Packet Reordering is Not Pathological Network Behavior

By

Bennet, Partridge, and Shectman

Presented By:

Dan Sarisky

Introduction

- Existance of packet reordering
- Extent of packet reordering
- Causes of packet reordering
  - Faulty network equipment
  - Local parallelism
    - Cost effective
    - No single point of failure
    - Lack of high bandwidth connections

Observations and Measurements

- Reordering is a complex phenomenon
- Amount of Reordering is a Function of :
  - Existance of parallel links in a path
  - Exact configuration of hardware and software nodes in the path
  - Load on those nodes

MAE-East

- Located in Washington DC
- Major peering point for ISPs
- Advantages of MAE-East
  - Knowledge of network leading to MAE-East
  - Diversity of equipment close to MAE-East
  - Publicly available information about MAE-East
  - MAE-East is heavily loaded
- Reordering is not confined to MAE-East

Test 1- General Reordering

- First send 5 56 byte ICMP packets
- If those got through then send 5 second burst of 50 ping packets
- Tests were run from 6-7pm Mondays

Results of test 1

Hosts receiving Packets	1-10	1-10	1-20	1-20
Date of Test		In order		In order
Early Dec 1997	117	7	97	5
Early Jan 1998	60	4	46	1

Amazingly high reordering rates. Over 90% of connections had a reordering incident

Test 2 – Reordering and Load

- Does high traffic load cause reordering?
- 1 site had a high degree of reordering from test 1.
  - Send 100 packet burst of 512 byte packets every minute for 4 days.

Reordering metric

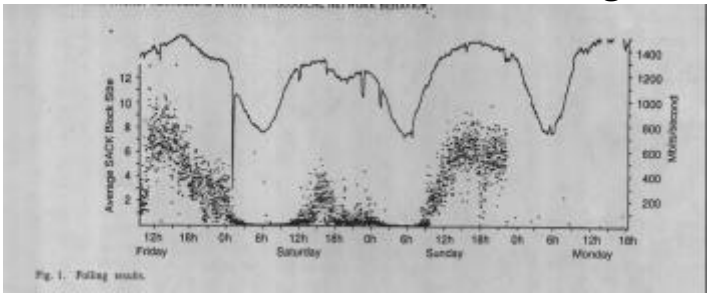
- Number of SACK blocks needed to cover the out-of-order replies.
  - Matched intuitive assessment of amount of scrambling
  - Higher if data passed through 2 scramblers than through 1
  - Independent of data loss and size of data set
  - Minimal value for in-order data

Reasons for these results

- Reordering is a function of network load, below a certain load very little occurs
- In 11/97 MAE-East reconfigured its network to encourage parallel links
- Major ISPs use private peering points which use fewer parallel links that MAE-East and thus have lower reordering rates

Private peering

- Midway through the 4 day test a private peering relationship was established between two ISPs in Nothern Virginia



Reordering effects on TCP

- Forward path reordering
  - Reordering of data packets
- Reverse path reordering
  - Reordering of ACKs

Forward Path Reordering

- Unnecessary retransmissions caused by fast retransmit of duplicate ACKs
- Congestion Window size is driven down successive retransmissions

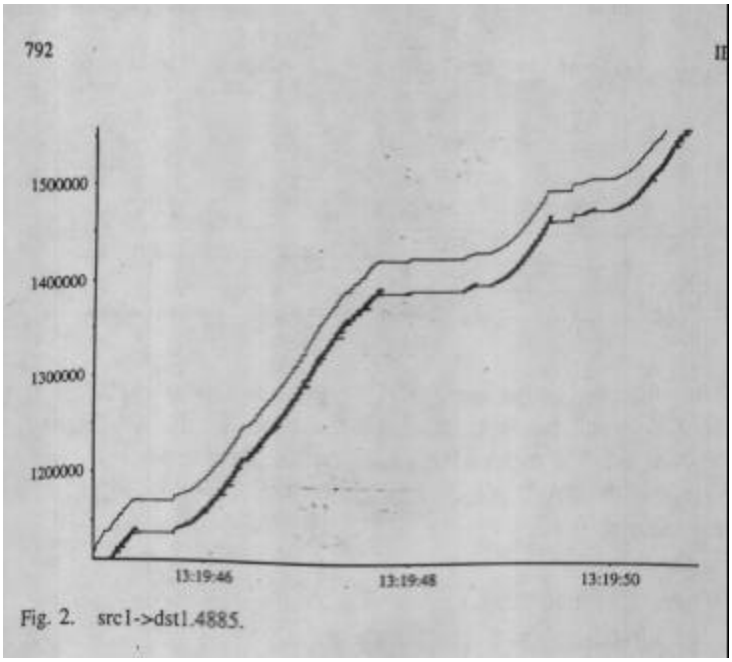


Fig. 2. src1->dst1.4885.

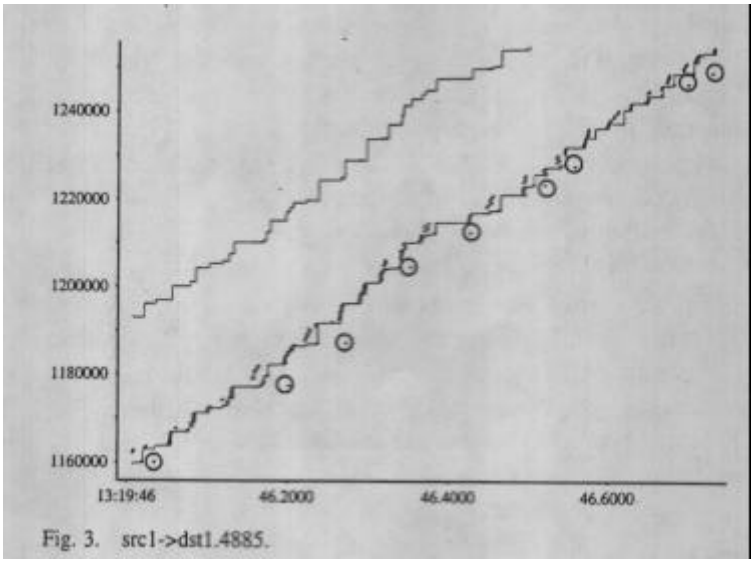


Fig. 3. src1->dst1.4885.

Continued

- Packet loss is obscured by strange pattern of ACKs
- Poor RTT estimation
  - Still works with TCP timestamps
  - Not enough data points otherwise
- Lowered receiver TCP efficiency caused by sorting, bursty traffic, and broken header prediction

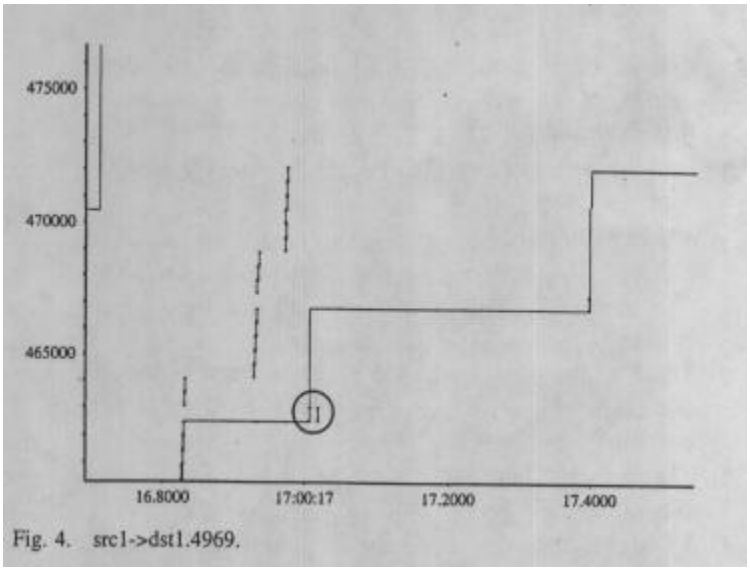


Fig. 4. src1->dst1.4969.

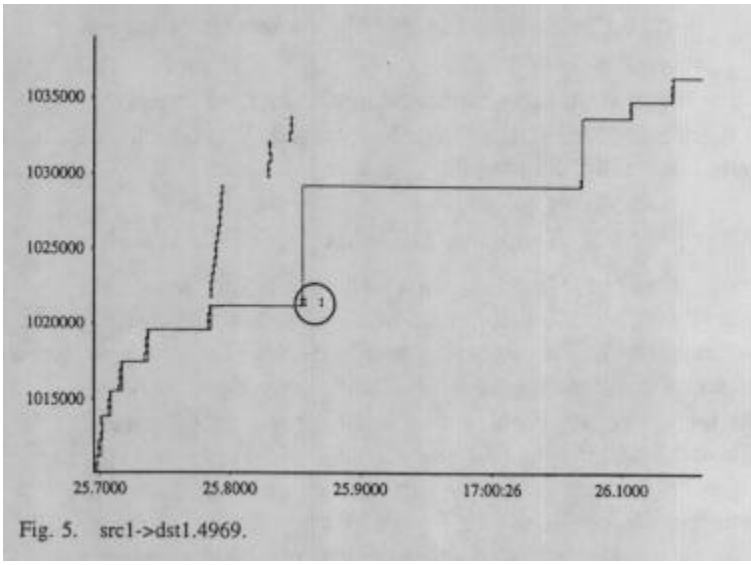


Fig. 5. src1->dst1.4969.

Reverse Path Reordering

- Loss of self clocking
  - Caused by sender seeing one big ACK, then lots of ACKs that is already "had"
- Congestion window grows too slowly
- Encourages bursty data traffic, which encourages more data reordering – circular problem

Why Re-Ordering occurs

- Equipment at MAE-East is a DEC Gigaswitch FDDI
- Allows for hunt groups – collection of actual ports that operate as a virtual port for increased bandwidth
- FIFO buffers – results in head-of-line blocking
- Connections are done by input queues bidding for output. Bids are answered one at a time

Continued

- DEC Literature says that re-ordering is a problem and that the duplicate ACK threshhold for fast retransmit should be set to 100 instead of 3.

Hardware based solutions

- Types of switches
  - Shared Buffer – Doesn't scale to high speed
  - Output Buffered – Same problem
  - Multi-Stage – highly varied delay
  - Input Buffered – Current solution
- Possibly aggregate all hunt groups onto same interface card, doesn't scale.

IP based solutions

- IP could be made aware of parallel links and can ensure that data between the same sender and host can always take the same path. This could be implemented in a hash table efficiently
- No one conversation can use more than 1 link.
- Uneven distribution is possible between links if hash function isn't just right

TCP-based solutions

- Detect erroneous fast-retransmits, and undo congestion window changes. Do this by watching for too quick response times for a fast-retransmitted packet
- TCP can support SACK. This will allow for better detection of which frames are coming out-of-order and which are actually lost

Numbering Packets

- There exist algoritms for doing this efficiently, but they are nonwork conserving. This idea has not yet been effectively applied to forwarding ASICs of switch schedulers yet.

Conclusion

- Reordering will happen, but it needs to be made less common. Algorithms like packet numbering can be used to minimize re-ordering
- Accept that re-ordering will occur and redesign protocols to recover more gracefully from re-ordering events. Observing SACK patterns may be a starting point