Using Packets to Guide Server Optimization

Josh Clark
Distributed Performance Engineer
https://github.com/je-clark/sf24_eu_server_optimization

About Me

- M.S. in Computer Engineering
  - Focus on networking, protocol design, and security
- Distributed Performance Engineer
  - Using packets to solve complex performance issues

https://www.jeclark.net
https://github.com/je-clark
https://github.com/je-clark/sf24_eu_server_optimization

Tech Companies Optimize Networking

A brief history of the QUIC protocol and Google

https://www.androidpolice.com/quic-protocol-guide/

An Argument for Increasing TCP's Initial Congestion Window

Nandita Dukkipati, Tiziana Refice, Yuchung Cheng, Jerry Chu
Tom Herbert, Amit Agarwal, Arvind Jain and Natalia Sutin
Google Inc.
Mountain View, CA, USA
{nanditad, tiziana, ycheng, hkchu, therbert, aagarwal, arvind, nsutin}@google.com

https://research.google/pubs/an-argument-for-increasing-tcps-initial-congestion-window/

Snap: a Microkernel Approach to Host Networking

Michael Marty, Marc de Kruijf, Jacob Adriaens, Christopher Alfeld, Sean Bauer, Carlo Contavalli*
Michael Dalton*, Nandita Dukkipati, William C. Evans, Steve Gribble, Nicholas Kidd, Roman
Kononov, Gautam Kumar, Carl Mauer, Emily Musick, Lena Olson, Erik Rubow, Michael Ryan,
Kevin Springborn, Paul Turner, Valas Valancius, Xi Wang, and Amin Vahdat
Google, Inc.
sosp2019-snap@google.com

https://research.google/pubs/snap-a-microkernel-approach-to-host-networking/

How Netflix tunes Ubuntu on EC2

Jorge O. Castro
on 11 August 2015

Tags: ec2 , Ubuntu

https://ubuntu.com/blog/how-netflix-tunes-ubuntu-on-ec2

facebookincubator/katran

A high performance layer 4 load balancer

C · 4.6k · Updated 1 hour ago

https://github.com/facebookincubator/katran

CLOUDFLARE The Cloudflare Blog

How to drop 10 million packets per second

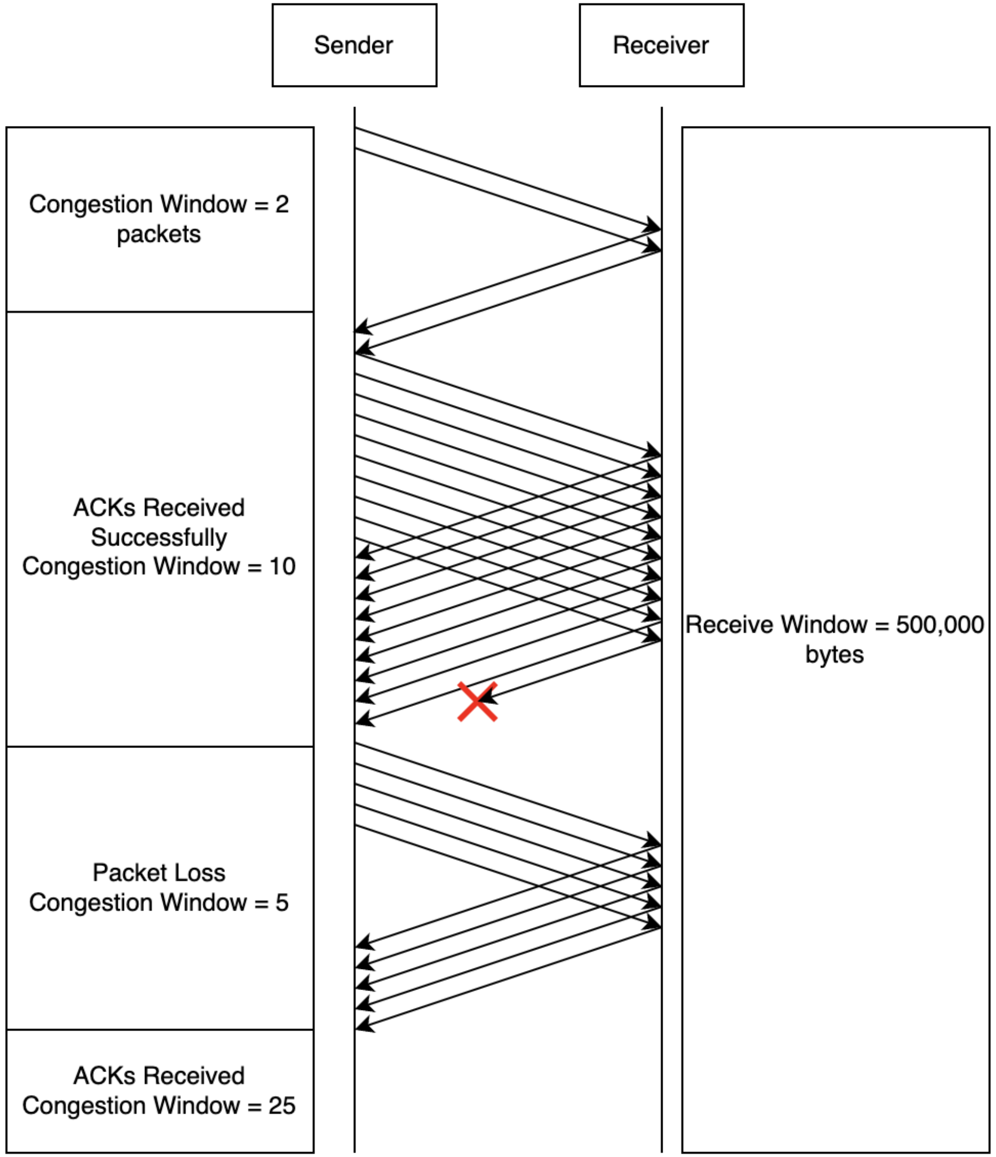
https://blog.cloudflare.com/how-to-drop-10-million-packets/

Today's Agenda

- TCP Windowing Review
- Baseline TCP Settings to Enable
- Evaluating and Optimizing Long TCP Flows
- Evaluating and Optimizing Short TCP Flows

TCP Windowing Review

TCP Windowing



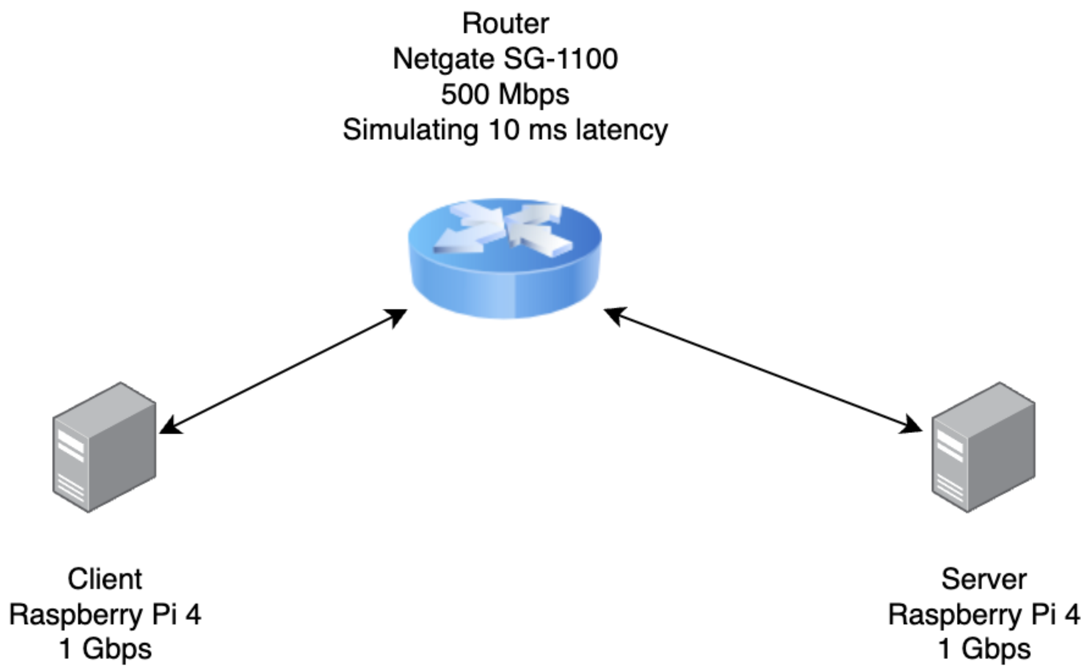
Baseline TCP Settings

Baseline TCP Settings


```
pi@server:~ $ sudo sysctl -a | grep tcp
net.ipv4.tcp_frto = 2
net.ipv4.tcp_sack = 1
net.ipv4.tcp_timestamps = 1
net.ipv4.tcp_window_scaling = 1
```

Long TCP Flow Optimization

Bandwidth Delay Product



BDP = bandwidth x latency

Receive Window Settings


```
net.ipv4.tcp_rmem = 4096    131072   6291456
```

Minimum   Default   Maximum

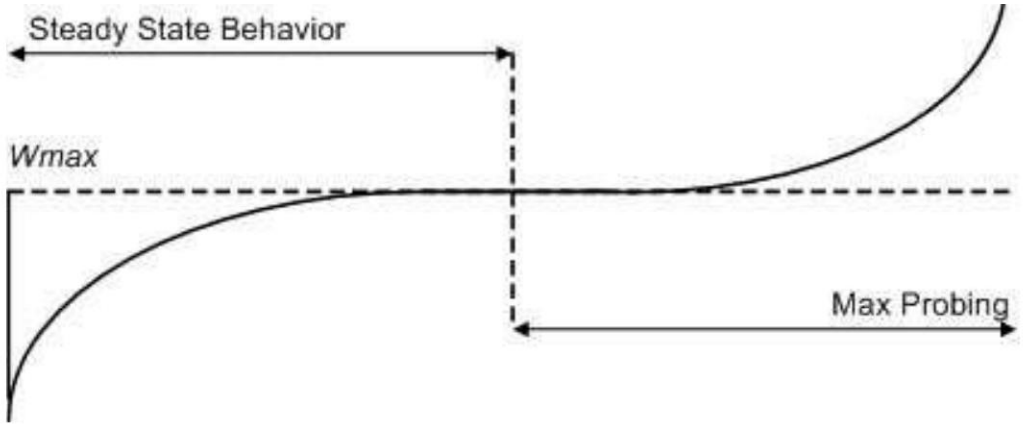
- Test Network BDP = 625 KB
- Linux can use up to 3x more memory than the receive window value
- Receive memory should be at least 1.8 MB

Capture 0

0_scp_200MB.pcap

Congestion Control - CUBIC

- Tracks a max window where loss occurs
- Logarithmic increase to max window, then cubic above max window
- Window x 0.2 when loss occurs

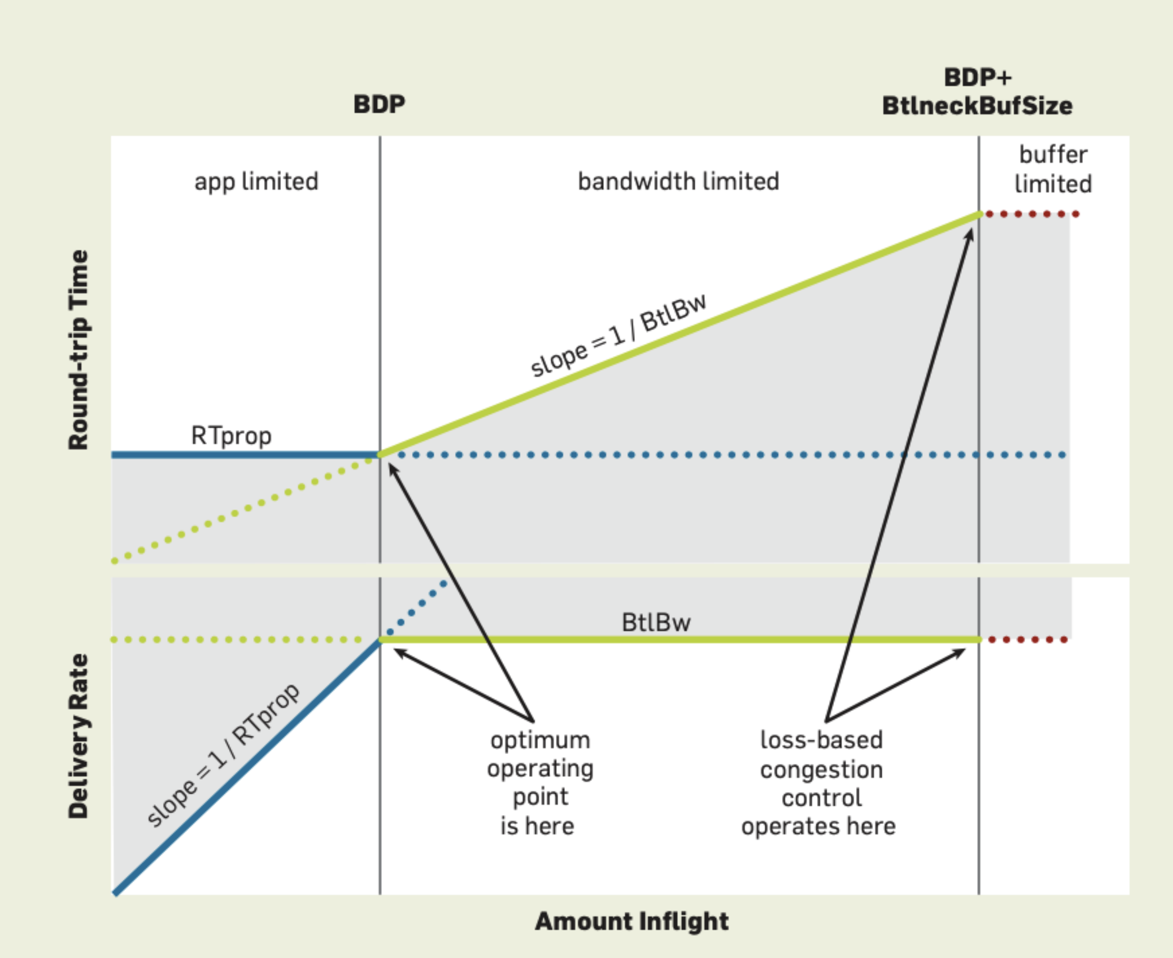


https://www.cs.princeton.edu/courses/archive/fall16/cos561/papers/Cubic08.pdf

Congestion Control - BBR

- Existing algorithms use packet loss to measure congestion, but that isn't accurate
  - Packet loss can happen randomly without congestion
  - Congestion-induced packet loss is a lagging metric for congestion

Figure 1. Delivery rate and round-trip time vs. inflight.

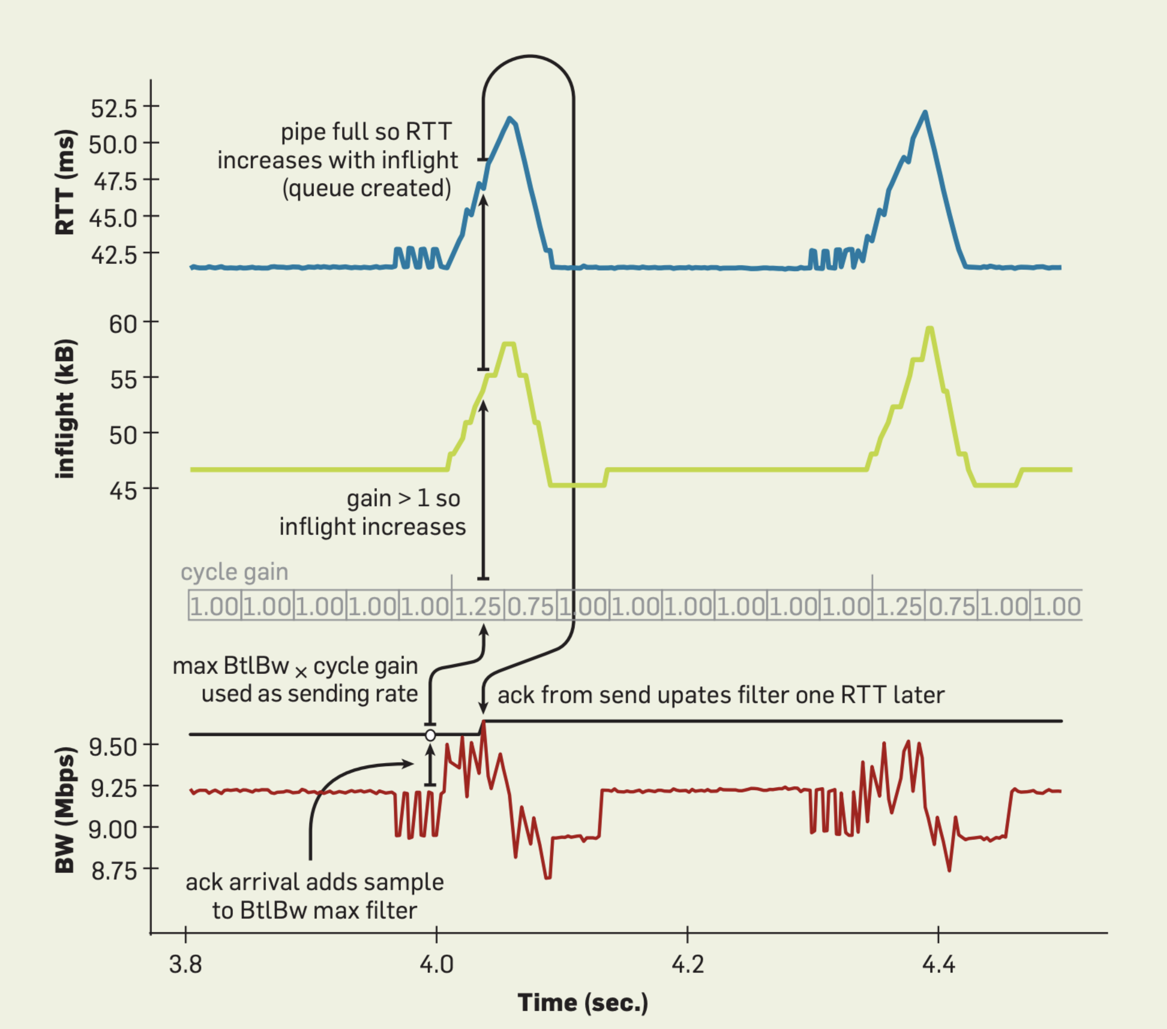


https://dl.acm.org/doi/pdf/10.1145/3009824

Congestion Control - BBR

- Estimates BDP using RTT variance
- Sends data at BDP rate
  - Periodically attempts a pacing_gain to see if bottleneck bandwidth has changed
- Ignores small amounts of packet loss

Figure 4. RTT (blue), inflight (green), and delivery rate (red) detail.



https://dl.acm.org/doi/pdf/10.1145/3009824

Capture 1

1_scp_200MB_bbr.pcap

Capture 2

2_scp_mystery.pcap

Dealing with Small Files

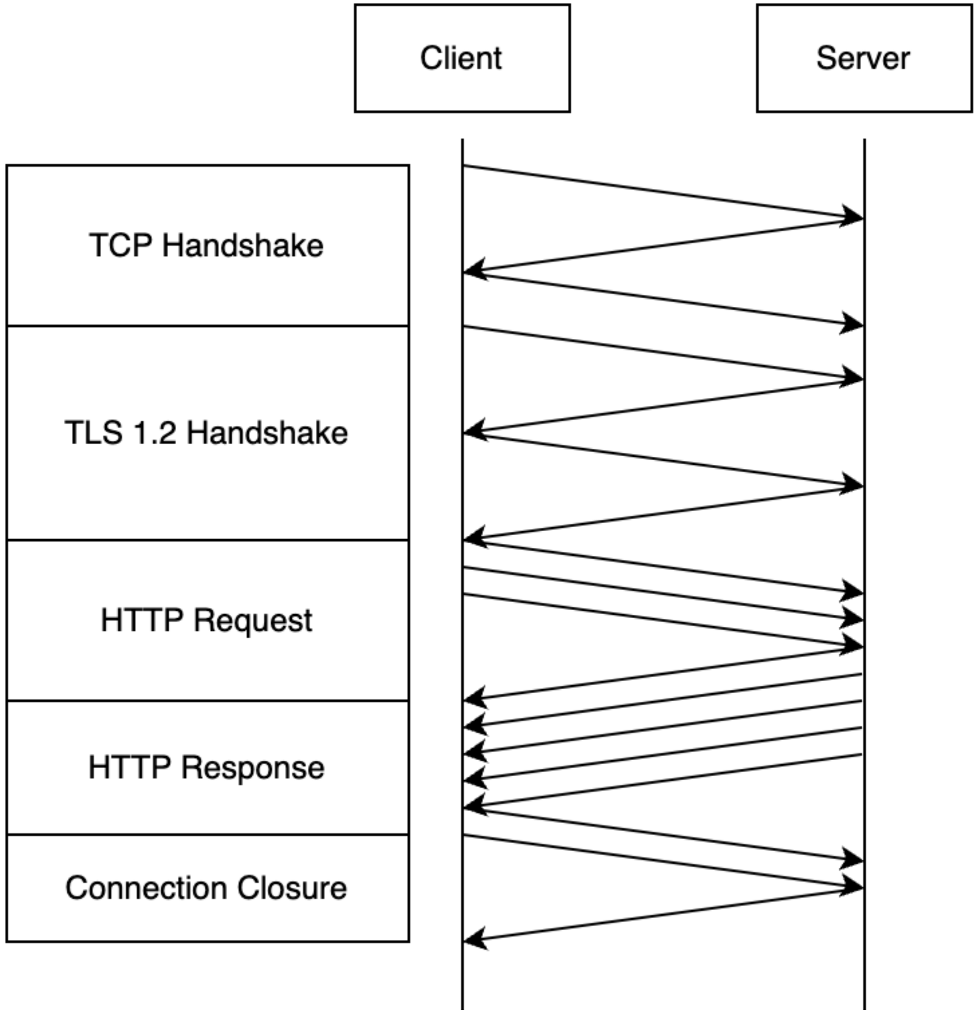

```
  adding: files/file_477 (deflated 27%)
  adding: files/file_366 (deflated 27%)
  adding: files/file_963 (deflated 27%)

real	0m0.398s
user	0m0.285s
sys	0m0.098s
```

```
pi@client:~ $ time scp -r pi@192.16
pi@192.168.2.15's password:
files.zip

real	0m2.286s
user	0m0.285s
sys	0m0.069s
```

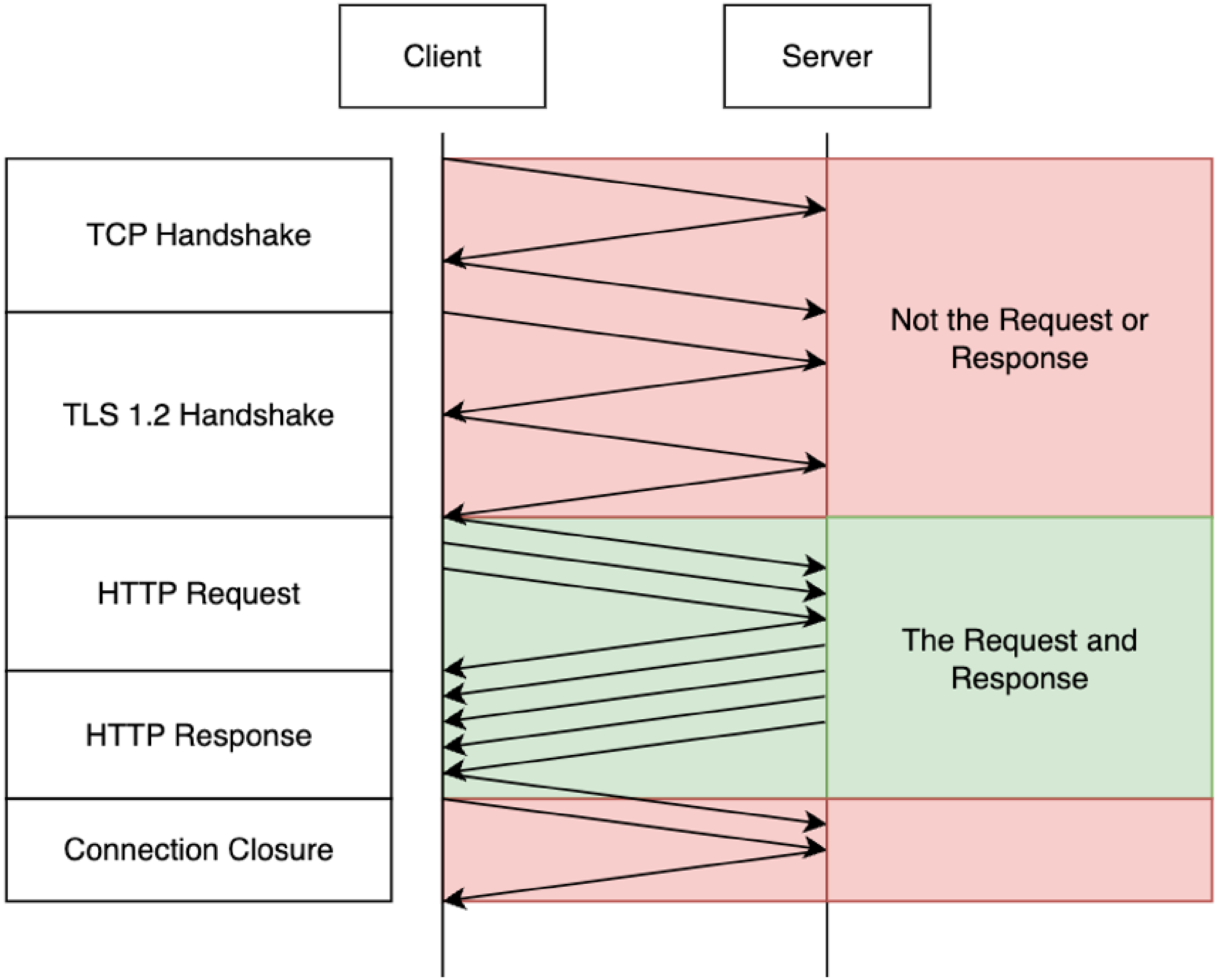
```
  inflating: files/file_477
  inflating: files/file_366
  inflating: files/file_963

real	0m0.244s
user	0m0.107s
sys	0m0.122s
```

Short TCP Flow Optimization

It’s All About the Turns



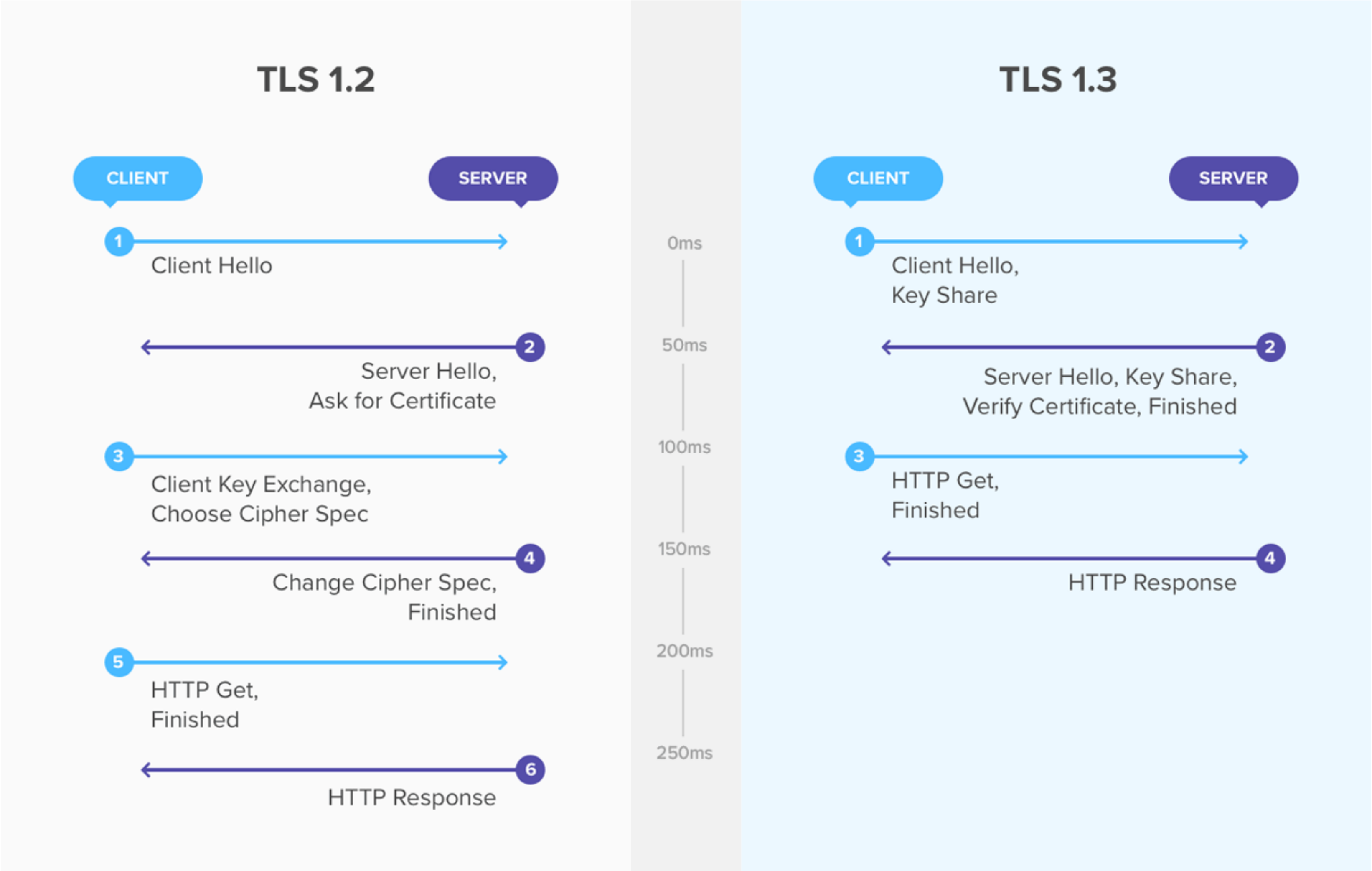
It’s All About the Turns



Capture 3

3_small_http_request.pcap

TLS 1.2 vs TLS 1.3



https://www.eyerys.com/articles/news/improved-internet-security-protocol-tls-13-has-been-approved

Capture 4

4_small_http_request_tls13.pcap

init_cwnd

- In 2011, Google changed init_cwnd to fit 90% of responses in 1 RTT

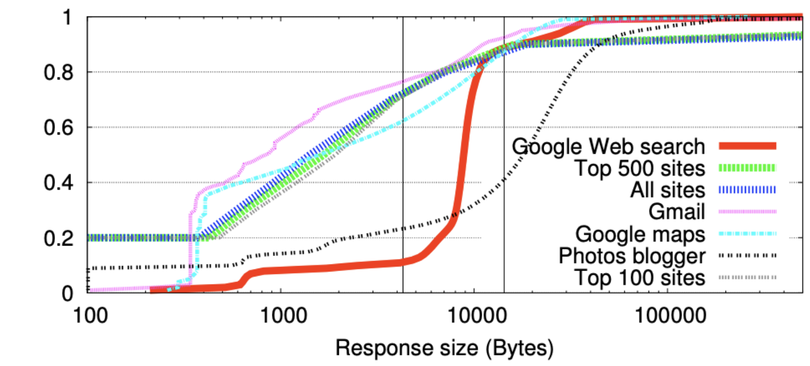


Figure 1: CDF of HTTP response sizes for top 100 sites, top 500 sites, all the Web, and for a few popular Google services. Vertical lines highlight response sizes of 3 and 10 segments.

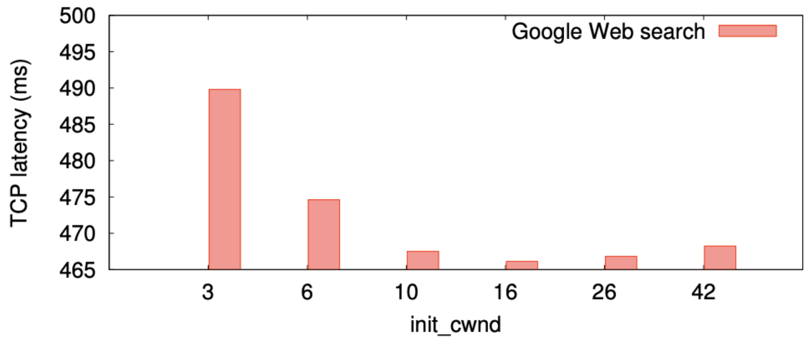


Figure 2: TCP latency for Google search with different *init_cwnd* values.

https://research.google/pubs/an-argument-for-increasing-tcps-initial-congestion-window/

init_cwnd


```
pi@server:~ $ ip route show
default via 192.168.2.1 dev eth0 proto dhcp src 192.168.2.15 metric 100
192.168.2.0/24 dev eth0 proto kernel scope link src 192.168.2.15 metric 100
pi@server:~ $ sudo ip route change default via 192.168.2.1 dev eth0 proto dhcp src 192
.168.2.15 metric 100 initcwnd 25
pi@server:~ $ ip route show
default via 192.168.2.1 dev eth0 proto dhcp src 192.168.2.15 metric 100 initcwnd 25
192.168.2.0/24 dev eth0 proto kernel scope link src 192.168.2.15 metric 100
```

Capture 5

5_small_http_request_initcwnd_45.pcapng

Potential Pitfalls

Potential Pitfalls

- Increased CPU utilization
- Increased memory utilization
- TCP Zero Windows

Feedback