Improved Selective Acknowledgment Scheme for TCP

Raj Kettimuthu & Bill Allcock
Globus Alliance / Argonne National Laboratory

Introduction

- Improved Selective Acknowledgment (ISACK) Scheme addresses limitations of TCP selective acknowledgment (SACK) mechanism
  - SACK can convey information about only 4 noncontiguous blocks of data received
  - Sender might have to unnecessarily retransmit some packets
- ASACK uses both SACK and ISACK to give optimal performance

Background

- TCP provides connection-oriented, reliable byte stream service.
  - Provides reliability by assigning a sequence number to each octet transmitted and by requiring a positive ACK
  - The acknowledgment mechanism is cumulative
- TCP provides flow control
  - Return a “window” with every ACK indicating a range of acceptable sequence numbers

Background

- Routers and slower links between sender and receiver may cause congestion
- Slow start, congestion avoidance, fast retransmit and fast recovery to deal with congestion
  - Congestion window (cwnd) and slow start threshold size (ssthresh) for each connection
  - Sender transmits up to min(congestion window, advertised window)

Congestion control algorithm

- Cwnd initialized to one segment and ssthresh to a large value
- If (cwnd <= ssthresh), TCP performs slow start; else it performs congestion avoidance
  - Slow start – cwnd begins at one segment and incremented by one segment for every ACK
  - Congestion avoidance – increase in cwnd is at most one segment per RTT

Congestion control algorithm

- Congestion indicated by timeout or reception of 3 consecutive duplicate ACKs
- When a timeout occurs, set ssthresh to max(window/2, 2) and cwnd to one segment
  - Note: window = min(cwnd, advertised window)
- On receiving 3 consecutive dupacks, fast retransmit and fast recovery are performed

Congestion control algorithm

- Fast retransmit – retransmits apparently missing segment, set ssthresh = max(window/2, 2), cwnd = ssthresh + 3 and enter fast recovery
  - Inflates cwnd by the number of segments that have left the network and that the other end has cached
  - Receipt of dupacks tells TCP more than just a packet has been lost – data is still flowing between the two ends

Congestion control algorithm

- Fast recovery – increments cwnd by segment size each time a dupack arrives and transmits a packet (if allowed)
- When next ACK arrives that acknowledges the retransmitted data, set cwnd = ssthresh and enter congestion avoidance
- TCP Reno includes this congestion control algorithm

Behavior of TCP Reno

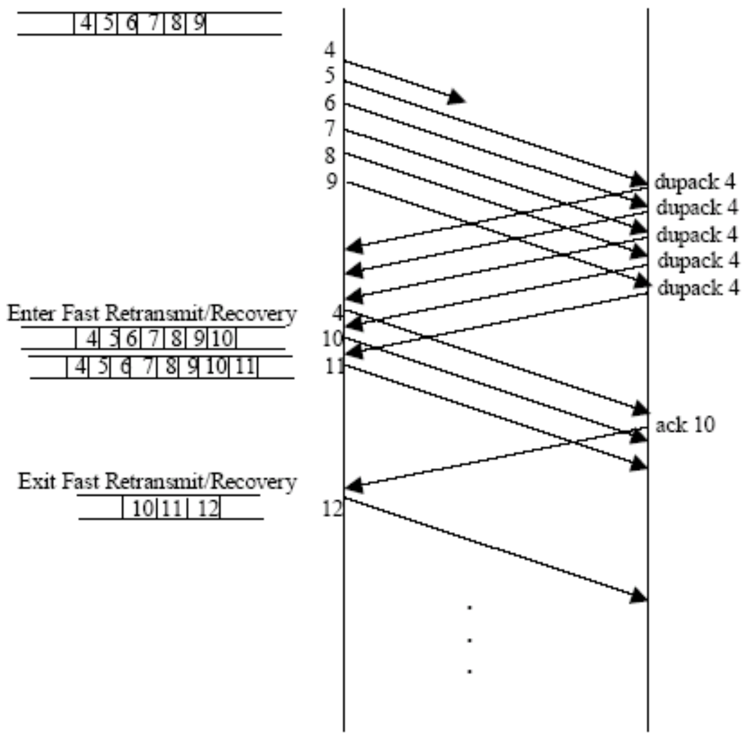

Figure 1. Behavior of TCP Reno in the presence of a single dropped segment in a window of data

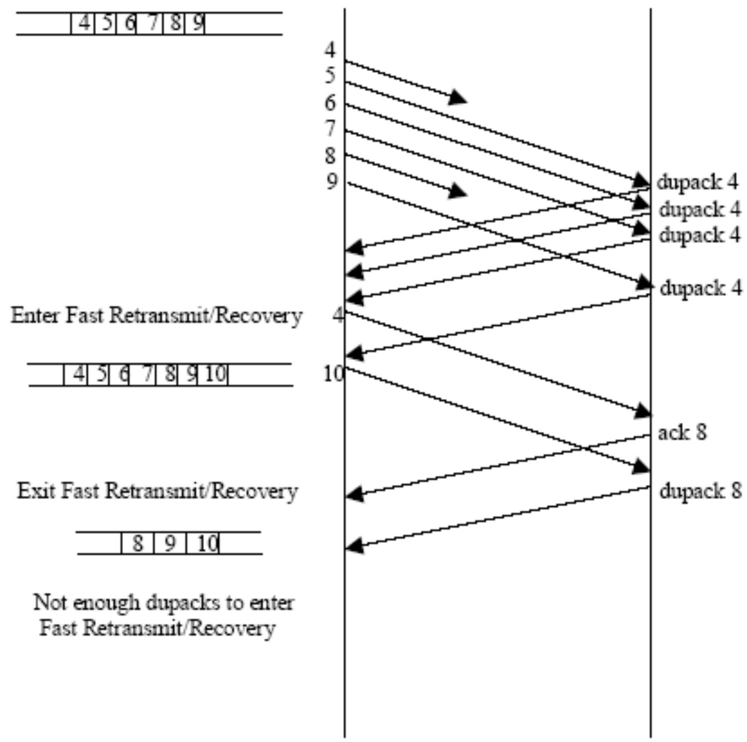

Figure 2. Behavior of TCP Reno in the presence of multiple dropped segments in a window of data

TCP New-Reno

- When a partial ACK is received, it retransmits the first unacknowledged segment in the window and remains in fast recovery
- Remains in fast recovery until all of the data outstanding when fast recovery was initiated has been acknowledged
- When multiple segments are lost from a single window, it recovers without a timeout, retransmitting one lost segment per RTT

SACK

- Retransmitting one lost segment per RTT is still slow
- SACK helps recover faster by providing additional information about the state of congestion
- Uses two new TCP options

Kind = 4	Length = 2

Figure 3. SACK-permitted option

SACK

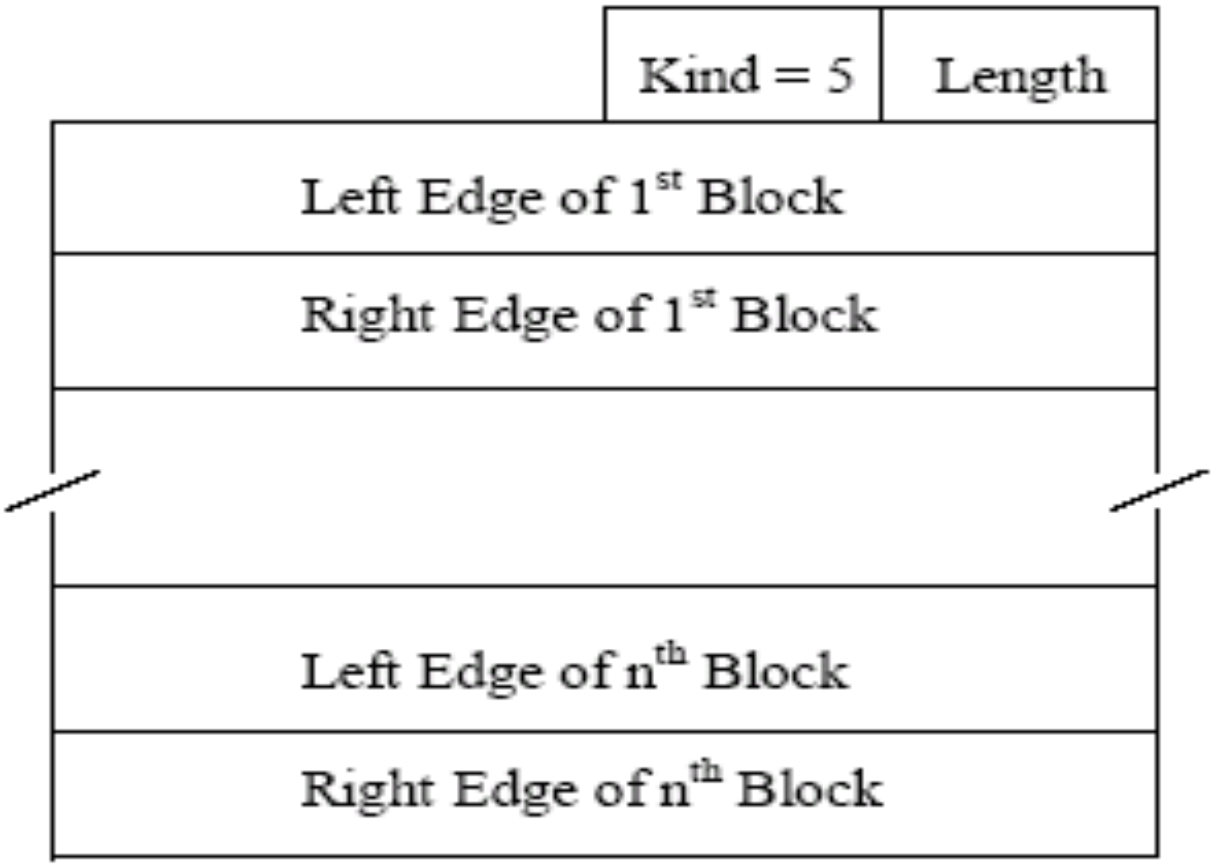

Figure 4. SACK option

Limitations of SACK

- SACK option that specifies “n” noncontiguous blocks will have a length of “8*n+2” bytes
- TCP options space – 40 bytes
- SACK can specify a maximum of 4 blocks
- SACK is often used with timestamp option, reducing the number of blocks to 3
- Introduction of new options may reduce it further

Limitations of SACK

Triggering Segment	ACK	1st Block		2nd Block		3rd Block	
		Left Edge	Right Edge	Left Edge	Right Edge	Left Edge	Right Edge
3500	4000						
4000 (lost)							
4500	4000	4500	5000				
5000	4000	4500	5500				
5500	4000	4500	6000				
6000	4000 (lost)	4500	6500				
6500 (lost)							
7000	4000 (lost)	7000	7500	4500	6500		
7500 (lost)							
8000	4000 (lost)	8000	8500	7000	7500	4500	6500
8500 (lost)							
9000	4000	9000	9500	8000	8500	7000	7500

← Sender would retransmit segment 4000 (beside row 5500)

← Sender would retransmit segment 6000 (unnecessary) (beside row 9000)

Figure 5. Limitation with TCP SACK

ISACK

- For each noncontiguous block, ISACK sends the offset of the left edge from the 32-bit "(cumulative) Acknowledgment Number" field
- Uses 2 new TCP options

Kind = 27	Length = 2

Figure 6. ISACK-permitted option

- Enabling option sent in SYN segment

ISACK option

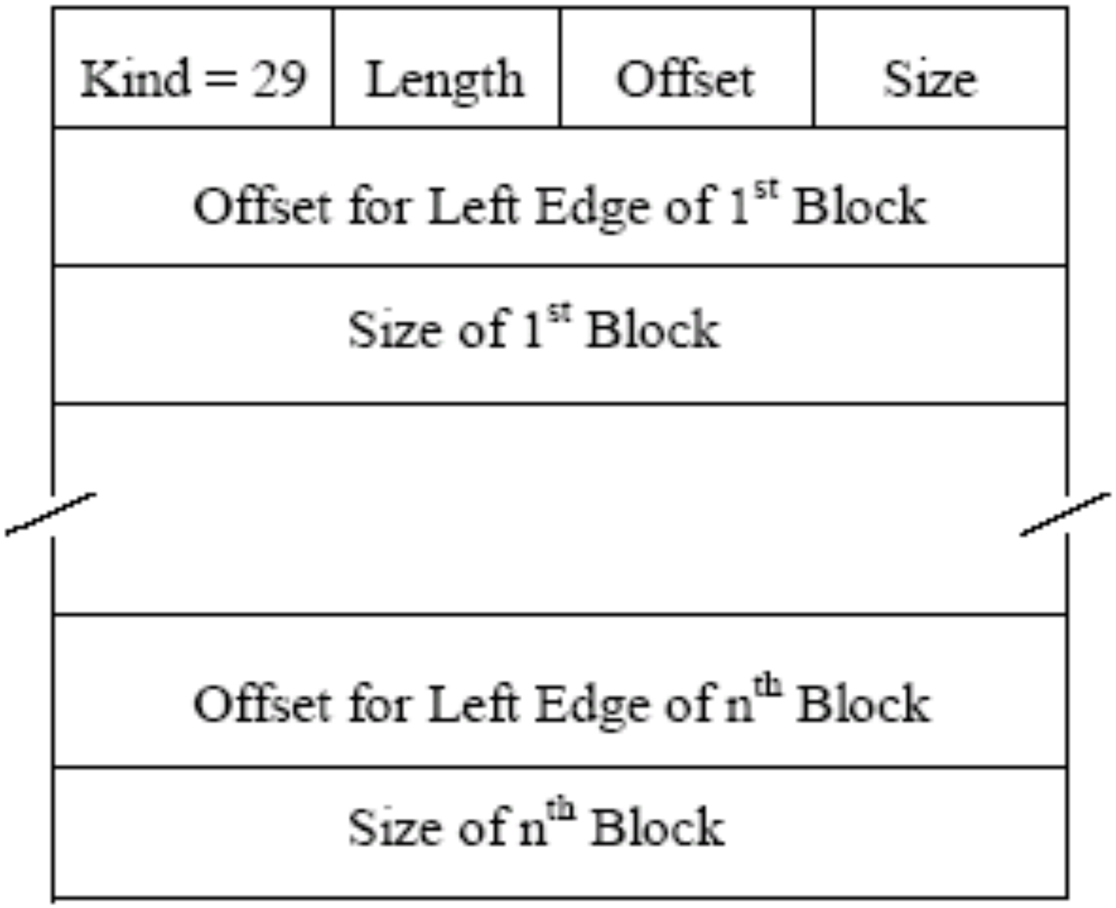

Figure 7. ISACK option

ISACK option

- "Offset" field specifies the number of bits used to represent the offsets of each left edge
  - Value is given by ceil($\log_2$(maxoffset)); maxoffset is the largest among the offsets
- "Size" field specifies the number of bits used to represent the size of each block
  - Value is given by ceil($\log_2$(maxsize)); maxsize is the size of the largest block

Behavior of ISACK

Table 1. Behavior of SACK

Triggering Segment	ACK	1st Block		2nd Block		3rd Block		4th Block	
		Left Edge	Right Edge	Left Edge	Right Edge	Left Edge	Right Edge	Left Edge	Right Edge
5000	5500								
5500 (lost)									
6000	5500	6000	6500						
6500 (lost)									
7000	5500	7000	7500	6000	6500				
7500 (lost)									
8000	5500	8000	8500	7000	7500	6000	6500		
8500 (lost)									
9000	5500	9000	9500	8000	8500	7000	7500	6000	6500
9500 (lost)									
10000	5500	10000	10500	9000	9500	8000	8500	7000	7500

Table 2. Behavior of ISACK

Triggering Segment	ACK	1st Block		2nd Block		3rd Block		4th Block		5th Block	
		Offset	Size	Offset	Size	Offset	Size	Offset	Size	Offset	Size
5000	5500										
5500 (lost)											
6000	5500	500	500								
6500 (lost)											
7000	5500	1500	500	500	500						
7500 (lost)											
8000	5500	2500	500	1500	500	500	500				
8500 (lost)											
9000	5500	3500	500	2500	500	1500	500	500	500		
9500 (lost)											
10000	5500	4500	500	3500	500	2500	500	1500	500	500	500

Behavior of ISACK

- Maxoffset = 4500
- Number of bits used to represent each offset = $ceil(log_2(4500)) = 13$
- Maxsize = 500
- Number of bits used to represent the size of each block = $ceil(log_2(500)) = 9$
- Total number of bits required by ISACK option to specify the 5 noncontiguous blocks = 8(Kind) + 8(Length) + 8(Offset) + 8(Size) + 5*13(offsets) + 5*9(sizes) = 142 bits(18 bytes)

ASACK

- ISACK imposes a little more processing overhead than does SACK
- Use ISACK only when SACK can not convey the information
- ASACK dynamically switches between SACK and ISACK to give optimal performance

Kind = 28	Length = 2

Figure 8. ASACK-permitted option