Performance Evaluation of Congestion Window Validation for DASH Transport

Sajid Nazir, *Ziaul Hossain*, Raffaello Secchi, Matthew Broadbent, Andreas Petlund, Gorry Fairhurst

NOSSDAV, Singapore, 19-20 March 2014

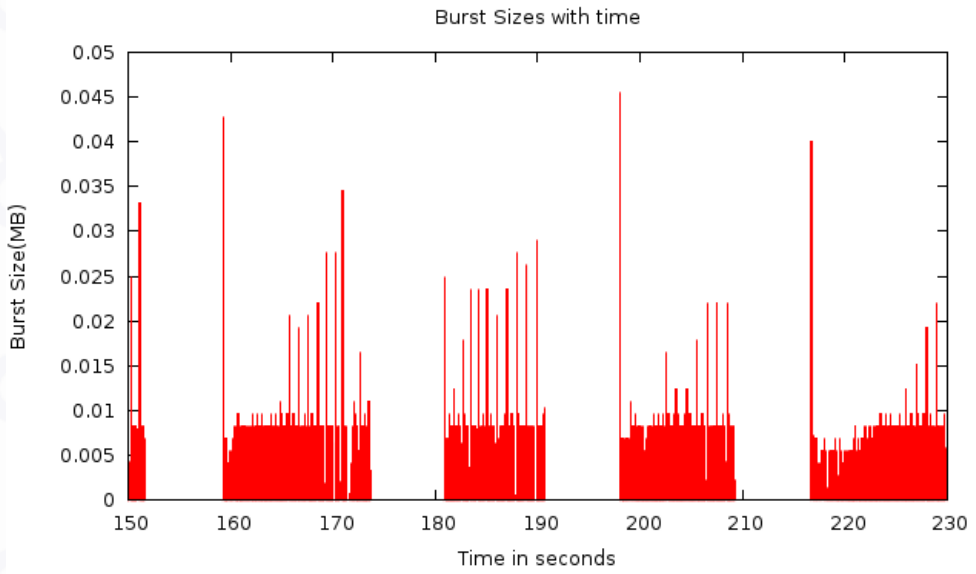
Motivation and Goal

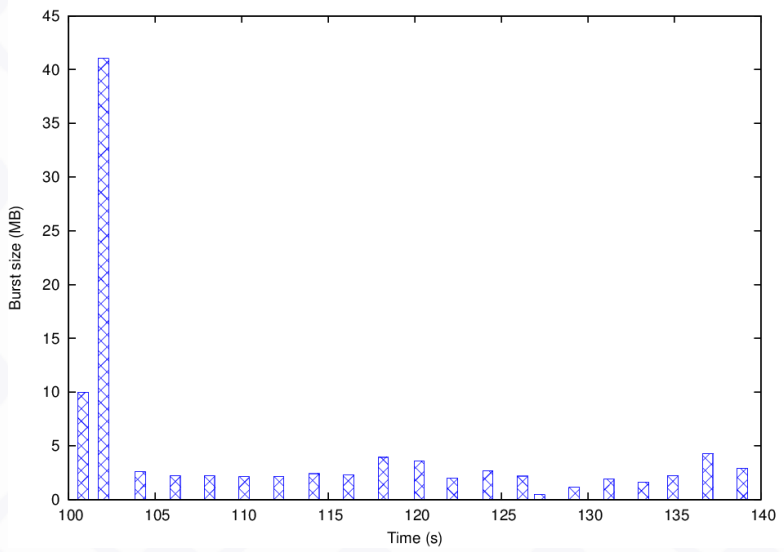
- Video traffic – more than half of the Internet
  - DASH is a significant standard
- DASH (Dynamic Adaptive Streaming over HTTP)
  - Segmentation of media content using various quality rates
  - Segments transfer with HTTP in bursts
  - HTTP uses TCP as the Transport Protocol
  - Bursty DASH unsuitable with standard TCP

- Design Smarter TCP for DASH
  - Support bursty DASH traffic
  - React appropriately to congestion
  - Better network sharing (not impacting others)

DASH Traffic Pattern

- ON/OFF pattern with idle period
- Application limited period
- How to transmit them with HTTP/TCP?

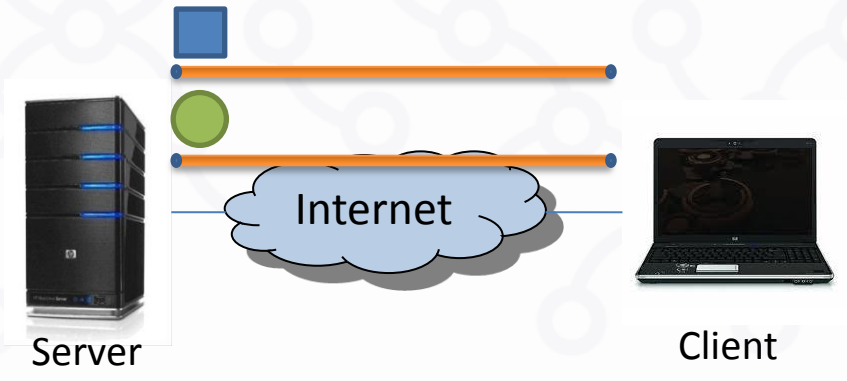




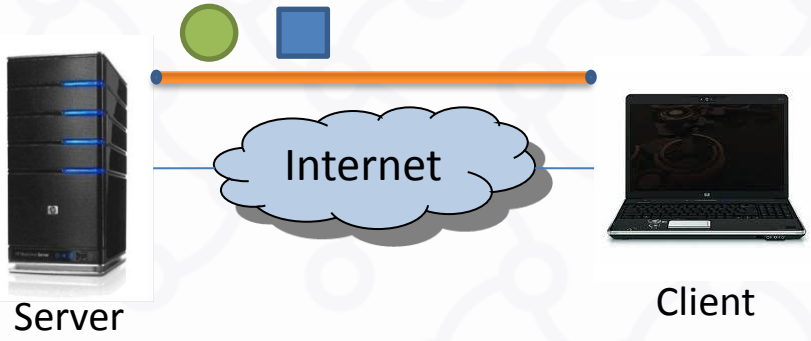
Persistent HTTP

- Moving towards persistent TCP (*e.g.* Google SPDY)
  - Server retain network status (RTT, windows)
  - Good for the network (more stable transmission)
  - Saves time for opening new connections

Different objects sent with different TCP Connections

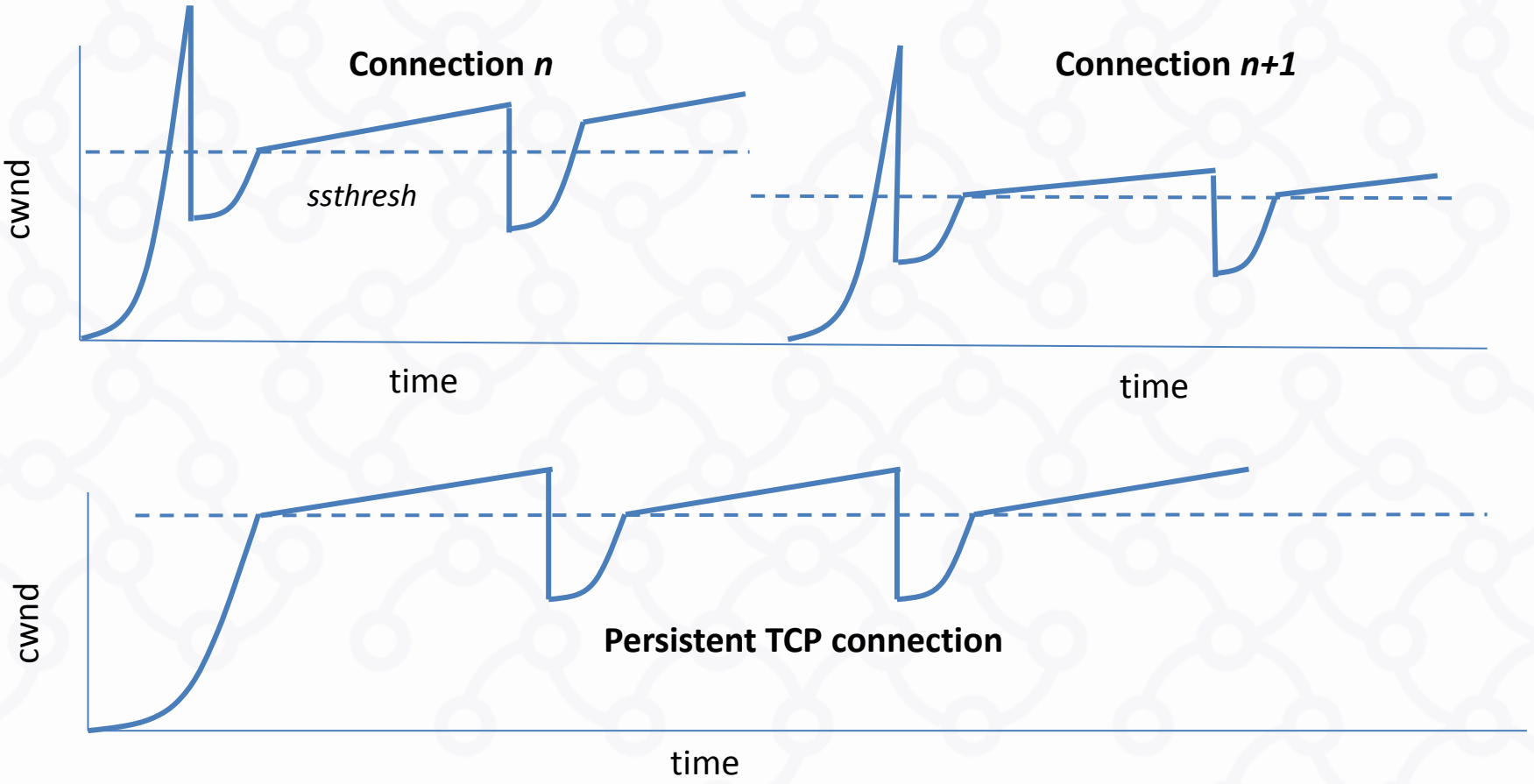


Persistency: All objects sent with same TCP Connection

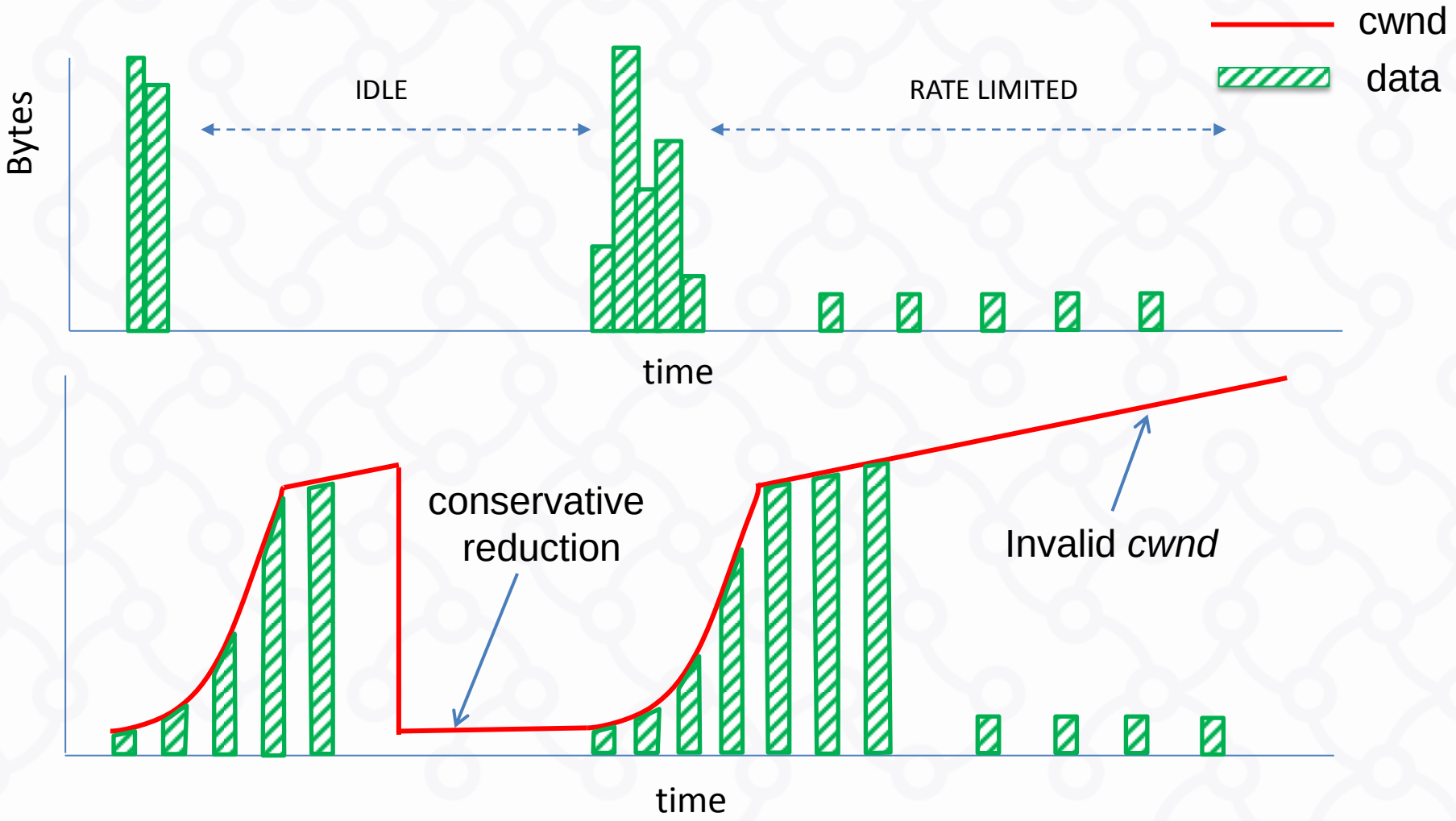


TCP *cwnd* evolution

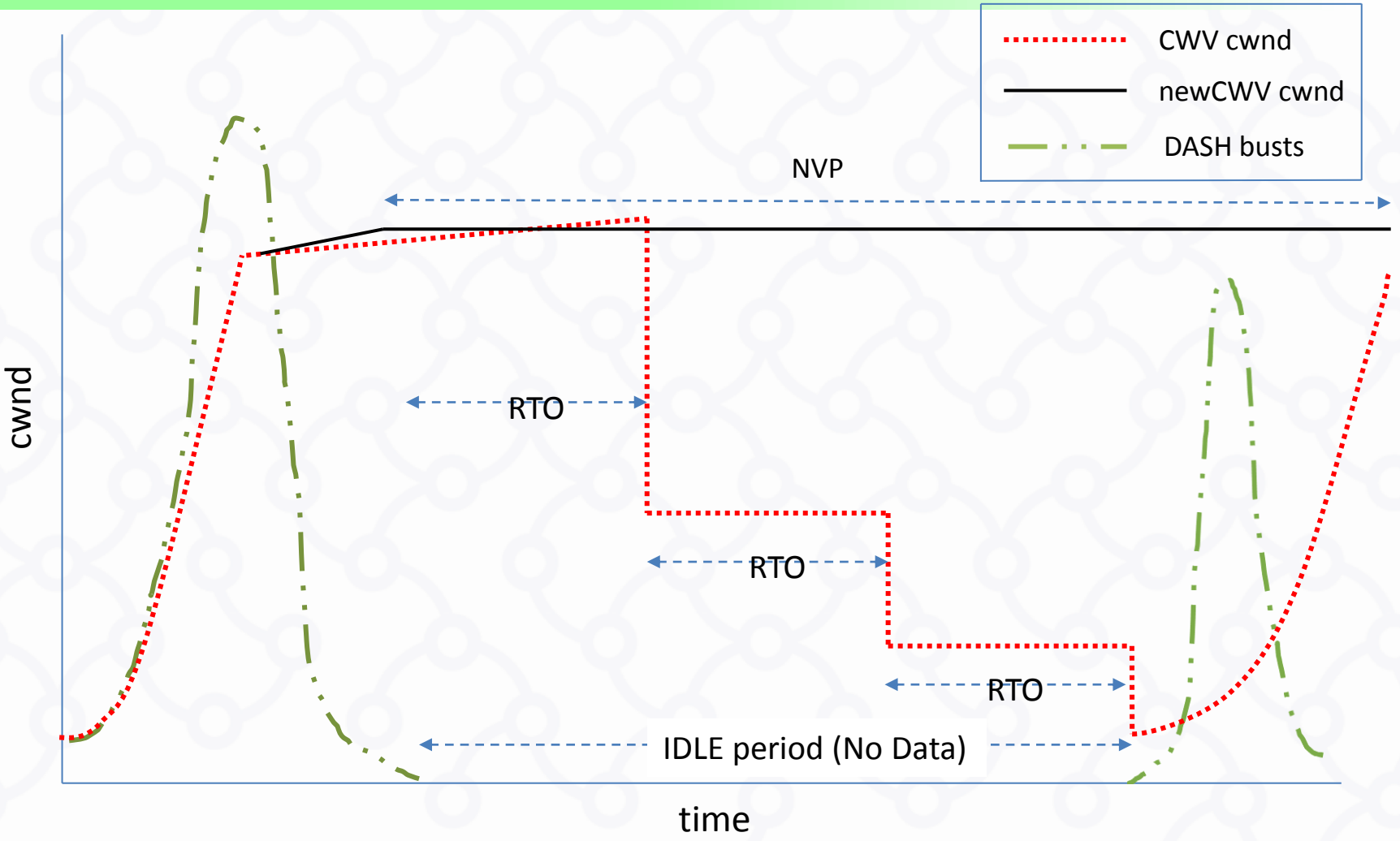
- *cwnd* determines the flow rate (fast increase from low value)
- Persistent TCP connection causes more stable traffic



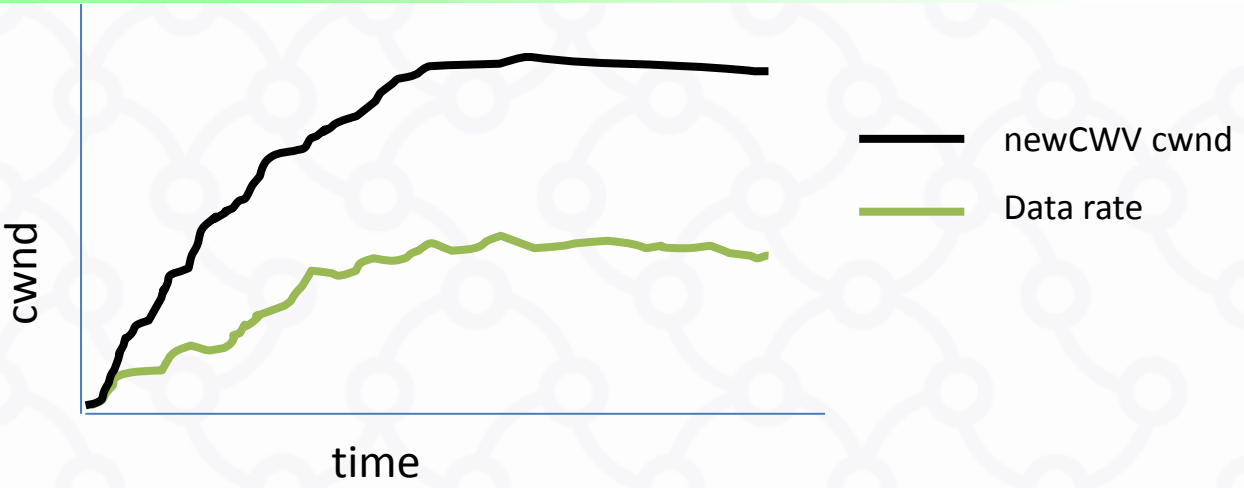
Standard TCP (NewReno) problems with persistent connection

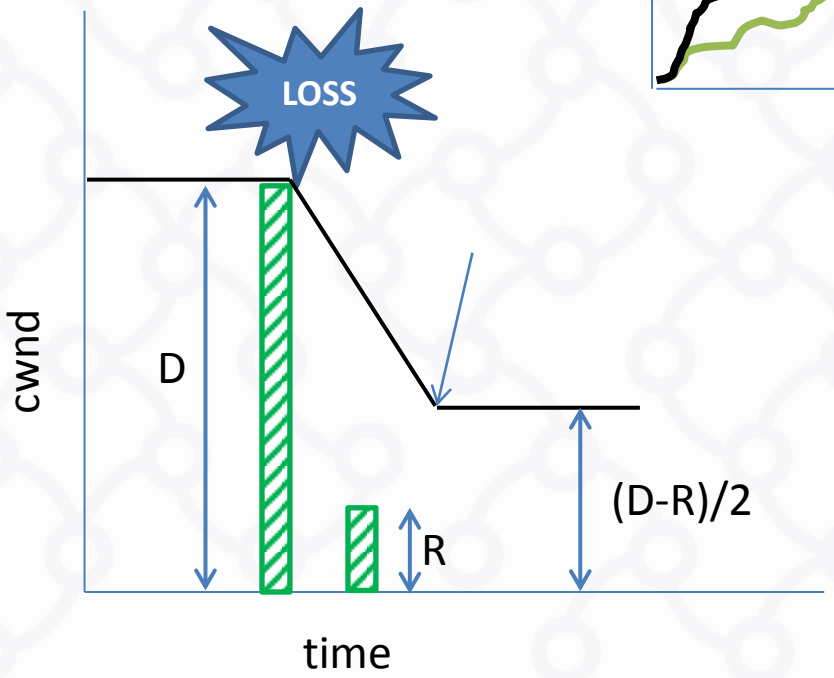


newCWV behaviour



newCWV behaviour(2)



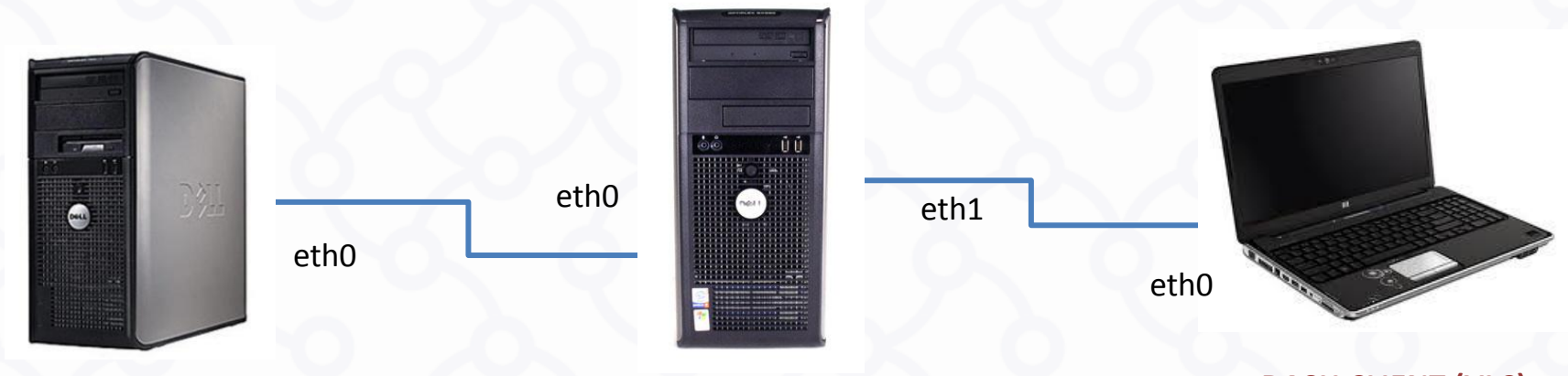


- *cwnd* grows to twice the data rate

- D-R was transmitted
- Safe reduction by half like TCP
- *cwnd* = (D-R)/2

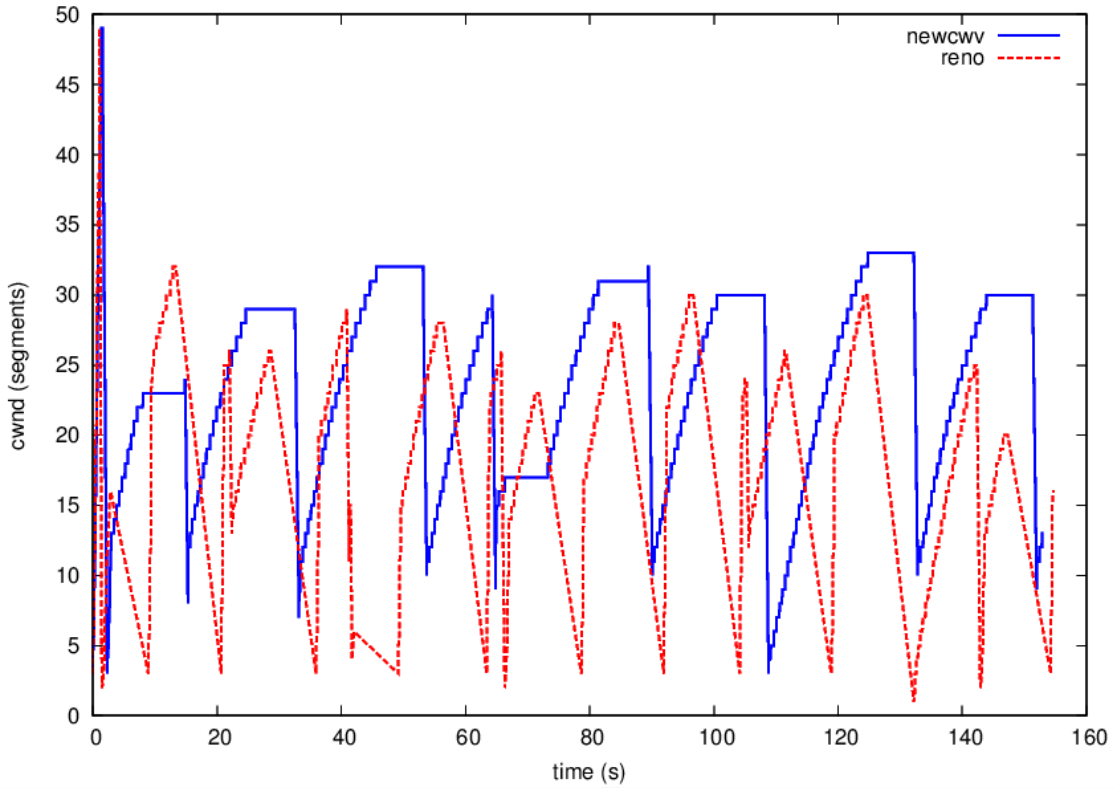
Experiments: Setup

- Linux Loadable Kernel Module developed for newCWV
- Bottleneck link of 1Mbps
- Delay 200ms
- Used VLC client and MPEG-DASH server



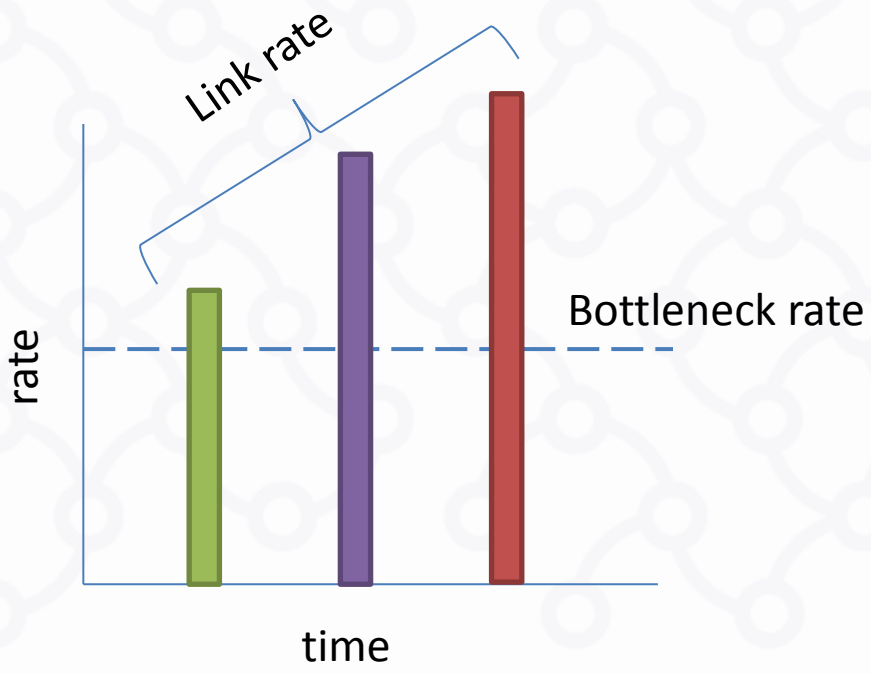
cwnd for NewReno and newCWV

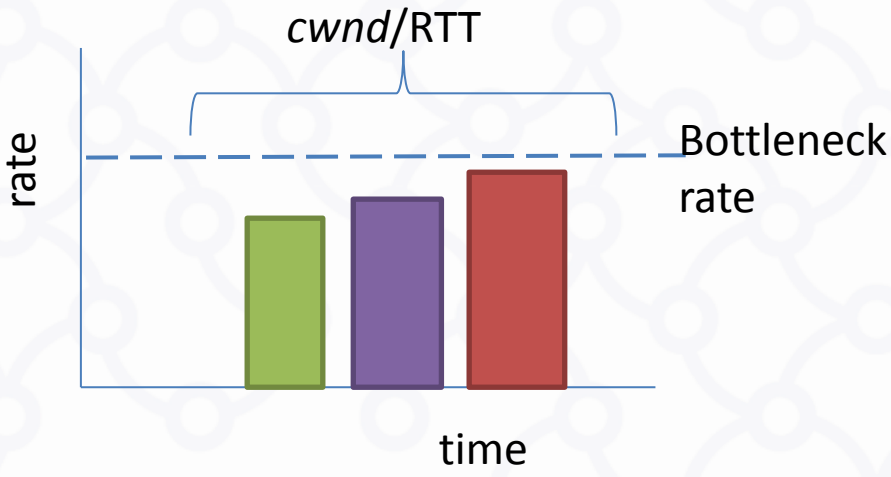
- cwnd collapses for NewReno while freezes for newCWV
- More stable *cwnd* with newCWV than oscillating NewReno



Burst & Burst-Mitigation

- High cwnd generates large bursts into network
  - Increases probability of burst losses
- Spreading a burst over a period of time helps
  - Pacing burst segments (cwnd/RTT)

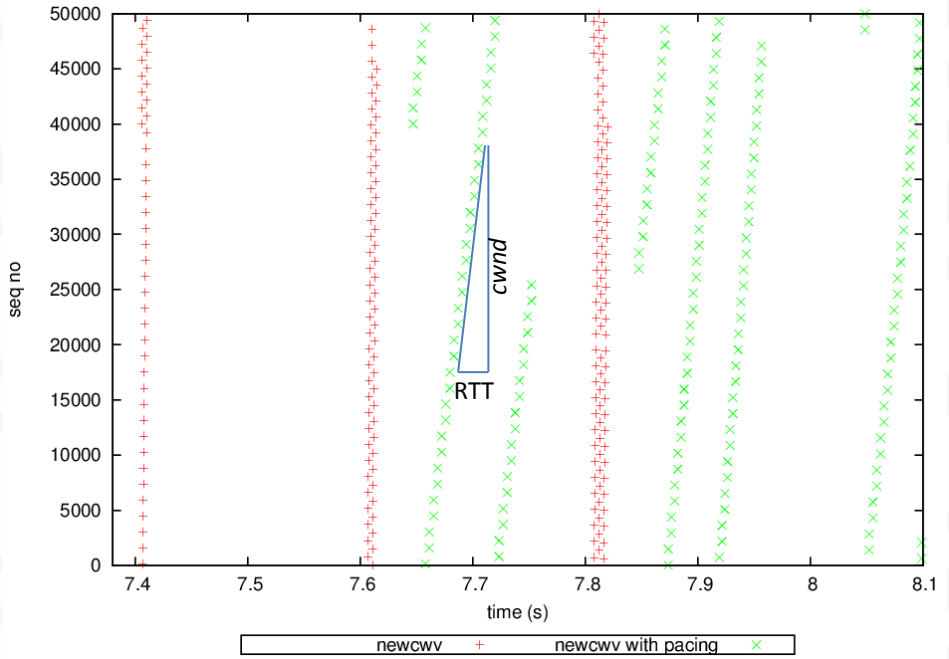




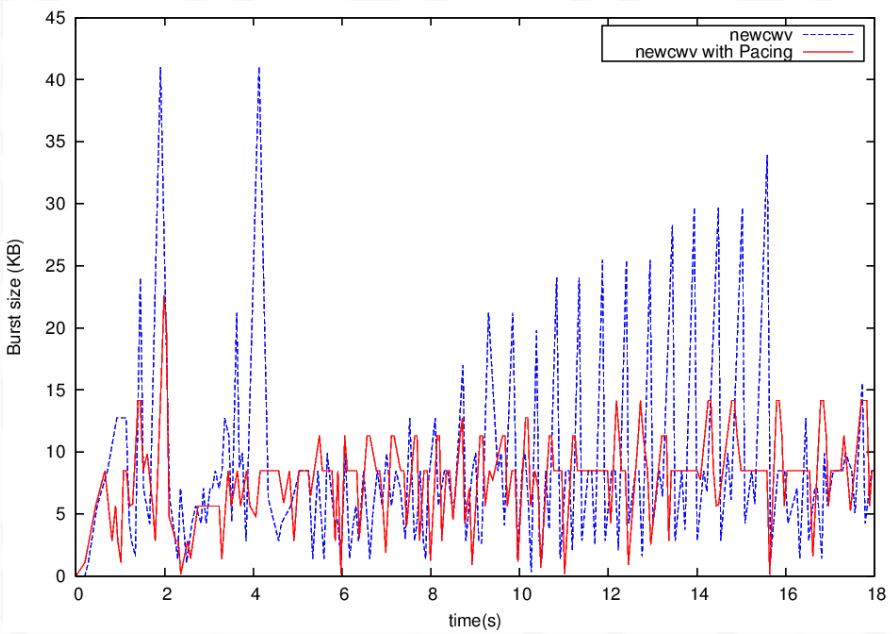
Experiments: Enabling Pacing

- Pacing was implemented by FQ module in Linux
- Paces a burst of data over the RTT

Pacing segments for a 200ms TCP flow

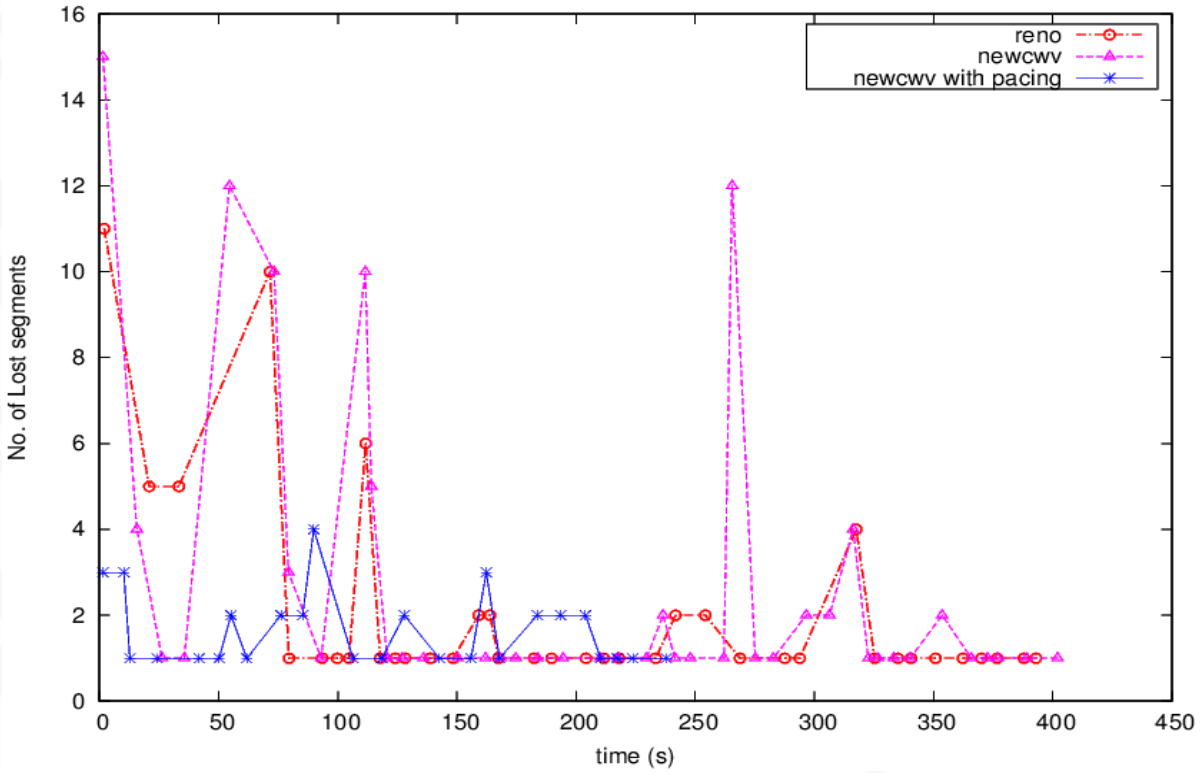


Pacing reduces burstiness of newCWV for DASH



DASH with newCWV & pacing

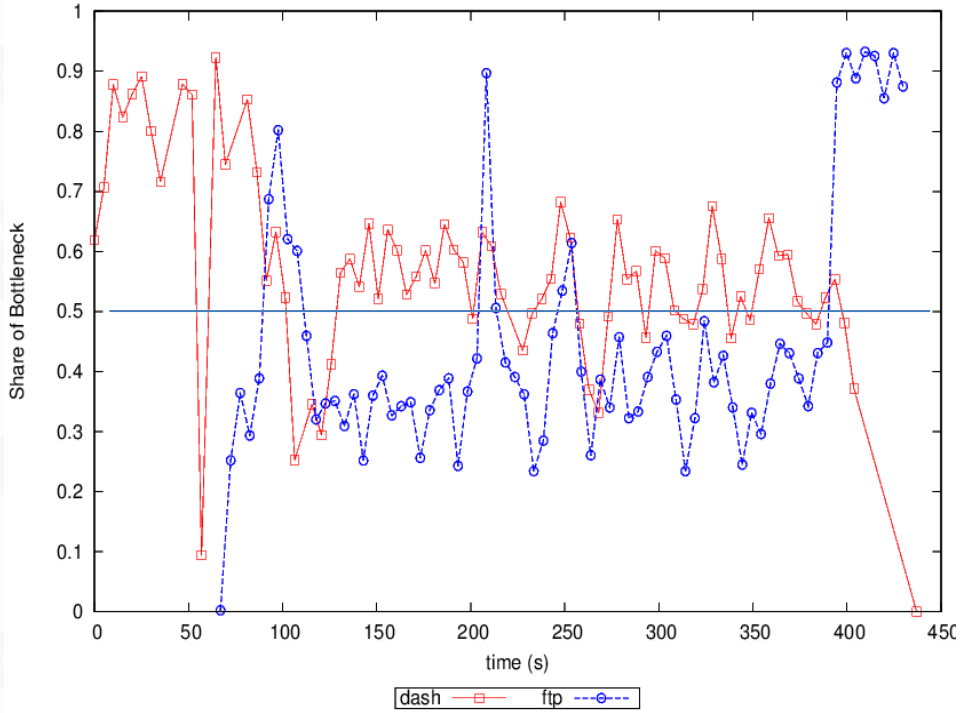
- Pacing reduces losses caused by newCWV bursts
- Higher quality segments could be requested

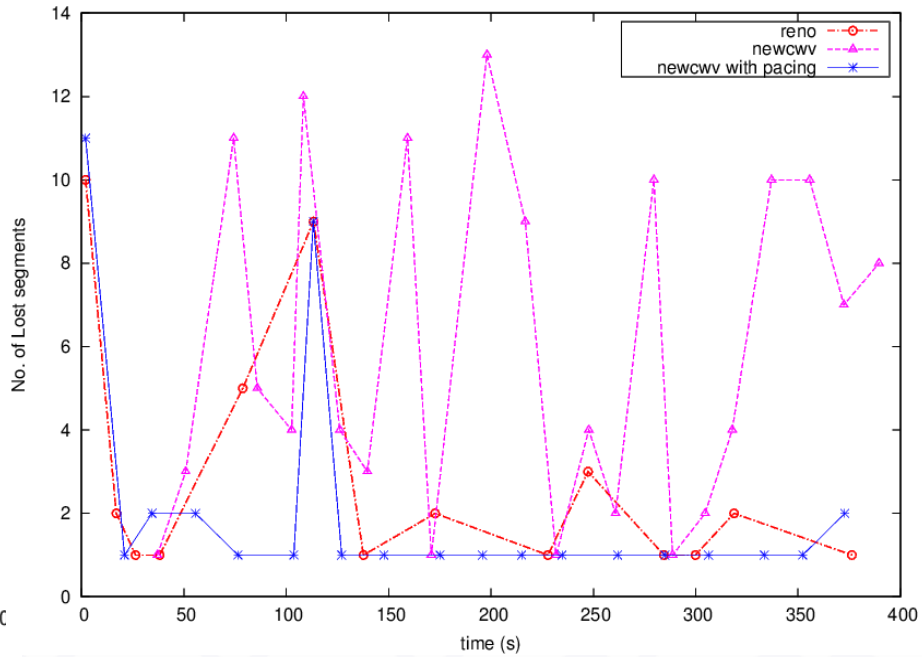


Method	Avg. Segment rate (kbps)
NewReno	803.40
newCWV	785.40
newCWV with pacing	836.26

Sharing Bottleneck

- Proportional sharing the link with FTP flow
- Pacing results in fewer losses





Conclusion & Future work

- newCWV features
  - Support bursty DASH traffic
    - newCWV with pacing is beneficial
  - React appropriately to congestion
    - TCP like congestion response with better burst transfer times
  - Better network sharing (not impacting others)
    - Lower loss while sharing bottleneck

- A range of clients (Adobe Flash, Apple iOS, MS Silverlight *etc.*)
- DASH and TCP cross-layer interaction

Thank you

This research was supported by the RCUK Digital Economy programme to the dot.rural Digital Economy Hub; award reference: EP/G066051/1; the EU FP7 OFELIA project (FP7-ICT-258365), the EU FP7 STEER project (FP7-ICT-318343) the NERC EVOp (NE-I002200-1) project and the RITE project (ICT-317700)

Questions?