Network Quality Estimation in Chrome

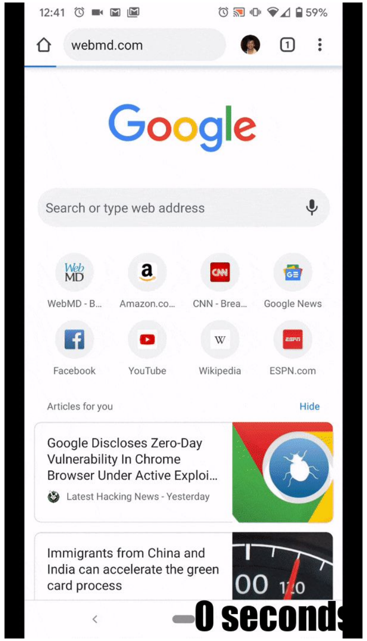
Web & Networks Interest Group
Feb 2020
Presented by: Tarun Bansal (Google Chrome)

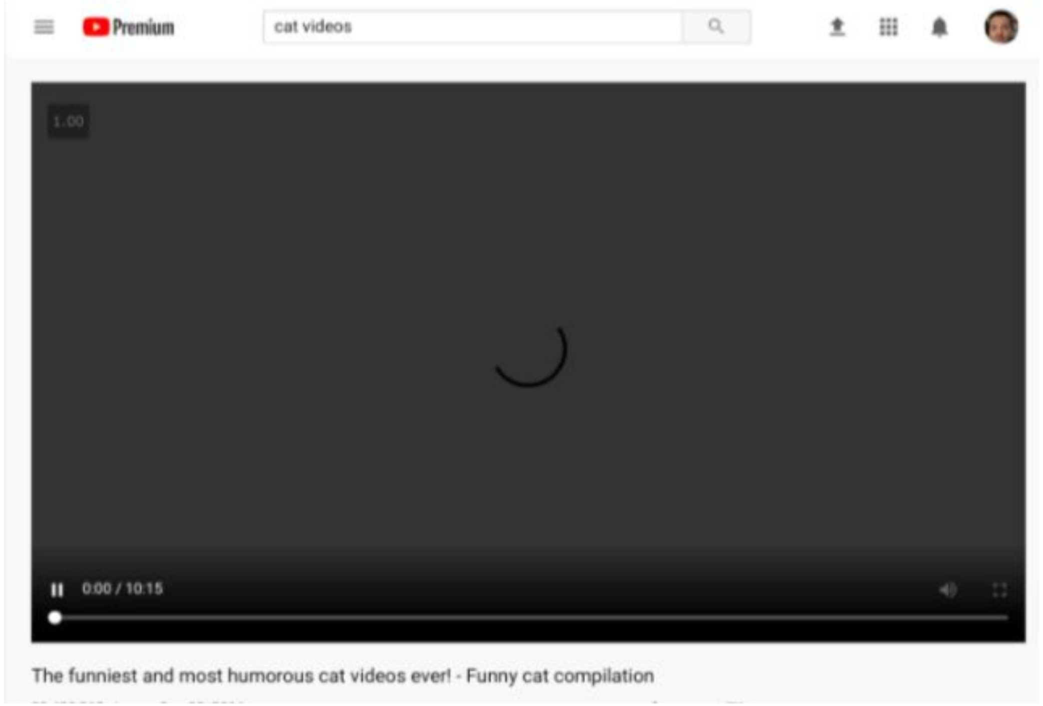
whoami

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

Slow web is not fun!

- ~20% of page loads on Chrome on Android are on 3G like connections.
    - Variable depending on the country: 5% - 40%


12:41
59%
webmd.com
Google
Search or type web address
WebMD - B...
Amazon.co...
CNN - Brea...
Google News
Facebook
YouTube
Wikipedia
ESPN.com
Articles for you
Hide
Google Discloses Zero-Day Vulnerability In Chrome Browser Under Active Exploi...
Latest Hacking News - Yesterday
Immigrants from China and India can accelerate the green card process
0 seconds


Premium
cat videos
0:00 / 10:15
The funniest and most humorous cat videos ever! - Funny cat compilation

NQE (Network Quality Estimator)

- Service within Chrome that provides continuous estimates of network quality
  - RTT and downlink estimates
  - Aggregated across all paths (browser to all web servers): Effectively, focuses on the common hop (browser to the network carrier).

- Rest of the talk
  - What's the use case for knowing network quality?
  - Implementation details of NQE

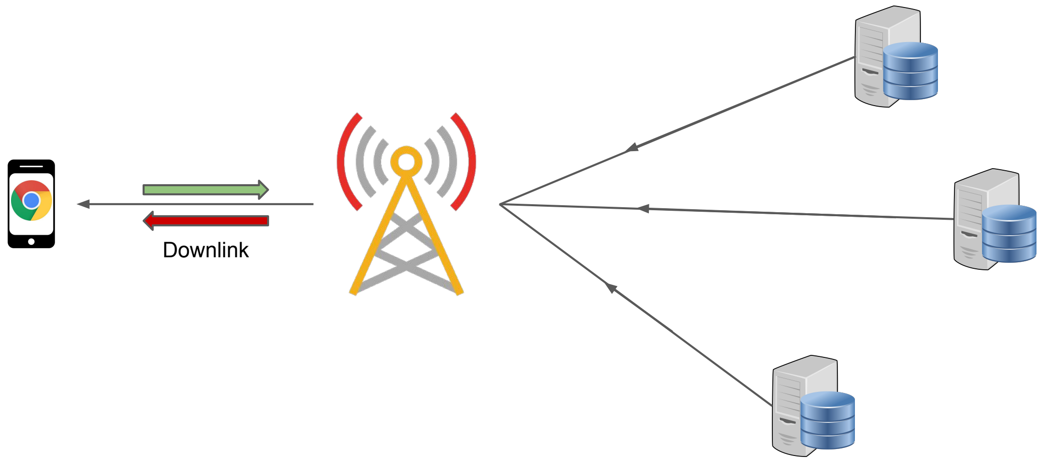
Help from lot of folks

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

Web page loading

- Performance-driven web development is difficult.
- The webpage may request loading of tens of resources even before first paint
    - CSS , JavaScript, Iframes, Images
- By default, Chrome loads all the resources in parallel.
- Not all resources are equal in importance
    - Less important: iframes, tracking pixels, async JavaScript, below the fold images etc.
    - Bad experience on bandwidth constrained networks (buffer bloats, packet losses)

~Optimal webpage loading



- Optimal use of the network
  - Keep the pipes full
  - 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
    - "An intervention is when a user agent decides to deviate slightly from a standardized behavior in order to provide a greatly enhanced user experience. Because of its nature, it must be done sparingly and with extreme prudence." (source)
    - Interventions break web specs to speed up loading
    - Important to trigger interventions only when loading is painfully slow.

- Feedback to the network stack
  - How long to wait to connect to proxy servers before retrying the next one?
  - Future use cases: Adapt initial TCP timeout based on network quality

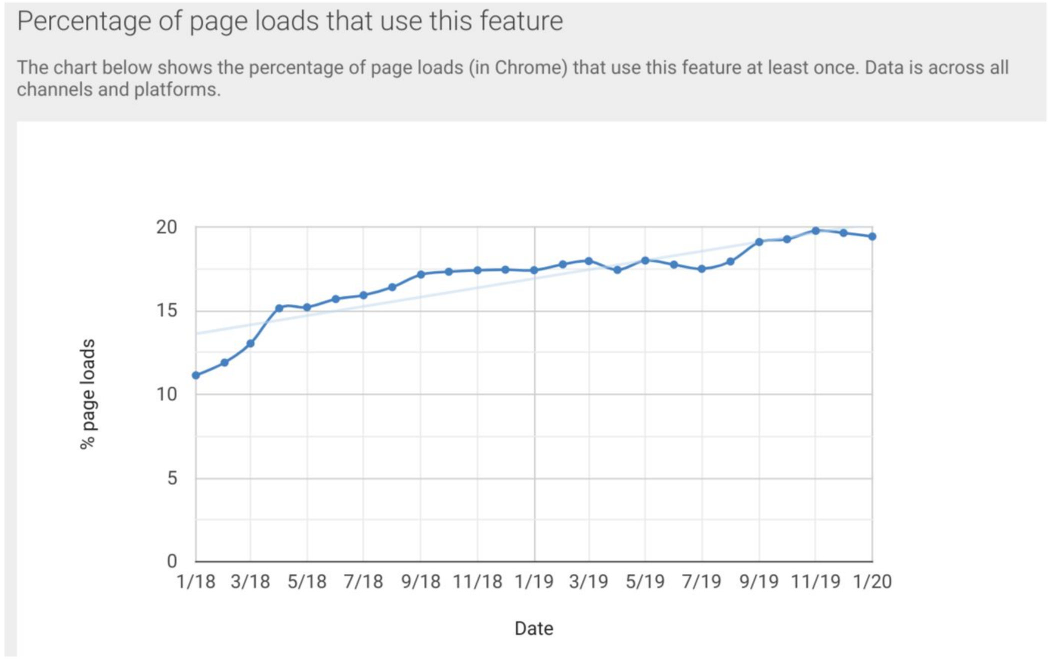
Lastly, web developers can use network quality estimates too!

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

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

Current usage # on Web

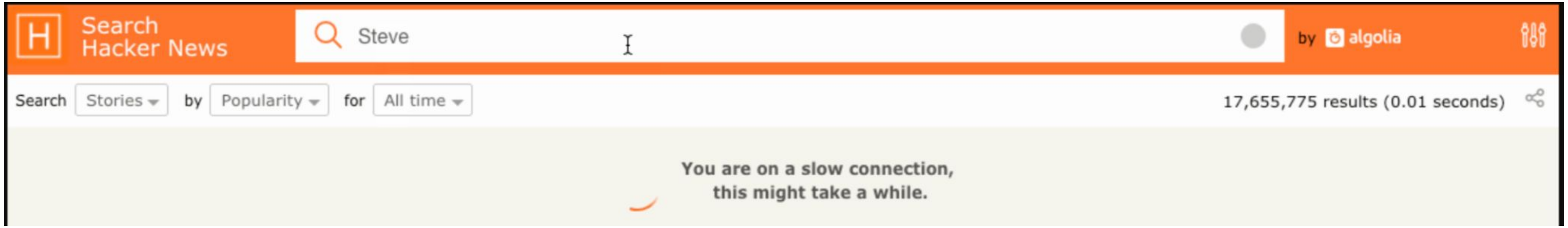
- ~20% of webpages across all platforms (source)


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?

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



Search Stories by Popularity for All time 17,655,775 results (0.01 seconds)

You are on a slow connection,
this might take a while.

Technical details

How to determine if the connection is slow?

- First hop (device to next hop) connection type is not a good indicator of the network quality
  - Wi-Fi != Always fast (depends on user’s Internet 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

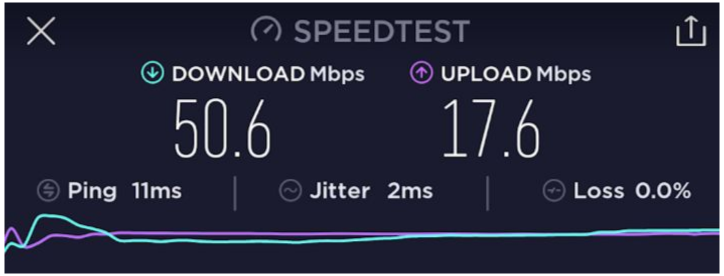
Challenges

- Multiple platforms: Difference in availability of APIs

chrome

Challenges (contd.)

- Algorithm must run on off-the-shelf devices
  - No access to radio state (SNR, different states) or TCP state in kernel
- Need to measure network quality passively.
  - Minimal privacy issues with passive measurements.
  - No overhead of maintaining a server.
  - Passive measurement automatically works with other browsers based on Chromium code base (Opera, Samsung, Edge etc.), Webview etc.


SPEEDTEST

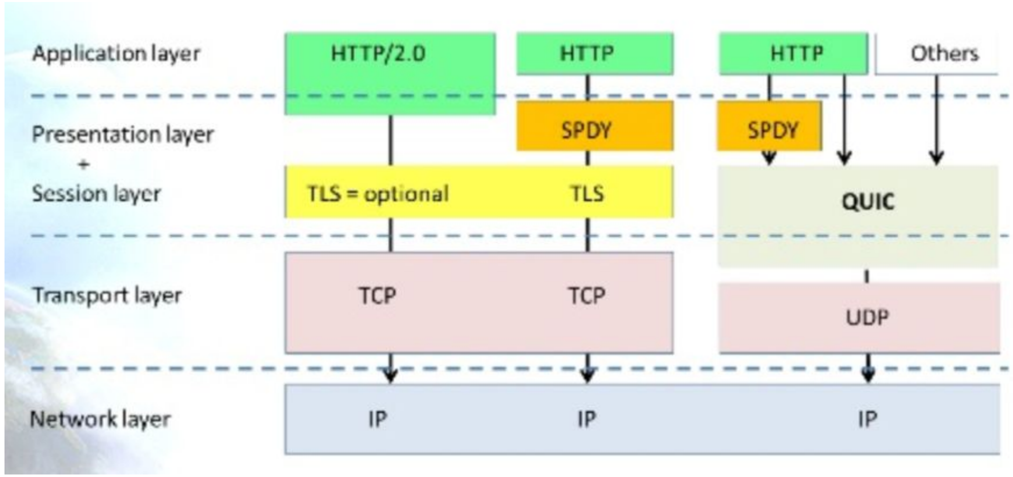
DOWNLOAD Mbps 50.6

UPLOAD Mbps 17.6

Ping 11ms | Jitter 2ms | Loss 0.0%

Drumroll...Estimating RTT

- NQE utilizes 3 sources of information (varies based on platform)
  - HTTP layer
  - Transport layer: TCP
  - Something in between: SPDY (aka H2), QUIC (aka H3).



First source: HTTP layer RTT

- RTT = Response headers received - request sent
- Available on all platforms
- Limitations
  - 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

Second source: 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
- Limitations
  - Does not take into account packet loss.
  - Unavailable for UDP sockets (aka QUIC connections).
  - Available only on POSIX platforms (Android, Linux, ChromeOS).
- Provides a lower bound on RTT estimate

Third source: QUIC/H2 PING

- Servers expected to respond immediately to PING frames
- Guaranteed to be not hanging
- **Most accurate**
- Limitations:
  - Not all servers support QUIC/H2
  - Not all QUIC/H2 servers support PING
  - PING frames may be queued behind other packets

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
    - Slightly different from TCP which uses weighted average: Median more resistant to outliers.
- Combine 3 RTT values (one from each source) to get a single value.
    - Heuristics for combining the 3 values.

Is RTT enough?

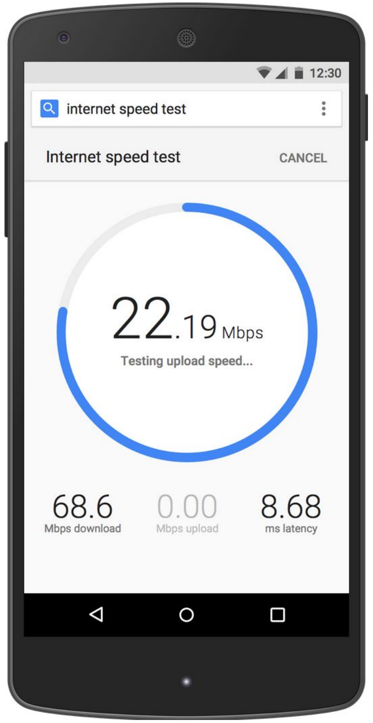
RTT does not reflect the capacity of the network

- Why is capacity important? Carriers may throttle if user transmits a lot of data in short time.

We need **Bandwidth** estimate

Measuring Bandwidth: State of the Art

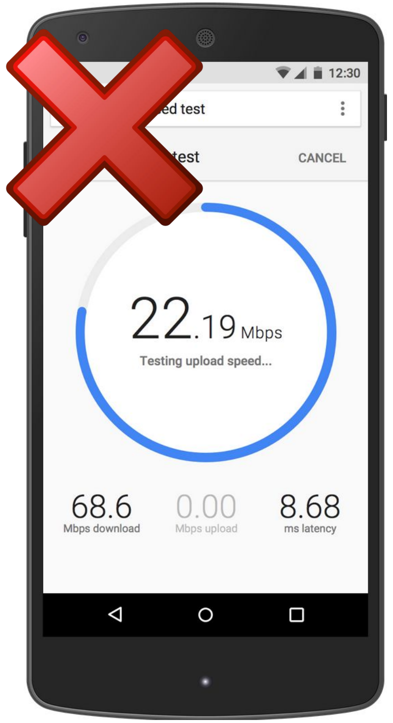
- Packet pair
- Packet train
- Physical layer characteristics
- Middleware (Network carrier estimates bandwidth)
- Download large file, Measure TCP stats



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)



Downlink Bandwidth computation in Chrome: Challenges

- No cooperation from sender side (web server)
  - Unknown TCP flavor
  - Unknown downlink packet loss rates: Difficult to differentiate between underutilized network and throttled network.

- Very little insight from the on-device TCP’s 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 (N = 5)

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)
- On Android, use wireless signal strength to improve responsiveness

Summary

- Lots of use cases for knowing network quality estimates and exposing to different layers of Web
- Lots of technical challenges :)

The End

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

Source code, NetInfo API, Demo Page

Measuring estimation accuracy

- Challenges
  - Ground truth unknown
  - Lab results
    - Does not reflect the challenges on real networks, but the accuracy is great there :)
- Approach
  - A/B tests
  - Ensure that user metrics (page load counts, performance) improve.
    - E.g., Improving the accuracy should improve the scheduling algorithm and page load performance