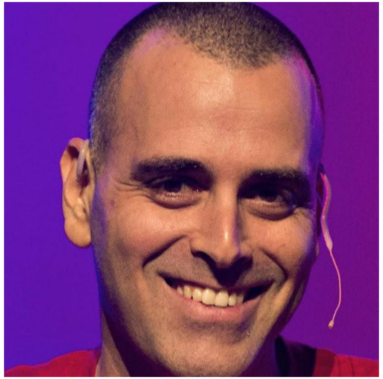
Network Quality Estimation in Chrome

Web & Networks Interest Group
Feb 2020

whoami

- Work on Chrome team
    - Area of networking, web page loading etc.
    - Focus on tail end where the web performance is really slow

Help from lot of folks



- Ilya, Yoav, Ben
- Chrome networking and loading team, devrels, privacy folks

NQE (Network Quality Estimator)

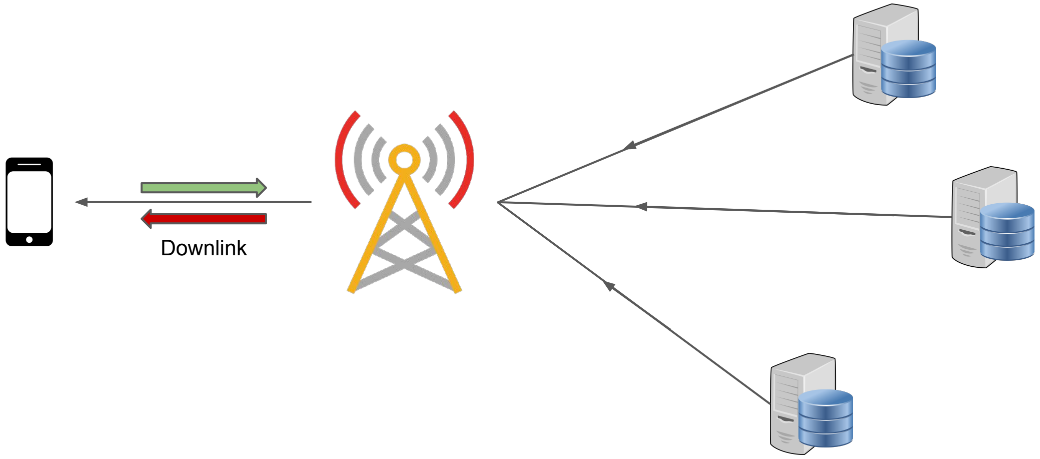
- Service within Chrome that provides continuous estimates of network quality
    - Focus only on the last hop (device to network carrier).

- Rest of the talk
    - What’s the use case for knowing network quality?
    - Technical details.

Actual Web page loading

Website requests loading 55 resources loaded before first paint
Browser loads 55 resources before first paint

~Optimal webpage loading



- Make best use of the network capacity
- Lower priority resources should not slow down loading of higher priority resources
- Estimating network capacity is a prerequisite to computing ~optimal loading schedule.

Other uses of network quality estimation

- If webpage is expected to load slowly
    - Use **interventions** to speed up loading
        - Interventions break web specs
            - Defer JavaScript (rendering does not block on JavaScript loading)
            - Do not load WebFonts etc.
        - Useful to trigger interventions only when necessary.

- Adjust time-outs at network layer
    - How long to wait to connect to proxy servers before retrying the next one?

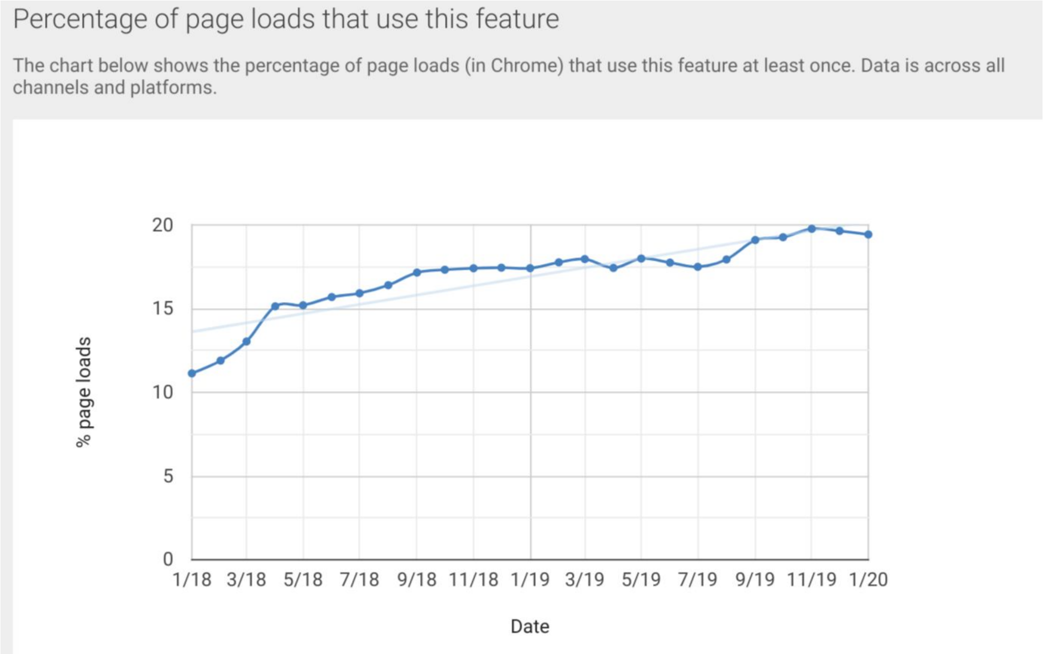
Web developers can use NQE too!

We exposed it to web developers to see what they do with it.

- RTT estimate
- Downlink bandwidth estimate
- `EffectiveConnectionType`: 4 easy to consume buckets

Current usage # on Web

- ~20% of webpages across all platforms (source)

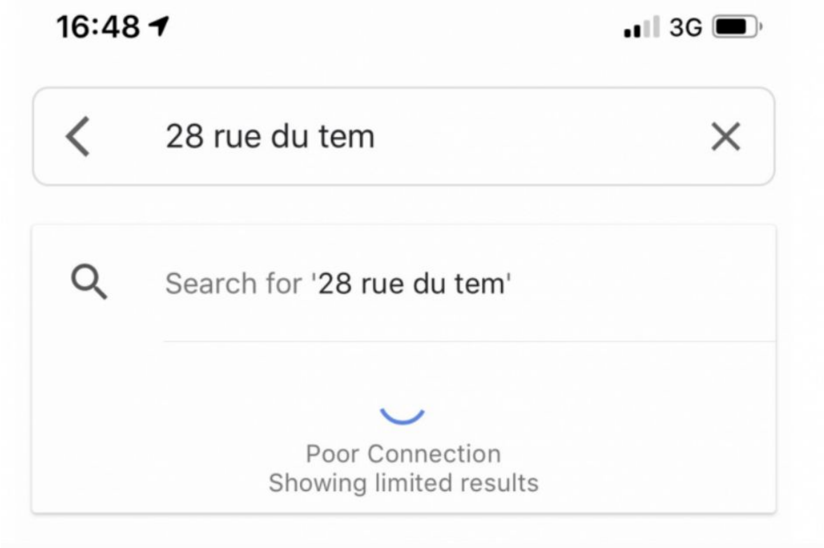


Percentage of page loads that use this feature

The chart below shows the percentage of page loads (in Chrome) that use this feature at least once. Data is across all channels and platforms.

How are web developers using it?

- Expose information to the web developers
  - Use the correct video resolution right from the beginning of the playback (Shaka, Facebook)
  - User Interface (“Your connection is slow…”) (link).



28 rue du tem

Search for '28 rue du tem'

Poor Connection
Showing limited results

Technical details

How do we know if the connection is slow?

- Connection type is not a good indicator of the network quality
  - Wi-Fi != Always fast (depending on the user’s subscription plan)
  - Most page loads on empirically slow networks are actually on Wi-Fi and 4g, not 2g and 3g.

- NQE (Network Quality Estimator) service within Chrome
  - Provides network quality estimates

Challenges

- Multiple platforms

Challenges

- Algorithm must run on off-the-shelf devices across most of the platforms
    - No access to radio state (SNR, different states) or TCP state in kernel

- Need to measure RTT passively.
    - Privacy issues with active probing.
    - No overhead of maintaining a server.
    - Passive measurement automatically works with other browsers based on Chromium code base (Opera, Samsung, Edge etc.), Webview etc.

Measuring RTT passively in NQE

- NQE utilizes 3 sources of information
  - HTTP layer
  - Transport layer
  - Something in between: H2/QUIC

HTTP layer RTT

- RTT = Response headers received - request sent
- Cons: Hanging GETs may have artificially high RTT, Computed RTT includes server processing time. For H2/H3 connections, request may be stuck behind other requests.
- Provides an **upper bound** on RTT estimate

Transport layer RTT

- At periodic intervals, make a syscall to all active TCP sockets to get current RTT estimates. Take a median of all values.
- Less noisy: Not susceptible to server processing delays or Hanging GETs
- Cons: Does not take into account packet loss.
- Provides a **lower bound** on RTT estimate
- Unavailable for UDP sockets (aka QUIC connections).
- Available only on POSIX platforms (not available on Windows, Mac OS X).

QUIC/H2 PING

- Servers expected to respond immediately
- Guaranteed to be not hanging
- **More realistic upper bound**
- Cons: Not all servers support QUIC/H2

Aggregating RTT Samples

- 3 sources of RTT
- For each source, aggregate all samples to get one RTT estimate per source.
- Aggregation algorithm
    - Weighted median, Recent samples have more weight

- Combine 3 RTT values (one from each source) to get a single value.

Is RTT enough?

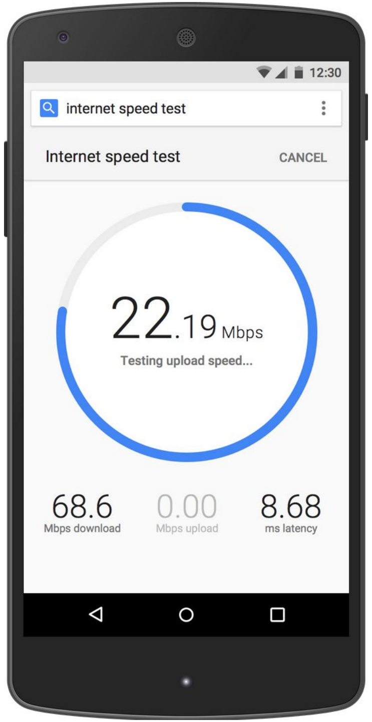
RTT does not reflect the capacity of the network

- Why is capacity important? Carriers may throttle if user transmits too much data in short time.

We need **Bandwidth** estimate

Measuring Bandwidth: State of the Art

- Packet pair
- Packet train
- Physical layer characteristics
- Middleware (Network carrier estimates bandwidth)
- Download large file, Measure TCP stats



internet speed test

Internet speed test

22.19 Mbps

Testing upload speed...

68.6 Mbps download

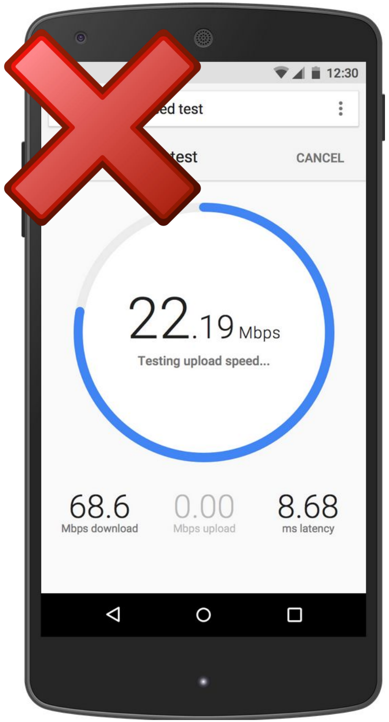
0.00 Mbps upload

8.68 ms latency

Measuring Bandwidth: State of the Art

- Existing Algorithms from academia and industry
  - Packet pair
  - Packet train
  - Physical layer characteristics
  - Middleware (Network carrier estimates bandwidth)
  - Download large file, Measure TCP stats

- Requirements
  - Measurement should not generate new traffic (Passive measurement)



Bandwidth computation in Chrome: Challenges

- Computing downlink bandwidth (from web server to device) without any cooperation from sender side
  - Can’t assume TCP flavor etc.
  - Downlink packet loss rate unknown: Difficult to differentiate between underutilized network and slow network.

- Very little insight in receiver’s TCP internal state

Bandwidth Estimation Algorithm

Goal: Provide an **estimate** of the available bandwidth

where bandwidth = Achievable goodput if Chrome were to download a large resource under current network conditions

Algorithm: Compute bandwidth over a time-window

Window properties:

- At least 128 KB large (tackle socket buffering)
- At least N active non-hanging requests at all times during the duration of the window

Responsiveness

- How quickly the estimates adapt to changing network conditions?
- RTT and bandwidth estimation algorithms work based on organic traffic
- Absence of organic traffic can lead to inaccuracies
  - User just started the browser: Missing network quality estimate
  - Network quality suddenly changed (parking lot): Stale network quality estimate

- Improving responsiveness
  - Store network quality estimates on disk keyed by network ID (network type, SSID, MCC/MNC, wireless signal strength)
  - Do not wait for organic traffic: Use wireless signal strength to improve responsiveness

Measuring estimation accuracy

- Challenges
  - Ground truth unknown
  - Lab results unreliable
- Approach
  - A/B tests
  - Ensure that user metrics (page load counts, performance) improve.

The End

Contacts:
tbansal@chromium.org, igrigorik@chromium.org, yoavweiss@chromium.org, bengr@chromium.org