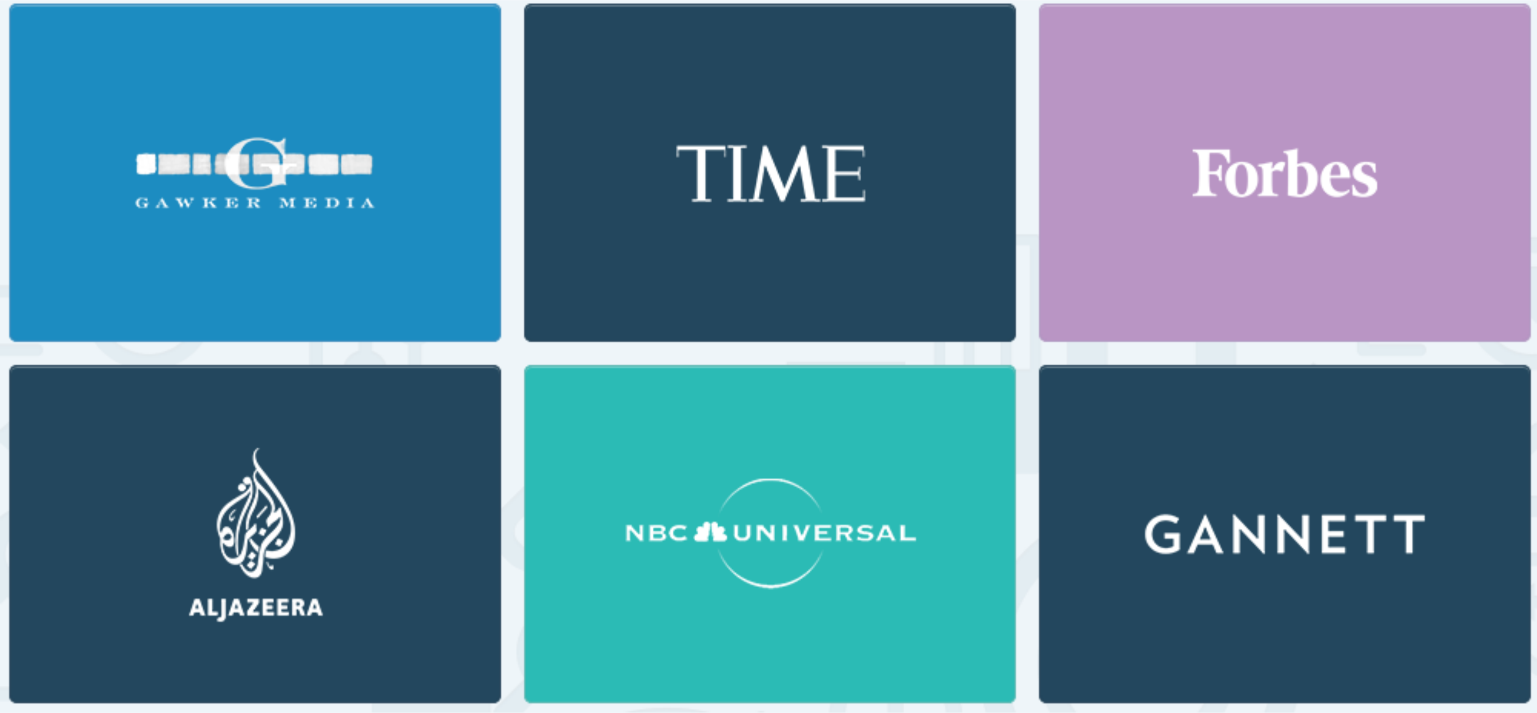
Tuning TCP and NGINX on EC2

Chartbeat

Who are we?

Chartbeat measures and monetizes attention on the web. Working with 80% of the top US news sites and global media sites in 50 countries, Chartbeat brings together editors and advertisers to identify in real time the active time an audience consumes articles, videos, paid content, and display advertising.



- Founded in 2009
- Hosted on AWS , 400-500 servers depending on time of day
- Around 180k - 220k req/sec
- 6 - 9 million concurrents

chartbeat.com/totaltotal

Who am I?

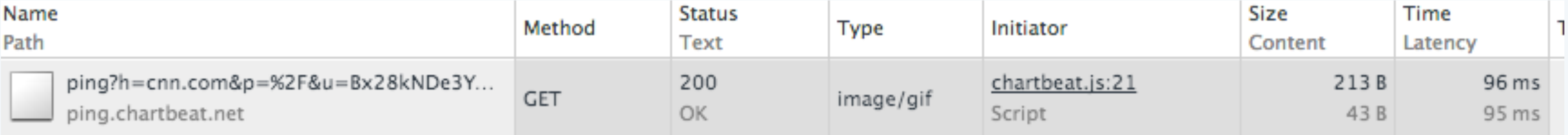
- Sr Web Operations Engineer
- Previously worked at
  - Bitly
  - TheStreet.com
  - Promotions.com

Traffic Characteristics

Every 15 seconds

213 byte, request size

43 byte, response size



Name Path	Method	Status Text	Type	Initiator	Size Content	Time Latency	T
ping?h=cnn.com&p=%2F&u=Bx28kNDe3Y... ping.chartbeat.net	GET	200 OK	image/gif	chartbeat.js:21 Script	213 B 43 B	96 ms 95 ms	

Problem

- Reports of slowness from some customers
- Taking 3 seconds to send data

Default Retransmission Timeout

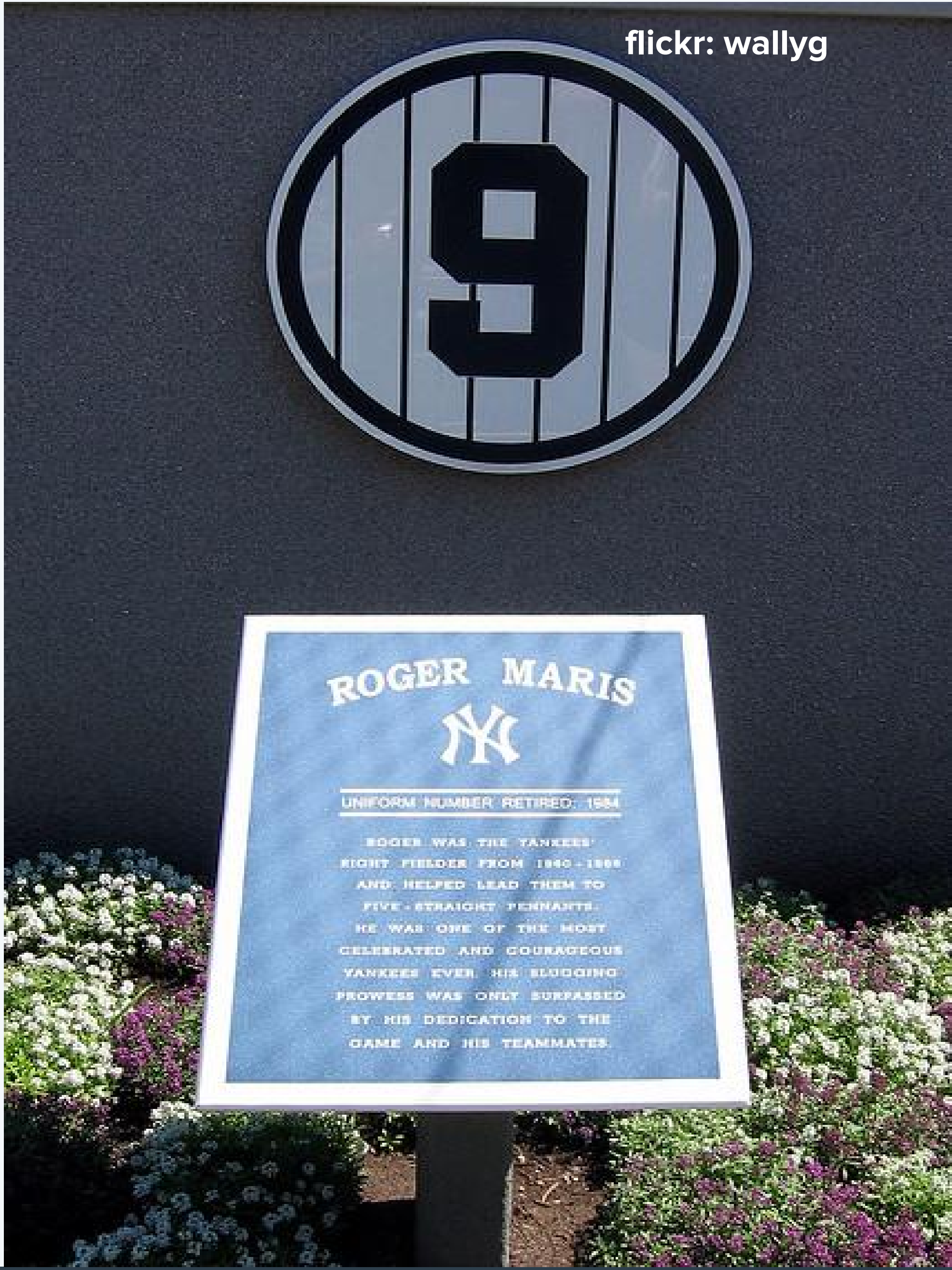
RFC 1122: Section 4.2.3.1

The following values SHOULD be used to initialize the
estimation parameters for a new connection:

(a) RTT = 0 seconds.

(b) RTO = 3 seconds. (The smoothed variance is to be
initialized to the value that will result in this RTO).

3
BABE RUTH
NY
UNIFORM NUMBER RETIRED: 1948
FROM 1920 - 1934, THE BABE
SINGLEHANDEDLY LIFTED BASEBALL
TO NEW HEIGHTS WITH HIS
UNLIMITED TALENT AND
UNBRIDLED LOVE FOR THE GAME.
HIS ENORMOUS CONTRIBUTIONS
TO BASEBALL AND THE YANKEES
MADE HIM THE MOST CELEBRATED
ATHLETE WHO EVER LIVED.

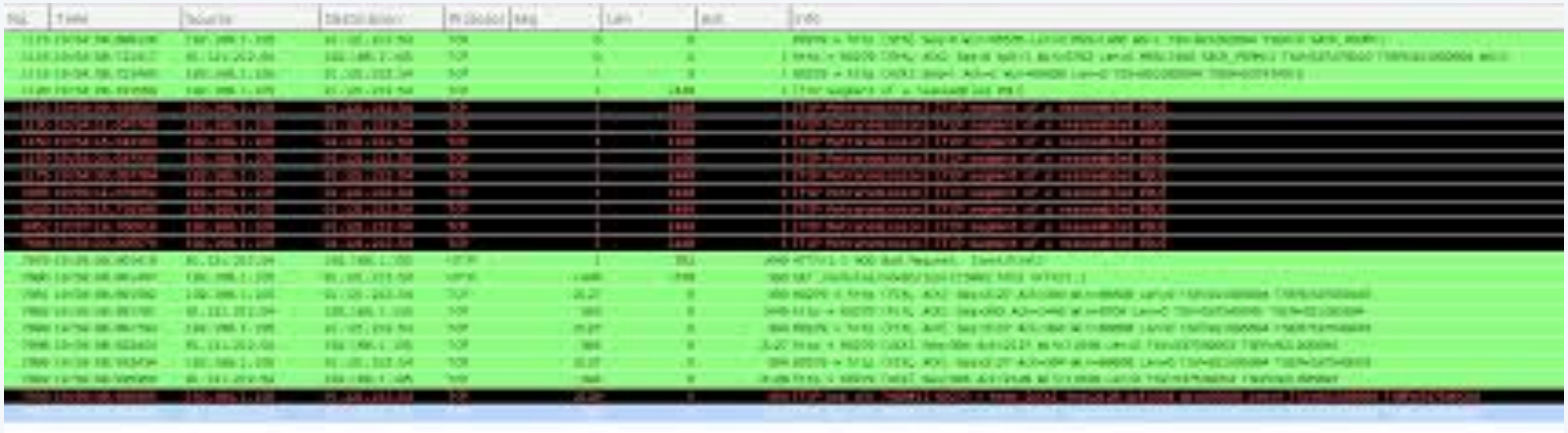

9
ROGER MARIS
NY
UNIFORM NUMBER RETIRED: 1984
ROGER WAS THE YANKEES'
RIGHT FIELDER FROM 1960 - 1966
AND HELPED LEAD THEM TO
FIVE - STRAIGHT PENNANTS.
HE WAS ONE OF THE MOST
CELEBRATED AND COURAGEOUS
YANKEES EVER. HIS SLUGGING
PROWESS WAS ONLY SURPASSED
BY HIS DEDICATION TO THE
GAME AND HIS TEAMMATES.

flickr: wallyg

flickr: oregondot

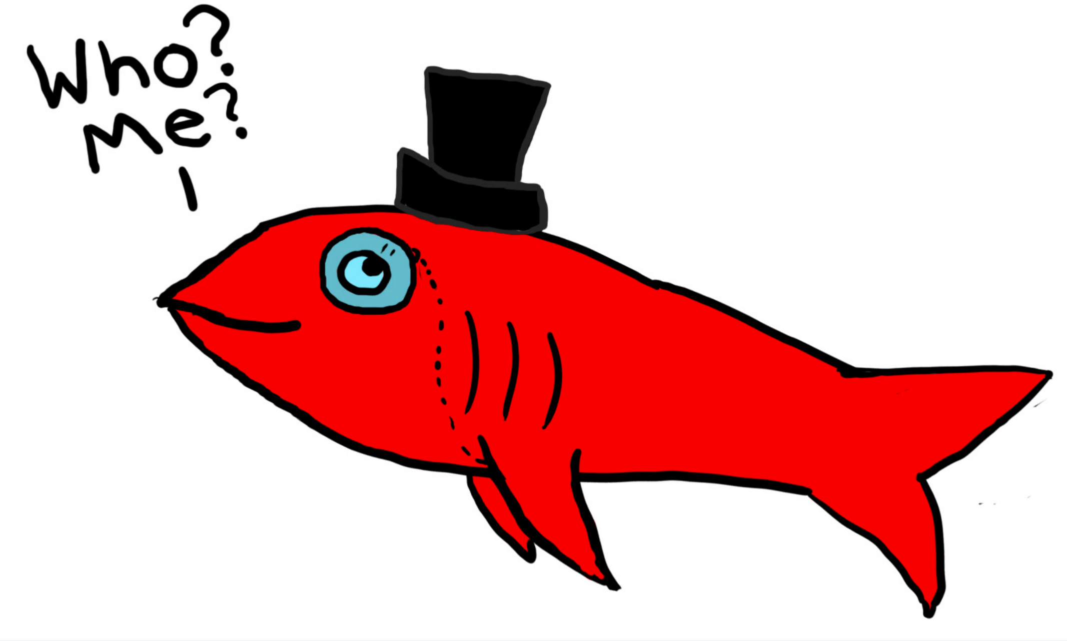
Now what?

TCPDump + Wireshark confirms retransmissions



DON'T GRAPH ALL THE THINGS

- Graph only relevant metrics
  - you'll end up with a ton of red herrings



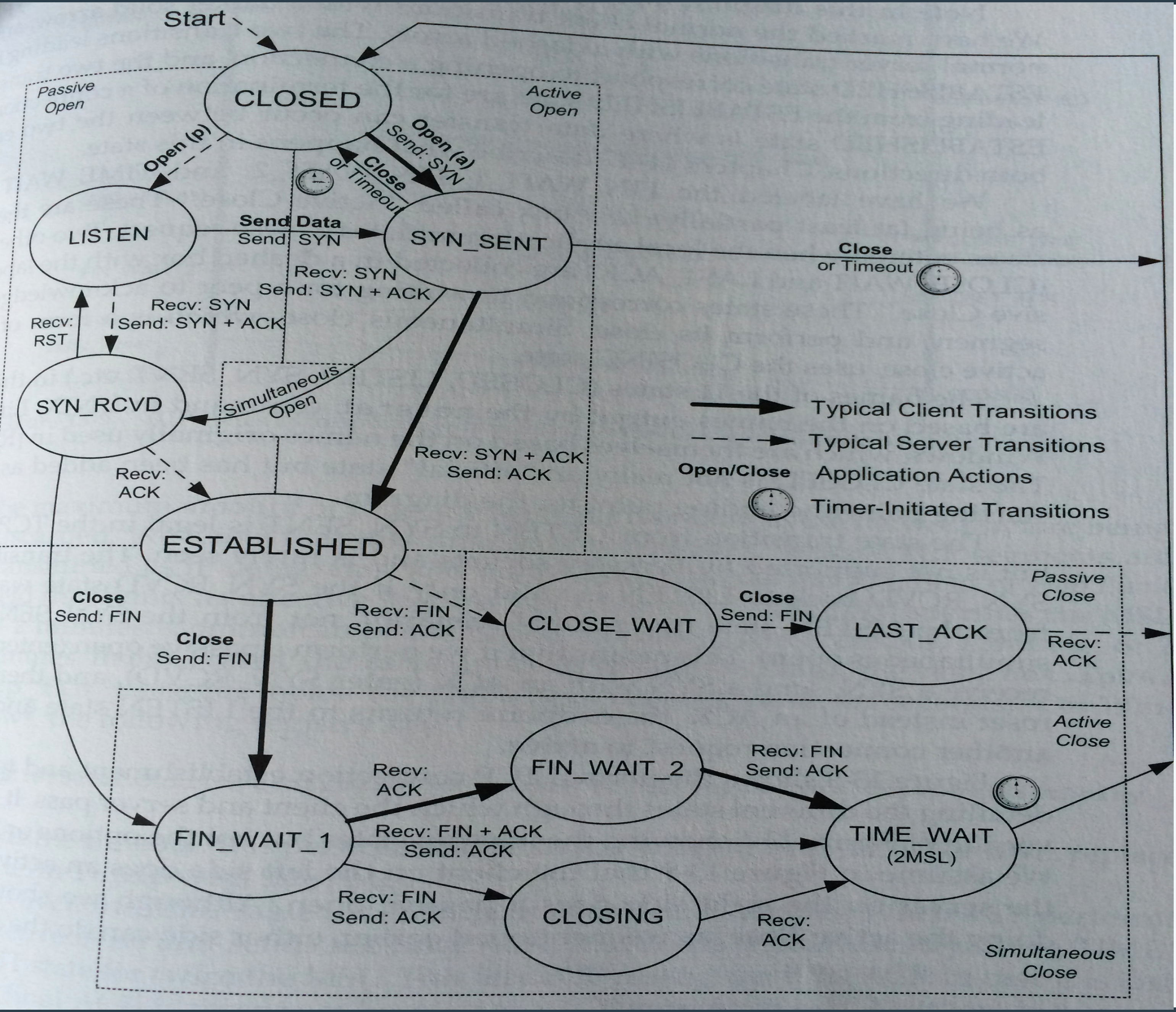
Sources of info

- ss -s
  - summary of socket statistics

```
TCP:   10678 (estab 2503, closed 8167, orphaned 0, synrecv 0, timewait 8167/0),
ports 0
```

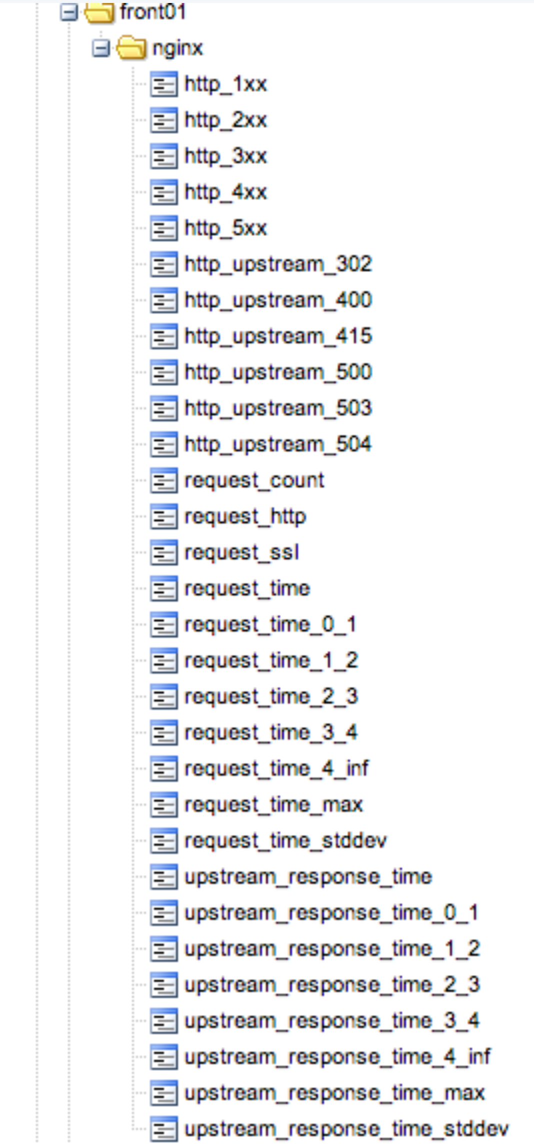
- netstat -s

"tcp_active_connections_openings",
"tcp_connections_aborted_due_to_timeout",
"tcp_data_loss_events",
"tcp_failed_connection_attempts",
"tcp_other_tcp_timeouts",
"tcp_passive_connection_openings",
"tcp_segments_retransmited",
"tcp_segments_send_out",
"tcp_syns_to_listen_sockets_dropped",
"tcp_times_the_listen_queue_of_a_socket_overflowed",



TCP/IP Illustrated Volume 1 Second Ed.

Logster + Graphite



- front01
  - nginx
    - http_1xx
    - http_2xx
    - http_3xx
    - http_4xx
    - http_5xx
    - http_upstream_302
    - http_upstream_400
    - http_upstream_415
    - http_upstream_500
    - http_upstream_503
    - http_upstream_504
    - request_count
    - request_http
    - request_ssl
    - request_time
    - request_time_0_1
    - request_time_1_2
    - request_time_2_3
    - request_time_3_4
    - request_time_4_inf
    - request_time_max
    - request_time_stddev
    - upstream_response_time
    - upstream_response_time_0_1
    - upstream_response_time_1_2
    - upstream_response_time_2_3
    - upstream_response_time_3_4
    - upstream_response_time_4_inf
    - upstream_response_time_max
    - upstream_response_time_stddev

https://github.com/etsy/logster

Tails logs, generates metrics and outputs to Graphite or Ganglia

FINDINGS

Sources of info

Values > 1, can't be good

- netstat -s

"tcp_active_connections_openings",

"tcp_connections_aborted_due_to_timeout",

"tcp_data_loss_events",

"tcp_failed_connection_attempts",

"tcp_other_tcp_timeouts",

"tcp_passive_connection_openings",

"tcp_segments_retransmited",

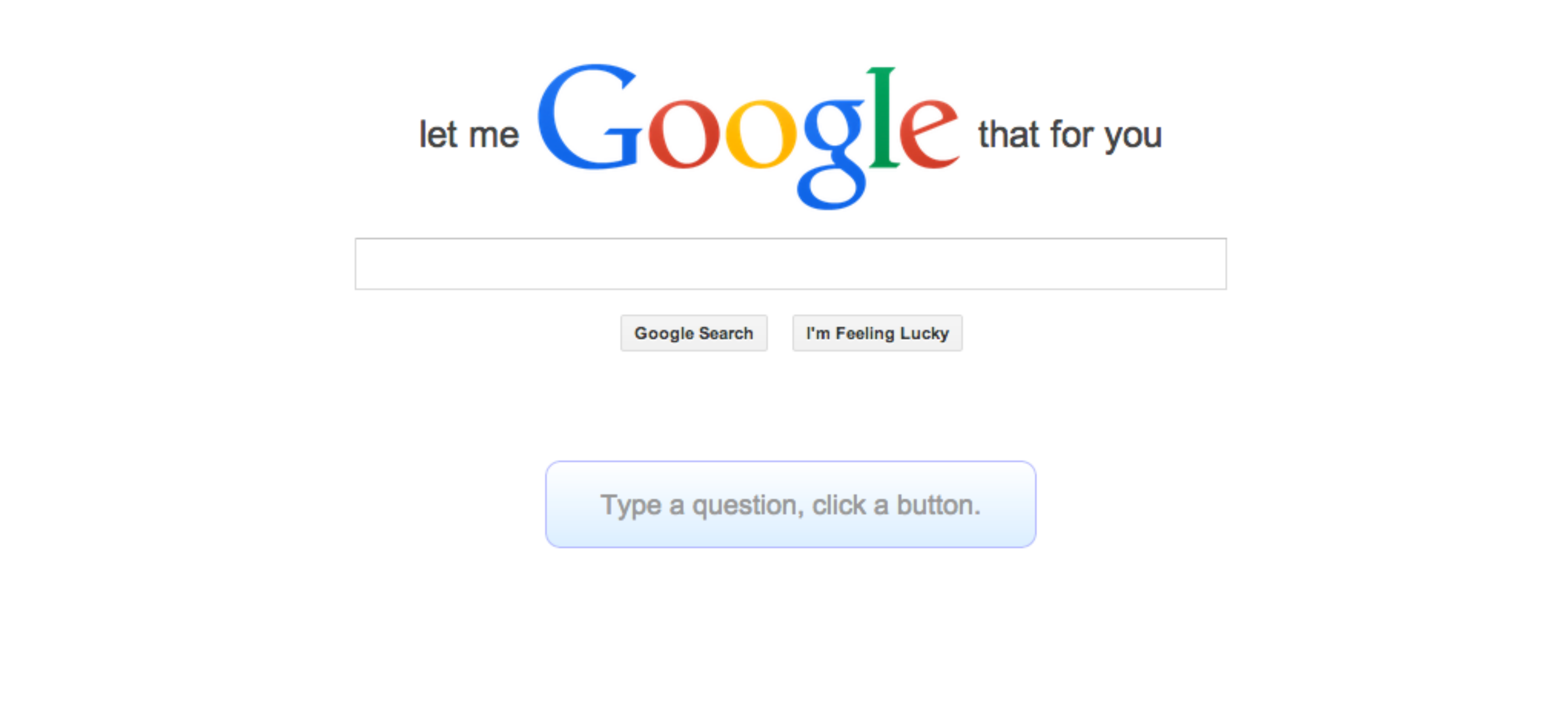
"tcp_segments_send_out",

"tcp_syns_to_listen_sockets_dropped",

"tcp_times_the_listen_queue_of_a_socket_overflowed",

Confirmed what we suspected

WHUT



let me Google that for you

Type a question, click a button.

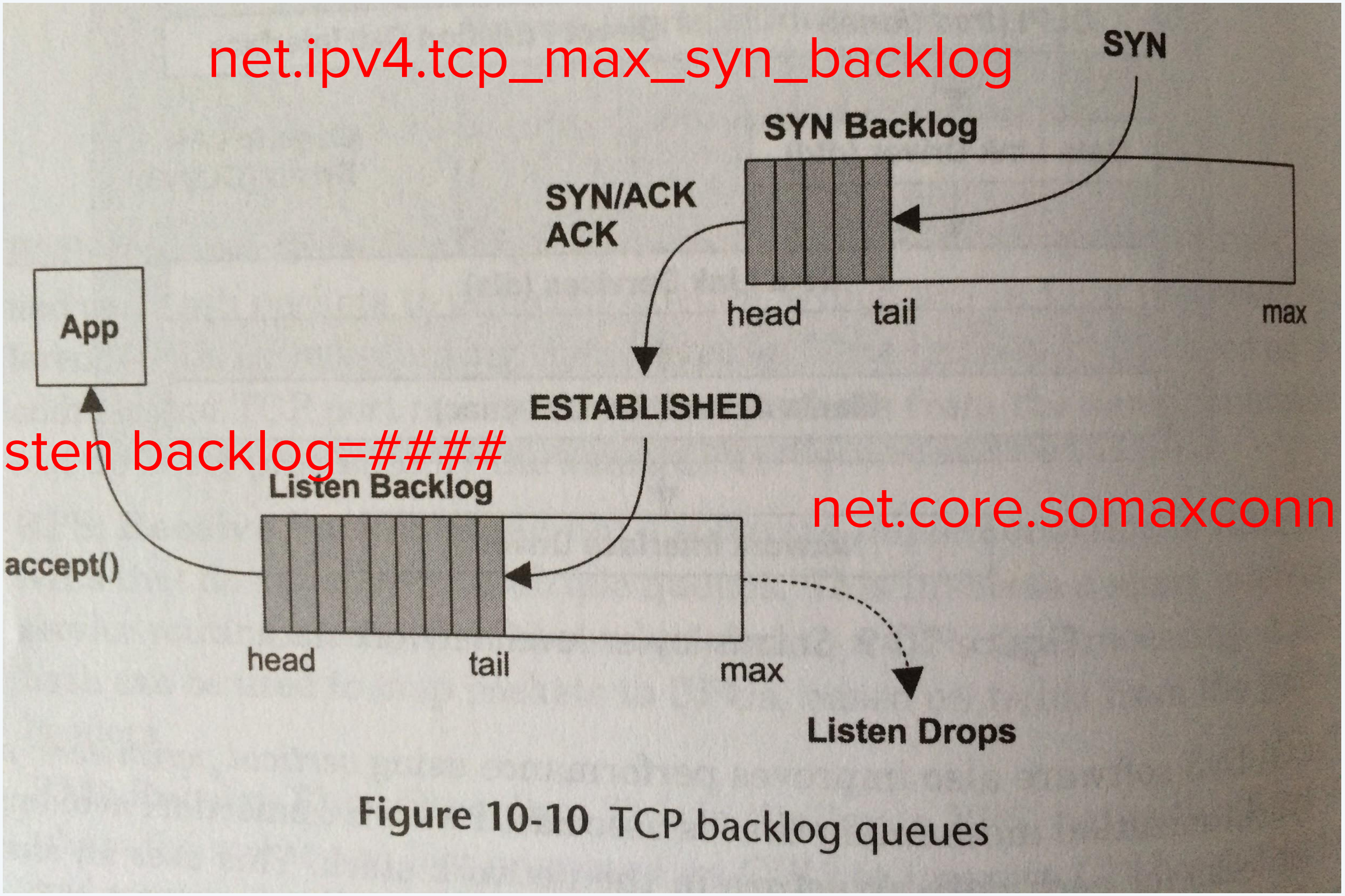


Figure 10-10 TCP backlog queues

Systems Performance

Enterprise and the Cloud by Brendan Gregg, pg 492

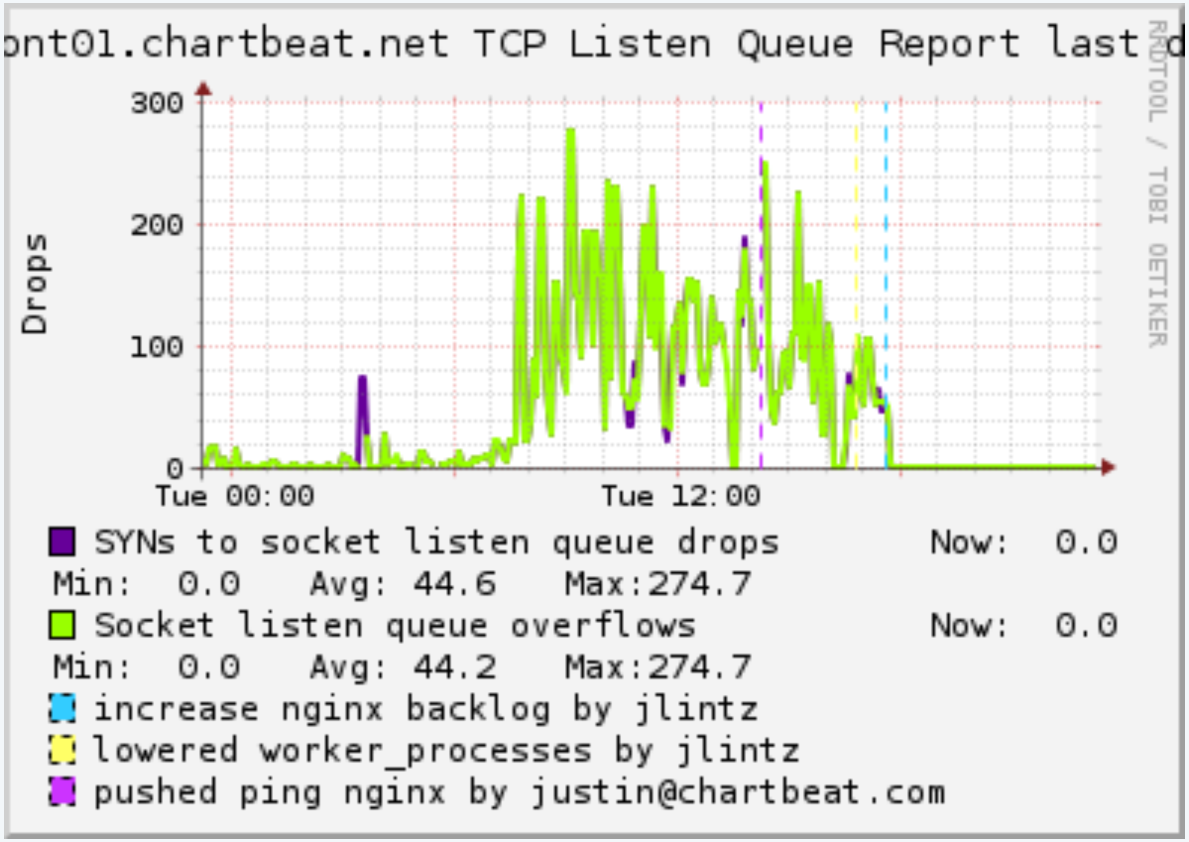
Insane Defaults

- `net.core.netdev_max_backlog` = 1000
  - Per CPU backlog?
  - Network Frames
- `net.ipv4.tcp_max_syn_backlog` = 128
- `net.core.somaxconn` = 128
- nginx listen backlog = 511 **?!?**
  - Silently truncated to `somaxconn` value

New Values

- `net.core.netdev_max_backlog` = 16384
- `net.ipv4.tcp_max_syn_backlog` = 65536
- `net.core.somaxconn` = 16384
- nginx listen backlog = 16384
    - should be <= `somaxconn`

Results


ont01.chartbeat.net TCP Listen Queue Report last d
Drops
300
200
100
0
Tue 00:00
Tue 12:00
RRDTOOL / TOBI OETIKER
■ SYNs to socket listen queue drops Now: 0.0
Min: 0.0 Avg: 44.6 Max:274.7
□ Socket listen queue overflows Now: 0.0
Min: 0.0 Avg: 44.2 Max:274.7
▨ increase nginx backlog by jlintz
▨ lowered worker_processes by jlintz
▨ pushed ping nginx by justin@chartbeat.com

Further settings explored

net.ipv4.tcp_slow_start_after_idle

net.ipv4.tcp_max_tw_buckets

net.ipv4.tcp_rmem/wrem

net.ipv4.tcp_fin_timeout

net.ipv4.tcp_mem

net.ipv4.tcp_slow_start_after_idle

Set to 0 to ensure connections don’t go back to default window size after being idle too long.

Example: HTTP KeepAlive

net.ipv4.tcp_max_tw_buckets

Max number of sockets in TIME_WAIT. We actually set this very high, since before we moved instances behind an ELB it was normal to have 200k+ sockets in TIME_WAIT state.

Exceeding this leads to sockets being torn down until under limit

net.ipv4.tcp_rmem/wrem

Format: `min default max` (in bytes)

The kernel will autotune the number of bytes to use for each socket based on these settings. It will start at `default` and work between the `min` and `max`

net.ipv4.tcp_fin_timeout

The time a connection should spend in FIN_WAIT_2 state. Default is 60 seconds, lowering this will free memory more quickly and transition the socket to TIME_WAIT.

This will NOT reduce the time a socket is in TIME_WAIT which is set to 2 * MSL (max segment lifetime)

net.ipv4.tcp_fin_timeout continued...

MSL is hardcoded in the kernel at 60 seconds!

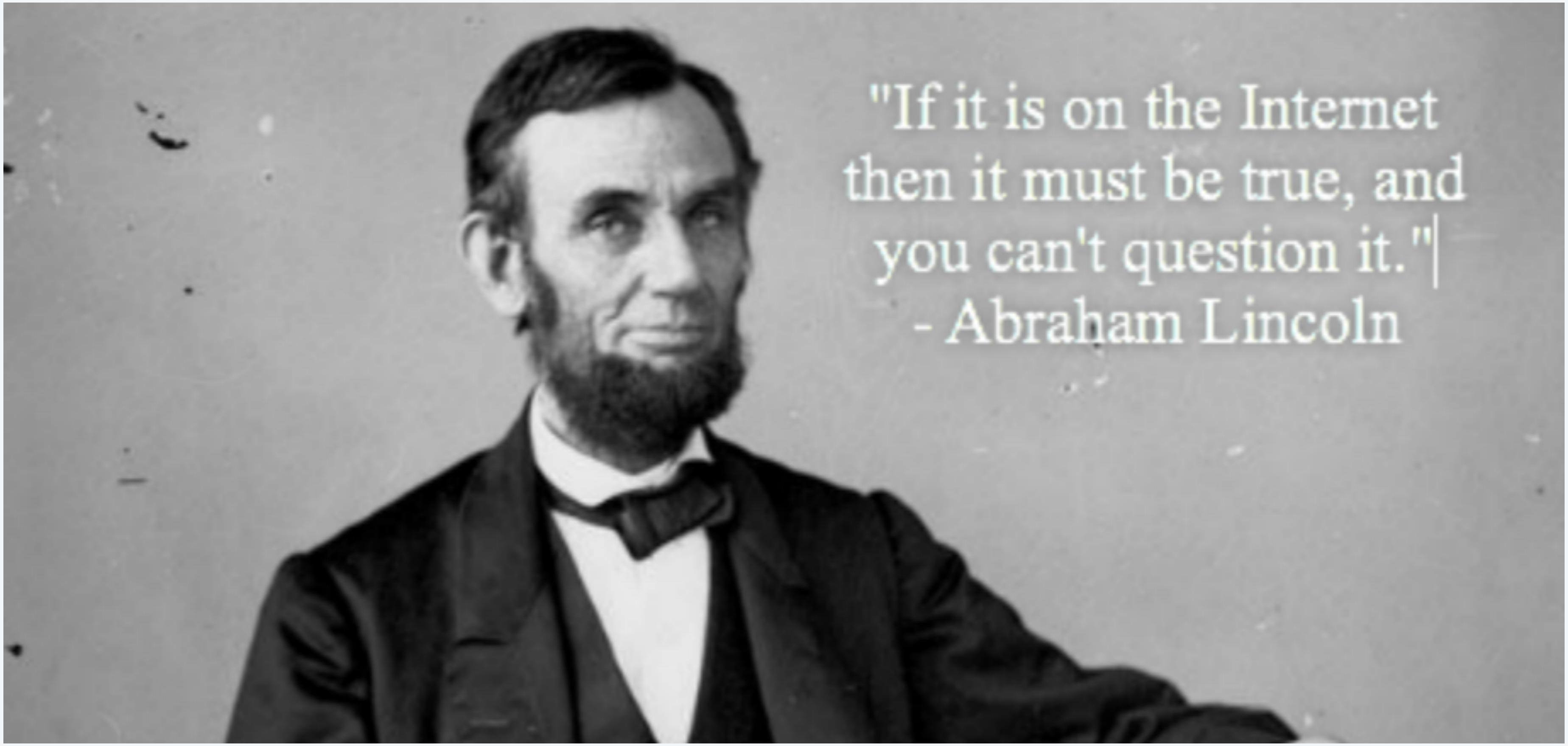
https://github.com/torvalds/linux/blob/master/include/net/tcp.h#L115

```
#define TCP_TIMEWAIT_LEN (60*HZ) /* how long to wait to destroy
TIME-WAIT * state, about 60 seconds  */
```

net.ipv4.tcp_mem

Format: `low pressure max (in pages!)`

Below `low`, Kernel won’t put pressure on sockets to reduce mem usage. Once pressure hits, sockets reduce memory until `low` is hit. If `max` hit, no new sockets.



"If it is on the Internet then it must be true, and you can't question it."

- Abraham Lincoln

REUSE

REDUCE

~~**RE**CYCLE~~

net.ipv4.tcp_tw_recycle (DANGEROUS)

- Clients behind NAT/Stateful FW will get dropped
- *99.99999999% of time should never be enabled

* Probably 100% but there may be a valid case out there

net.ipv4.tcp_tw_reuse

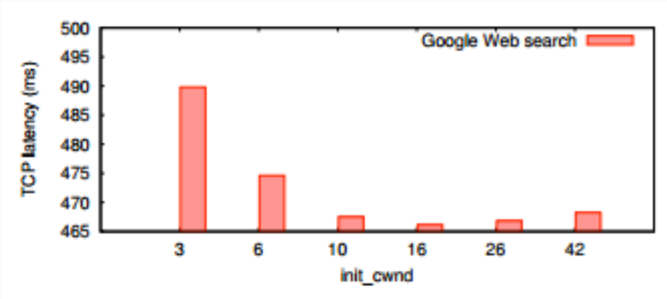
- Makes a safer attempt at freeing sockets in TIME_WAIT state.

Recycle vs Reuse Deep Dive

http://bit.ly/tcp-time-wait

One last thing...

TCP Congestion Window - initcwnd (initial)



Starting in Kernel 2.6.39 , set to **10**
Previous default was **3**!

http://research.google.com/pubs/pub36640.html

Older Kernel?

```
$ ip route change default via 192.168.1.1 dev eth0  proto static initcwnd 10
```

NGINX

listen statement

- backlog
    - limited by `net.core.somaxconn`
- defer
    - `TCP_DEFER_ACCEPT` – Wait till we receive data packet before passing socket to server. Completing TCP Handshake won’t trigger an `accept()`

server block

- `sendfile`
  - Saves context switching from userspace on read/write.
  - “zero copy” , happens in kernel space
- `tcp_nopush`
  - TCP_CORK
  - allows application to control building of packet, e.g pack a packet with full HTTP response
- `tcp_nodelay`
  - Nagle’s Algorithm
  - Only affects keep-alive connections
- `multi_accept`
  - Accept all connections on listen queue at once

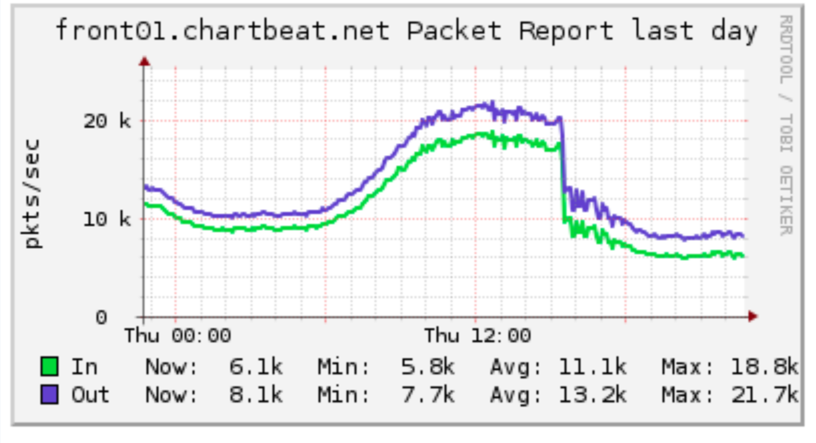
Nagle’s Algorithm (`tcp_nopush`)

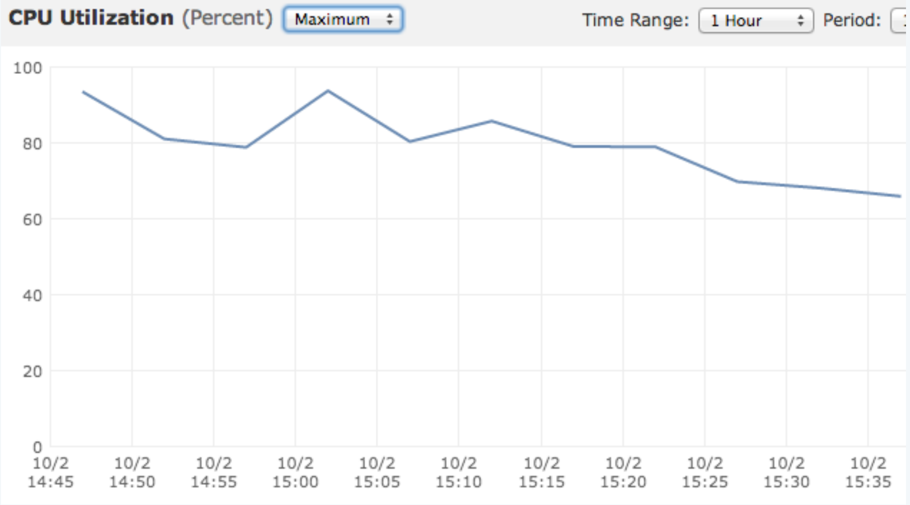
Small payload + need for low latency?

Disable

HTTP Keep-Alive

- Enabled once behind ELB
- Given small payload and 15 seconds between data, waste of resources for us to enable exposed directly to net




CPU Utilization (Percent) Maximum

Time Range: 1 Hour

Period:

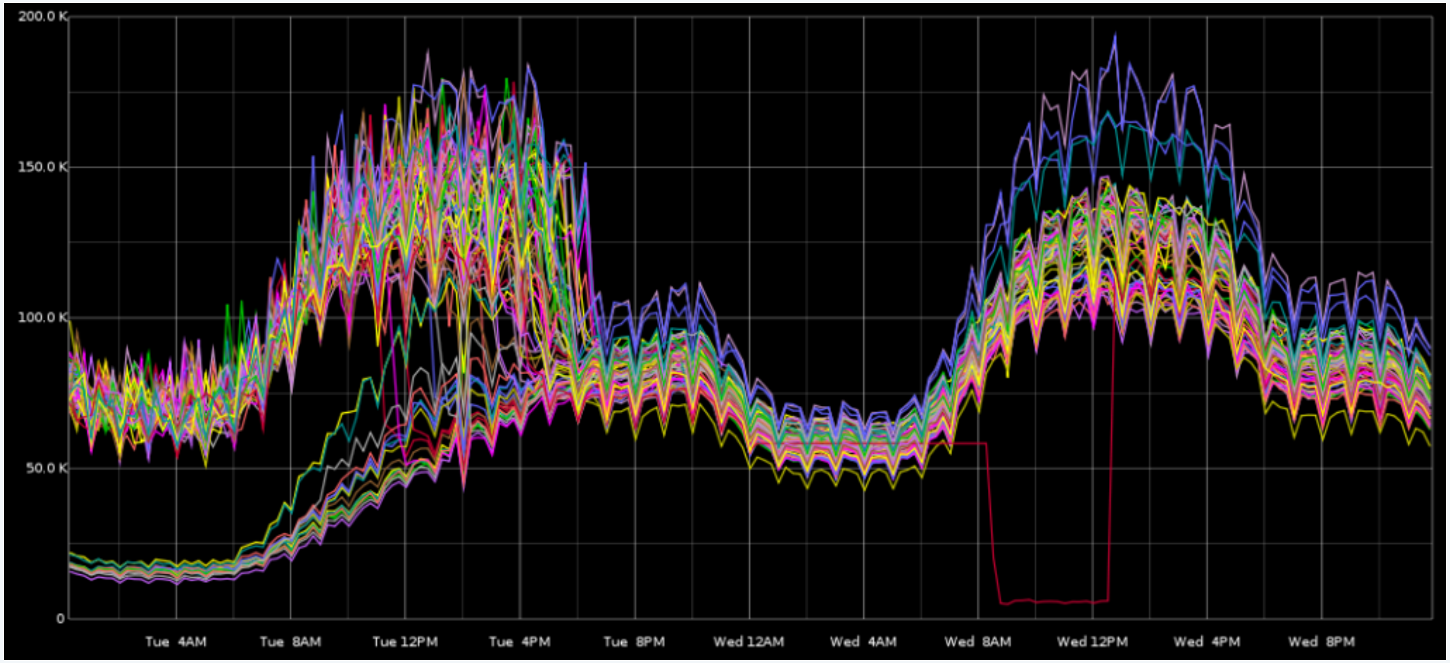
HTTP Keep-Alive Cont..

- Also enable on upstream proxies
    - Available since 1.1.4
    - *cough* had to upgrade Nginx and Fix memory leak dealing with libevent and keepalives before we could get this fully setup

ELB

Cross-Zone load balancing

Ensures requests to each ELB in each AZ go to ALL instances in ALL AZs



Idle Connection Timeout

- Defaults to 60 seconds
- Finally tunable via API.
- Tweak if doing anything long lived , e.g. Websockets, or ensure you are sending “pings”

Connection draining

“Graceful” removal of node from ELB, will ensure existing connections can finish instead of a hard cutoff (old behavior)

Metrics to monitor

- SurgeQueueLength (Not Good)

A count of the total number of requests that are pending submission to a registered instance.

- SpilloverCount (BAD)

A count of the total number of requests that were rejected due to the queue being full.

Conclusions

- The internet is full of lies
- With enough traffic, tweaking system and application defaults are a necessary
- Find trusted sources (Me? Maybe?) for settings and test in staging environments
- Measure impact and understand what metrics may be impacted by your tweaks
- Don’t get lost in all the `sysctl` settings
- TCP is complicated

FIN

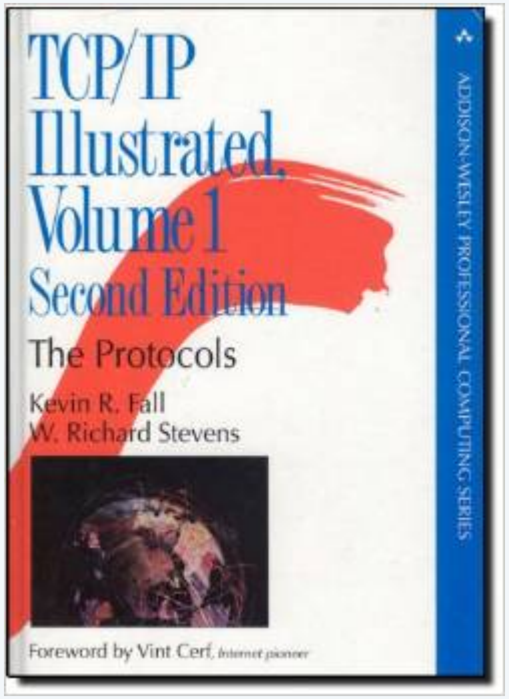
FIN_WAIT_1

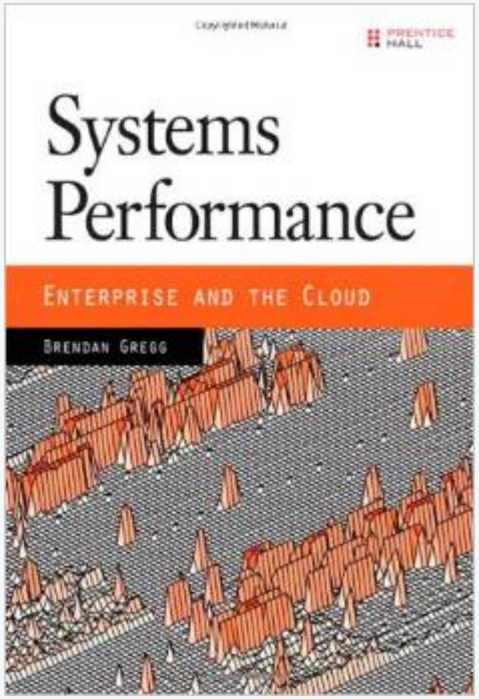
FIN_WAIT_2

TIME_WAIT

Resources and References

https://www.kernel.org/doc/Documentation/networking/ip-sysctl.txt


TCP/IP Illustrated, Volume 1 Second Edition
The Protocols
Kevin R. Fall
W. Richard Stevens
Foreword by Vint Cerf, Internet pioneer
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES



```
man tcp(7)
```

Additional reading

http://engineering.chartbeat.com

Full story about experiences with our architecture and material discussed in slides

Questions / Comments?

@Lintzston

justin@chartbeat.com