Effects of PRR after Slow Start

tcpm – 87. IETF Berlin – July 30, 2013

Mirja Kühlewind <mirja.kuehlewind@ikr.uni-stuttgart.de>

Overview

Proportional Rate Reduction for TCP (RFC 6937 exp)

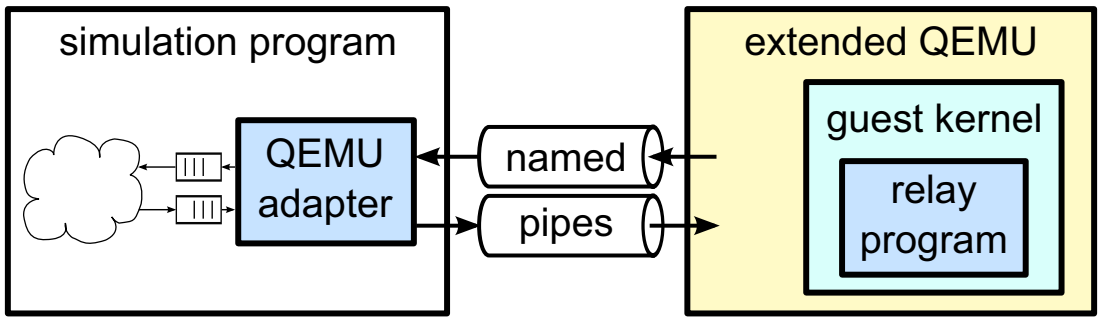
- Alternative to Fast Recovery/Rate-Halving
- Goal: Actual congestion window (cwnd) size at the end of reduction should be as close as possible to ssthresh
- Implemented in Linux kernel since version 3.2

Simulation Study with Linux kernel version 3.9.0 (PRR)

- (Correct) reduction to half the cwnd after exponential increase in Slow Start causes further congestion
- Burst sending after large number of losses possible

Simulation Framework

Event-driven Simulation using Real Kernel Code



Integration of virtual machines (using QEMU) in the IKR SimLib

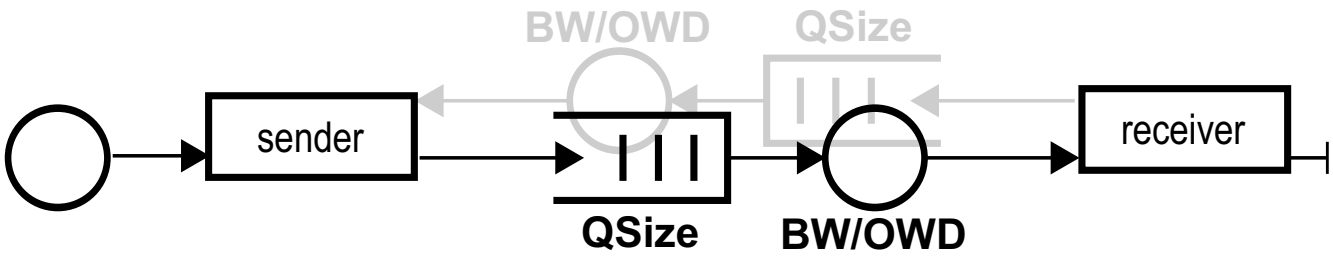
- Java-based, event-driven simulation framework (IKR SimLib)
- Kernel code integration by using a Linux OS in a virtual machine (QEMU)
- Control of timing and interfaces for event handling in the simulation program
- Relay program in the guest OS to generate TCP traffic
- Real TCP/IP packets with dummy data are forwarded over simulated network

→ Simulation framework with realistic TCP behavior

→ Easy updates to latest kernel versions possible

Simulation Model

Single Connection

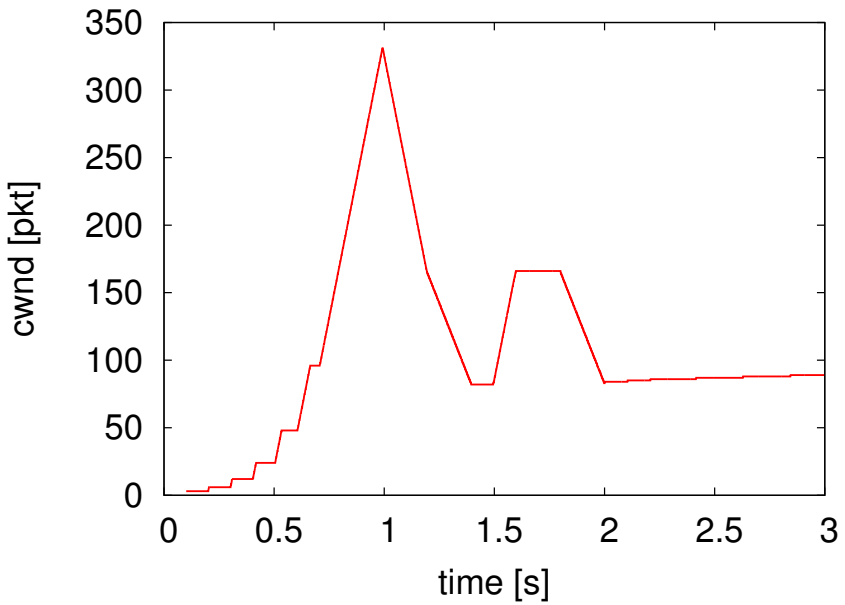


- Bottleneck bandwidth: 10 Mbps
- One-Way-Delay: 50ms (after bottleneck queue)
  → Loss notification after one RTT of delay at sender (worst case)
- Queue size: 125000 Byte (Bandwidth-Delay-Product)
- Congestion Control: TCP Reno

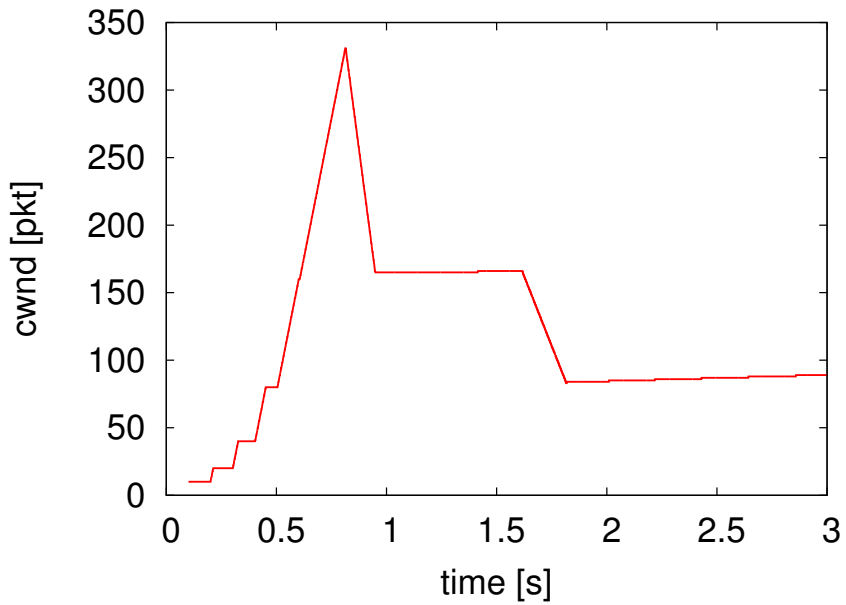
- Receive Window Limitation (2. Scenario):
  net.ipv4.tcp_rmem = 8192 109312 3497984

PRR Reduction after Slow Start

Linux 2.6.35.16 (Rate-Halving)



Linux 3.9.0 (PRR)



→ With PRR after Slow Start cwnd is reduced exactly to the sending rate when the loss happened (as cwnd was doubled during last RTT until loss was notified at receiver)

→ First increase (of 1 packet) in Congestion Avoidance leads (again) to packet loss

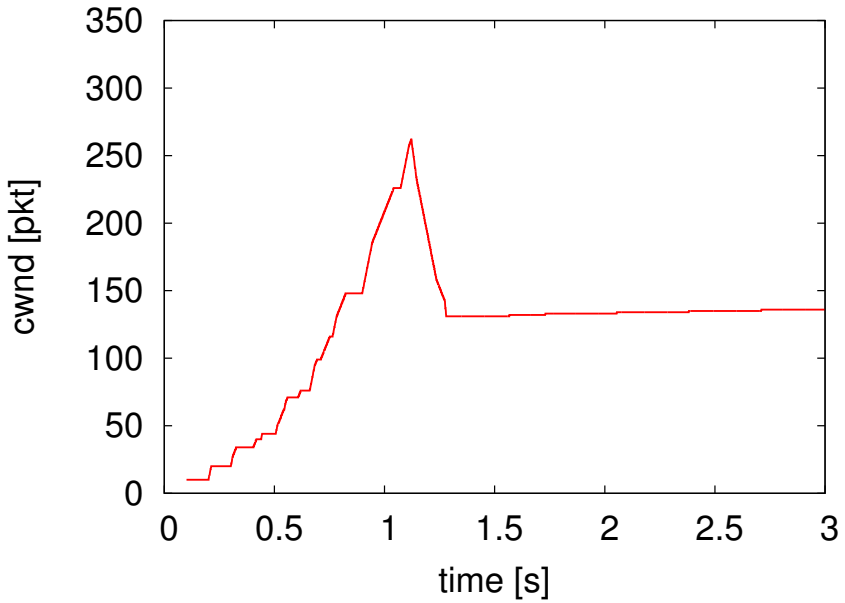
→ Reduction is performed one RTT later (incl. max. queuing delay)

→ Congestion and thus a full buffer remains longer with PRR

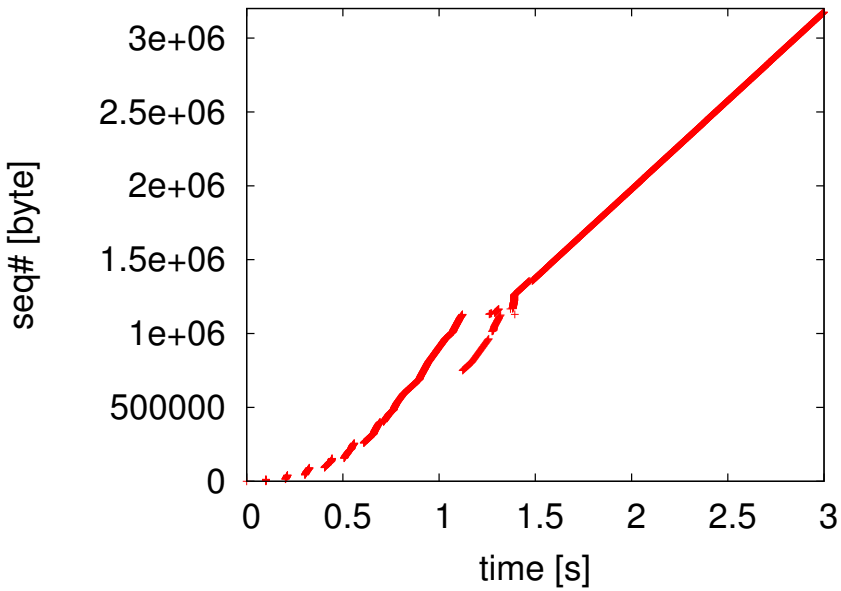
→ Simple solution approach: reduce cwnd to 1/4 after Slow Start..?

PRR after Slow Start with rcv Window Limitation

Linux 3.9.0 (PRR)



Linux 3.9.0 (PRR)



→ Receive window limitation during Slow Start causes different behavior and smaller maximum cwnd

→ Cwnd can grow much larger than flight size

→ Large burst of packets is sent out at once when accumulated ACK is received (here: 27 pkts)

→ Implementation problem?

→ General burst limitation needed? Or pacing?