

Explicit Congestion Notification (ECN)

- ECN for TCP and IP

Explicit Congestion Notification

- TCP has inferred congestion through measurement
 - TCP Reno and Cubic → packets lost when queues overflow
 - Problematic because it increases delay
 - Problematic because of non-congestive loss on wireless links
 - TCP Vegas → increase in delay as queues start to fill
 - Conceptually a good idea, but difficult to deploy when competing with TCP Reno and Cubic
- **Why not have the network tell TCP congestion is occurring?**
 - **Explicit Congestion Notification (ECN)** field in IP header
 - Tell TCP to slow down if it's overloading the network

ECN and IP

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Version = 4				Header Len			DSCP				ECN		Total Length																		
Fragment Identifier										DF MF		Fragment Offset																			
TTL				Upper Layer Protocol				Header Checksum																							
Source Address																															
Destination Address																															
(Options – variable length, padded to 32 bit boundary)																															
Transport Layer Data – variable length																															

- Sender sets ECN bits on transmission:
 - 00 = Doesn't support ECN, 01 = ECN capable
- If congestion occurs, **congested router** sets ECN bits to 11 (“congestion experienced”)
 - Indicates that a queue of packets for some link is getting full, but has not yet overflowed – **signal of congestion set by the network**
 - If the queue overflows, packets are dropped
- Cooperation between network and endpoints

K. Ramakrishnan, S. Floyd, and D. Black, “The Addition of Explicit Congestion Notification (ECN) to IP”, IETF RFC 3168, September 2001. <https://datatracker.ietf.org/doc/rfc3168/>

Diagram shows IPv4, but ECN is also present in IPv6 and works the same

ECN and TCP (1/3)

- Receiver may get a TCP segment within an IP packet marked ECN Congestion Experienced (ECN-CE)
- ECN Echo (ECE) field in TCP header allow it to signal this back to the sender

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Version = 4				Header Len				DSCP				ECN		Total Length																	
Packet Identifier																DF		MF		Fragment Offset											
TTL				Upper Layer Protocol				Header Checksum																							
Source Address																															
Destination Address																															
Source Port																Destination Port															
Sequence Number																															
Acknowledgement Number																															
Data Offset				Reserved				CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Receive Window Size															
Checksum																Urgent Pointer															
[options - variable length]																															
[data - variable length]																															

ECE = 1 on next TCP packet sent by receiver after ECN-CE received; tells TCP sender that congestion occurred

ECN and TCP (2/3)

- Sender reduces congestion window on receiving acknowledgement with ECE = 1 **as if the packet had been lost**
- Sets CWR = 1 in next packet to inform network and receiver it has done so

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31		
Version = 4				Header Len				DSCP				ECN		Total Length																			
Packet Identifier																DF		MF		Fragment Offset													
TTL				Upper Layer Protocol				Header Checksum																									
Source Address																																	
Destination Address																																	
Source Port																Destination Port																	
Sequence Number																																	
Acknowledgement Number																																	
Data Offset				Reserved				CWR		ECE		URG		ACK		PSH		RST		SYN		FIN		Receive Window Size									
Checksum																Urgent Pointer																	
[options - variable length]																																	
[data - variable length]																																	

CWR = 1 in next TCP packet generated by sender after receipt of packet with ECE set

ECN and TCP (3/3)

- ECN lets TCP react to congestion before packet loss occurs
 - Routers signal congestion **before** queues overflow
 - Independent of choice of TCP congestion control algorithm
 - Smaller queues → reduces latency, beneficial for video conferencing and gaming

- ECN extensions also exist in QUIC and RTP

ECN and TCP: Deployment

- TCP endpoints needed to be updated to support ECN → done
- Endpoints now support ECN and **many** now enable ECN by default
 - ECN was widely implemented but disabled-by-default for many years, due to concerns that it wouldn't pass through firewalls
 - Apple forced firewalls to be fixed: iOS 9 enabled ECN for 5% of connection; iOS 10 for 50%; iOS 11 and later for all connections
- Most routers support ECN, relatively few enable it by default
 - Endpoints can react to ECN signals, but network is currently unlikely to provide them
 - Starting to change – e.g., recent DOCSIS cable modems enable ECN
 - Will take time, but latency reduced as ECN is enabled throughout the network

Explicit Congestion Notification (ECN)

- ECN for TCP and IP