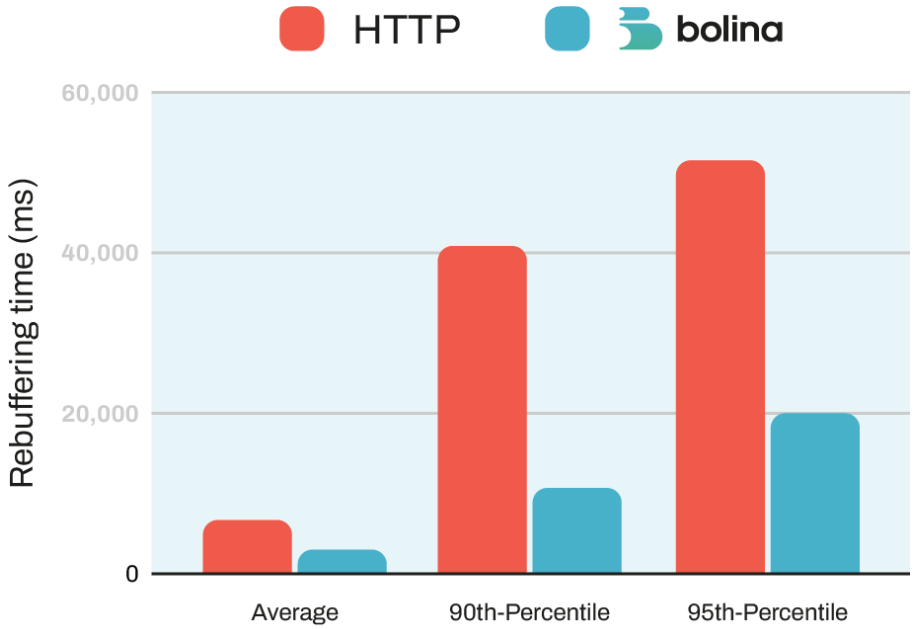
Use case: Video streaming in India
Data from real users

KPI: Rebuffering time
HLS stream with a single video quality (3.5Mbps)
Bolina vs HTTP (in AWS Cloudfront)

bolina improvements:

Average rebuffering time: **-55%**

90th-percentile rebuffering time: **-73%**



Faster Video Startup

Use case: Video streaming in India
Data from real users

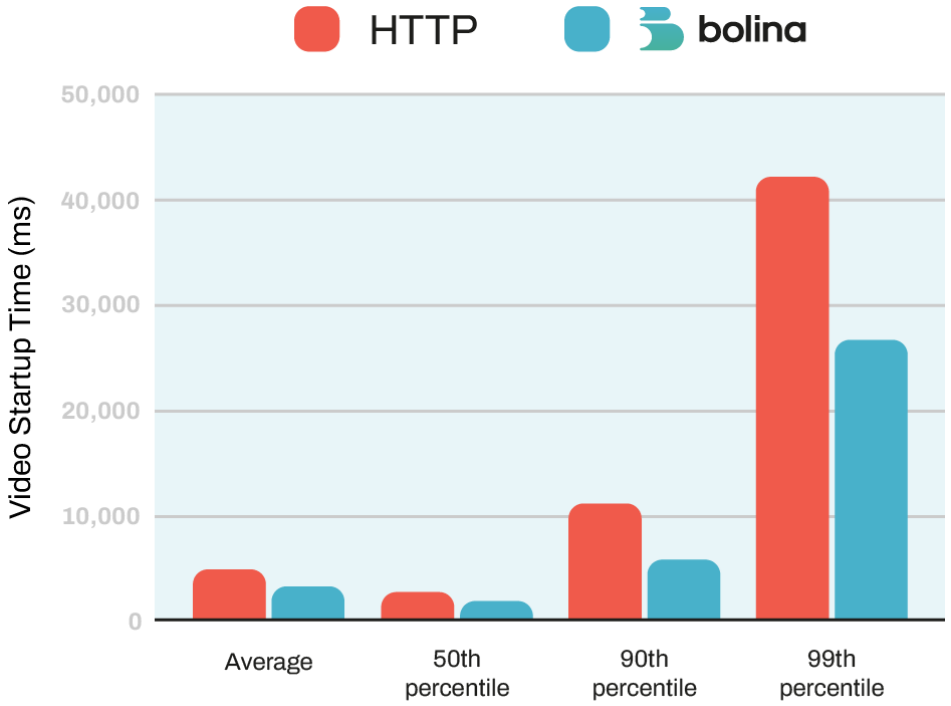
KPI: Video startup time
HLS stream with a single video quality (3.5Mbps)
Bolina vs HTTP (in AWS Cloudfront)

bolina improvements:

Average video startup time: -32%

Median video startup time: -26%

Video startup time **standard deviation: -64%**



Higher Video Quality

Use case: Video streaming in India
Data from real users

KPI: Video Quality

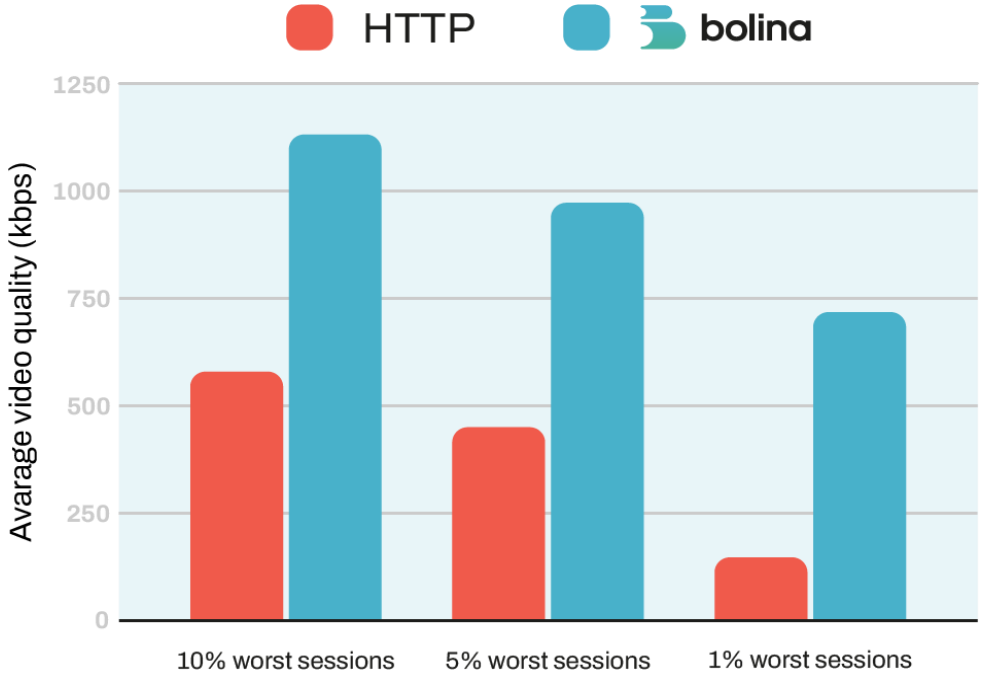
bolina improvements:

+93% higher video quality for the 10% worst sessions

+384% higher video quality for the 1% worst sessions

Median **video quality: +27%**

Time in **highest video quality: +17%**



Ready to speed up your mobile video delivery?

Rui Costa

CEO and Co-Founder

codavel.com

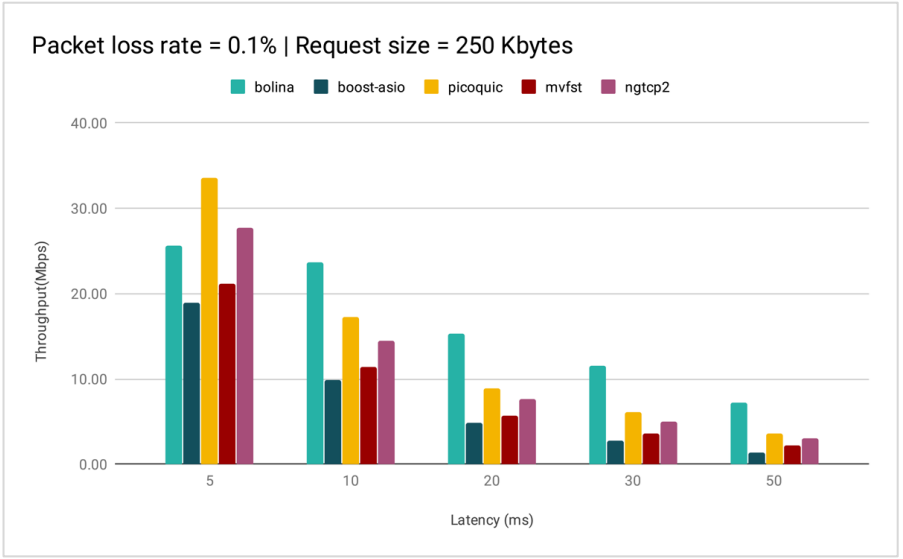
/in/ruifcosta/

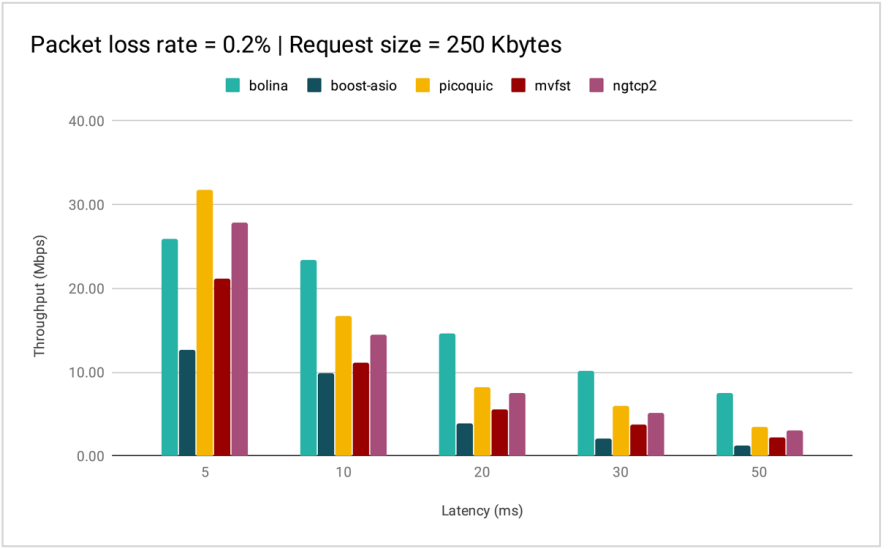
rcosta@codavel.com

APPENDIX

Bolina vs QUIC

Smaller requests





Bolina vs QUIC

Larger requests