Intelligent Source Buffer Management to Minimize Total Network Latency

Stuart Cheshire, Apple

Tuesday 9th December 2024

Source-Device Bufferbloat

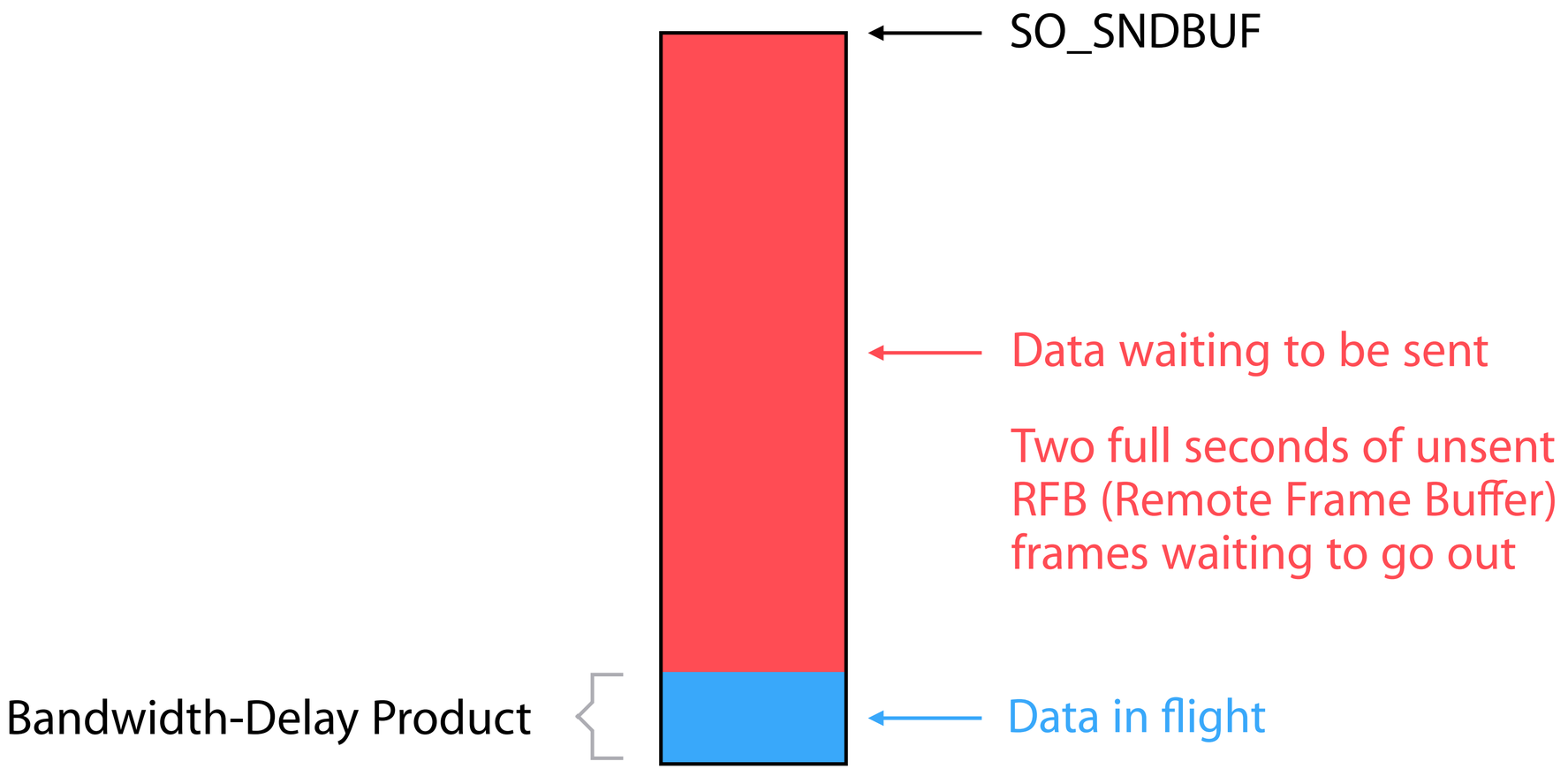
April 2011

Mac OS Screen Sharing sluggish on slow networks

Network Bufferbloat suspected

Real cause was excessive buffering by the sender

Sluggish Screen Sharing



TCP_NOTSENT_LOWAT

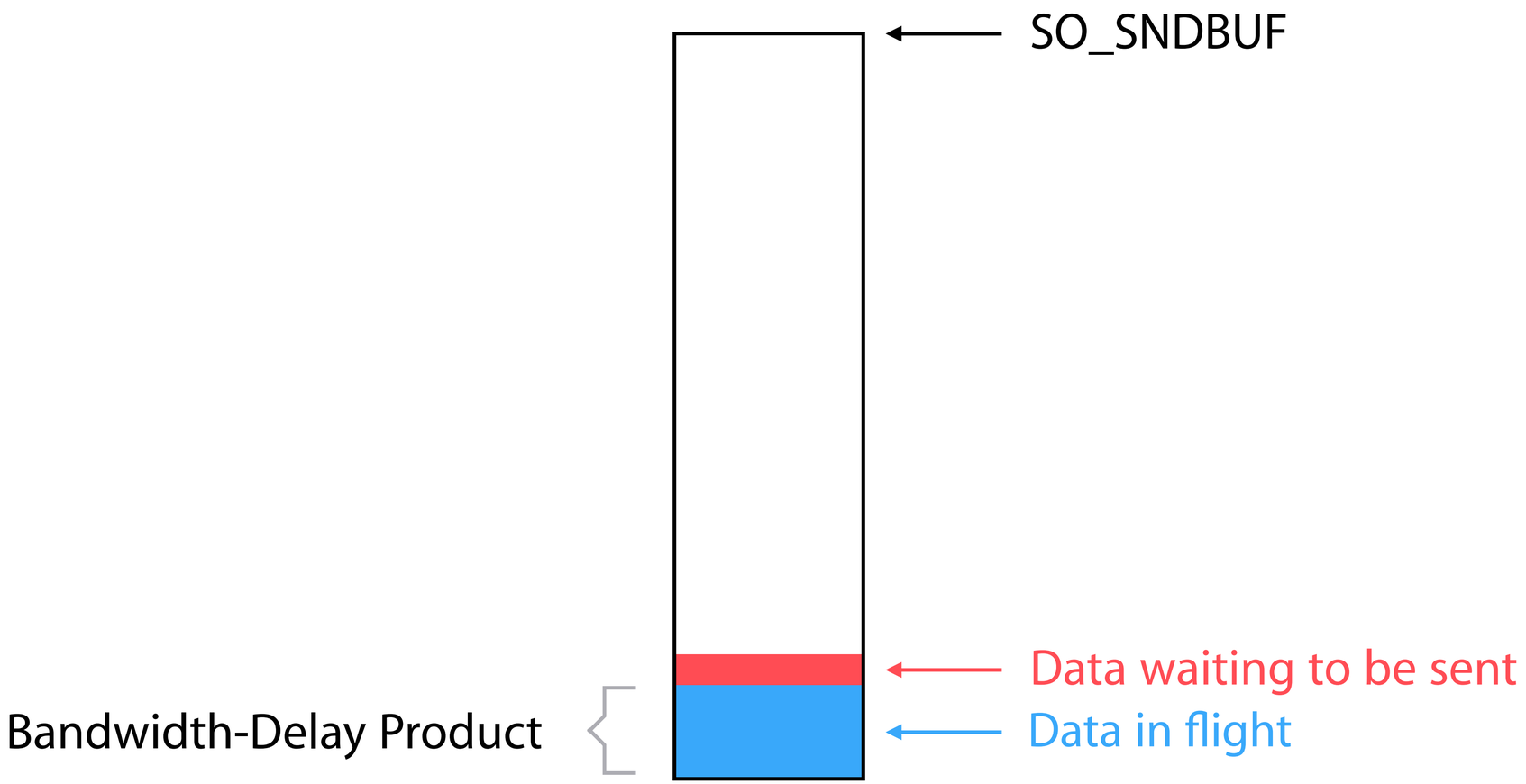
May 2011

TCP Not-Sent Low-Water Mark

kevent() doesn’t signal application to generate a new compressed frame
until TCP is almost ready to need more data

Fixes excessive sender-side buffering for real-time delay-sensitive applications

Snappy Screen Sharing
Using TCP_NOTSENT_LOWAT



Screen Sharing with TCP_NOTSENT_LOWAT

June 2015 Apple Worldwide Developer Conference

On-stage demo

WWDC 2015 Session 719 “Your App and Next Generation Networks”

Improving TCP_NOTSENT_LOWAT

Low-Water Mark was specified in bytes

16 kilobytes (about ten Ethernet frames) works pretty well, but…

- Can be too much on low-rate networks (e.g., 250 kb/s and less)
- Can be too little on high-rate networks (e.g., Gb/s and above)
- Would be better if specified in time (milliseconds, or microseconds) indicating how much notice the application needs to generate next chunk of data

Improving TCP_NOTSENT_LOWAT

Different on Linux

Bob McMahon adding working-latency measurement to iperf2

- Used TCP_NOTSENT_LOWAT to minimize source delays
- On Linux, throughput was severely limited
- Discovered that Linux implementation has different semantics

Improving TCP_NOTSENT_LOWAT

Different on Linux

On Mac OS and iOS, socket option determines low-water mark

- When unsent backlog falls below low-water mark, application is *signaled* (e.g., via kqueue) to generate more data
- Application can then atomically write as much as makes sense for that application, up to SO_SNDBUF

On Linux, socket option determines high-water mark

- Application is *prevented* from writing more than high-water mark
- Can severely reduce throughput if TCP_NOTSENT_LOWAT set to 16 kB

Updating TCP_NOTSENT_LOWAT

Goals

Consistent behavior across all operating systems and network stacks

Indicate application notification requirement in units of time, not bytes

Also specify equivalents for other transport protocols, like QUIC,
and other network APIs not based on BSD Sockets

- Media over QUIC (MOQ) could really use this, to avoid unnecessary delays

TCP_REPLENISH_TIME

Replacement for TCP_NOTSENT_LOWAT

New IETF work on Source Buffer Management

New Internet Draft: draft-cheshire-sbm-00

New IETF mailing list requested: sbm@ietf.org (If approved, should be active in a few days)

To be presented at Transport and Services Working Group (TSVWG) meeting
at IETF 122 in Bangkok, March 2025

TCP_REPLENISH_TIME

Call to Action

Watch for announcement of sbm@ietf.org mailing list

Read draft-cheshire-sbm and send feedback

Join TSVWG meeting in Bangkok

Implement TCP_REPLENISH_TIME in your networking stacks

Use TCP_REPLENISH_TIME in your networking applications

Spread the word about how to reduce delays (and kernel memory usage!)

Intelligent Source Buffer Management to Minimize Total Network Latency

Stuart Cheshire, Apple

Tuesday 9th December 2024