Measuring the State of ECN Readiness in Servers, Clients, and Routers

Steven Bauer
MIT CSAIL
bauer@mit.edu

Robert Beverly
Naval Postgraduate School
rbeverly@nps.edu

Arthur Berger
MIT CSAIL / Akamai
awberger@csail.mit.edu

ABSTRACT

Better exposing congestion can improve traffic management in the wide-area, at peering points, among residential broadband connections, and in the data center. TCP's network utilization and efficiency depends on congestion information, while recent research proposes economic and policy models based on congestion. Such motivations have driven widespread support of Explicit Congestion Notification (ECN) in modern operating systems. We reappraise the Internet's ECN readiness, updating and extending previous measurements. Across large and diverse server populations, we find a three-fold increase in ECN support over prior studies. Using new methods, we characterize ECN within mobile infrastructure and at the client-side, populations previously unmeasured. Via large-scale path measurements, we find the ECN feedback loop failing in the core of the network 40% of the time, typically at AS boundaries. Finally, we discover new examples of infrastructure violating ECN Internet standards, and discuss remaining impediments to running ECN while suggesting mechanisms to aid adoption.

Categories and Subject Descriptors

C.2.3 [**Computer Communication Networks**]: Network Operations—*network monitoring*; C.2.1 [**Computer Communication Networks**]: Network Architecture and Design

General Terms

ECN, Measurement, Experimentation

Keywords

Explicit Congestion Notification, Router-Assisted Congestion Control

1. INTRODUCTION

The Internet's traditional end-to-end approach to congestion control, TCP, attempts to simultaneously optimize efficiency and fairness between traffic flows[14]. Cooperating end host TCPs must *infer* congestion via packet loss. As a distributed control problem, on the whole, TCP works remarkably well and has scaled through orders of magnitude speed increases. However, a large body of prior work recognizes several shortcomings.

First, the TCP rate equation biases against particular flows, particularly those that are short-lived or with high delay. Second, given the ubiquity of wireless devices, loss may be an inappropriate congestion signal. Third, the TCP feedback control loop necessarily results in under-utilization and oscillatory behavior (the TCP "sawtooth"). Finally, large bandwidth-delay flows are unable to fully utilize available bandwidth as the bandwidth-probing is conservative [14].

A well-studied approach to congestion is active queue management. Routers along a path know precisely their buffer occupancy and may signal an end host to slow its sending rate in the face of incipient congestion, i.e. without inducing packet loss. Explicit Congestion Notification (ECN) is a standardized TCP addition providing such feedback [24].

The benefits of ECN are only realized when end-hosts and routers support ECN. Today, equipment vendors largely support ECN and all major operating systems (Windows, Linux, OSX) support ECN. After more than a decade, much of the Internet has the capability to leverage ECN. However, few networks employ ECN and few devices at the edge initiate ECN connections.

The adoption of ECN may change given a renewed interest in managing congestion (e.g. [12]). Recent work illustrates how ECN may be leveraged to provide new Internet primitives and semantics. In addition to IETF standards leveraging ECN [10, 23], the IETF congestion exposure (conex) group is developing experimental standards that expose the source and location of congestion along a path potentially enabling new economic and billing models [7]. Among service providers, congestion, rather than traffic volume, is a potential alternative basis for interconnection and peering agreements [8]. Similar arguments apply to using induced congestion to bill and monitor residential broadband connections rather than adopting volume caps [4].

Finally, managing congestion is increasingly important within data centers and cloud providers. The common practice of partitioning a problem (e.g. web query) and aggregating the results must be performed within stringent latency requirements. Long-lived heavy bandwidth flows hamper this ability by inducing large queues. Recent proposals based on an analysis of real-world data center traffic rely on ECN to better handle such workloads [2].

In light of recent system and infrastructure adoption of ECN and the growing realization that ECN provides potential benefits to managing congestion [15], we perform large-scale measurements to reappraise the Internet's ability to properly support ECN. Our primary contributions include:

1. An examination of ECN support among various server populations, including updated measurements of a previously surveyed population [17, 22].
2. Analysis of millions of Internet paths to identify where ECN fails or is mis-marked, and the implications for congestion safety.
3. Case studies of infrastructure ECN misbehavior; implications on deployment and protocol design.
4. A novel measurement of client-side, e.g. eyeball, ECN support.

2. BACKGROUND

Explicit Congestion Notification (ECN) [24, 26, 6] defines an enhancement to TCP/IP that provides router-assisted congestion control in the Internet. The fundamental idea is to allow routers along a path, devices with explicit knowledge of their output queue occupancy, to *mark* packets of a TCP flow. This mark is *echoed* back to the source TCP. The source TCP thus receives notification of incipient congestion, allowing it to reduce congestion *before* loss occurs.

Mechanically, ECN adds two additional bits in the flags portion of the TCP header: ECN Echo (ECE) and Congestion Window Reduced (CWR), and changes the original semantics of the 8-bit type-of-service (ToS) field in the IP header. Instead of a byte of ToS, six bits are used for the Differentiated Service Code Point while two remaining bits encode three possible ECN states: no-ECN '00', ECN Capable Transport (ECT) '10' or '01', and Congestion Experienced (CE) '11'. As shown in §4, legacy handling of these bits as the old ToS field is a frequent cause of ECN failure.

ECN capability is *negotiated* between TCP end-points by setting the ECE and CWR flags in the SYN and the ECE flag in the SYN/ACK of TCP's three-way handshake. If negotiated, TCP sets ECT in the IP header of data segments. Routers along the path may, in response to congestion, set CE on packets marked as ECN capable with ECT. Upon receiving a segment with CE, a TCP sets ECE on all acknowledgments until the source reduces its congestion window and sets CWR.

ECN requires both TCPs and routers along an end-to-end path of a flow to support ECN. There are several points of potential failure or misbehavior. In this paper, we examine all of these points. In particular, we ask several questions:

1. What fraction of hosts successfully negotiate ECN?
2. When TCP negotiates ECN, is the connection marked as ECN capable at the IP layer?
3. Do routers and/or middleboxes erroneously clear or set congestion marks?
4. If a synthetic IP congestion signal (CE) is introduced, is a corresponding TCP ECE observed?
5. If a synthetic TCP ECE is sent, is the corresponding TCP CWR observed?

Previous measurement efforts [17, 22] found little support for ECN: on the order of 2% of servers or less, and an even smaller percentage of flows with ECN apparently enabled [19]. These results were not surprising as ECN was turned off by default on most operating systems due to an ECN blackhole problem. Of servers tested [22] in 2008, TCP SYN packets with ECN set were silently dropped 0.56% of the time. However, the recent widespread adoption of, and support for, ECN in operating systems suggests that a reappraisal of previous studies is warranted. For instance, server-mode ECN was introduced into the Linux kernel (2.6.29) in 2009, and enables ECN for incoming TCP connections that indicate ECN capability. Thus, while ECN is finally part of all major operating system TCP/IP stacks, it is being deployed conservatively – clients must be explicitly configured to initiate ECN for outgoing TCP connections, presenting a challenge we address in §3.3.

Given the potential network impact of these widespread changes, we reexamine and extend previous studies of ECN. In addition to enhanced testing of all the ECN messages two hosts can exchange, we also sought to test other parts of the network for ECN readiness, not just popular web servers. These include client-side (e.g. eyeball) networks such as residential broadband networks and tests that provide insight into the use of ECN by mobile devices. Mobile infrastructure is of interest not only because of the particular utility of ECN in wireless environments, but also because operators often control both the handset and network proxies.

3. METHODOLOGY

3.1 Server Testing

We measure three web server populations: 1) the top one million web servers [1] as ranked by Alexa; 2) a collection of approximately 7500 university and college web servers identified by [11]; and 3) web servers that handle requests from mobile devices (which we discuss separately below).

For each domain, we retrieve the complete web page (HTML and all page resources hosted by the same domain) using a client configured to initiate outgoing ECN connections. We use the native TCP stack of the measurement host's Linux operating system to retrieve the resources. Using Linux's iptables firewall connection tracking and mangling rules we artificially introduce all ECN signals.[1] We simulate congestion along the outgoing path by setting CE on all data packets sent by the measurement host and we simulate congestion along the incoming data path by reporting ECE to the remote sender. Our implementation guarantees that all data packets, including retransmissions, sent by our client have CE set until ECE is heard. Similarly it guarantees ECE is sent in response to all incoming data packets until CWR is heard. If a server fails to respond to SYNs with ECN, we retry the server with a non-ECN enabled SYN to identify possible ECN blackholes. A benefit of our implementation is that we can easily test any application or service which relies upon TCP – a feature we leverage to test additional populations described below.

We capture all packets that occur during the exchange and analyze the resulting trace. The analysis of whether or not ECN is negotiated on a particular connection is straightforward – does the response SYN/ACK from the server have the ECE flag set. Next, we examine whether the ECN capable transport code point (ECT) is set in the IP header of data packets for those flows which negotiate ECN. Note, re-transmitted packets and packets without data do not set ECT. However, if no data packets are marked with ECT,

[1]See `http://ecn.csail.mit.edu` for full details.

Table 1: Server population results; "-" indicates no data available

Year/Population	2004	2008	Alexa	University	Mobile
Hosts classified	80,498 (100%)	1,349,711 (100%)	541,885 (100%)	7563 (100%)	3,591 (100%)
ECN-capable	1,765 (2.2%)	14,407 (1.1%)	93,232 (17.2%)	1061 (14.0%)	559 (15.6%)
Not ECN-capable	78,733 (97.8%)	1,335,304 (98.9%)	448,653 (82.8%)	6,502 (86.0%)	3032 (84.4%)
Possible ECN blackhole	814 (1.0%)	7,627 (0.6%)	3293 (0.6%)	37 (0.5%)	5 (0.1%)
ECT in non-ECN flow	-	-	698 (0.15%)	13 (0.2%)	9 (0.3%)
no ECT in ECN flow	758 (42%)	-	5,598 (6.0%)	302 (28.5%)	63 (11.3%)
no ECE for CE	1,302 (1.6%)	-	3,995 (4.3%)	267 (25.2%)	21 (3.8%)
no CWR for ECE	-	-	55 (0.05%)	0 (0.0%)	1 (0.2%)

we show with our traceroute measurements that something along the path likely is improperly clearing the ECN bits. (We discuss some of the causes for this misbehavior in later sections.) We believe that all major operating systems implement ECN functionality correctly

Since we set CE on the client's packet that contained the HTTP request (and all potential retransmissions), we expect to see a ECE in all traces where the web server responded with a HTTP status message. The analysis of CWR however is dependent upon the number of data packets we receive from the server since some data packets will already be in flight by the time the server receives the first ECE message (which we only send in response to data packets from the server). We attempt to maximize the number of packets a server sends by setting the TCP's maximum segment size option to 300. We also turn off all segmentation and receive offload features in Linux.

We report on full ECN functionality tests from a single vantage point at MIT, but have conducted limited testing from other vantage points, in particular from non-Internet2 connected networks. We describe the ECN tests of paths that we run from > 125 PlanetLab [25] locations below.

3.2 Mobile Infrastructure

ECN can be particularly beneficial in wireless environments where loss does not necessarily indicate congestion. We therefore seek to characterize the ECN capabilities and behavior of web server side wireless-specific infrastructure by obtaining a population of servers dedicated to hosting content for wireless mobile devices, e.g. cellular smartphones.

We note that the content provided from a web server to a client is often tailored to the properties of the client device, especially for mobile devices. Additionally, mobile devices are frequently redirected to a different, dedicated server for mobile content. We leverage this common practice to obtain a population of servers dedicated to serving mobile content for subsequent ECN analysis.

For all domains in the Alexa 1M population, we issue five different HTTP GET queries, each with a distinct "User-Agent" HTTP header string. The user agent string mimics a desktop, and four different mobile phones. If the queried web server returns an HTTP 3xx redirect to a location different than the desktop query for any of the phone queries, we record the location. For example, a query from a desktop to `www.bloomberg.com` is not redirected, while a query from a phone user agent is redirected to `mobile.bloomberg.com`; these hostnames correspond to different IP addresses.

Using this technique, we find ∼82,000 sites performing HTTP redirection on the basis of the user agent. Of those, 7,422 redirect to distinct mobile infrastructure, i.e. redirect to a URL that resolves to a different IP address. We use the later population for analysis of mobile web sites.

3.3 Client Testing

Understanding client-side ECN support is important for several reasons, notably since clients represent "eyeballs" in the network, i.e. humans sensitive to congestion. However, while prior work (§2) investigates the prevalence of server-side ECN, client-side support has received little attention.

Maier et al. observe a negligible number of hosts initiating ECN capable TCP in a large residential broadband network [19]. The lack of ECN in Maier's study is unsurprising as operating systems that support ECN, with typical default settings, did not and still do not negotiate ECN for outgoing connections. Because passive measurements provide only a limited view into client-side ECN capabilities, we develop a new hybrid passive/active method.

Our technique measures a large section of two peer-to-peer (P2P) networks – where nodes act as both clients and servers – to capture the ECN behavior of a population of Internet *clients*. We build BitTorrent and Gnutella crawlers on an ECN-enabled measurement host under our control. Our BitTorrent [9] crawler discovers torrents via aggregated RSS feeds, connects to the torrent's tracker(s), and obtains a list of torrent peer IP addresses. The crawler attempts a BitTorrent handshake with each peer, thereby initiating ECN and sending data segments without transferring content. Simultaneously, we capture all packets for analysis. For Gnutella, we use an existing crawler [27].

Our crawlers negotiate ECN for all connections to remote nodes in the P2P network, the majority of which reside in residential networks. By observing the behavior of the negotiation, and subsequent response to synthetic ECN signals introduced using the aforementioned iptables rules (§3.1), we gain insight on the prevalence of ECN on an end-to-end basis in the Internet.

3.4 End-to-End ECN Path Testing

If a router does not have any features turned on that leverage the ECN field, it should not modify the ECN field in any way. From conversations with network operators, none reported turning on ECN markings. However, we leverage traceroute to identify links where the ECN bits in the IP header are being modified. ICMP TTL-exceeded messages include the first 28B of the packet that expired, the so-called ICMP "quotation." This quotation provides path-level visibility into the ECN field. We make use of this feature by setting the ECT code point in a series of tests from our PlanetLab nodes to the web server populations described above.

While most ICMP quotations are reliable and accurate, like [20] we find some small evidence of quotation error.

Filtering differences raise the question of which type of probe (ICMP, UDP, TCP-SYN, TCP-ACK) should test a path. We did not find any apparent differences in how routers responded to packets with the ECN field set in a comparison of probe types to a 10,000 node sample set. Therefore we used Scamper's [18] ICMP-paris traceroute mode [3] (even though ICMP packets would not normally be ECN marked) since it was likely to discover more hops.

4. RESULTS

4.1 Server results

Table 1 provides our results of ECN testing to the web server populations in September of 2011. Where applicable, we include results from previous studies in 2004 [21] and 2008 [17] for comparison purposes. The number of servers which negotiated ECN rose to between 14% to 17% in all web server populations. A possible ECN SYN blackhole existed in less than 0.6% of tests. But we caution that this may over-estimate the percentage as false positives could occur both in our methodology and the previous studies if a lossy link happened to drop the ECN enabled SYN packets but not the subsequent SYN packets without ECN. Definitively identifying ECN blackholes would require repeated probing of the same server with both ECN and non-ECN SYNs.

The rest of the table rows report various ways in which the ECN congestion feedback loop can be broken. We conservatively report on not receiving a CWR in response to ECE only if we receive more than 10 data packets from the server i.e. we receive a subsequent flight of packets after the server must have received ECE notifications. The variation across different populations is striking – the ECN field in the IP header on flows to and from university networks is being cleared 25.2% and 28.5% of the time respectively.[2] This is far higher than the comparable percentages of paths to Alexa web hosts – 3.8% outbound and 11.3% inbound.

In limited testing from different vantage points, the percentages of ECN capable servers across target populations remained consistent. The results characterizing the broken ECN feedback loop showed more variation. This is expected given that these results depend upon the network path between our measurement vantage points and the target servers. Our traceroute measurements below shed additional light on these variations.

4.2 Client-side results

Table 2 shows that fewer clients negotiated ECN on incoming connections; 4.2% to less than 0.1% depending on population. To better understand this discrepancy, we employ stack fingerprinting to the TCP SYN/ACK packets [5] to infer the operating system of the client. Table 3 shows a striking difference between the ECN capable and non-ECN capable populations: the vast majority of ECN capable hosts are Linux (88.4%).

4.3 End-to-End ECN Paths

We collected two sets traces to determine *where* on the path the ECT codepoint was cleared, and related aspects. The first trace set includes one randomly-chosen destination

[2]When there is no ECE for a CE, either the outbound CE could be being cleared or the inbound ECE.

Table 2: Client Population Data

Population	Gnutella	BitTorrent
Hosts classified	126,984 (100%)	16,602 (100%)
ECN-capable	106 (0.1%)	701 (4.2%)
Not ECN-capable	126,878 (99.9%)	15,901 (95.8%)
ECT in non-ECN flow	2,153 (1.7%)	452 (2.8%)
no ECT in ECN flow	44 (42%)	117 (17%)
no ECE for CE	10 (9%)	167 (24%)
no CWR for ECE	0 (0.0%)	0 (0.0%)

Table 3: Inferred O/S of Client Populations

O/S	Gnutella (All)	Gnutella (ECN)	BT (All)	BT (ECN)
Windows	29.9%	0.0%	24.6%	0.0%
Mac/BSD	3.9%	29.5%	8.9%	4.4%
Linux	1.1%	46.6%	7.0%	88.4%
Other	65.1%	23.9%	59.5%	7.14%

from each of the globally routable BGP prefixes ($\simeq$ 367,000) in order to touch all origin AS's and be broadly representative of Internet paths. The second trace set represents "popular" destinations, and thus used the 542,000 unique addresses from the Alexa top 1 million [1]. For the prefix-traces, we used 127 PlanetLab nodes, from 29 countries and 5 continents, to initiate the traces. And for the website-traces, we were able to use some additional nodes for a total of 140. Each PlanetLab node executed a trace to each of the destinations, for a total of 117 million traces collected.

Percent of Traces which passed the ECT codepoint

The location of any devices that modify the ECT codepoint relative to our tracing vantage points is important. At one extreme, ECT is cleared on 100% of the traces originating from particular PlanetLab nodes due to all traces passing through a misbehaving device at the first or second hop. Traces from these nodes are not reflective of the whole Internet, but are relevant for the clients at that location.

ICMP filtering behavior impacts our macro analysis as well. For a few vantage points, the vast majority of their traces are blocked after a few hops, and for the hops that did respond, the ECT codepoint remained set; thus these traces could be grouped with those that preserved the codepoint, though in reality farther along the path the codepoint might have been cleared.

Across all traces, the ECT codepoint is unmodified for 83% and 82% of the popular website and prefix traces respectively. Restricting the analysis to those traces where all hops respond, including the destinations, does not qualitatively affect our results, yielding 84% and 80% respectively. To remove the impact of nodes where the ECT codepoint is cleared on almost all of the traces, if we omit the 20% of nodes that had the fewest traces where the ECT remained set, then for each of remaining nodes, the ECT codepoint remained set on 94% to 98% of the website-traces and 90% to 99% of the prefix-traces.

Location on the path where the ECT codepoint is cleared

When the ECT codepoint is modified, we wish to attribute such misbehavior to an interface hop on the path. However, there is ambiguity in identifying the responsible device. For example, while sending probes with ECT set, suppose interface hops 1-4 respond with ICMP TTL exceeded quotations

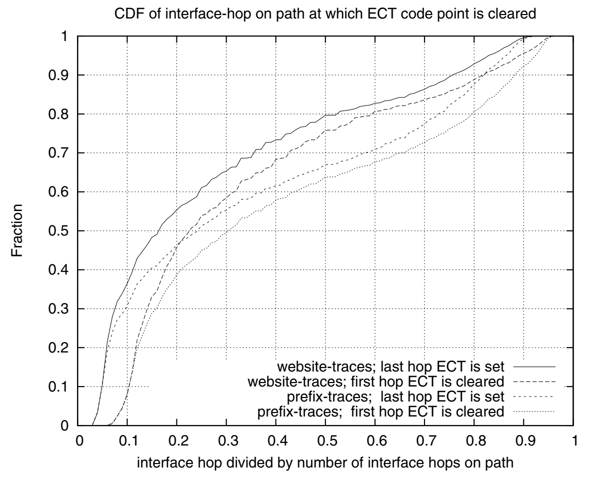

Figure 1: CDF of interface-hop on path at which ECT codepoint is cleared

that preserve ECT, while the fifth hop's quotation shows ECT cleared. The responsible device could be the router at the fifth hop, the router at the fourth hop, or some middlebox (e.g. a firewall, switch, or load balancer) between the two that does not decrement TTL. The case of the prior router being responsible occurs when routers decrement TTLs and expire packets before performing other processing on the incoming packet. As mentioned in Section 3.4, there is also the possibility of ICMP misquotations.

To resolve this ambiguity, we collect into a set "A" the interfaces that did not clear ECT, i.e. interfaces where, on any trace, the response from the next hop preserved ECT. Assuming that an interface/router is consistent as to whether it clears or preserves ECT, then for all the traces, the interfaces in "A" should not clear ECT. We find cases where: (1) the interface first reporting ECT cleared is in "A," and (2) the prior interface is in "A." In our traces, one case does not dominate the other. More interestingly, we discover a third case where both interfaces are in "A," which suggests the action of middleboxes, and/or casts into doubt the assumption of consistency. In the following we do not assume which box cleared the codepoint, but report the two bounding possibilities (given that the returned value is accurate).

For traces where all interface hops respond, and where ECT is cleared, we record the hop number of the last router interface that preserves ECT, and of the interface that first reported the cleared codepoint. By normalizing by the number of interface hops on the path, Figure 1 shows the CDF of the relative position of ECT misbehavior along probed paths. Examining at the bottom curve, "prefix-traces; first hop ECT is cleared," for 40% of the traces the interface hop which first reports ECT cleared is at 20% or less along the path. At the upper end of the CDF, for 20% of the traces, the hop which first reports ECT cleared is in within the last 20% of the path. Thus 40% of the traces fall in-between. All four CDF's show that there is a substantial portion of the mass in the middle range of the path – suggesting that there is significant misbehavior within the core of the network. (For both the website and prefix traces, the median number of interface hops was 16 and only 4% of the paths had 10 or fewer hops.) Since the last hop where the ECT is set always comes before the first hop where ECT is cleared, the CDF of the former lies above that of the latter. Figure 1 shows that, overall, the ECT is cleared earlier on paths to the popular websites.

AS's where the ECT codepoint is cleared

Attributing cleared ECT to an AS introduces a well-known ambiguity: router interfaces between AS's may be assigned addresses from either peer AS, or even use outside or private addresses. We perform a basic disambiguation by scanning across all traces and looking for instances where the subsequent hop's interface address is from the address space originated by the same AS as the address of current hop. With this requirement, many interface hops are not resolved to an AS, and are simply labeled as "unknown." With this determination, we then scan through the traces a second time, looking for where the ECT codepoint is cleared. In the future, we will perform a more thorough procedure to determine border routers [13].

In analogy to Figure 1, we would like to estimate where on the AS path the ECT codepoint is first cleared. For the set of prefix-traces where all hops responded, we revisit the subset on which the ECT codepoint was cleared, and the pair of router-interface hops between which this first occurred. Using our restricted inference technique, frequently one, or both, of these router interfaces can not be associated with certainty to an AS. This is expected since some providers still clear the legacy ToS byte at their network ingress, i.e. between AS's. We find that 55% these pairs of router-interface hops have one of the AS's classified as "unknown," and 11% have both so classified, and thus 34% have both hops associated with an AS. When the two AS's are known, for 99% of the traces, the two AS's are the same; and for these, 32% are the same as the source AS, 17% are the same as the destination AS, and the remainder, 51%, are distinct from the source and destination AS. Thus we do see cases where the ECT codepoint is cleared interior to an AS (that is not the source AS).

Of all the AS's seen on the prefix-traces, only about 1% were detected to have cleared the ECT codepoint. In particular, of the over 37,000 AS's seen on the prefix-traces, only 177 AS's were detected to have definitely cleared the codepoint (the two AS's that bound the clearing of the codepoint are the same); and only an additional 343 AS's might have cleared the codepoint. Though, if the codepoint had not been cleared at the given interface, it possibly would have been cleared at a subsequent interface/AS that was not detected by our experiment.

In order to proceed with the estimation, we assume an interface whose AS has been classified as "unknown" is actually in the nearest, previous AS that was known on the path. Since we know the AS of the source, this assumption allows us to assign an AS number to every interface on the path. Note that this assumption biases the location at which the ECT codepoint is cleared towards earlier on the AS path.

The results are presented in Table 4 (This is analogous to the plot "prefix-traces; first hop ECT is cleared" in Figure 1.) We restrict the table to AS-paths lengths that accounted for at least 1% of the traces. For a given AS-path length (a given row in the table) and for each AS hop on the path, we report the percent of traces at which the ECT codepoints is first observed to be cleared. For paths of length 4, the most popular length, the ECT codepoint was cleared at an intermediate AS on 30% of the traces, and for paths of length

Table 4: AS hop at which ECT is cleared

Number of AS hops on path	Percent of traces with given number of AS's	For each AS hop on path, precent of traces in which ECT codepoint is first cleared
2	1	58 42
3	19	60 25 15
4	40	58 11 19 12
5	28	44 16 12 19 9
6	10	37 8 28 11 12 4
7	1	37 4 35 6 11 6 1

5, the percentage is 47%. If the calculation is repeated where we no longer require that all interfaces on the path respond, but only that the destination responds, the percents in Table 4 are roughly the same.

Note that since our assumption of assigning "unknown" AS's to the previous known AS on the path introduces a bias towards earlier on the path, the percents in Table 4 underestimate how far along the path the ECT codepoint is cleared. Also recall that Table 4, and Figure 1, report the location where the ECT codepoint is *first* cleared; and thus if the codepoint had not been cleared at the given interface, it possibly would have been cleared at a subsequent one. Thus, the percents in Table 4 should be viewed as approximations. However, the implication from Table 4, consistent with Figure 1, is that the ECT codepoint is not only being cleared in the origin (and destination AS), but also at intermediate AS's.

Other notable behaviors

For 99% of the website and prefix traces, the quoted ECN field from probe responses had either ECT set for all hops or had ECT cleared at some hop and the field remained cleared. For the remaining 1% of the traces, we observe a variety of behaviors. On some traces, the reported ECT codepoint changes several times between set and cleared; a few traces had as many as 14 switches in value. We also find instances where the Congestion Experienced (CE) codepoint is set. Across the two trace sets, there were 98 unique hop pairs for which the ECN field switched to CE. For these pairs, the CE codepoint is often set in conjunction with the diffserv codepoint having a non-zero value, thus in all likelihood the network is making use of the legacy ToS field.

5. MITIGATION OPTIONS

While ECN nonce[26] is often seen as a mechanism to ensure that a receiver correctly reflects the congestion signal carried in the ECN bits of the IP header, it can also serve as a mechanism for ensuring correct ECN behavior along a path (see section in 6.2 [26]). Network devices that mangle the ECN bits destroy the ability of the receiver to correctly calculate the ECN nonce sum. How a sender responds to incorrect nonce sums is a matter of policy, but ceasing to set ECT in outgoing segments would be one obvious response. Across all of our data sets, we see no instances of ECN nonce support. Manual inspection of arriving packets with ECT1 set reveals that these flows always set ECT1, even when ECN is not negotiated by TCP. (Thus, TCP continues to rely on cooperative and friendly behavior.)

One other mitigation strategy for dealing with paths that mangle the ECN bits in the IP header is to check upon receipt of an ECN enabled SYN packet that the ECN field is zero and refusing to enable ECN if it is not. While RFC 5562 suggests adding ECN capabilities to SYN/ACK packets (thus allowing a non-zero ECN field), TCP SYN packets should never be ECN marked [16]. While this approach does not help on paths that clear the ECN field, which is the common error, it does address some particularly problematic manglings of the ECN field like always setting CE (i.e. signaling constant congestion.)

6. CONCLUSION

While some impediments to wider use of ECN appear to be diminished (i.e. servers have increasingly enabled it), an important finding of this paper is the presence of a non-trivial number of paths that mangle the ECN field in the IP header. This problem compromises a carefully designed congestion feedback loop and potentially raises concerns about the congestion safety or fairness of using ECN if senders do not receive the signal to slow their rate. Eventually a sender would overflow the bottleneck queue and then back off, but other flows that share partial paths would be disadvantaged.

Similar concerns exist in the case of congestion echo (ECE) not making it to the sender. While the possibility of such behavior is mentioned in [24], repeated tests to servers demonstrate this is not just a random event but rather can be a persistent problem along a path. If the CWR is being cleared on the other hand, the problem is instead that the sender receives a constant stream of ECE flagged messages and therefore never opens its congestion window – resulting in very low throughput.

While the existence of these problems is potentially discouraging we have found that the problems can often be rapidly fixed. We identified problems on both our lab and our residential broadband network. Both networks were quickly fixed when we supplied packet captures and traceroutes illustrating the problem. In the case of our lab, the intent had been to copy the 802.1p field from Ethernet to DSCP but an Ethernet switch instead ended up overwriting all eight bits of the ToS field. In the case of the broadband provider, the intent had been to clear the diffserv field but instead the entire old ToS field was being set to zero. We suspect that many of the problems we found have similar origins. Other potential sources of error include NATs and home routers, load balancers, tunnels, and other types of middle-boxes. Our tools and methodology will hopefully aid in quickly finding and fixing these problems.

Acknowledgments

We thank Jim Gettys and all the anonymous reviewers for their feedback and insights. R. Beverly was partially supported by a Cisco Research Grant. Views and conclusions are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. government.

7. REFERENCES

[1] Alexa. Top 1,000,000 sites, 2011. `http://www.alexa.com/topsites`.

[2] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan. Data center tcp (dctcp). In *Proceedings of ACM SIGCOMM*, 2010.

[3] B. Augustin, X. Cuvellier, B. Orgogozo, F. Viger, T. Friedman, M. Latapy, C. Magnien, and R. Teixeira. Avoiding traceroute anomalies with paris traceroute. In *Proceedings of the 6th ACM IMC*, 2006.

[4] S. Bauer, D. Clark, and B. Lehr. The evolution of internet congestion. In *Proceedings of the 37th TPRC*, 2009.

[5] R. Beverly. A robust classifier for passive tcp/ip fingerprinting. In *Proceedings of the 5th Passive and active network measurement conference*, pages 158–167, 2004.

[6] B. Briscoe. Tunnelling of Explicit Congestion Notification. RFC 6040 (Proposed Standard), Nov. 2010.

[7] B. Briscoe, A. Jacquet, C. Di Cairano-Gilfedder, A. Salvatori, A. Soppera, and M. Koyabe. Policing congestion response in an internetwork using re-feedback. In *Proceedings of ACM SIGCOMM*, 2005.

[8] B. Briscoe and S. Rudkin. Commercial models for ip quality of service interconnect. *BT Technology Journal*, 23:171–195, April 2005.

[9] B. Cohen. Incentives build robustness in bittorrent, 2003.

[10] P. Eardley. Metering and Marking Behaviour of PCN-Nodes. RFC 5670 (Proposed Standard), Nov. 2009.

[11] K. Forster. Universities worldwide, 2011. `http://univ.cc/`.

[12] J. Gettys. Bufferbloat, 2011. `http://www.bufferbloat.net/`.

[13] B. Huffaker, A. Dhamdhere, M. Fomenkov, and K. Claffy. Toward topology dualism: improving the accuracy of as annotations for routers. In *Proceedings of the 11th Passive and active measurement conference*, 2010.

[14] D. Katabi, M. Handley, and C. Rohrs. Congestion control for high bandwidth-delay product networks. *SIGCOMM Comput. Commun. Rev.*, 32, August 2002.

[15] A. Kuzmanovic. The power of explicit congestion notification. In *Proceedings of ACM SIGCOMM*, 2005.

[16] 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.

[17] A. Langley. Probing the viability of tcp extensions, 2008. `http://www.imperialviolet.org/binary/ecntest.pdf`.

[18] M. Luckie. Scamper: a scalable and extensible packet prober for active measurement of the internet. In *Proceedings of the 10th ACM IMC*, IMC '10, 2010.

[19] G. Maier, A. Feldmann, V. Paxson, and M. Allman. On dominant characteristics of residential broadband internet traffic. In *Proceedings of the 9th ACM IMC*, 2009.

[20] D. Malone and M. Luckie. Analysis of icmp quotations. In *Proceedings of the 8th Passive and active network measurement conference*, 2007.

[21] A. Medina, M. Allman, and S. Floyd. Measuring interactions between transport protocols and middleboxes. In *Proceedings of the 4th ACM IMC*, 2004.

[22] A. Medina, M. Allman, and S. Floyd. Measuring the evolution of transport protocols in the internet. *SIGCOMM Comput. Commun. Rev.*, 35, April 2005.

[23] T. Moncaster, B. Briscoe, and M. Menth. Baseline Encoding and Transport of Pre-Congestion Information. RFC 5696 (Proposed Standard), Nov. 2009.

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

[25] N. Spring, L. Peterson, A. Bavier, and V. Pai. Using planetlab for network research: myths, realities, and best practices. *SIGOPS Oper. Syst. Rev.*, 40, January 2006.

[26] N. Spring, D. Wetherall, and D. Ely. Robust Explicit Congestion Notification (ECN) Signaling with Nonces. RFC 3540, June 2003.

[27] D. Stutzbach, R. Rejaie, N. Duffield, S. Sen, and W. Willinger. On unbiased sampling for unstructured peer-to-peer networks. *IEEE Trans. Netw.*, 17, April 2009.

Summary Review Documentation for

"Measuring the State of ECN Readiness in Servers, Clients, and Routers"

Authors: S. Bauer, R. Beverly, A. Berger

Reviewer #1

Strengths: The results: hard, new datapoints on ECN readiness over much Internet infrastructure (still a good way to go but starting to build) and data on various pathologies that is useful to work out how to best improve ECN readiness (such as knowing that the core is clearing ECT relatively often). A well-executed measurement study in a very readable paper that looks at an impressive number of Internet paths over a variety of sub-populations.

Weaknesses: Well, you could say that the paper is small in scope, just giving better datapoints on one TCP/IP mechanism, and unimportant, given that the mechanism is not much used in practice. But I think this would be a bit unfair.

Comments to Authors: These comments are intended to help you improve your paper further. (1) What I liked most was your division of Internet devices into different sub-populations and the exploration of "what went wrong". Both help to move us beyond a simple but unhelpful statistic ("we find that the Internet is 15% ECN ready"). Where you could go further is with ways to improve the status quo based on your findings. Are there simple tests that TCP/IP implementations should perform to avoid being adversely affected by the pathologies (detect and turn off?) or do they not matter much? Note that the ECN nonce can potentially help here by serving as a robust testing mechanism for correct implementations along a path -- if used properly it will detect middleboxes interfering with the ECN bits as well as "cheating hosts". (2) Also, the material in looking for the location of problems along the path did feel rather speculative, since you cannot come to firm conclusions, but I appreciated it nonetheless.

Reviewer #2

Strengths: Repetition of old results with new outcomes. ECN deployment is still underway, but there are new challenges. Comprehensive testing for a short paper.

Weaknesses: None.

Comments to Authors: Nice short paper. Good coverage of a moderately interesting topic. Most of the interest is in the fact that there is a positive trend, but it's disheartening to see how slow the positive trend is.

The information in S4.2 would have been much more interesting if divided by AS instead of by hop-count. Once the network is blocking ECN, IMHO it's more important who is doing it rather than where.
It would have been useful to correlate the BitTorrent implementations types (by bittorrent id) or the Gnutella implemenations (by user-agent) to the ECN-capable nodes to better understand which types of nodes are ECN-capable.

Reviewer #3

Strengths: Fairly broad study of ECN. Results are interesting. Good positioning of results and contribution into the historical context.

Weaknesses: Methodology is confusing; esp description in 2 and the state machine in Figure 1 seem to not match later text. While results are interesting, their significance can be improved, for example, by listing out the top fewcauses of ECN misbehavior today.

Comments to Authors: Writing is mostly good, though another pass would help. Results are interesting. A more precise breakdown of the types of idiosyncracies that are happening, their potential causes, and fixes would be very useful. In the sense that, they would prescribe how ECN can be better supported. (The examples at the of Sections 4 and 5 are steps along the right path.)
State machine in Figure 1 is cryptic. What you really mean to do, is to use iptables rules to mimic a congestion event along the path as a router would have otherwise done (mark CE on packets). However some of the transitions are based on packets coming in from different directions; ECE would come from the local client, CWR would come from the remote host. Also, I can't tell what the left path of the tree is doing.
OSes have ECN support but don't turn it on by default, i.e., if the other end does not request ECN they wouldn't use it. Is this accurate? Can you quantify how many clients/ OSes are like this?
There seems to also be a fair bit of re-use of the ToS field, some of which doesn't seem to interact correctly with ECN. It would help to be more concrete about these anomalies.
In 3.4, wouldn't using TCP @port 80 get you better pass through rates from proxies?
From Table 1, it isn't clear that mobile servers are doing any better on ECN.Neither are the more popular servers or university clients. Is there bias in the experiments, with the measuring host, or the first few links on the path to the measurement host(s) dominating the results here?
In 4, Is 'hop' an IP hop, as visible from ICMP? That would be vulnerable to not seeing mpls segments or other circuits.
The ambiguity, when ECT is cleared, whether it is cleared on the outgoing part of the previous hop or the incoming part on the next hop doesn't seem to be worth chasing down. Localizing it to that pair would be helpful in itself. You might even get away by looking at multiple pairs that share an endpoint to get a better handle on which of those two end points may be clearing that bits.

Using p2p traffic to measure client eyeballs is cool, but probably not a big deal.

Reviewer #4

Strengths: Keeping track of the ECN capabilities of Internet devices is important to improve TCP's behavior across the Internet.

Weaknesses: Given that lack of ECN capabilities of the measured end-hosts, I do not see a point in this paper. The methodology for the mobile infrastructure is weak, while this is the environment where ECN is likely to matter more.

Comments to Authors: To understand mobile devices' ECN, the authors can't simply go to the mobile version of websites, but also need to conduct measurements from mobile devices themselves or use NAT devices as used by mobile operators.
Why exactly are passive measurements not a reliable indicator of the prevalence of client-side ECN support? Is it the number of observed end-hosts? I don't quite understand. Why would a P2P-biased view be better? If the authors want to make claims a bout some bias, then please argue.
From table 2, given that most hosts are not ECN-capable, why should we care about what routers do? If end-hosts were ECN capable and impeded by routers on the way, I would understand. But given the current deployment of ECN, the paper is weak.
What is the relationship between the paths that you are probing and the original end-hosts for which you wanted to check the ECN capabilities?
Adding to Figure 2 the absolute hop count and AS hops where ECT is cleared would help. The discussion on figure 2 is too short. The authors spend much time in the paper on discussions and methodological considerations, while not really giving a take-home message. I would see this as a CCR paper, not so much a research paper

Reviewer #5

Strengths: This paper presents a thorough study of the current support for ECN. It measures all three parts: servers (including servers for mobile clients), clients, and network paths. Nice technique to measure client support by crawling Bittorrent and Gnutella networks.

Weaknesses: Having updated numbers is interesting, but the results are not that surprising. The paper lacks more analysis of the causes for the small ECN support and the reasons for network elements to mistakenly clear ECN bits.

Comments to Authors: This paper presents a nice update on ECN support. It is important to study servers, clients, and network paths to fully understand the potential for ECN utilization. At the end the results and the analysis are not very novel.
The paper does present a separate characterization of web servers for mobile devices, but the results are similar to other websites and there is no further discussion about this point.
The technique to crawl p2p clients to study client support for ECN is nice. In general, it is hard to measure the client side. The results are not very telling, because the paper presents no characteristics of these clients. Is there a way to find out the operating system and other configuration details of these clients? It would be interesting to complete these tests with hosts where you know the OS, so that you can study of different OSes for ECN.
Sec. 3.4 says that it uses ICMP traceroute probes, because the use of ECN doesn't change depending on the protocol. It would be nice to have at least a short description of this experiment.
In Sec. 4.2 paragraph 6, is it true that routers are consistent in the way they treat the ECN bits? In this section, it would be nice to have a discussion of why routers sometimes clear the ECN bits. Maybe bringing the last paragraph of the discussion to this section would help the readers understand this point.
The results in Fig. 2 are biased because after a router clears the ECN bits, if another router in the path also clears these bits, you cannot detect it. So, it is expected that this curve is shifted to the left. The paper should at least discuss this bias or even change the presentation of this result. It makes more sense to discuss the ASes or pair of ASes where ECN bits are cleared. The analysis in terms of fraction of the total number of hops is confusing.
Also in Sec. 4.2, last paragraph under "AS's where the ECT codepoint is cleared," why use the restricted inference technique? What is the fraction that is not labeled? Instead of trying to pinpoint an AS, you could characterize pairs of ASes.

Response from the Authors

We valued and addressed the feedback from all reviewers. The most valuable contribution came from Reviewer 1 who points out the fact that an ECN nonce (RFC 3540) can serve as a robust testing mechanism for correct ECN behavior along a path not just correct ECN behavior by receivers. This is noted in RFC 3540 but we had overlooked it.
A number of the reviews noted confusion on how we implemented the server testing using the iptables functionality of Linux. There are a number of details to the implementation that are subtle particular if one is not familiar with iptables rules and capabilities or the exact protocol details of ECN. We updated the description of our methodology to hopefully make it more clear but perhaps more importantly we have all our configuration scripts and testing tools available for any reader wishing to replicate our measurements. A benefit of our implementation is that one can easily test any application or service that relies upon TCP -- a feature we leverage to test both web servers and hosts running p2p software without needing any additional implementation effort.
Some of the reviews noted a general lack of interest in our ECN measurements. Our interests simply differ. When we started this research we intended it as an email to the IETF Conex WG. The discovery of the devices along the path mangling the ECN bits both at our home and lab networks suggested there was a more interesting phenomenon worthy of investigation. We were encouraged by a number of individuals to conduct a careful study. Even if one is not interesting in ECN in particular, there are interesting lessons in regards to the challenges associated with reusing fields (ToS to diffserv/ECN) or using previously reserved bits (TCP's CWR and ECE flags). Also our traceroute technique of diagnosing ECN problems can be used for other purposes, one of which is a survey of diffserv code points being set by each AS (a study we are currently conducting.)