ECN and TCP Slow Start – A motivation for a more scalable congestion control

ConEx – IETF Maastricht – 27th July 2010

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

University of Stuttgart, Germany

This work is partly funded by the German Research Foundation (DFG) through the Center of Excellence (SFB) 627 "Nexus".

Outline

Implementation issues of re-ECN

→ Input for the specification of ConEx modification to TCP

Evaluation in different simulation scenarios

→ Input for ConEx use-cases document

- Measuring of exposed congestion
- Motivating less aggressive congestion behaviour

Implementation of re-ECN

Implementation

- On **Linux kernel 2.6.26** according to **draft-briscoe-tsvwg-re-ecn-tcp-08**
- Counting on a per-packet base; but no re-ECN mark after lost packets
- Modes: Not-ECT, RECN and RECN-Co

Required Modifications

- TCP ECN-Nonce has not been implemented: new **TCP flag** introduced
- **Usage of the reserved bit 48** (when DF-flag is set, often the whole 8 bits get reset...)
- Modifications on all ECN methods (mostly separate methods or even own files)
- Some modifications in other methods (e.g. tcp_ack(...) fast path processing)

→ **Most of the re-ECN processing was simple to realize**

Open issues

- Handling of **IP-Fragmentation** and **GSO/TSO** is not specified
- **SYN-Cookies** are not implemented for ECN in Linux
- Recommendations on FNE packets (see later)

Evaluation of re-ECN

Kühlewind, M., Scharf, M.: Implementation and Performance Evaluation of the re-ECN Protocol. To appear in: Proc. Economic Traffic Management Workshop 2010, Sep., Amsterdam (2010)

→ **Focus here on ECN marking and TCP Slow Start effects**

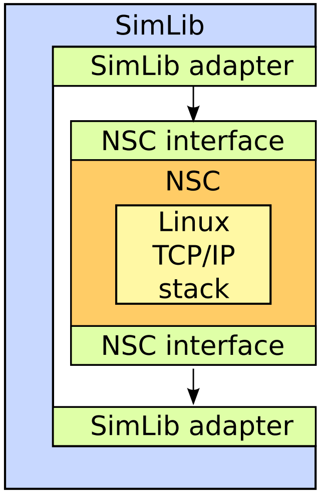
Simulation Setup

- Network Simulation Cradle (NSC):

  http://research.wand.net.nz/software/nsc.php
- Simulation Library of Institute of Communication Networks and Computer Engineering, University Stuttgart:

  http://www.ikr.uni-stuttgart.de/en/Content/IKRSimLib/

→ Event-driven simulation with real Linux kernel code



Overview

Three scenarios

- **Simple scenario:** One sender, one receiver, RED Router and bulk traffic transfer

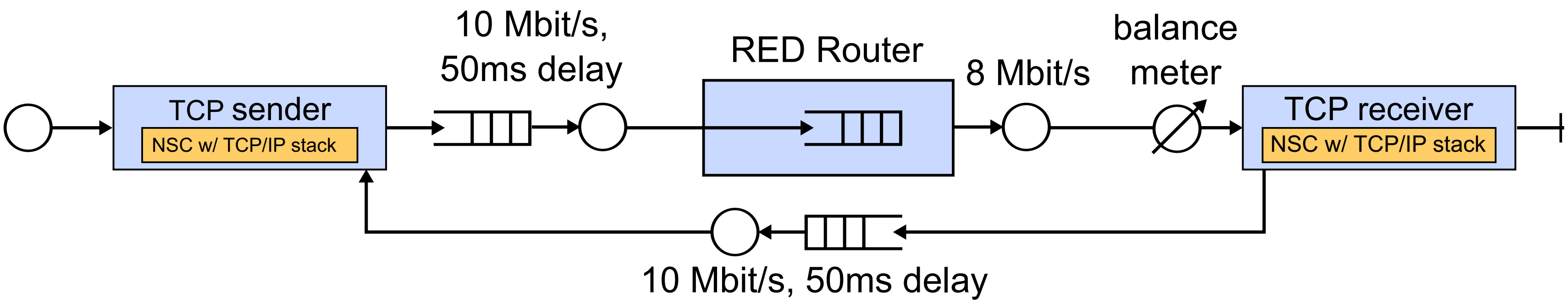
  → How does re-ECN look like?
- **CBR cross traffic:** Several runs with variation in data rate

  → Does Congestion Exposure work?
- **Internet traffic traces:** Replay with an neg. exp. Inter Arrival Time (IAT) and all connections sharing one bottleneck

  → Slow-Start effects – The need for a more scalable Congestion Control?

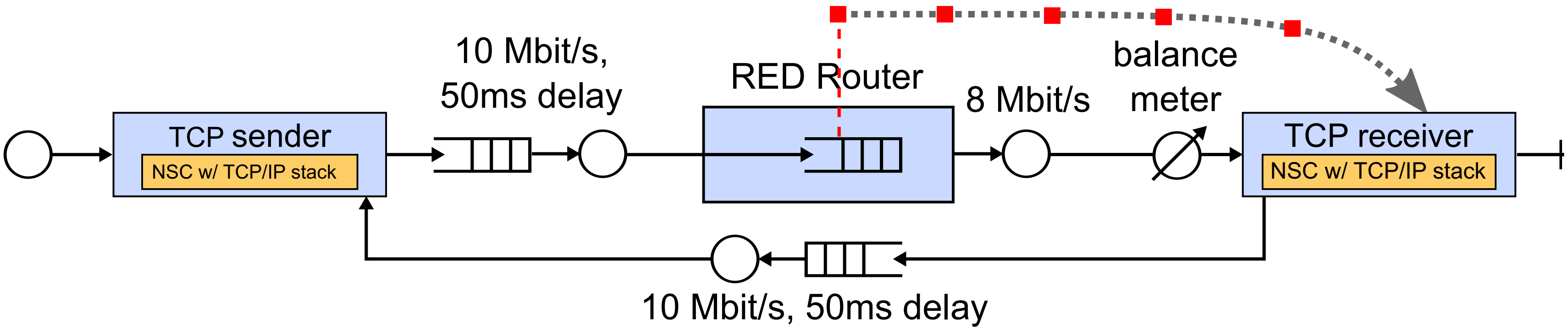
Scenario 1

One Sender, one Receiver, RED Router, Bulk Traffic



Scenario 1

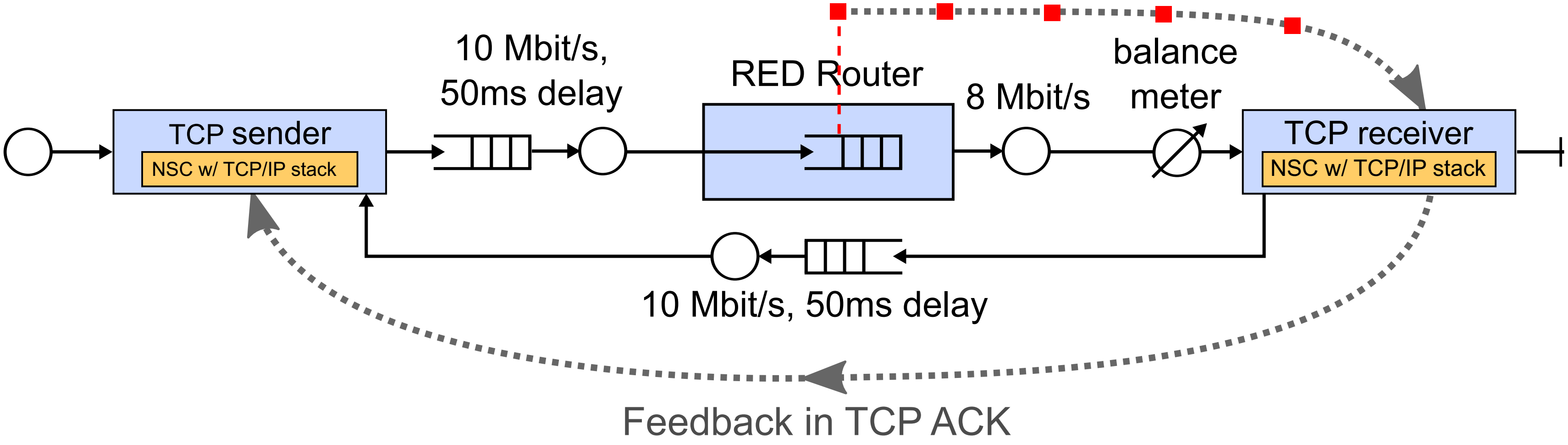
One Sender, one Receiver, RED Router, Bulk Traffic



- Router marks packets as negative (debit – red ECN marks) with certain marking probability depending on the queue length (RED)

Scenario 1

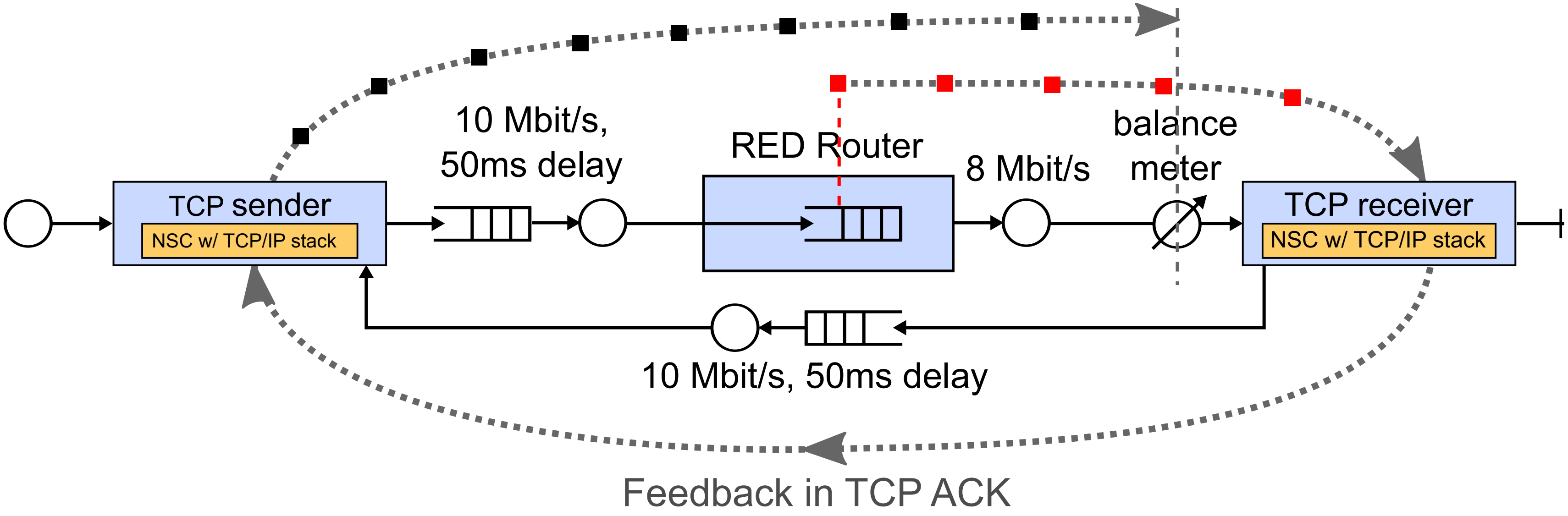
One Sender, one Receiver, RED Router, Bulk Traffic



- Router marks packets as negative (debit – red ECN marks)
  with certain marking probability depending on the queue length (RED)
- Receiver feeds this signal back to the sender

Scenario 1

One Sender, one Receiver, RED Router, Bulk Traffic



- Router marks packets as negative (debit – red ECN marks)

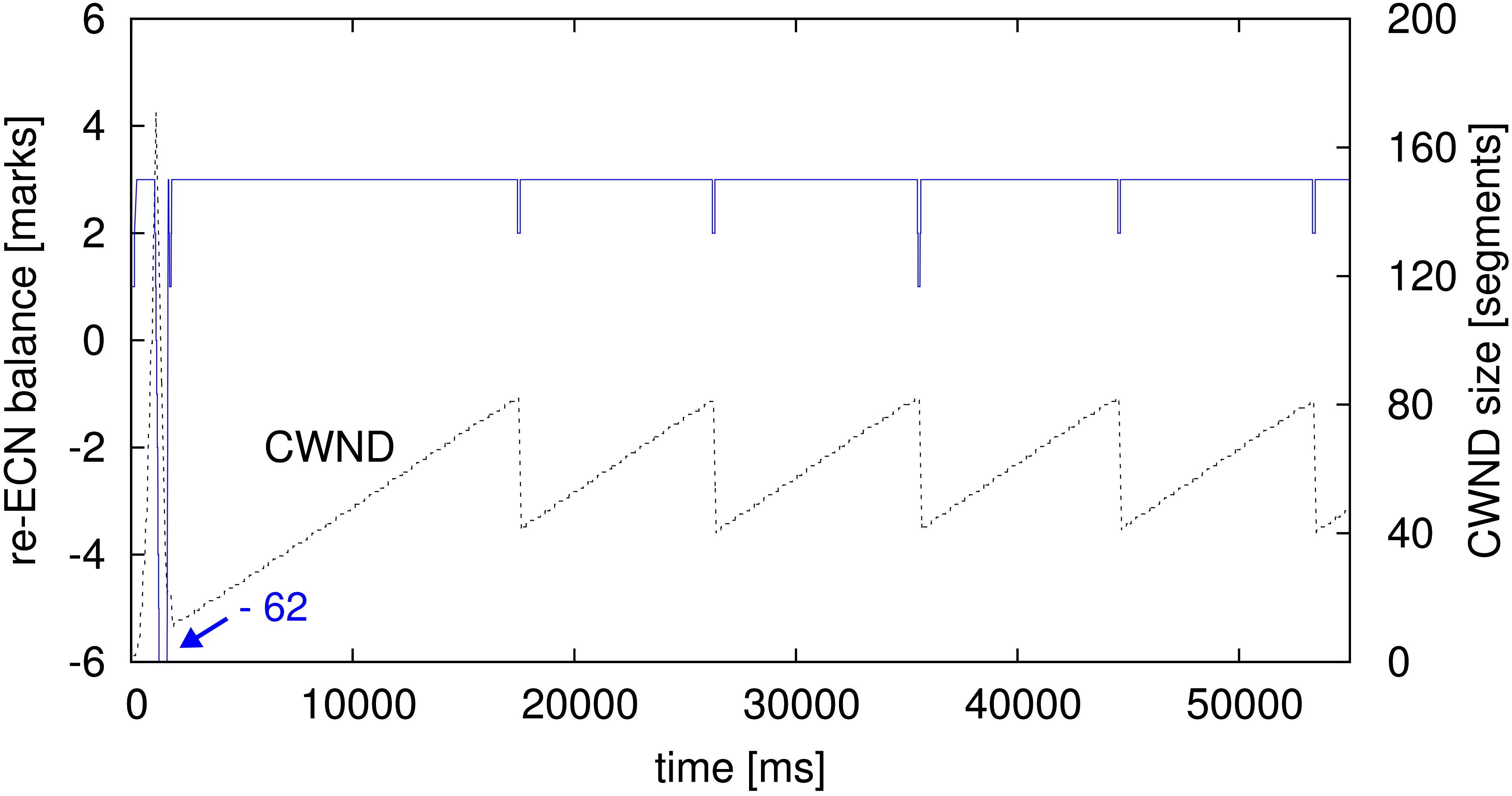
  with certain marking probability depending on the queue length (RED)
- Receiver feeds this signal back to the sender
- Sender re-inserts this information as positive marks (credit – black re-ECN marks)

  and reacts with TCP Congestion Control (Reno) as specified with ECN

→ Number of negative and positive packets should be about the same at network egress (balance meter)

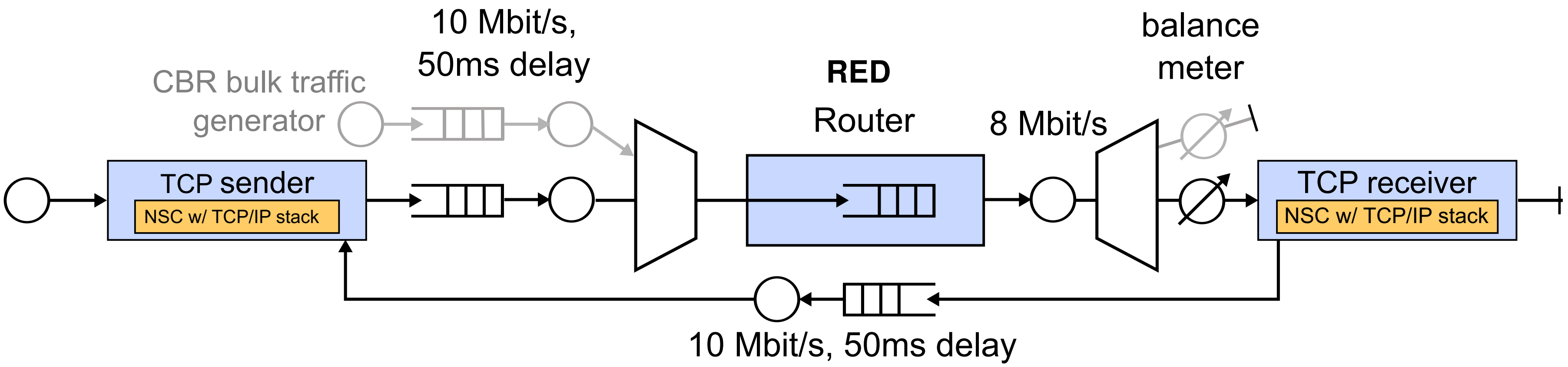
re-ECN Balance of a TCP Connection at Egress

→ How does a congestion signal look like?



Scenario 2

CBR Cross Traffic, RED Router

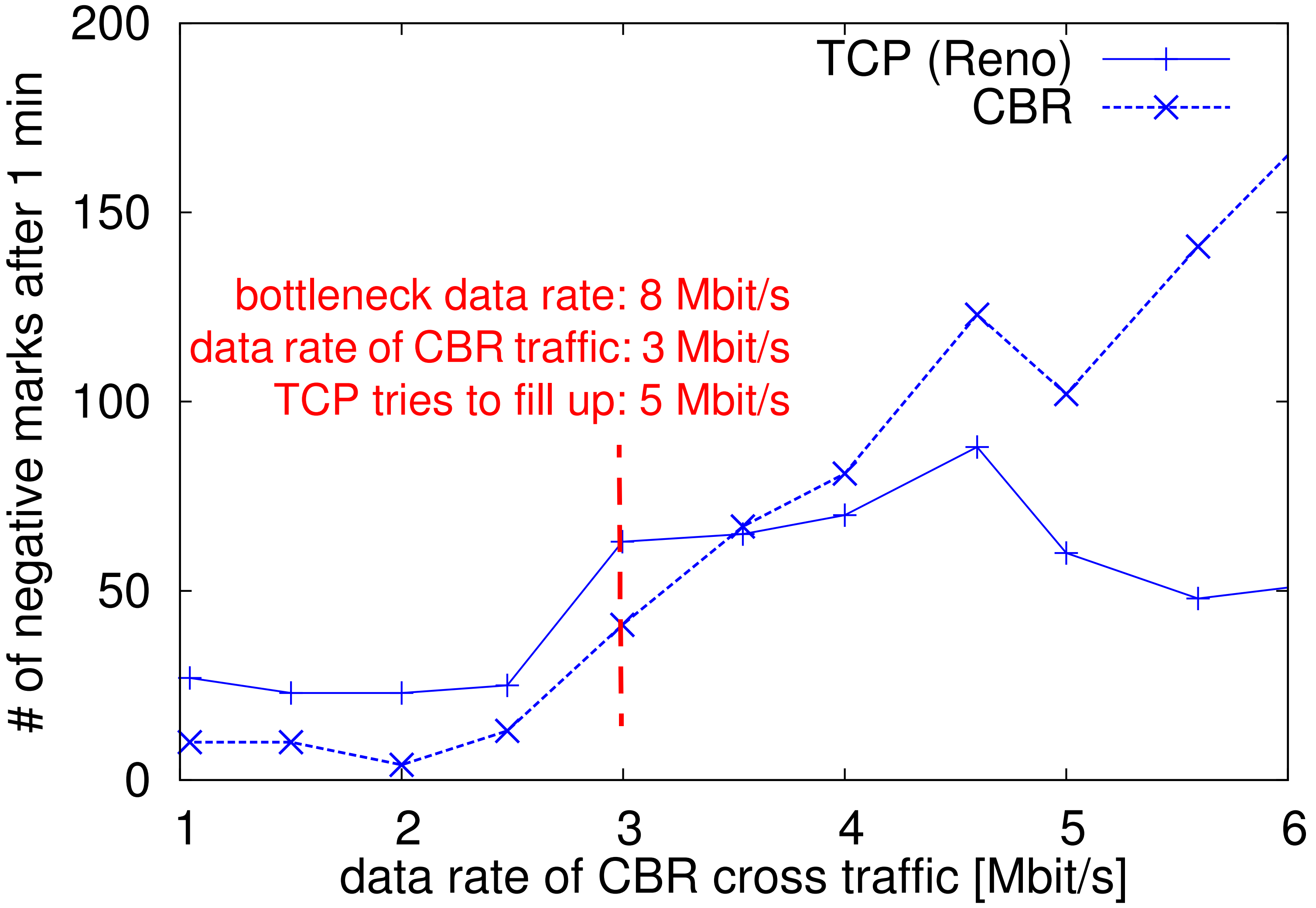


Several simulation runs with 1 minute transmission time each

- Bottleneck capacity of 8 Mbit/s will be shared between one TCP (Reno) and one constant bit rate (CBR) connections (2 connections in total)
- Different date rate for the CBR cross traffic in each run
- TCP connection tries to fill up the rest of the capacity

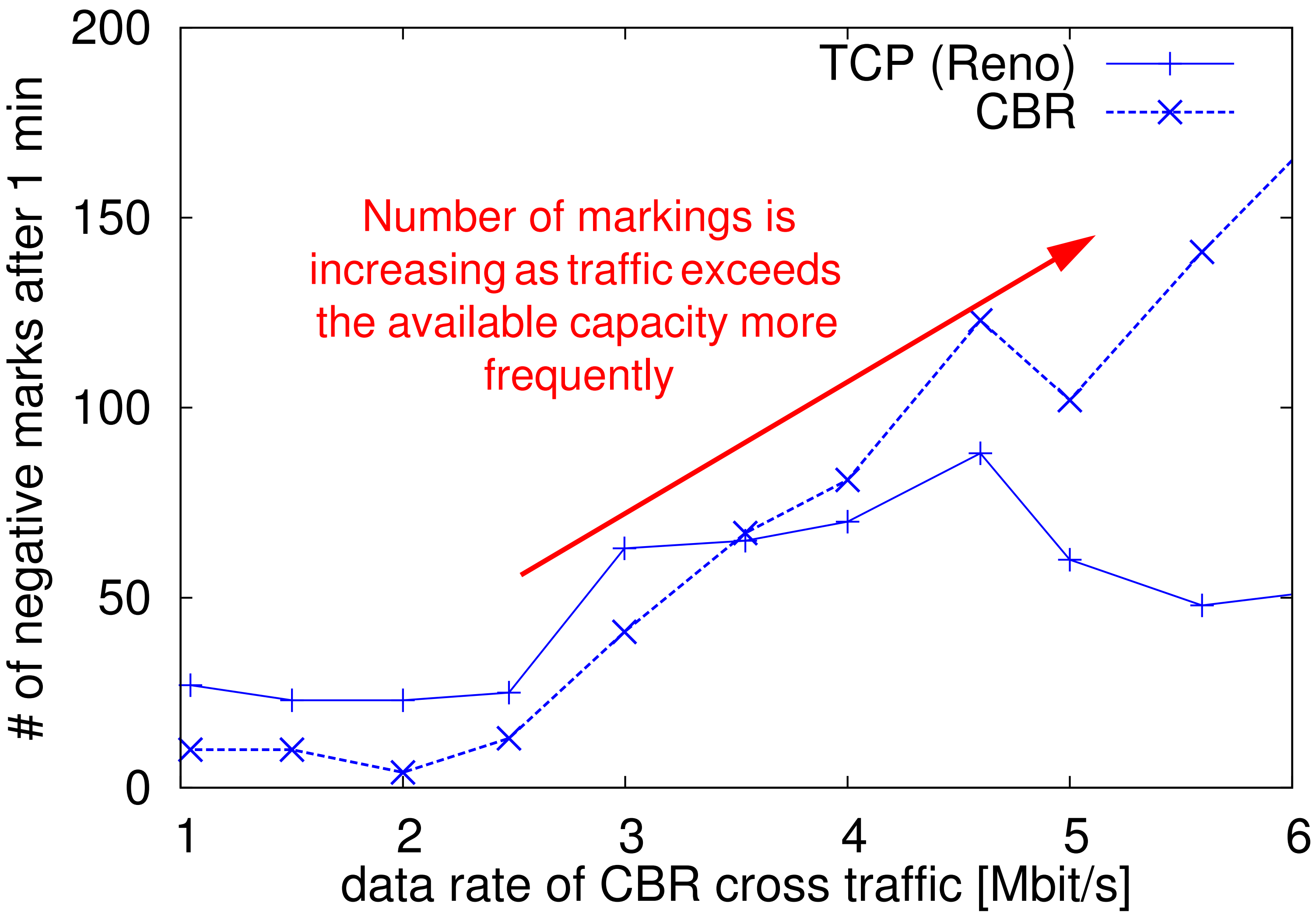
Number of Negative Marks with CBR Cross Traffic

TCP Reno Sender, 8 Mbit/s Capacity



Number of Negative Marks with CBR Cross Traffic

TCP Reno Sender, 8 Mbit/s Capacity

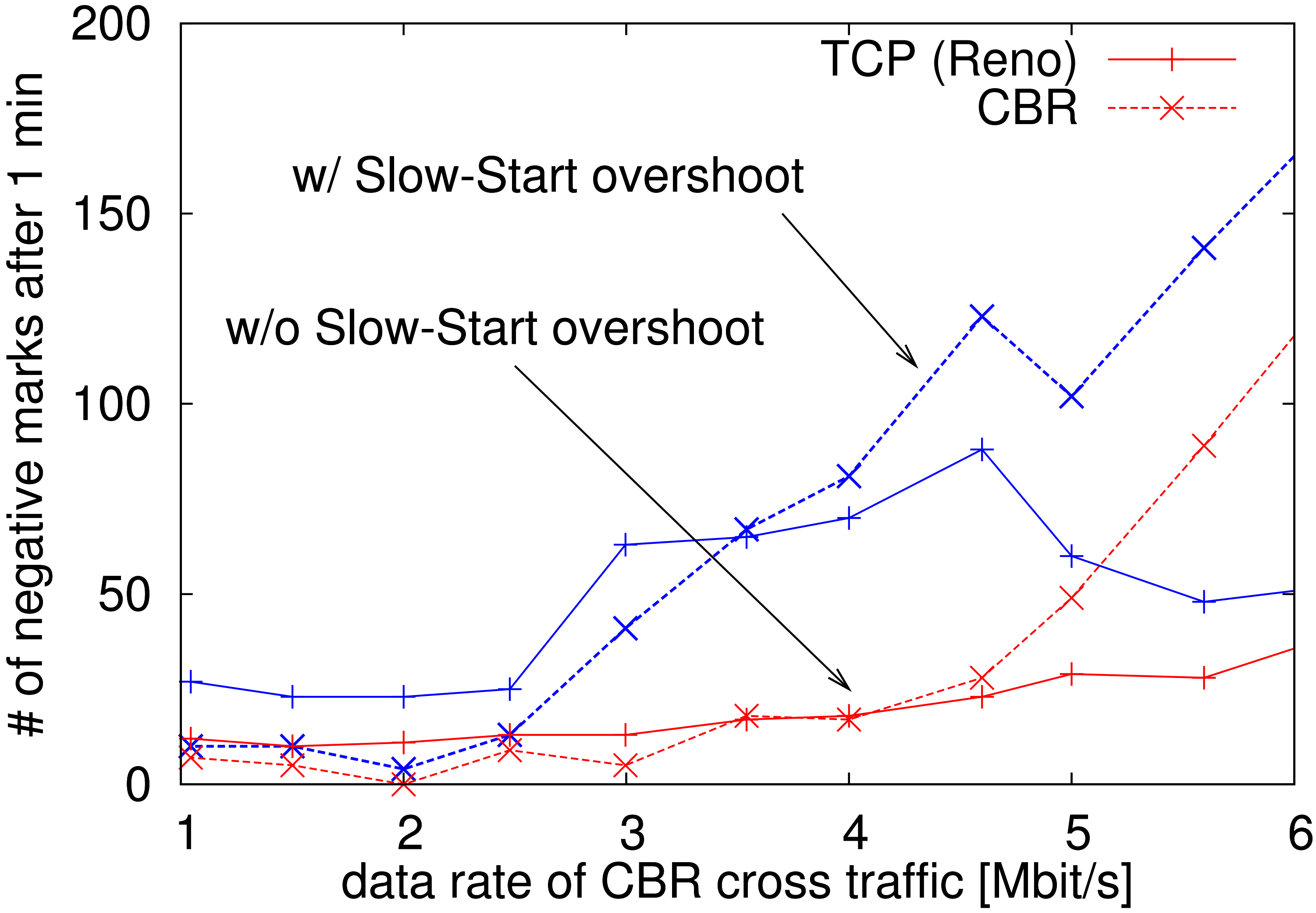


Number of markings is increasing as traffic exceeds the available capacity more frequently

→ **re-ECN works as a Congestion Exposure protocol!**

Number of Negative Marks with CBR Cross Traffic

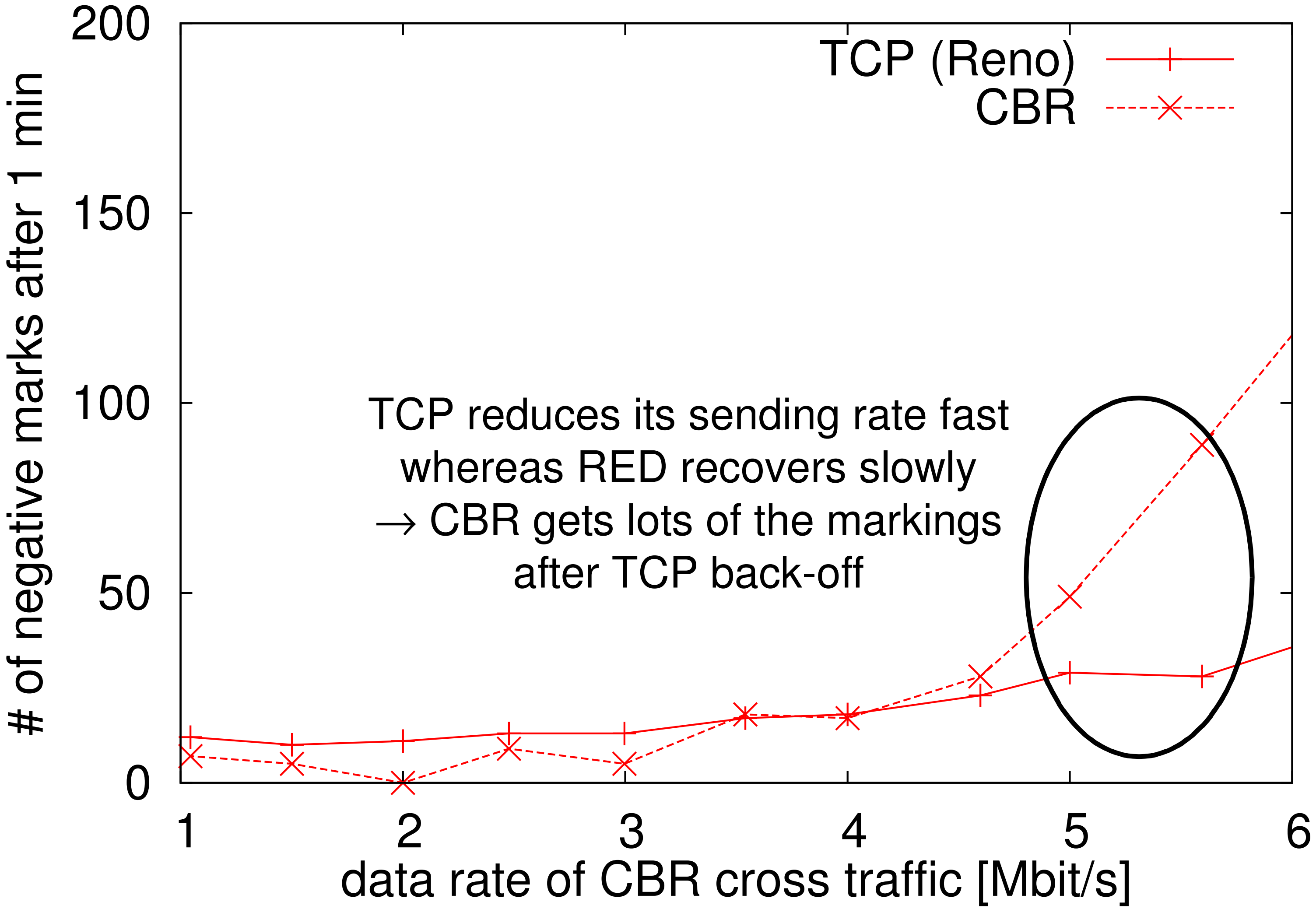
TCP Reno Sender, 8 Mbit/s Capacity



→ Most of the markings are caused by TCP Slow-Start overshoot

Number of Negative Marks with CBR Cross Traffic

TCP Reno Sender, 8 Mbit/s Capacity



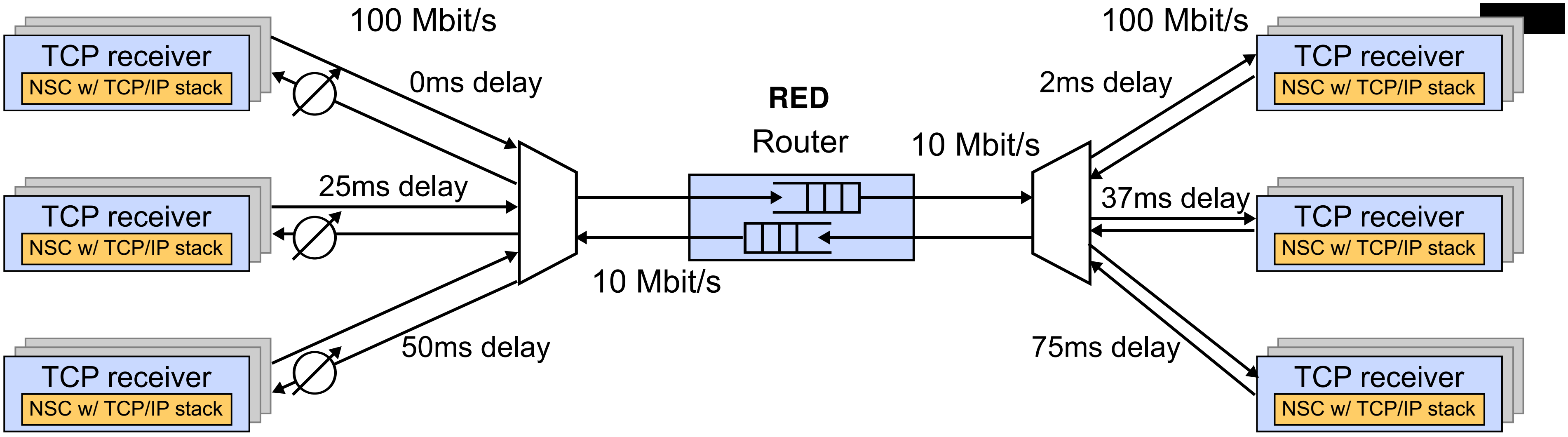
TCP reduces its sending rate fast
whereas RED recovers slowly
→ CBR gets lots of the markings
after TCP back-off

→ CBR cannot adapt to congestion while TCP backs off quickly

Scenario 3

Replay of Traffic Traces with a neg. exp. distributed IAT of 100ms

→ Mean load of 45% in one hour simulation time at bottleneck (RED Router)



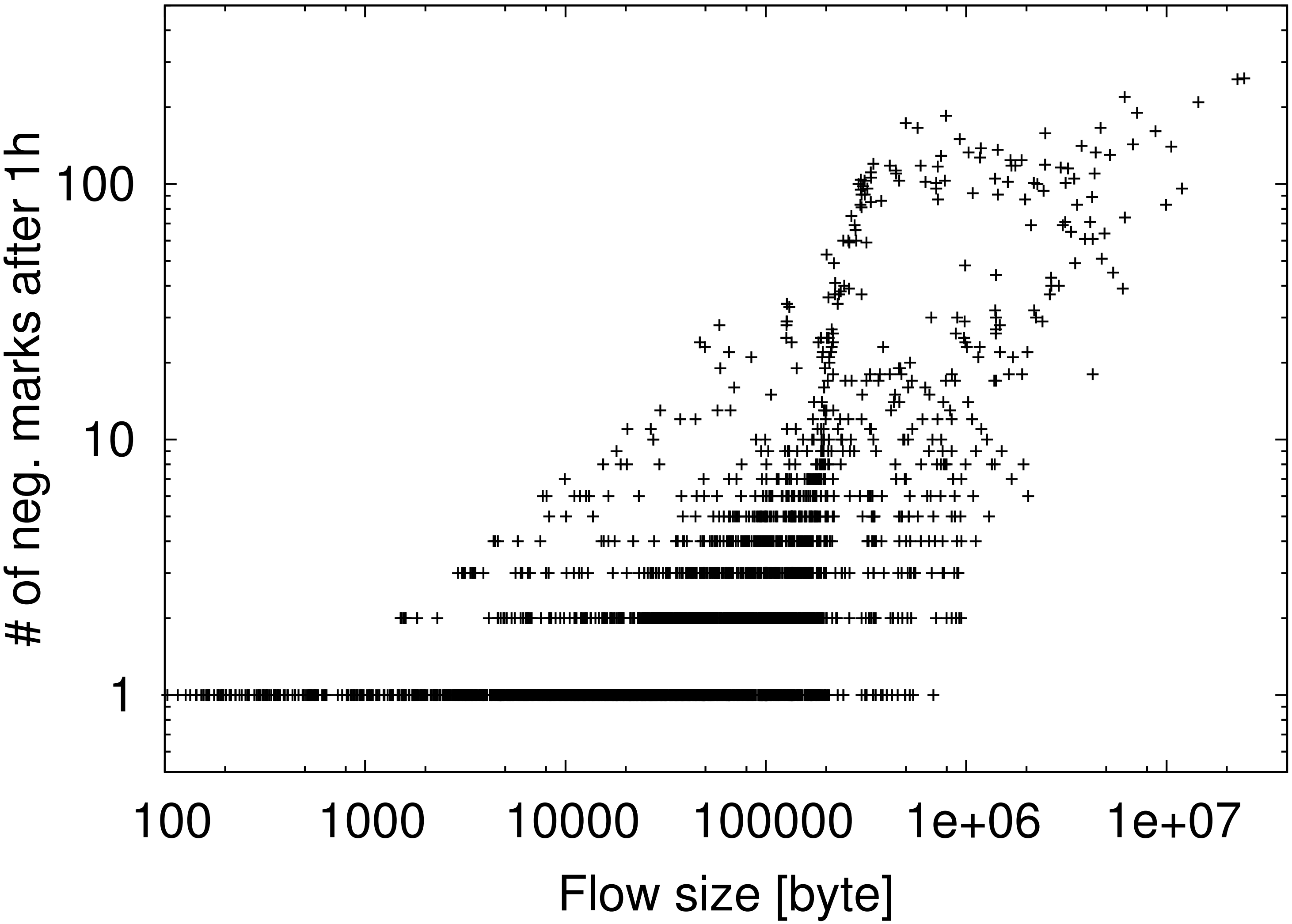
[redacted]

Recommended in Andrew, L., Marcondes, C., Floyd, S., Dunn, L., Guillier, R., Gang, W., Eggert, L., Ha, S., Rhee, I.: Towards a common TCP evaluation suite. In: Proc. PFLDnet. (2008)

Traffic Traces form Website: WAN in Lab – Traffic Traces for TCP Evaluation. http://wil.cs.caltech.edu/suite/TrafficTraces.php

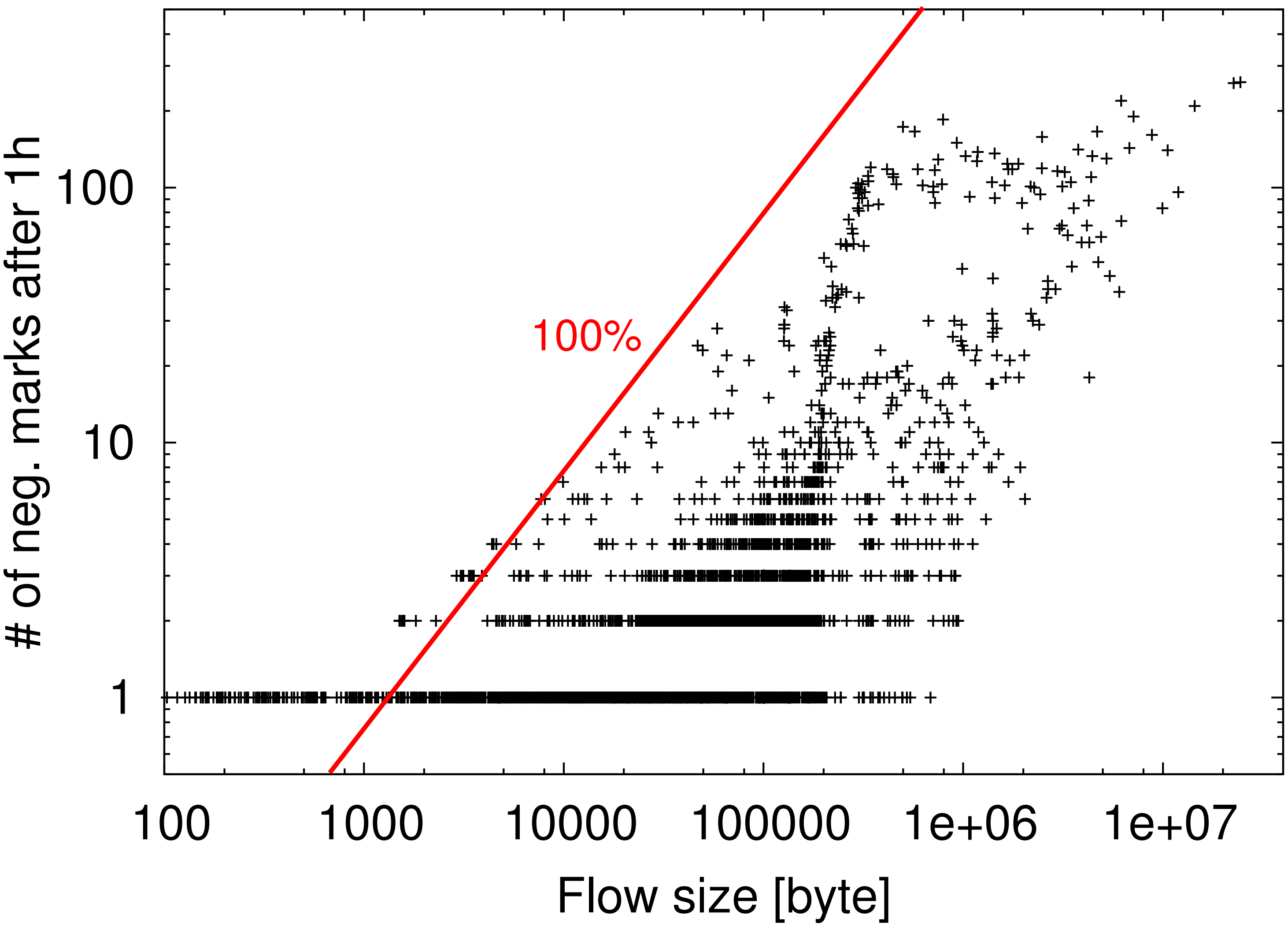
Number of Negative Marks over Flow Size

One point for every completed flow



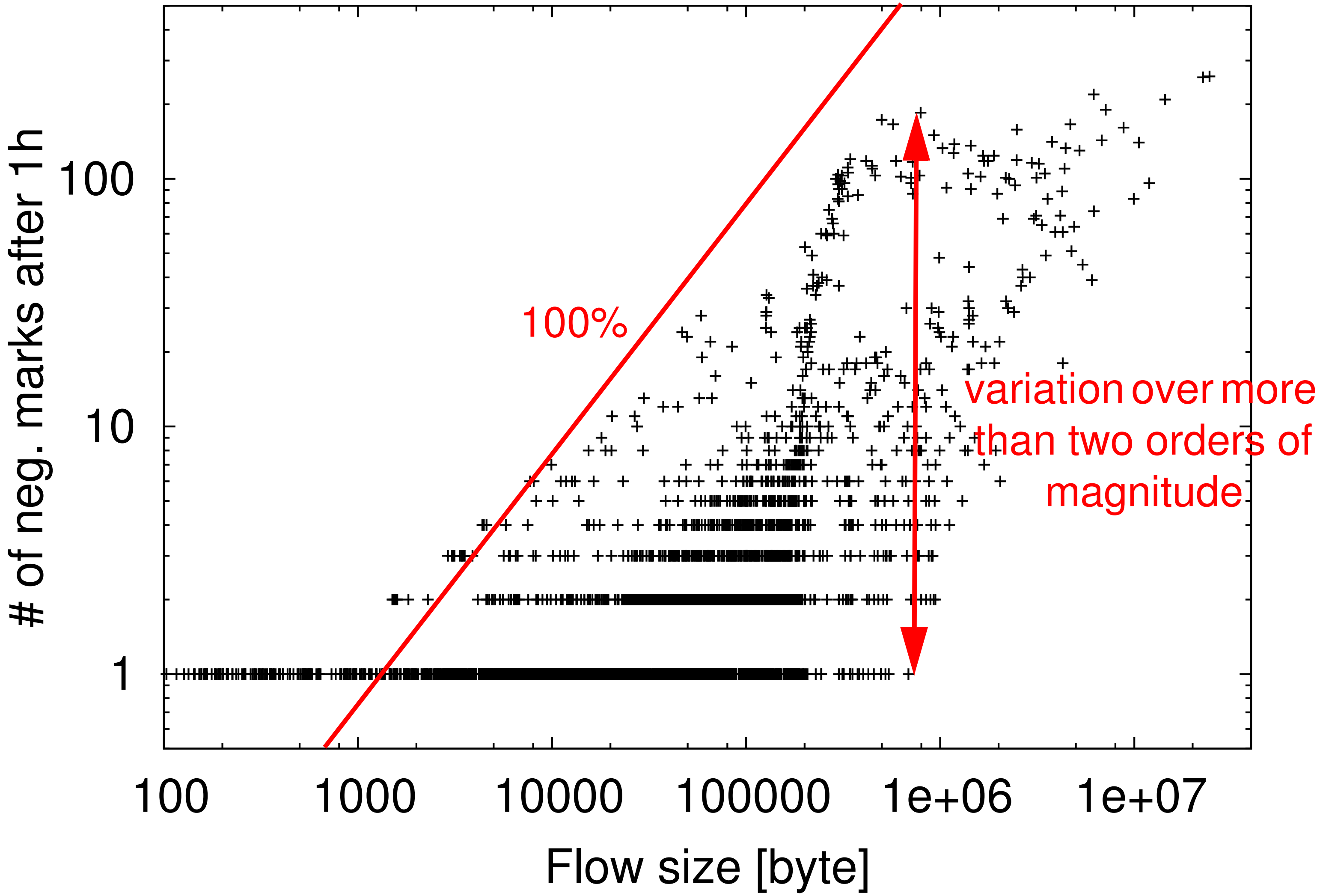
Number of Negative Marks over Flow Size

One point for every completed flow



Number of Negative Marks over Flow Size

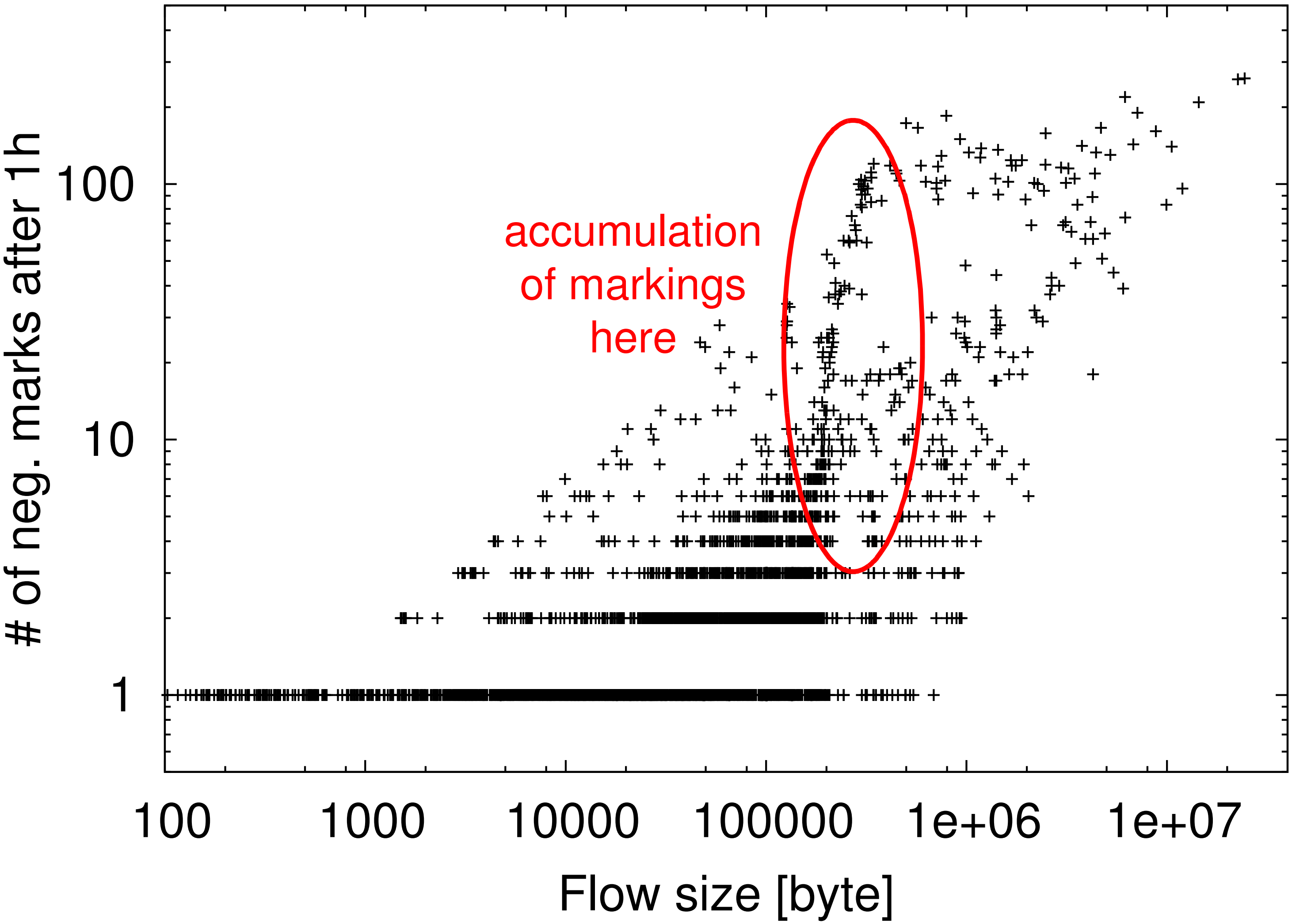
One point for every completed flow



→ Strong diversion in network load leads to high variations in the congestion level

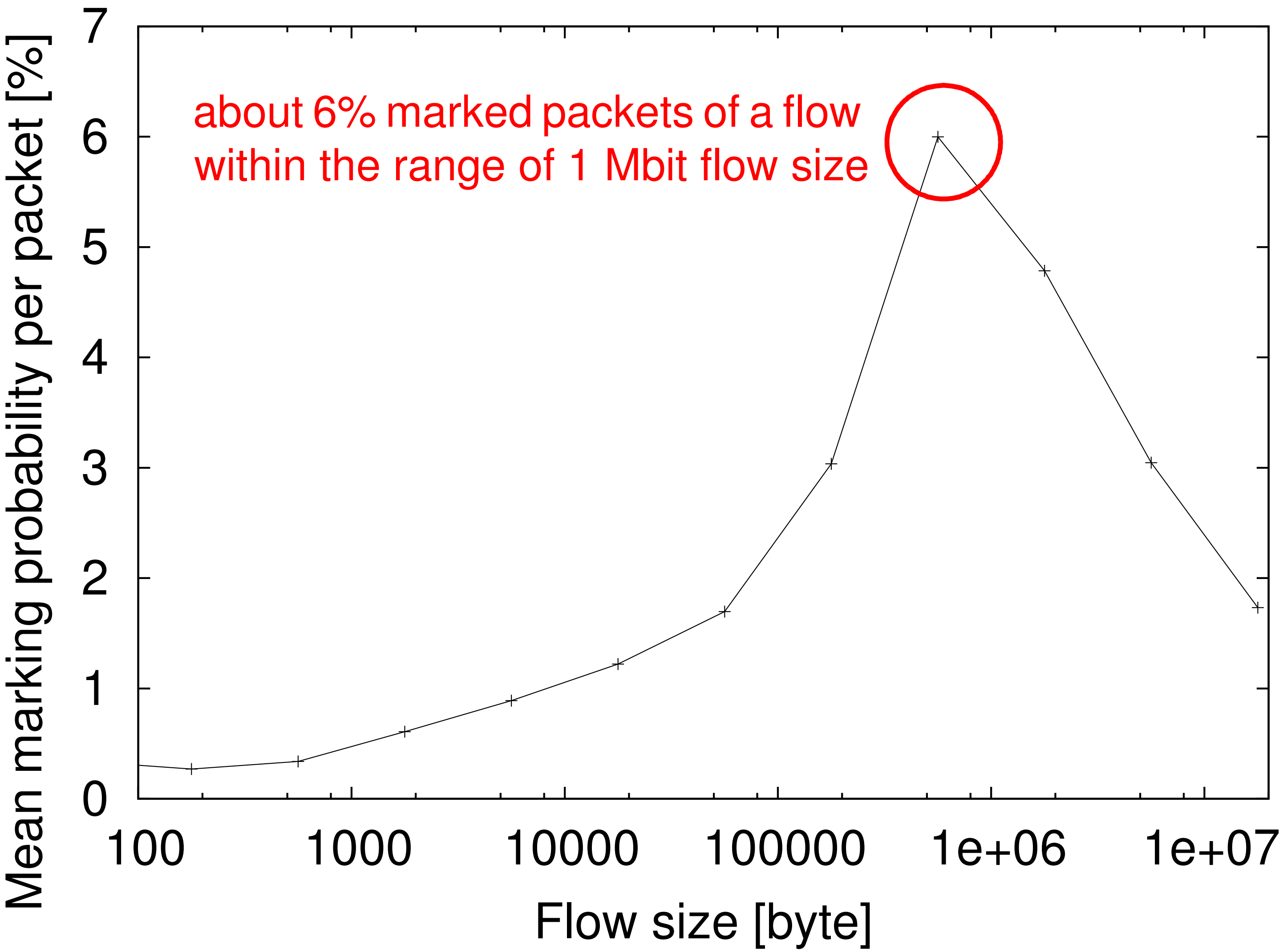
Number of Negative Marks over Flow Size

One point for every completed flow



Mean Probability for a Negative Marking

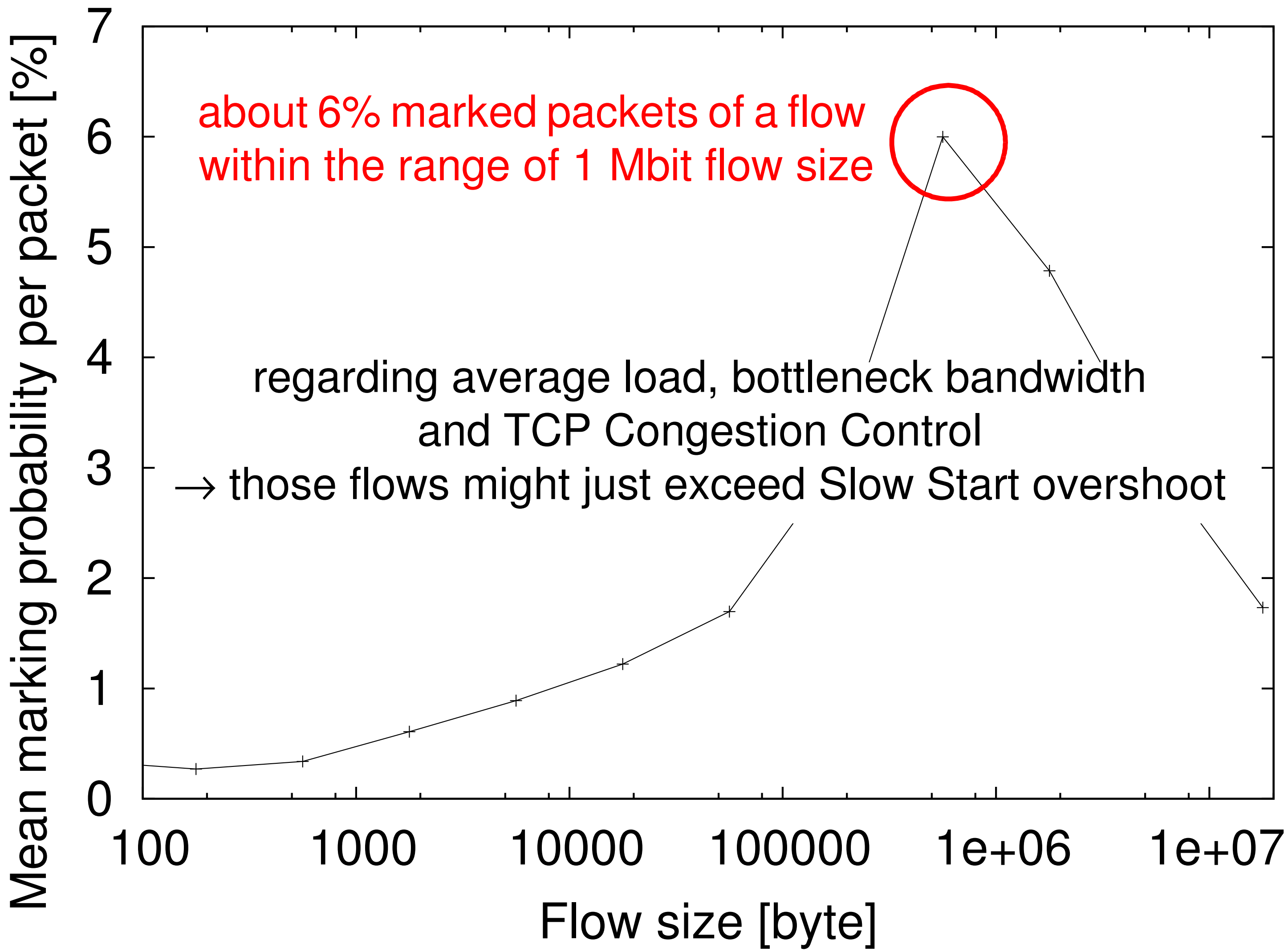
per Data Packet over Flow Size



about 6% marked packets of a flow within the range of 1 Mbit flow size

Mean Probability for a Negative Marking

per Data Packet over Flow Size

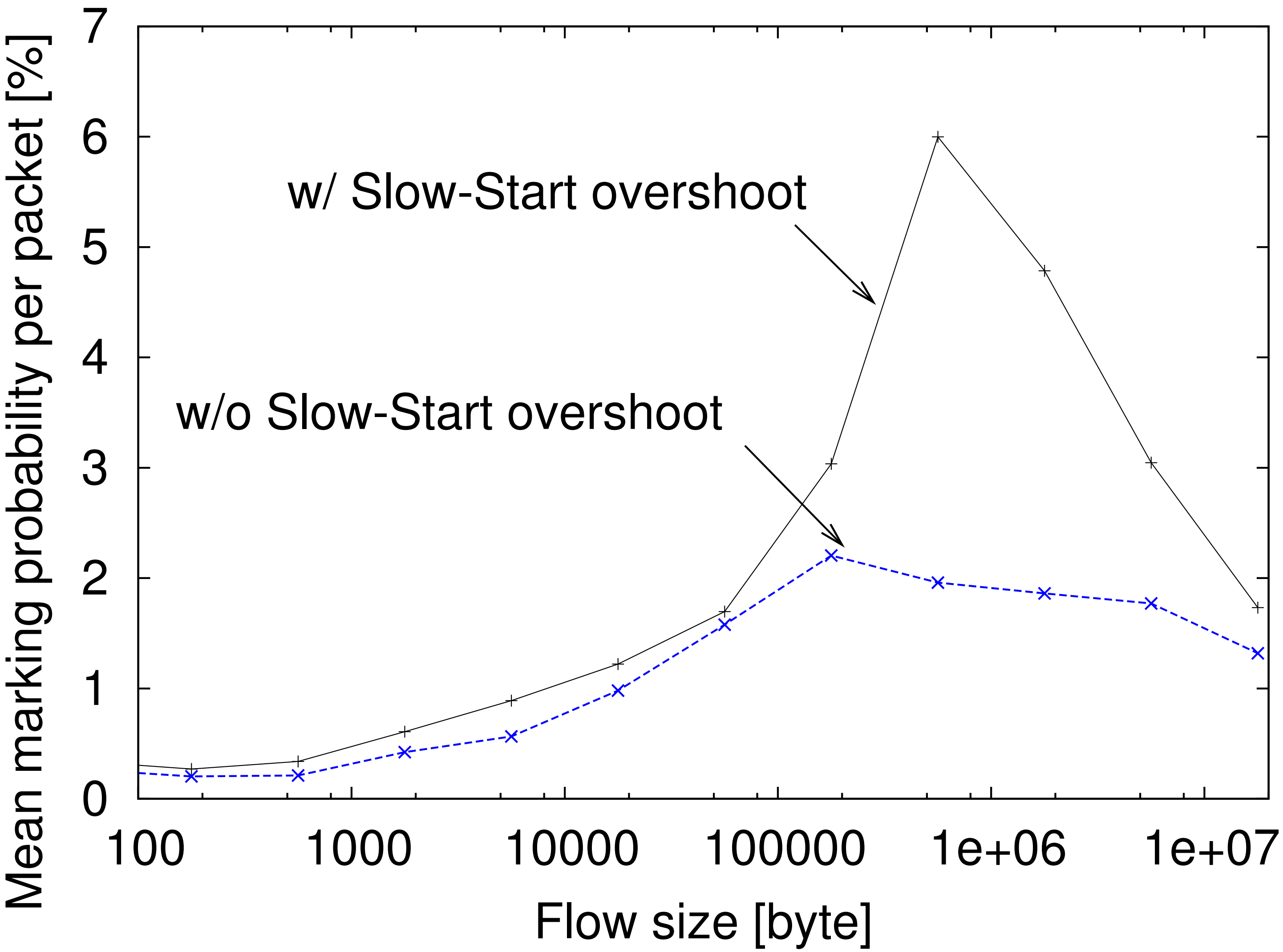


about 6% marked packets of a flow within the range of 1 Mbit flow size

regarding average load, bottleneck bandwidth and TCP Congestion Control
→ those flows might just exceed Slow Start overshoot

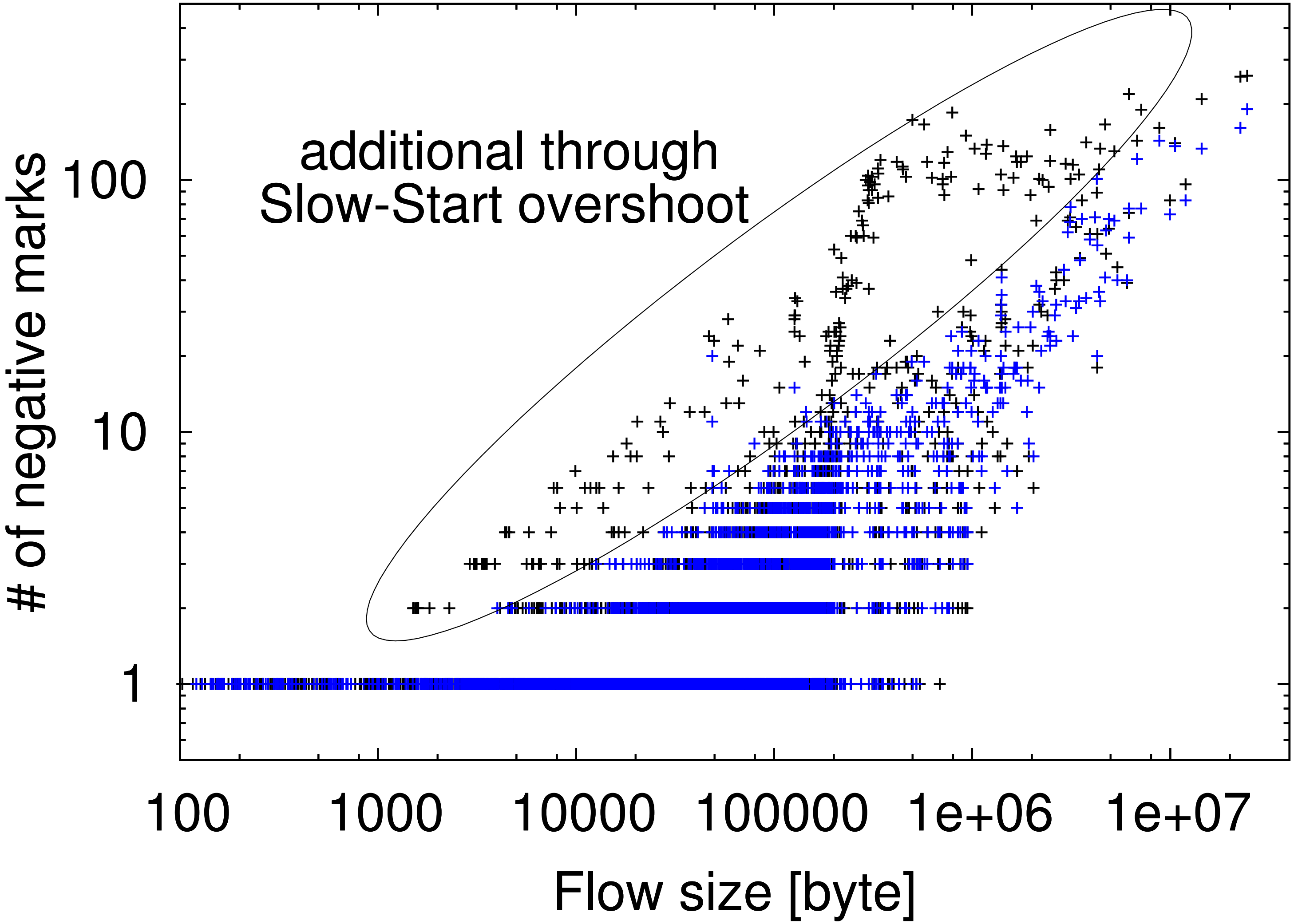
Mean Probability for a Negative Marking

Blue Line with lower SS Threshold to avoid SS Overshoot



Number of Negative Marks over Flow Size

Blue Points with lower SS Threshold to avoid SS Overshoot



Conclusion

Implementation

Most of the re-ECN processing was simple to realize in the Linux network stack (v2.6.26)

ECN marking

- TCP probing introduces frequent congestion as a feedback signal
- The number of congestion marking one flow receives depends on the current load and on the aggressiveness of the Congestion Control used

  → Congestion Exposure reveals the aggressiveness of a flow

TCP Slow-Start

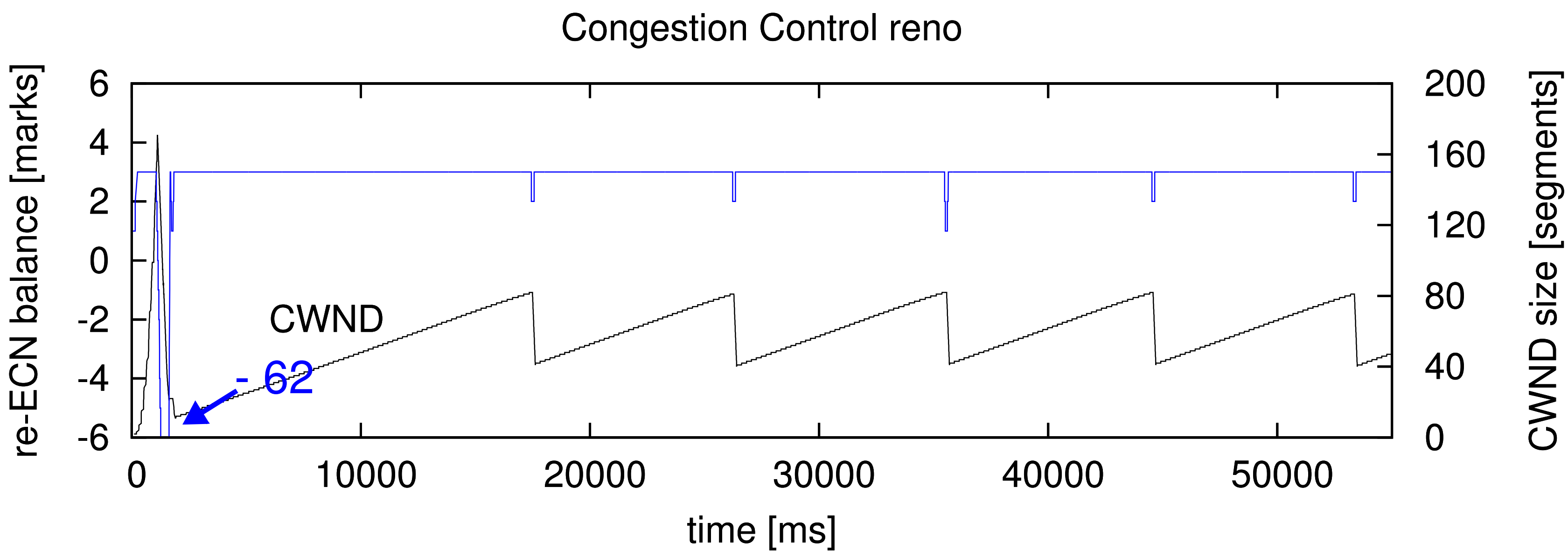
- TCP Slow-Start Overshoot causes lots of the congestion and discriminates certain flow length
- Congestion-volume counts the absolute number of markings (whereas TCP uses only one signal per RTT)

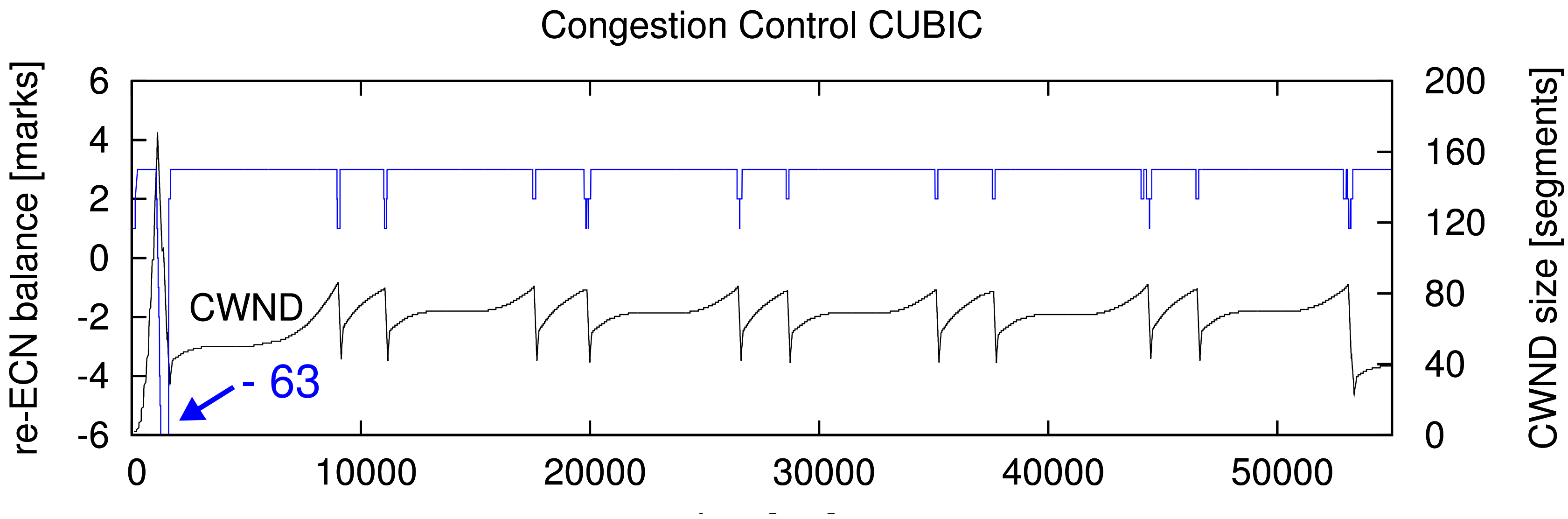
  → Regarding congestion as a metric for a more scalable Congestion Control

→ **Use congestion as a signal for adaption,**

but avoid causing more congestion than necessary!

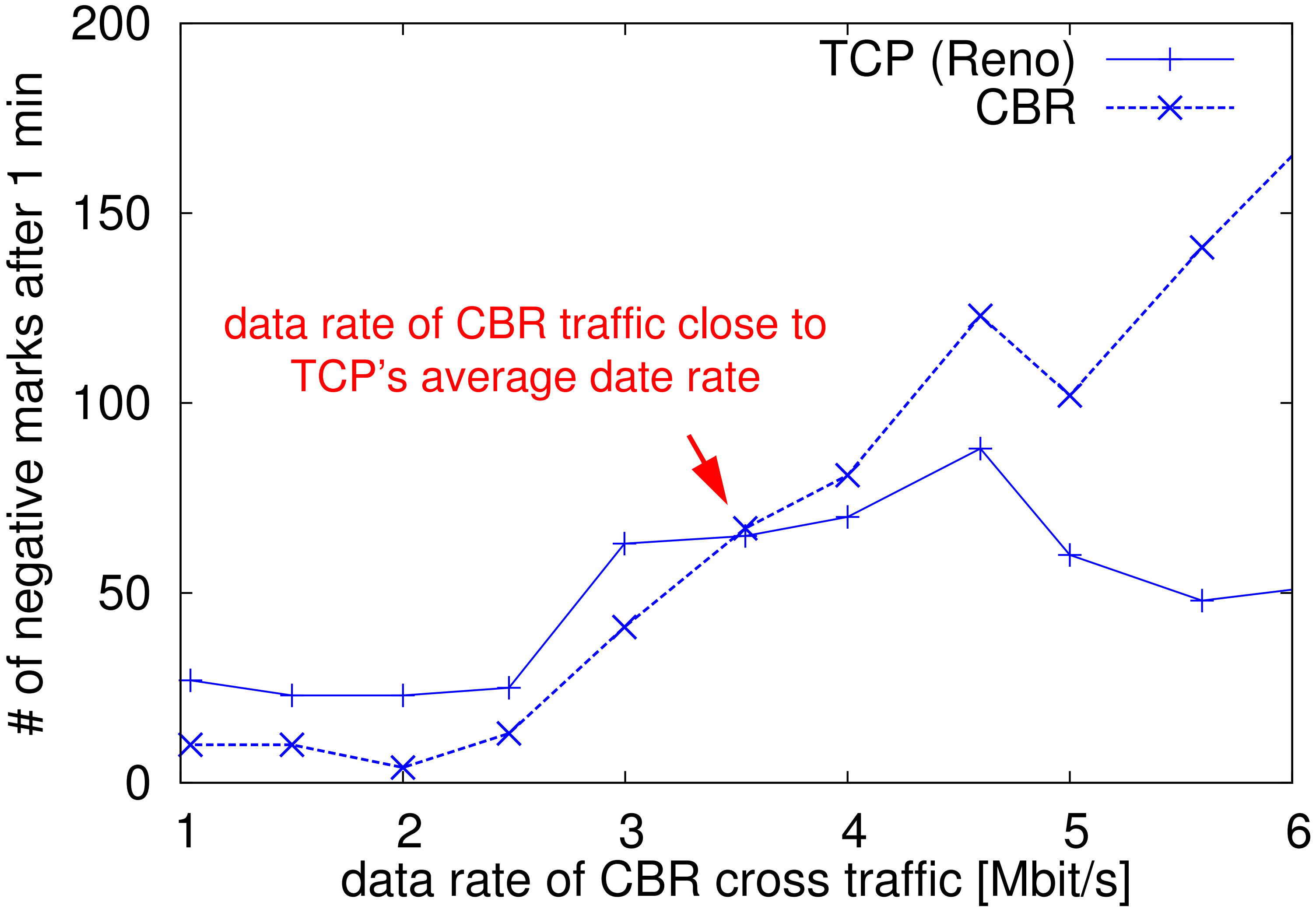
re-ECN Balance of a TCP Connection at Egress





Number of Negative Marks with CBR Cross Traffic

TCP Reno Sender, 8 Mbit/s Capacity



data rate of CBR traffic close to TCP's average date rate