TCP Performance Implications of Network Asymmetry

Venkata N. Padmanabhan

Microsoft Research

http://www.research.microsoft.com/~padmanab

Joint work with Hari Balakrishnan (MIT)

draft-ietf-pilc-asym-00.txt

PILC Working Group, IETF, Nov 1999

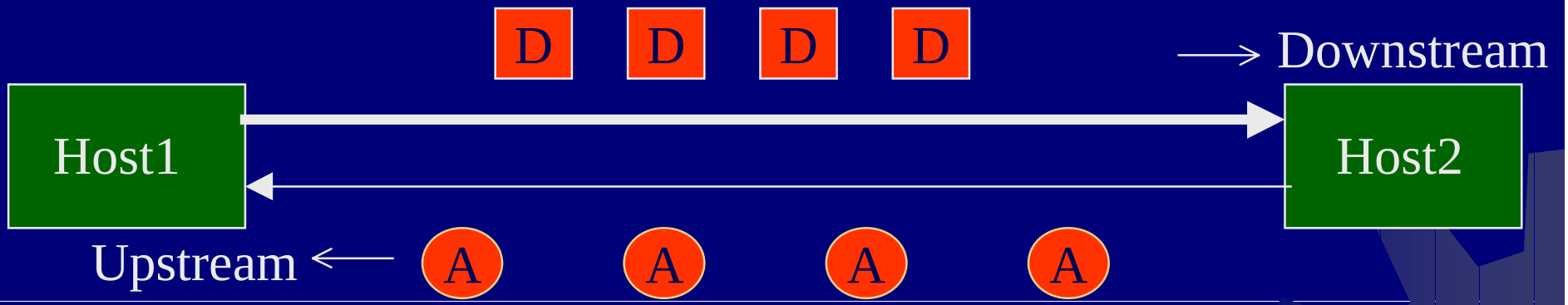
Outline

- What is network asymmetry?
- Problem
  - ack flow disrupted ⇒ sender adversely affected
- Solution
  - minimize disruption of acks
  - make sender robust to ack disruption
- Summary

Network Asymmetry

- Significant disparity in network characteristics in opposite directions
  - bandwidth
  - media access
  - error rate
- TCP performance in one direction is often limited by network/traffic characteristics in the opposite direction

Model for Unidirectional Transfer



- Normalized asymmetry (*k*) [LMS97]
  - bandwidth ratio divided by packet size ratio
  - divide further by 2 for TCP delayed acks

Impact of Asymmetry on TCP

- $k > 1 \Rightarrow$ acks queue up at upstream bottleneck $\Rightarrow$ ack clocking breaks down
- If buffer not full $\Rightarrow$ ack dilation
  - spacing does not reflect downstream bottleneck
  - sender clocks out new data at slower pace
- If buffer full $\Rightarrow$ infrequent acks
  - only 1 ack in *k* is delivered to sender
  - sender increments *cwnd* slowly
  - sender could burst out packets

Solution

Two aspects

- Minimize disruption in ack stream
  - header compression
  - ack filtering/compaction/compression
  - ack reconstruction/expansion/decompression
  - ack congestion control
- Make sender adapt to infrequent acks

Header Compression

- Reduces size of ack-only pkts substantially
  - 40-byte → 6-7 byte [RFC-1144]
  - vulnerable to packet loss/reordering
- *Twice* algorithm [RFC-2507]
  - primarily addresses single packet loss case
    - but burst losses are not uncommon
  - new protocol machinery for burst loss case
- Still need mechanisms to shield/adapt sender

Ack Filtering [BPK97]

- Discard “redundant” upstream acks
  - acks with higher cumulative ack # subsume acks with smaller ack #
  - discard the latter in favor of the former
    - deterministic or random
  - reduces bandwidth and buffer requirement
  - but makes acks infrequent
- No filtering when packet loss happens
  - duplicate acks not filtered
  - SACK that indicates hole(s) not filtered

Ack Reconstruction [BPK97]

- Restore “original” ack stream at far end
  - hold on to ack for a little while waiting for next one
  - if no next ack ⇒ send ack
  - if next ack arrives
    - insert as many acks as delayed acks would imply
    - space apart uniformly over inter-ack time gap
- Pros and cons
  - no new protocol machinery
  - shields sender
  - hard to restore original stream perfectly
  - increases the effective RTT

Alternative Schemes

- Ack compaction & expansion [Sam99]
  - compacter conveys to expander the # of deleted acks and the # of bytes they acknowledge
  - avoids need for guessing # deleted acks
  - but new protocol machinery
- Ack compressor & decompressor [Joh99]
  - compressor conveys to the the decompressor the arrival times of acks
  - avoids even need for guessing ack spacing
  - but new protocol machinery

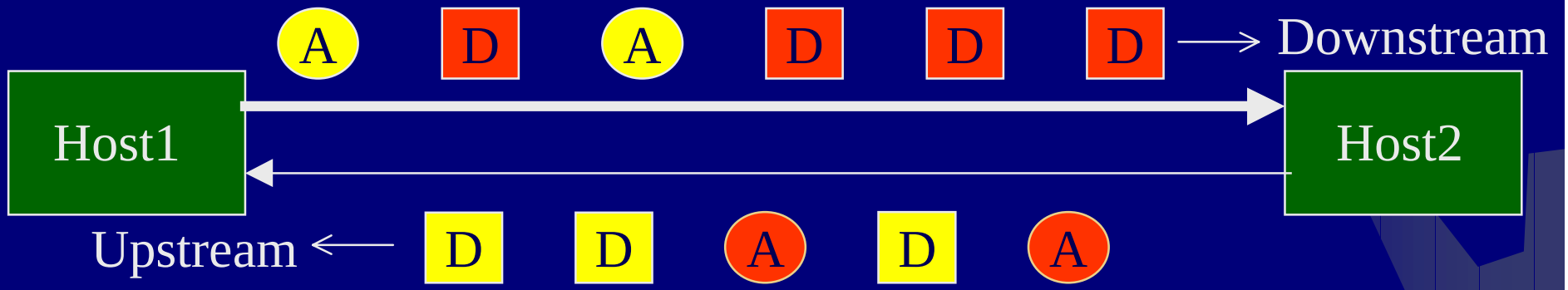
Ack Congestion Control [BPK97]

- End-to-end alternative to ack filtering
  - generalized delayed acks (one ack for *n* packets)
  - receiver uses ECN to modulate ack frequency
  - no ECN $\Rightarrow$ decrease *n* linearly (within limits)
  - ECN $\Rightarrow$ increase *n* multiplicatively (within limits)
- Pros and cons
  - very general (not tied to specific links)
  - end-to-end $\Rightarrow$ no IPSEC issues
  - but is this a sledgehammer to kill a fly?
  - requires sender to adapt to infrequent acks

Sender Adaptation

- Congestion window growth
  - consider the amount of data acknowledged rather than just the number of acks
- Burstiness
  - limit size of mini-burst (to say 3 by default)
  - space mini-bursts apart using estimate of connection rate (*cwnd/srtt*)

Model for Bi-directional Traffic



- Large data packets can hold up acks on the upstream link

Solutions

- *Acks-first* scheduling [BPK97,LVR98]
  - could cause starvation of upstream data packets
  - ack congestion control alleviates this
- Back-pressure to limit data pkt queuing [LVR98]
  - assumes upstream link attached directly to host
  - performance dependent on downstream connection
- “Fair”-scheduling of data/acks [LMS97,LVR98]
- Link-level fragmentation of data packets

Other Situations Where Decreasing Ack Frequency Might Help

- Reduced MAC overhead
  - RTS/CTS style MAC
  - Example: packet radio, 802.11, cable modem
- Backbone routers
  - fewer packets to forward per second
- Internet servers
  - reduced network interrupt/ack processing overhead at the cost of increased timer overhead

Security Issues

- IPSEC ESP ⇒ neither read nor write access
  - rules out header compression, ack filtering and reconstruction
  - only end-to-end schemes work, e.g., ack congestion control and sender adaptation
- TF-ESP/AH ⇒ read-only access
  - allows ack filtering
- IPSEC AH ⇒ read-only + reconstructable
  - allows ack filtering
  - allows (modified) header compression [RFC2507]

Summary

- Disruption of ack stream causes problems
- Solution:
  - minimize disruption
    - header compression
    - ack filtering/reconstruction
    - ack congestion control
  - make sender adaptive
- Potential benefits extend beyond (bandwidth) asymmetric contexts

TCP Performance Implications of Network Asymmetry

Venkata N. Padmanabhan

Microsoft Research

http://www.research.microsoft.com/~padmanab

Joint work with Hari Balakrishnan (MIT)

draft-ietf-pilc-asym-00.txt

PILC Working Group, IETF, Nov 1999