

ECN IS FINE - BUT WILL IT BE USED?

M. Małowidzki

Military Communication Institute
05-130 Zegrze, Poland
malowidz@wil.waw.pl

ABSTRACT

The paper discusses Explicit Congestion Notification (ECN) in the context of the commercial Internet. It presents possible threats to ECN and shows the gain malicious ECN sources could achieve from ECN subverting. Besides, the paper analyses the relationships between ECN and Service Level Agreements (SLAs).

KEYWORDS

ECN, TCP, IP, congestion control, SLA

INTRODUCTION

Explicit Congestion Notification is a congestion avoidance scheme that uses marking packets instead of dropping them in the case of incipient congestion. The receivers of marked packets should return the information about marked packets to the senders, and the senders should decrease their transmit rate. To avoid heavy congestion, routers mark packets with probability depending on an average queue length.

The idea is not new and has been studied in various networks at least since the beginning of the 90s (see [2] for a reference). For example, ECN in ATM networks takes the form of the EFCI bit set in ATM cell headers; the information about congestion is returned by the receiver in management cells. Similar mechanisms are present in Frame Relay, too. In IP networks, ECN [3] relates mainly to TCP traffic but both IP and TCP headers are used to forward ECN-related information.

The idea looks promising and its advantages have been verified in [4]. However, in the context of IP networks, which currently support mainly best-effort mode with very limited QoS support, some questions may be asked. First, if the scheme is secure enough that malicious users cannot exploit it to gain better performance. Second, how ECN relates to Service Level Agreements (SLAs) used in the Internet?

The paper is organized as follows: First, ECN is described in greater detail. Second, the possible threats to ECN are discussed and the possible results of an attack are presented. In the following sections, ECN is discussed in the context of the best-effort and SLA-based

commercial Internet. The final section contains conclusions.

EXPLICIT CONGESTION NOTIFICATION

Random Early Detection (RED) [1] is a proposed scheme to react to congestion in its initial phase. Instead of waiting for the output queues in routers to overflow (and then drop all packets according to TailDrop approach), routers start dropping packets from some threshold with increasing probability, based on the average length of the output queue. ECN refines this approach and uses packets marking instead of dropping. In both cases, affected TCP flows should back off, which would allow avoiding the state of heavy congestion.

ECN mechanism ([3], see also Fig. 1) uses two bits in the IP header (in the former ToS byte, now redefined by DiffServ to be the DSCP field) and two (or, as proposed in [5], three) bits in TCP header. The two IP bits carry information about if the flow the packet belongs to is ECN-capable (the two equally valid ECT codepoints) and, if it is, whether congestion has been experienced (the CE codepoint). The two TCP header bits, ECE (ECN-Echo) and CWR (Congestion Window Reduced), are used to inform the sender about congestion or that the receiver has reduced its congestion window, respectively. Moreover, these bits are used during TCP connection setup phase to negotiate the willingness to use ECN. The third proposed bit, NS (Nonces Sum), is discussed later in the paper.

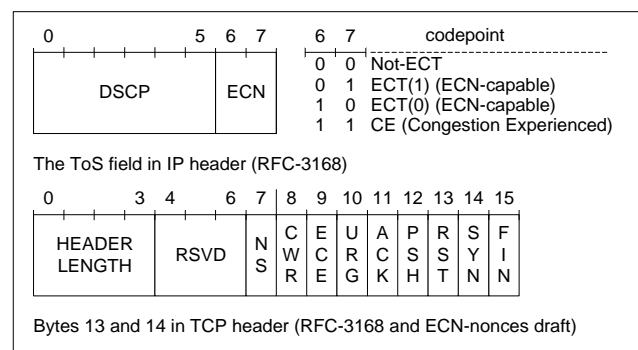


Fig. 1: The new definitions of some fields in IP and TCP headers

For ECN to work it is required that TCP endpoints and routers cooperate. Routers using ECN RED in the case of incipient congestion mark ECN-capable packets (with one of the ECT codepoints set) or drop non-ECN-capable packets (with the Not-ECT codepoint). Packets already marked are simply forwarded. If the TCP endpoint receives a marked packet, it sets the ECE bit in TCP acknowledgements traveling back to the sender until it receives a packet with the CWR bit set. On the other side, if the TCP endpoint sees the ECE bit set, it should react in the same way as in the case of a single packet being dropped, i.e. halve its congestion window *cwnd* and reduce the slow start threshold *ssthresh*.

The advantages of ECN have been verified in [2], [4]. Nonetheless, they are also checked in the paper. The immediate observation is that ECN allows to avoid packet drops (at least while congestion is still relatively mild) and reduces packet delay due to keeping queues short. Moreover, TCP endpoints can distinguish between packet loss due to transmission errors and congestion signals – this would be especially useful in networks with relatively high bit error rates (e.g., radio networks).

ECN starts to be supported in TCP implementations as well as in networking equipment. For example, Linux kernel 2.4 or later has full support for ECN; Cisco routers also support ECN from IOS Release 12.2(8)T. Some web sites have already enabled ECN, so it is not impossible to observe ECN-capable packets traveling in the Internet. On the other hand, [12] lists software and hardware incompatible with ECN, e.g. firewalls dropping packets with ECN-related flags set.

THE THREATS TO ECN

Before we proceed, we will define a reference model for further discussions (see Fig. 2). Three sites, A, B and C, are connected to a carrier network offering *IP connectivity*. They have SLAs with carrier network's operator and use its network to connect to the Internet. The fourth site, D, is connected to the Internet through site A.

Besides, we would like to introduce the concepts of a *macro flow* and a *micro flow*. A macro flow is generated by a network operator's customer (e.g., the sites from Fig. 2) and consists of many micro flows, that is, single TCP connections.

In the paper, we assume that most of the traffic belongs to TCP. This assumption is valid currently and seems to remain valid for the next few years, at least to the moment of VoIP traffic explosion.

RFC-3168 contains especially complex section about security and discusses possible ways to subvert ECN. Generally, there are two possible attacks: a *micro-scale attack* (that is, "improved" TCP implementation) and a

macro-scale attack (altered router's behavior). This will be next discussed in details.

A Micro-Scale Attack

A modified TCP implementation could:

- not modify its congestion window in the presence of ECE bit set and/or
- not set ECE bit in ACKs after a marked packet has been received
- finally, what also can be regarded as a form of an attack, resign from verifying if the peer endpoint is honest

The first possibility is obvious: A malicious endpoint simply ignores congestion indication. The second approach does not allow the peer to back off and is especially attractive when most data is transmitted in the reverse direction (to the malicious endpoint, as it is in the case of HTTP or FTP clients). Similarly, the third approach could be attractive for commercial web sites – they could be indulgent to cheating clients.

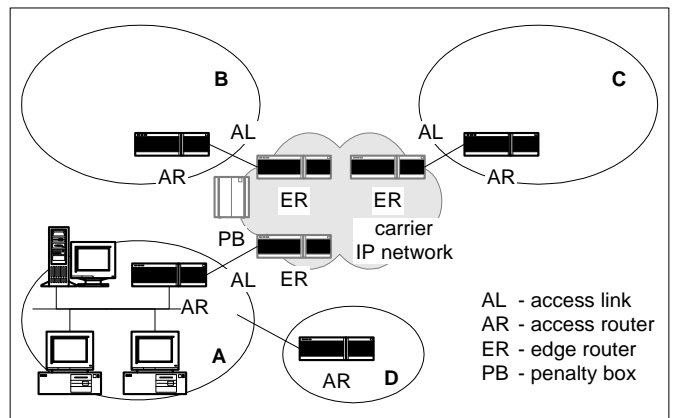


Fig. 2: The reference model

There are reliable ways of defense against first two "improvements" (described in [3]) provided that *the peer endpoint is honest and compliant*. A compliant endpoint could monitor its peer's behavior or use *CE probes* (outgoing packets with CE codepoint set) to verify if the peer returns congestion information. A better way would be to use *ECN nonces* [5] which make use of the fact that there are two possible values for ECN-capable codepoint. The sender generates nonces, that is, random one-bit values encoded in the two ECT codepoints (ECT(0) represents a nonce of 0, and ECT(1) – a nonce of 1). The receiver is required to set the NS bit to the value of the one-bit sum of all nonces over the byte range represented by the acknowledgement. A mistake allows guessing about a marked packet (setting CE codepoint erases information about the original ECT codepoint). The third attack described above would assume that an endpoint refrains from using any of the two mechanisms.

Yet another line of defense is so-called *penalty box*, proposed in [3]. Penalty boxes would operate somewhere in the network (usually, at its edges), trace traffic and penalize misbehaving ECN flows by dropping their packets. We are going to comment more on penalty boxes later in the paper. However, here we only note that due to predicted incremental deployment of possibly various ECN schemes appearing gradually (e.g. [8]), their task in distinguishing compliant flows from non-responsive ones may be especially difficult.

We mention here about micro-scale attacks for completeness mainly; we are more interested in macro-scale ones. This is generally more difficult to alter TCP implementation (a source code must be available). Moreover, we believe that future TCP will be better protected from possible attacks ([7] describes successful attacks against today's TCP implementations and a reliable defence, i.e. adding a nonce to each TCP segment. The same idea is used by ECN nonces [5]). Besides, subverted TCP implementations could compete with other TCP flows *within the same macro flow*, and this might be highly undesirable. Finally, we believe that in most situations single subverted flows could only cause a limited damage.

A Macro-Scale Attack

A macro-scale attack involves the router (typically, the site's access router) with malicious software. The router can perform two kinds of subvertive actions:

- intercept TCP connection setup messages and modify TCP header bits to block ECN (and disable its verification mechanisms)
- modify ECN-related bits in packets (e.g., falsely mark packets as ECN-capable or clear congestion indication, that is, change CE codepoint to one of the ECT codepoints and clear the ECE and NS bits)

These attacks may be detected by a compliant ECN implementation observing ECN bits and using CE probes or nonces. However, if two malicious sites cooperate in this way (*coordinated macro-scale attack*), penalty boxes would be the only defense. Usually, the routers involved in the attack would enforce non-ECN connections for their local TCP endpoints (no need to maintain per-connection state information) and later mark packets as ECN-capable, but the attack is feasible even if full ECN features would be to be used (in such a case, the routers would have to care for fooling ECN probing and nonces checking performed by their local endpoints).

For example, assume that in our reference model (Fig. 2), site C offers some services (WWW, mail servers, recently popular *instant messaging* servers etc). to other sites. For traffic originating from A, C may behave as a complaint ECN site. However, B and C could secretly cooperate in ECN subverting and C could take part in a coordinated attack to give B better communication

performance. A real-world example would be a company that uses the Internet to connect its branches.

THE GAIN FROM ECN

To assess and verify the gain from using ECN when compared to drop-based RED, we performed tests using real-world equipment and software (Fig. 3). We used Linux system with kernel version 2.4.7-10, which implements NewReno-style TCP with SACK and offers full support for ECN (ECN may be enabled or disabled at runtime with a simple command). The FTP server used ECN-compliant TCP while the FTP clients (marked as A and B in Fig. 3) used various TCP implementations, depending on the performed test. Cisco router 1720 had installed IOS Release 12.2(8)T with ECN support. Its configuration was the same during all tests – it used RED with ECN marking with *minth* and *maxth* thresholds set to 20 and 40 packets, respectively. The *exponential weight* parameter was set to 9.

Two modes of operation were examined (as in [4]): bulk and transactional transfers. In the first case, a long, 2 Mbyte file was downloaded by clients. Various link capacities and delays were examined. In the second case, clients downloaded small files (8 kB) one after another. This time, link capacity was set to 64 kb/s and round-trip delays varied from 20 to 100 ms. In all cases, the tests lasted for 5 minutes. To verify that everything worked as expected (i.e. the router really marked packets, and that the subverted implementation – see below – was really subverted), a sniffer collecting packets with any of the ECN flags set was used.

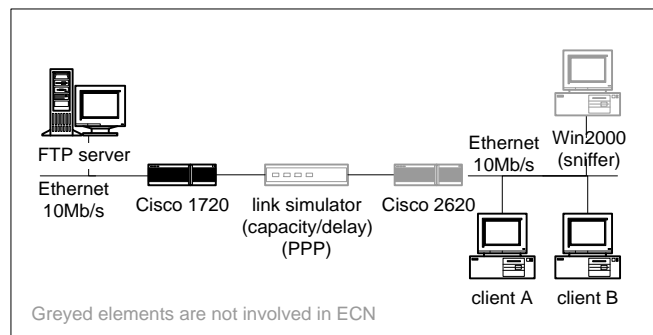


Fig. 3: The testbed

We took another approach than that in [2] (and closer to that in [4]). Instead of measuring low-level parameters (average length of output queue, packet delays or TCP throughput), we concentrated on *service performance* perceived by a user. Thus, we used FTP as an example of a popular Internet service and we measured *FTP goodput*, that is, how fast files were downloaded. This fact has some important consequences: An FTP session uses two TCP connections [10], the first (*ftp*) one is a control connection, which commands are sent over, and the second (*ftp-data*) one is established for file transfer. We

intentionally measured FTP goodput taking into account only bytes transferred over the ftp-data connection, but we took the whole time needed to run the session, since this is how a user sees the performance.

Every FTP client worked in the following way: First, it opened the control connection and issued the following commands: USER, PASS, TYPE, PASV (requires the server to open server socket) and RETR. Then, it opened the ftp-data connection and transferred a file. Finally, after sending QUIT, both connections were closed and the procedure was repeated.

The first task was to assess the gain from using ECN over non-ECN TCP. During the bulk transfer test, both client machines A and B opened two simultaneous FTP sessions. In the first test, only non-ECN TCP was used; in the second test, TCP used ECN. The transactional case was similar except that 10 simultaneous sessions were active on each client machine. We do not present the results here, since we did not notice any serious difference between both schemes (in total goodput of all sources). This may seem somewhat surprising at first, but a reasonable explanation exists: Since the sources were required to react to marked and dropped packets in the same way, probably the gain from not dropping some number of packets in the ECN test was modest. This would suggest that for ECN to show its advantages fully new ways of reaction to congestion signals should be developed. And of course, we do not question results from previously cited works (which show differences at the lower level) – perhaps our scenario was too simple (the same sources, delays etc).

MALICIOUS ECN SOURCES

The subsequent tests were performed to assess the impact of subverted ECN sources to conformant traffic. Thus, a malicious TCP implementation was prepared (Linux kernel was altered appropriately) which pretended ECN-compliant endpoint but ignored congestion information in both directions. In the bulk transfer test, there were two FTP sessions made from every client machine, but this time the sessions were started one after another, with 30-second periods, and the malicious sessions were run as the last ones. Client B kept using subverted ECN implementation while client A used non-ECN TCP (the third test) or ECN-compliant TCP (the fourth test). The exemplary results for link capacity 256 kb/s and round-trip delay 100 ms are presented in Fig. 4, Fig. 5 and Table 1 (the results obtained for other delays were similar).

During the transactional test, 10 compliant sessions were started on client machine A. One minute later, 10 subverted sessions were run on client machine B. As in the bulk test, client A used non-ECN TCP (the fifth test) or ECN-compliant TCP (the sixth test). The results for link capacity 64 kb/s and round-trip delays set to 20 and 100 ms are shown in Table 2 (for the last 4 minutes).

The results are self-commenting. In the bulk test, malicious sessions started later but easily throttled “legal” ones and achieved much better performance. On the other hand, the transactional case showed that short-lived ECN sessions were resistant to malicious traffic and received almost half of the aggregate goodput (but the Non-ECN ones suffered seriously and achieved about 35-40% of the aggregate goodput). However, we admit that although these results show that the threat really exists, one cannot map the results onto the Internet scale in a direct way. That is, a test involving macro-scale traffic and real Internet links would be needed to make the assessment complete.

		Client A		Client B	
Client A Type	Attempt	Session 1	Session 2	Session 1	Session 2
Non-ECN (Test III)	1	8.8	2.8	14.6	10.0
	2	8.1	3.7	14.5	10.0
ECN (Test IV)	1	8.2	2.1	17.5	8.4
	2	8.6	5.7	9.3	12.5

Table 1: The bulk test: average goodput rates [kB/s]

		Delay 20 ms		Delay 100 ms	
Client A Type	Attempt	Client A	Client B	Client A	Client B
Non-ECN (Test V)	1	2.08	3.75	2.49	3.68
	2	2.45	3.72	2.56	3.67
ECN (Test VI)	1	3.06	3.16	3.03	3.13
	2	2.97	3.18	3.05	3.09

Table 2: The transactional test: aggregate goodput rates for all 10 sessions for last 4 minutes [kB/s]

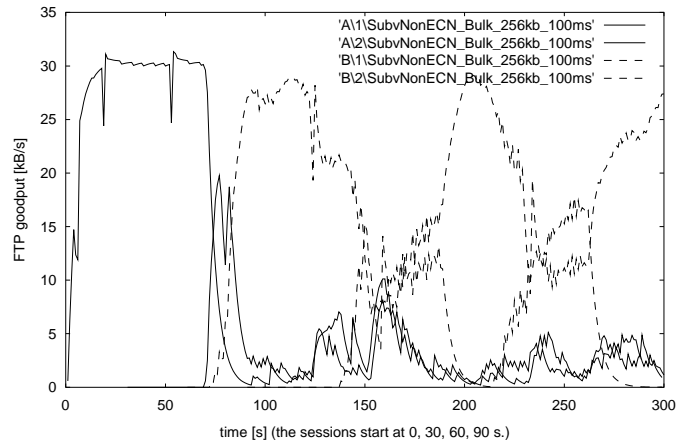


Fig. 4: Test III: Subverted sessions (dashed lines) started later but succeeded to preempt non-ECN ones (solid lines)

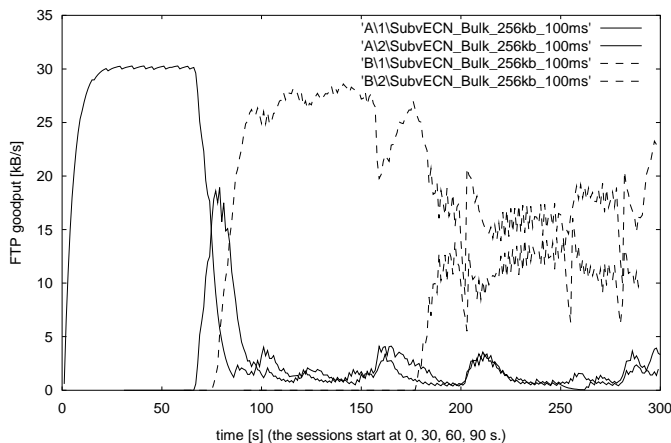


Fig. 5: Test IV: Subverted sessions (dashed lines) also succeeded to preempt ECN sessions (solid lines)

To complete the discussion about the gain from ECN subverting, we make two more remarks. First, the attack is most successful if it does not cause heavy congestion (routers do not drop packets yet) but makes responsive flows back off gradually. Second, RFC-3168 states that ECN will not significantly increase the current vulnerability of the IP architecture to unresponsive flows. We would only like to comment that in our opinion ECN-based attacks are especially “neat,” that is, do not generate additional traffic (like “brute-force” subverting based on packet duplication performed by a router) and there is no danger that the congestion window will be opened too wide (and too much traffic will be injected, like during attacks based on TCP acknowledgments mentioned in [7]). In contrast to non-ECN-based attacks, ECN subverting allows treating malicious macro-flows as higher-priority ones, while responsive flows will be gradually yielding.

THE BEST-EFFORT INTERNET: WILL NETWORK OPERATORS MARK?

This vision dictates that the Internet is a “common good” that is shared by well behaving users, and that fairness is one of the most desired properties (see [9], for example). ECN would help keeping the network in a good (uncongested) shape, so all users should contribute to it.

The problem is that the coordinated macro-scale attack could cause a serious damage to this ideal vision. The sources may use encrypted channels with traffic masking, which further complicates penalty boxes’ tasks – they would have to operate on a macro-flow basis.

SLAs AND MARKING

At present, a shift may be observed from best-effort service towards SLA-based commercial Internet. The competition on the market enforces network operators to assure some level of QoS parameters for their customers

and one can expect this trend to develop. Thus, it is impossible to separate the technology (e.g., congestion avoidance mechanisms or QoS mechanisms), QoS guarantees specified in SLAs and, finally, pricing users for using network resources – all these three so different planes must be considered *together*, with the analysis of their interactions (see [9] for an in-depth discussion about marking in the context of economy). In this section, we comment on the impact of SLAs on marking, so we will describe shortly a typical SLA.

The basic service is defined as *providing IP connectivity*. That is, a network operator commits itself to forwarding IP packets between two edges of its network. This is a very important fact from our point of view: Network operators *do not care* about which transport-level protocols are used or if their implementations are conformant or not (in fact, they may gain if the implementations are conformant, but the choice belongs to the users). They just forward IP traffic; all QoS parameters relate to the IP layer. This is another reason against using penalty boxes: The sites pay for access link and can generate any kind of traffic they find useful.

A typical SLA concerns edge-to-edge QoS parameters, defined in three planes:

- service availability
- packet loss ratio
- average packet delay

They are average parameters, defined over some period. The period is usually long, “safe” from network operator’s point of view, but it is likely that the competition on the market will enforce stronger guarantees in the nearest future. SLAs provide for fines paid by operators when these parameters are violated.

We are interested in packet loss ratio only, although marking could affect packet delays, too. The question is if network operators should mark packets instead of dropping them. We assume that customers use mainly conformant TCP implementations, which back off on packet loss or mark. In such a case, during congestion, all customers (sites) should experience the same deterioration of network’s performance, the same – relatively to traffic they transmit. This makes guaranteed packet loss ratio least likely to be exceeded.

A coordinated macro-scale attack could jeopardize this scenario, and conformant sources could experience especially bad performance. This could even lead to QoS violation and fines paid by operators.

But let us assume that network operators do use ECN for congestion control. Then, a question appears how marked packets should be treated. Should they be treated in the same way as dropped packets? At last, they have been delivered successfully and have used some network

resources. Perhaps they should be treated as normal (unmarked) ones? Obviously not, since users are required to back off and may feel disappointed. So, marked packets should be taken into account in a SLA (e.g., as packet marking ratio). Furthermore, it becomes necessary to control if users are responsive (penalty boxes must be acquired and installed). Moreover, it becomes important *where* packets have been marked (unlike dropped packets, they could be marked prior to entering a given domain) – a need for accounting appears. So, is the commercial Internet ready for marking?

Are there any alternatives? Unfortunately, the topic is complex and in-depth discussion is beyond the scope of the paper. However, we mention a few (at least theoretically) possible alternative ways, just to sketch the directions to go. First, proper network *over-engineering* may be the simplest way to cope with congestion in the backbone. Second, *traffic engineering* (e.g., in the form of MPLS) may help with efficient use of network resources. Third, pricing schemes (e.g., *congestion pricing*) could effectively encourage users to being responsive and replace questionable “trace and penalize” approach. The basic network service, IP forwarding, could be kept, possibly augmented with appropriate QoS mechanisms.

WHERE ECN COULD HELP?

In our opinion, ECN would be most useful in scenarios, when the two following assumptions hold:

- congestion is often present
- the environment is trusted

Both postulates are valid in a LAN environment: Typically, LANs are fast (10-100 Mbps), but the Internet access links are much slower (usually a few Mbps) and may be often congested (most often, they would be congested in the *reverse* direction, from the Internet to the site, so web servers should participate in ECN). This is an ideal setting for ECN to show its advantages. Note that LAN administrators could easily detect and penalize users that subvert ECN.

Refer again to Fig. 2. Site A can use ECN for its outgoing TCP traffic competing for access link’s bandwidth. However, as site D’s Internet traffic flows through site A’s network and its main router, ECN is not sufficient in itself, because both macro flows should be separated. A possibility would be to use a classifier and Weighted Fair Queuing on the site A’s main router to guarantee the separation by assuring access link capacity percentage for both sites.

CONCLUSIONS

In the paper, we aim at discussing ECN in the context of the current Internet. It is *not* our goal to convince that ECN should not be deployed – while one should realize

possible threats, they are not a reason to reject ECN and all the advantages it gives. Despite somewhat controversial title of the paper, we believe that ECN will be finally deployed. It would be especially useful as a congestion avoidance mechanism on the Internet access links. However, this is still questionable if network operators should use ECN, at least in the context of currently used Service Level Agreements, and due to the fact of inherently malicious nature of human beings.

ACKNOWLEDGMENT

The author would like to acknowledge the financial support provided by the Foundation for Polish Science (FNP).

REFERENCES

- [1] S. Floyd, V. Jacobson, Random Early Detection Gateways for Congestion Avoidance, *IEEE/ACM Transactions on Networking*, August 1993
- [2] S. Floyd, TCP and Explicit Congestion Notification Available: http://www.icir.org/floyd/papers/tcp_ecn.4.pdf
- [3] K. Ramakrishnan, S. Floyd, D. Black, The Addition of Explicit Congestion Notification (ECN) to IP, RFC-3168, September 2001
- [4] J. H. Salim, U. Ahmed, Performance Evaluation of Explicit Congestion Notification (ECN) in IP Networks, RFC-2884, July 2000
- [5] N. Spring, D. Ely, Robust ECN Signaling with Nonces, IETF Internet Draft, October 2001
- [6] B. Braden, et al., Recommendations on Queue Management and Congestion Avoidance in the Internet, RFC-2309, April 1998
- [7] S. Savage, N. Cardwell, D. Wetherall, T. Anderson, TCP Congestion Control with a Misbehaving Receiver, *ACM Computer Communication Review*, October 1999
- [8] A. Misra, T. Ott, Jointly Coordinating ECN and TCP for Rapid Adaptations to Varying Bandwidth, *Proceedings of MILCOM*, October 2001
- [9] D. Wishik, How to mark fairly, January 2001 Available: <http://www.statslab.cam.ac.uk/~djw1005/Stats/Research/markings.ps>
- [10] W. Stevens, *TCP/IP Illustrated, Vol. I*, Addison-Wesley 1994
- [11] ECN (Explicit Congestion Notification) in TCP/IP: <http://www.icir.org/floyd/ecn.html>
- [12] ECN-under-Linux Unofficial Vendor Support Page: <http://gtf.org/garzik/ecn/>