Reconsidering the usage of stand-alone packets for Congestion Indication

S. Harhalakis, N. Samaras and V. Vitsas

February 13, 2010

Abstract

With the introduction of Active Queue Management and Explicit Congestion Notification, Internet Routers' queues have become an active part of Congestion Control. This letter points out that data flows with one packet per-direction are not allowed to take advantage of ECN, meaning that they will experience unprivileged handling by intermediate routers and presents a method for reducing the unintended packet drops of AQM when it is combined with ECN. The method is backwards compatible and can be deployed incrementaly without introducing security risks. The method is applied on DNS traffic showing significant improvement of user experience.

1 Introduction

Router queues consist a shared resource [7]. In most cases, when the used portion of a queue reaches a certain limit some or all of the newly arrived packets are dropped. Transport layer protocols like the Transmission Control Protocol (TCP) that support flow control interpret packet loss as congestion indication and reduce their transmission rate [1]. Generally, packet loss acts as a signal that indicates congestion and may indirectly notify the transmitting party to reduce its transmission rate. This behavior is common among all protocols that support congestion control and is also the accepted behavior for user-space congestion-control implementations [4].

Drop-tail queues drop packets whenever they become full while Active Queue Management (AQM) [2] techniques start indicating congestion before a queue is full. When using AQM, packet drops are the most common approach to congestion indication. Random Early Detection (RED) [2] is probably the most popular AQM mechanism. In order to avoid congestion and prevent global synchronization [3] RED probabilistically starts dropping packets whenever the average queue length reaches a limit and becomes a drop-tail mechanism whenever the queue length reaches a hard limit. This way RED indicates congestion when it starts to occur instead of waiting for the queue to become full.

Explicit Congestion Notification (ECN) [6] is an addition to AQM that allows for congestion indication without actually dropping packets. When using ECN, RED queues mark instead of dropping packets unless they have reached their maximum queue size. Since packets aren't dropped there are fewer retransmissions which result in better network efficiency and protocol behavior. ECN's specification dictates that only packets that carry an ECN capable upper layer protocol should be marked instead of being dropped. Those IP packets are distinguished by an ECN Capable Transport (ECT) codepoint that is set by the transmitting node. When those packets traverse AQM-based queues they are marked with the Congestion Encountered (CE) codepoint instead of being dropped, unless there is no available buffer space. This marking is handled by the upper layer protocol at the receiver which informs the transmitter for impeding congestion. In the case of TCP, this is accomplished by using the "ECN Echo" (ECE) flag. Non-ECT traffic is never marked and is always dropped in order to indicate impeding congestion.

Even though all packets are candidates for drops, only a subset of them are candidates for marking. This means that ECN capable transport layer protocols are treated in an privileged way since their packets are only dropped whenever the queue becomes drop-tail while all other packets may be dropped a lot earlier. Looking at ECN's specification, it is obvious that it was not intended to act as a quality of service (QoS) method. However, because of the way it is used it results in a behavior where packets with an ECT codepoint be-

come drop-candidates much later than packets without the ECT codepoint.

2 Stand-alone packets

A fundamental assumption of all deployed congestion control methods is that a data flow has more than one packet per direction. This means that it is impossible to have flow control with only one packet per-direction. Since ECN can only be used with an upper layer protocol that supports congestion control (like TCP), we denote that stand-alone packets cannot take advantage of ECN and are thus treated in an unprivileged way. This happens because stand-alone packets cannot be subjects of congestion control.

Having in mind the Domain Name System (DNS) protocol, we note that there are delay sensitive (i.e. with high impact on user experience) data traffic exchanges with only one packet per direction that cannot take advantage of ECN. We thus consider all traffic that:

1. Has at-most one packet per direction,
2. When dropped will trigger a retransmission after a timeout,
3. When dropped will result in user observable delay, degrading user experience, and
4. When not dropped will not significantly increase the average queue size. This means that it must not be produced in bursts and must not consist a significant portion of Internet traffic.

and we refer to its packets as "special stand-alone packets". Based on that definition we introduce two proposals: An extension of ECN's usage and the application of that extension to a subset of DNS traffic.

3 ECN Proposal

We consider the current approach of AQM somehow unfair to special stand-alone packets. There is no documented intention for AQM to drop those packets since dropping them when there is available buffer space will not serve congestion control. Multiple methods can be proposed in order to alleviate this problem, but a realistic proposal should (a) require as few modifications as possible from the underlying network and the remote endpoints and (b) ensure backwards compatibility.

It is proposed that special stand-alone packets be marked with the ECT codepoint by transmitting nodes. This will instruct intermediate queues that use AQM to mark them instead of dropping them when there is available buffer space. The proposed ECN approach is backwards compatible and requires no modification in the underlying network. Existing AQM and ECN enabled routers will be able to handle this kind of traffic. Even communicating endpoints do not need to both support this method, meaning that it is possible for such a solution to be deployed incrementally.

The classification of packets as special stand-alone should be based on the rules of the previous section. When a proposal is made for the packets of a kind of traffic (e.g. UDP-based DNS queries) to be considered as special stand-alone, it must also:

- Prove that the average queue size of intermediate queues will not be significantly affected,
- Prove that other traffic will not be disrupted and
- Justify that it will not introduce security problems.

4 DNS Proposal

DNS is considerd one of the most crucial protocols of the Internet with high impact on end-users. Most DNS queries are performed using the UDP protocol in order to reduce delay and overhead. By definition, UDP-based DNS queries and their replies are one-packet per direction data flows[1]. DNS resolving is usually the first step of most world-wide-web transactions and its delay is almost the delay that a user experiences before (e.g.) a web page starts to download. DNS queries are retried after a predefined, system-wide time-out that depends on the operating system. For Windows systems, the timeout of the first retry is 1 second and for Linux (glibc) systems it is 5 seconds.

Based on the above observations, we consider UDP-based DNS queries and their replies as special stand-alone packets because:

- They are one-packet per-direction data flows,
- Each query is retransmitted after a predefined timeout. Replies are indirectly retransmitted after a new (repeated) query,

[1] DNS specification dictates that queries that require more that one datagram must be performed using TCP.

- Packet drops result in user-observable delay that affects user experience and
- DNS traffic consists a very small portion of Internet traffic

We thus propose that the ECN extension be used for UDP-based DNS queries and replies. IP packets that carry UDP datagrams should be marked with an ECT codepoint even though the traffic cannot react to congestion control. We choose the ECT(0) codepoint for this purpose and leave the ECT(1) codepoint unspecified for future use as recommended by [6] for cases where only one ECT codepoint is required.

Attempts	w/o #	w/o %	w/ #	w/ %	T/out	Improvement
1	1262	74.37 %	1507	88.08 %	5	19%
2	306	18.03 %	173	10.11 %	5	43%
Failed	129	7.60 %	31	1.81 %	-	76%

(a) Linux (glibc)

Attempts	w/o #	w/o %	w/ #	w/ %	T/out	Improvement
1	1205	68.70 %	1485	84.18 %	1	23%
2	393	22.41 %	233	13.21 %	2	41%
3	113	6.44 %	40	2.27 %	2	65%
4	36	2.05 %	5	0.28 %	4	86%
5	6	0.34 %	1	0.06 %	8	83%
Failed	1	0.06 %	0	0 %	-	100%

(b) Windows

Table 1: Effects of ECN on DNS query attempts

5 Test results

We performed of simulations using a modified version of the NS-2 simulator in order to determine the effects of applying the ECN extension on DNS traffic. The modifications allowed for ECN marking of MessagePassing packets which were used to simulate DNS queries and replies. The simulations tested the network behavior and DNS effectiveness when adding the ECT codepoint to DNS packets while operating over a congested link. Congestion was achieved by using PackMime TCP sessions with ECN enabled. The purpose was to examine (a) whether the average queue size is affected and (b) whether the proposed modification actually improves user experience. The behavior of both Windows and Linux (glibc) was tested by simulating their DNS resolvers. The test scenario consisted of:

- one 10MBps link that was congested on one-direction only
- RED queues
- 20 DNS clients
- one DNS server.

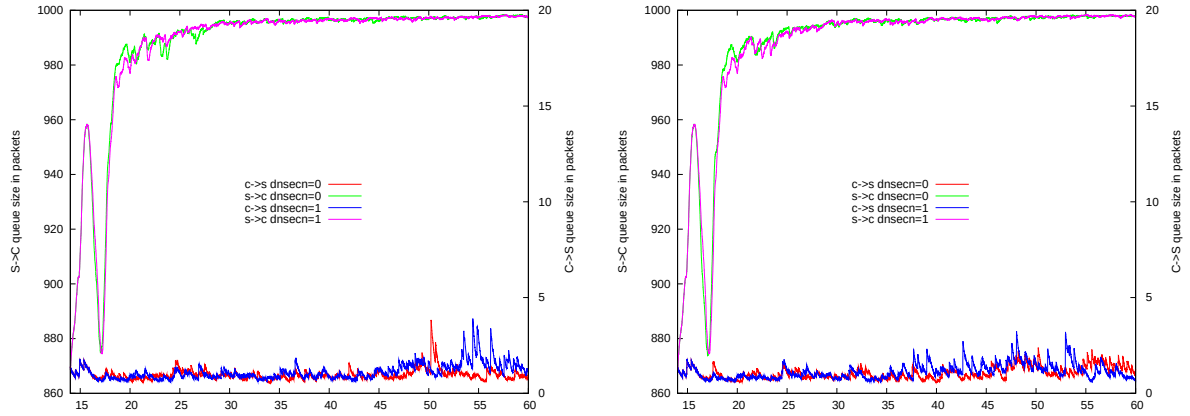
Each DNS client performed a new query every half a second. Linux queries timed-out after their second attempt and windows queries timed-out after their fifth attempt. The tests covered a 15-seconds warm-up period and a 45-seconds measuring period. Many of the simulation parameters were based on the parameters that were used for the tests of [5] which proposes the addition of ECN to TCP's SYN/ACK packets and seem to be well accepted. For the targeted load of the intermediate link, 95% was used as a somehow modest approach for a congested link.

Figure 1 shows the effects on the queue size from the ECN usage for both directions. The graphs indicate that there is no increase in the average queue size. Table 1 shows the number of required attempts for a DNS resolving as an absolute number (#) or a percentage (%), with (w/) and without (w/o) ECN. It also shows the improvement.

When not using ECN, more than 25% of DNS queries required a retry causing delays of at least 1 second for Windows and 5 seconds for Linux systems. This delay is well in the user-observable limits, meaning that the loss of a DNS query or reply packet is far worse than the loss of a TCP segment. The simulations also showed reduced DNS traffic since there were fewer retransmissions and no significant effect on the average queue size or the background traffic.

6 Conclusions

AQM and ECN are two related promising improvements for the Internet. However, it seems that their combination is somehow unfair to non ECN capable traffic. This letter points out that there are stand-alone packets that may be worth of taking advantage of ECN marking even though they are currently not allowed to. Those packets are currently considered as drop-candidates for the purpose of congestion indication de-



(a) Linux resolver (glibc)

(b) Windows resolver

Figure 1: Effects on queue size

spite the fact that this indication will not be utilized by the endpoints and that such a drop may negatively impact user experience. It is proposed that ECN semantics are extended and ECN marking be allowed for special stand-alone packets, as they are defined in this letter. UDP based queries and replies of the DNS protocol are a good example since they are one-packet per-direction data transfers that have high impact on user experience. Currently, they are drop candidates for congestion indication even though it seems impossible for a resolver or a server to ever support flow control effectively. Future work may examine whether there are other Internet protocols who's packets can be considered as special stand-alone packets.

References

[1] M. Allman, V. Paxson, and W. Stevens. TCP Congestion Control. RFC 2581 (Proposed Standard), April 1999. Obsoleted by RFC 5681, updated by RFC 3390.

[2] B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Partridge, L. Peterson, K. Ramakrishnan, S. Shenker, J. Wroclawski, and L. Zhang. Recommendations on Queue Management and Congestion Avoidance in the Internet. RFC 2309 (Informational), April 1998.

[3] Sally Floyd and Van Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, 1(4):397–413, 1993.

[4] M. Handley, S. Floyd, J. Padhye, and J. Widmer. TCP Friendly Rate Control (TFRC): Protocol Specification. RFC 3448 (Proposed Standard), January 2003. Obsoleted by RFC 5348.

[5] A. Kuzmanovic, A. Mondal, S. Floyd, and K. Ramakrishnan. Adding Explicit Congestion Notification (ECN) Capability to TCP's SYN/ACK Packets. RFC 5562 (Experimental), June 2009.

[6] K. Ramakrishnan, S. Floyd, and D. Black. The Addition of Explicit Congestion Notification (ECN) to IP. RFC 3168 (Proposed Standard), September 2001.

[7] Rayadurgam Srikant. *The Mathematics of Internet Congestion Control (Systems and Control: Foundations and Applications)*. SpringerVerlag, 2004.