Analysis of the State of ECN on the Internet

Chun-Xiang CHEN\(^{\text{a)}\} \text{ and Kenichi NAGAOKA}^{\text{\textsuperscript{†}}, \text{Members}}\)

SUMMARY ECN, as a decisive approach for TCP congestion control, has been proposed for many years. However, its deployment on the Internet is much slower than expected. In this paper, we investigate the state of the deployment of ECN (Explicit Congestion Notification) on the Internet from a different viewpoint. We use the data set of web domains published by Alexa as the hosts to be tested. We negotiate an ECN-Capable and a Not ECN-Capable connections with each host and collect all packets belonging to the connections. By analyzing the header fields of the TCP/IP packets, we dig out the deployment rate, connectivity, variation of round-trip time and time to live between the Not ECN-Capable and ECN-Capable connections as well as the rate of IPv6-Capable web servers. Especially, it is clear that the connectivity is different from the domains (regions on the Internet). We hope that the findings acquired from this study would incentivize ISPs and administrators to enable ECN in their network systems.

key words: Explicit Congestion Notification, active queueing, congestion control, connectivity

1. Introduction

TCP congestion control has also been an open problem so far. In the early Internet, the slow-start with AIMD (Additive Increase/Multiplicative Decrease) worked pretty well since the network bandwidth was not high at that time. With the rapid development of transmission technology, however, it has been approved that TCP slow-start mechanism (so-called TCP Tahoe or TCP Reno) could not make good use of the transmission bandwidth. To address this issue, a lot of modified TCP congestion control schemes have been developed, for example, HighSpeed TCP, Scalable TCP and Fast TCP, etc.\(^{[1]}\), \(^{[2]}\). These schemes go to congestion phase and activate their control mechanisms only after packet-loss happens in the intermediate network. Another approach is based on packet delay such as TCP Vegas, etc.\(^{[3]}\). The advantage of these schemes is that they do not require any changes at the middle points on the transmission path, so they work compatibly with the existing schemes, which have been employed on the Internet. However, these schemes are not able to drastically solve the problem existing in the loss-based or delay-based schemes.

To completely solve the problem, another approach so-called Explicit Congestion Notification (ECN) has been proposed\(^{[4]}\). Unlike the traditional loss-based or delay-based schemes, ECN extends the TCP/IP packet headers and lets the intermediate points on the path of the network indicate their congestion state to the receiver. Then the receiver echoes the state back to the transmitter so that the transmitter can adjust the sending rate before the network gets crowded. Since ECN-Capable routers on the path are able to report the impending congestion before the congestion really happens, ECN is highly expected as a decisive solution to avoid dropping packets. The effectiveness of using ECN has been shown in the previous works\(^{[6]}\), \(^{[7]}\). Nevertheless, other works\(^{[8]}\)–\(^{[11]}\) show that the deployment of ECN on the Internet is much slower than its expectation even though it was standardized as RFC 3168\(^{[4]}\) in 2001. Although ECN has been implemented with TCP/IP in most modern operating systems (see Sect. 6.4), it is almost inactivated at initial values. It seems that the ISPs do not have enough incentive to enable ECN by default setting.

In this paper, we practically investigate the state of art of deployment of ECN on the Internet. Through this investigation, we have managed to answer the following questions which are still unclear in the previous works: (1) What is the current deployment ratio of ECN on the Internet; (2) Is there a marginal (or extra) risk when a host negotiates a connection with ECN-Capable; (3) Are there any benefits for end users to use ECN-Capable TCP connection; (4) Can we provide any convincing data for ISPs and/or end users to incentivize them to enable ECN in their networking devices?

The rest of this paper is organized as follows. Section 2 describes the relative works. Section 3 gives an examined results of ECN in a test bed network. The deployment of ECN is measured in Sect. 4. The analytical results and discussion are illustrated in Sect. 5 and Sect. 6, and concluding remarks are given in Sect. 7.

2. Related Works

2.1 Previous Works

The benefits of ECN could be achieved under the condition that routers on the path between the transmitter and receiver operate on active queueing management (AQM)\(^{[26]}\), where the routers are able to accurately measure the occupancy of the queueing and indicate the status of congestion to the receiver\(^{[22]}\), \(^{[23]}\). The ECN-Capable receiver feeds the congestion status back to the transmitter so that the transmitter

---

Copyright © 2019 The Institute of Electronics, Information and Communication Engineers
can adjust the sending rate before the network becomes congested. This is an ideal scenario. However, it is difficult to make the changeover from the autonomous distributed Internet to a whole ECN-Capable network. As a matter of fact, the prevalence of ECN is much slower than expected.

There are many factors which might obstruct the deployment of ECN. ECN works as well as long as every point along the path is available for supporting ECN. Unfortunately, it is likely that a lot of old-model routers, still working on the Internet, are incapable of dealing with the ECN fields in TCP/IP packet header. In extreme cases, they might scrape away ECN option or discard the packet with ECN option. Also, security gateways or firewalls along the path might block the SYN packet with the ECN option in TCP packet header and make the ECN negotiation unsuccessful [5]. Nevertheless, ECN has been considered as an approach that is able to avoid the congestion as well as to achieve high throughput efficiency in high-bandwidth and/or long-distance networks.

To know the current state of ECN, several academic researches have been done [8]–[12]. [8] examined the ECN-capability about 84,394 web servers on the Internet. The percentage of ECN-capability was 1.1% in 2000 and 2.1% in 2004. [9] also shows that the percentage was about 2.2% in 2004 and 1.1% in 2008. [10] reported the state of ECN and TCP options on the Internet measured by another approach. It shows that 25.16% of web servers were ECN-Capable in April 2012, and 29.48% in August 2012. [11] shows the results measured in 3 different vantage points (in London, New York and Singapore). In [12], the authors measured and analyzed the deployment of ECN in Jan. 2017. It shows that the ratio of ECN-capability was around 68.11%. However, the number of the web servers used to be tested is relative small (50000 web servers), and the measurement was performed at one point.

It is clear from the earlier works that the deployment of ECN on the Internet is gradually increasing. However, measuring the ECN-capability is a hard work. It depends on the point of measurement as well as the distribution of the hosts to be measured. To catch the deployment of ECN in its entirety, measurement over a wide area is necessary [11]. Based on the authors’ knowledge, there is no similar work done at the point in Japan. Our work is a supplement to this issue. It will reveal various different results from the previous works.

2.2 Summary of the Addition of ECN to TCP/IP

Different from the traditional loss-based or delay-based schemes, ECN requires the support of the routers along the path as well as the endpoints. To achieve this objective, ECN uses two right-most bits of the DiffServ field in IP packet header to express four codepoints. The combinations mean: 00=Non ECN-Capable Transport; 01=ECN Capable Transport (ECT), ECT(0); 10=ECN Capable Transport (ETC), ECT(1) and 11=Congestion Encountered, CE. Senders are free to use either ECT(0) or ECT(1) codepoint to indicate ECT. ECN-Capable routers treat ECT(0) and ECT(1) codepoints as equivalent [4]. If a router supporting the ECN along the path forwards the data packet with codepoint ECT(0) (or ECT(1)), it is allowed to change the codepoint from ECT(0) (or ECT(1)) to CE instead of dropping the packet according to the occupation of its active queuing.

In transport layer, ECN is implemented as an option in TCP packet header. Two bits in the reserved field of TCP header are assigned to ECN. The first one is ECN-Echo flag (ECE), and the other one is congestion Window Reduced (CWR) flag. When a TCP source negotiates an ECN-Capable connection, it sets the ECE and CWR bits as well as the SYN bit (ECN-setup SYN packet [4]) to indicate that the sender expects an ECN-Capable connection. As reply to the ECN-setup SYN packet, the TCP receiver replies by setting the ACK and ECE bits if ECN is available. When an ECN-Capable connection has been established, the TCP sender sets an ECT codepoint in the IP packet header of data packet to express that ECN is capable. Thus the ECN-Capable routers along the path could mark the packet with CE codepoint to indicate an imminent congestion to the TCP receiver. The receiver echoes back this congestion notification by marking all acknowledgment packets with the ECE flag. When the TCP sender receives the acknowledgment packet with ECE flag, it can appropriately adjust its sending rate to avoid the occurrence of congestion.

3. Verifying the Effectiveness of ECN

Performance of ECN and its effectiveness was evaluated by computer simulation [6] and by testbed [7] as it was proposed. However, as hardware and software have greatly changed in network environments in the last two decades, the effectiveness of ECN should be verified again at a testbed close to the current Internet.

In this section, we build a concrete network (testbed) with delay emulation to verify the effectiveness of ECN by actually measuring the throughput and by observing the variation of the congestion window size of the TCP connection.

The topology of the network for test is shown in Fig. 1. The specifications of the equipments are shown in Table 1. The kernel version was updated with 4.1.35.

To observe the behavior of ECN, the features of the testing environment (Fig. 1) are adjusted as follows. PC<sub>C</sub>, PC<sub>S</sub> and hosts in side networks are customized to be available to support ECN. To build up a bottleneck link, the bandwidth of the interface (eth<sub>1</sub>) of ECS is limited to 100Mbps.
Table 1 Specifications

| Description     |
|-----------------|
| SHUB            | Gigabit Ethernet switching hub. |
| DE              | Delay emulator, NetEm (Linux kernel 4.1.35). |
| N1, N2          | Side networks, ECN-Capable. |
| ECS             | ECN-Capable switch. CentOS 6.8, Linux kernel 4.1.35 with mptcp, Cubic, 8GB |
| PC, PC          | ECN-Capable hosts. CentOS 6.8, Linux kernel 4.1.35 with mptcp, Cubic, 8GB |

The bandwidth of other links is 1Gbps. The delay emulator (Linux NetEm) is configured as a switch and is used to introduce transmission delay (100ms round trip time, 50ms at each way). In order to support ECN at ECS, the queue management algorithms in both interfaces eth1 and eth0 are changed from the default pfifo_fast to fq_codel, an active queue management algorithm. The hard limit on the real queue size of fq_codel in eth1 is shortened to 3750 packets. The Linux kernel of ECS is configured to be ECN-Capable (by setting net.ipv4.tcp_ecn = 1).

After we tune the various parameters of the experimental network, eth1 will firstly get congested when traffic load becomes heavy.

The effectiveness of ECN is verified by comparing the total network throughput and the number of packets dropped (or marked by ECN) in ECS. Throughput between PC and PC is measured in one minute with iperf, which is a tool to perform network throughput for either TCP or UDP throughput [18]. In our work, iperf is used in TCP mode.

Based on the measurements in the cases that ECN is not capable and capable, it is shown that the throughput of a TCP connection between PC and PC is about 85 Mbps and that 10 packets are dropped when ECN is not capable at ECS (Fig. 2) (there is no traffic within side networks at this moment). On the other hand, the TCP throughput is about 87 Mbps and no packets are dropped, and 14 ecn_marks happen instead (Fig. 3). Note that “dropped 10” in Fig. 3 is not the number of dropped packets when ECN is capable. This parameter inherits the number from the previous testing in Fig. 2 since the measurements are done consecutively. Figure 3 illustrates the accumulated results. It is obvious that ECN marks the packets to indicate an imminent congestion before the congestion really occurs.

The changes of TCP congestion window size in Linux cubic at host PC are illustrated in Fig. 4. It is clear that the TCP sender deals with ecn_mark feedback in the same way as it does when the packet loss happens.

We have repeatedly investigated the effectiveness of ECN in various cases (by running several iperf connections among PC, PC and side networks). It is shown that the total throughput efficiency of the network (the sum of the throughput of all iperf connections) is improved when ECN is capable. However, the throughput efficiency for a particular ECN-Capable TCP (a single iperf) connection is not always improved if it is unfortunately “ecn_marked” by the ECN-Capable switch. Effectiveness of using ECN can be found in [6], [7], [27], and the detailed figures and discussion are omitted here due to space limitations.

4. Data Sets and Methodology of Measurement

To measure and analyze the features concerned with ECN, in this section, we first give the data sets which include the domain entries of web hosts to be tested. During the translation from domain entries to IPv4 web hosts and IPv6 web hosts, we incidentally analyze the ratio of IPv6 capable web hosts on the Internet. The methodology of measurement for ECN is described in 4.2.

4.1 Data Sets of Web Servers

The data set of the targeted web servers should be carefully chosen since the distribution of these servers might strongly affect the results of investigation. We select two types of data set to measure the readiness of ECN.

The first data set includes one million domains of Web servers on the Internet provided by Alexa [16] (called ALL1M set hereafter). The list of domains is downloaded at each time when the measurement is done. This set is used to reveal the state of ECN at the major web servers over the whole Internet.
The second data set (called PUH set) is collected from the access logs of the users at the author’s university within one month.

First, we translate every entry in the data set to fully qualified domain name (FQDN) of the corresponding entry by adding the most popular prefix www to the head of every domain. For example, if the domain entry is google.co.jp, it is translated to www.google.co.jp. Then the IP address of the FQDN is looked up. Thus we get the FQDN entry corresponding to its IPv4 address. Meanwhile we obtain its IPv6 address if it has. For instance, the entry of domain google.co.jp is translated as follow: www.google.co.jp, \, \
172.217.161.35, 2404:6800:4004:80a::2003
It is likely that some hosts might have multiple FQDN that corresponds to the same IP address. We rule out such duplicate FQDN from the entries and exclude the invalid hosts. Here, the “invalid hosts” means that their FQDN point to a private IP addresses. In the end, we obtain the unique entries (say Unique Entry).

To show the deployment progress of ECN on the Internet, we made the measurements three times on Mar. 22, 2017, Nov. 2, 2017 and Jun. 15, 2018, respectively. Table 2 numerically shows the ratios of IPv6 and IPv4 hosts corresponding to the whole domain entries in Mar. 2017, where the percentage of IPv6 or IPv4 hosts is calculated by

\[
\text{IPv6\%} = \left( \frac{\text{IPv6 hosts}}{\text{Unique Entries}} \right) \times 100\% , \quad \text{IPv4\%} = \left( \frac{\text{IPv4 hosts}}{\text{Unique Entries}} \right) \times 100\%
\]

The number of IPv4 hosts in ALL1M is 13 fewer than that of the unique entries because there are 13 servers which only have IPv6 address. However, it seems that these web-like hosts do not really serve as web servers at this time. Furthermore, to clarify the dependence of IPv6 hosts on the top domains, some major top domains drawn from ALL1M have also been focused on. For instance, the top domains .com .net .jp .cn etc., are shown in Table 2.

| Table 2 | IPv4 and IPv6 hosts in data sets (Mar. 21, 2017) |
|---------|-----------------------------------------------|
| Unique Entries | IPv4 Hosts | IPv6 Hosts | IPv4 % | IPv6 % | IPv6 only hosts |
| ALL1M | 604393 | 604380 | 54970 | 100% | 9.10% | 13 |
| .com | 286669 | 286665 | 27277 | 100% | 9.52% | 4 |
| .net | 30925 | 30924 | 3712 | 100% | 12.00% | 1 |
| ... | ... | ... | ... | ... | ... | ... |
| .jp | 17644 | 17642 | 334 | 100% | 1.89% | 2 |
| .cn | 9269 | 9269 | 79 | 100% | 0.85% | 0 |
| PUH | 27921 | 27921 | 1446 | 100% | 5.18% | 0 |

*Rounded to decimal places

![Fig. 5](image)

4.2 Methodology of Measurement

The measurements in our work were performed with a Linux host in our lab. OS of this host is CentOS 6.8 with the customized Linux kernel 4.1.35 supporting ECN (say testing host).

To measure the connectivity and other features concerning with ECN, we partially use the tool called ecn_spider provided at [15]. The procedures at the testing host are summarized as follows. Note that tcpdump must be run for capturing the packets before starting the following procedures.

1. Disable ECN by setting `net.ipv4.tcpecn=0`.
2. Pick up next entry (server A) from the unique entries of the data set; Negotiate an http connection (TCP port 80) with server A by setting the flag SYN (SYN only). If the negotiation succeed, issue a request to load top page from server A; Close the connection.
3. Enable ECN by setting `net.ipv4.tcpecn=1`.
4. Negotiate an http connection (TCP port 80) with server A by setting the flags (SYN, ECN, CWR). If the negotiation succeed, issue a request to load top page from server A; Close the connection.
Table 3 Deployment of ECN (Mar. 22, 2017)

| Symbols | ALL1M          | JP          | CN          | PUH          |
|---------|---------------|-------------|-------------|--------------|
|         | IPv4 | IPv6 | IPv4 | IPv6 | IPv4 | IPv6 | IPv4 | IPv6 |
| $T_{B0}$ | 592909 | 52341 | 17360 | 299 | 8921 | 56 | 27718 | 1170 |
| $N_{ECN}$ | 442438 | 49261 | 9078 | 176 | 4372 | 41 | 13691 | 894 |
| $N_{NECN}$ | 150471 | 3080 | 8282 | 123 | 3549 | 15 | 14027 | 275 |
| $N_{NR}$ | 102467 | 2629 | 282 | 35 | 348 | 23 | 203 | 47 |
| $\beta_{ECN}$ | 74.62% | 94.12% | 52.29% | 58.86% | 49.0% | 73.21% | 49.39% | 76.41% |
| $\beta_{NECN}$ | 25.38% | 5.88% | 47.71% | 41.14% | 51.0% | 26.79% | 51.61% | 23.59% |
| $\beta_{NR}$ | 1.90% | 4.78% | 1.60% | 10.48% | 3.75% | 29.11% | 0.73% | 19.09% |

5. Repeat the above procedures until all web servers in unique entries are visited.

6. Terminate tcpdump after all the hosts in the domain entries are visited.

The next work we should do is to meticulously analyze the packets captured with tcpdump.

5. Analysis and Results

In this section, we show the analytical results by analyzing the packets of each connection captured by tcpdump. To analyze the whole behavior of ECN implemented in IP and TCP packet headers, we first extend the packets to whole plain text, then analyze them with the tools we developed.

5.1 Ratio of ECN Deployment

Table 3 numerically shows several features concerned with ECN connections for ALL1M and PUH and domains .JP .CN, where $T_{B0}$ is the number of web servers, which reply to the negotiation for establishing connection, and $N_{ECN}$ ($N_{NECN}$) is the number of web servers which are ECN-Capable (Not ECN-Capable). $N_{NR}$ is the number of web servers which won’t reply to the negotiations no matter whether the connection is negotiated with ECN-Capable or without ECN-Capable. $\beta_{ECN}$ and $\beta_{NECN}$ are the ratios of them corresponding to $T_{B0}$. $\beta_{NR}$ is the ratio of $N_{NR}$ to the total hosts. They are calculated by the following formulas.

$\beta_{ECN} = \frac{N_{ECN}}{T_{B0}} \times 100\%$

$\beta_{NECN} = \frac{N_{NECN}}{T_{B0}} \times 100\%$

$\beta_{NR} = \frac{N_{NR}}{T_{B0} + N_{NR}} \times 100\%$

The breakdowns of ECN-Capable ratios by domains are illustrated in Fig. 6 and Fig. 7 for IPv4 and IPv6 web hosts, respectively.

From Table 3 and Figs. 6 and 7, (1) the deployment of ECN on the Internet is steadily increasing for IPv4 web hosts as well as IPv6 web hosts; (2) the deployment ratio of IPv6 web host is much higher that that of IPv4 web hosts; (3) Compared with other domains, $\beta_{ECN}$ of domains .cn, .jp and .tw are quite low (These domains are distributed in Asia region).

5.2 Connectivity of ECN-Capable

Table 4 numerically shows the connectivity of connections negotiated with ECN-Capable, where $N_{11}$ is the number of the web servers, which definitely reply to the negotiation with ECN-setup SYN packet. They reply to the negotiation with ECN-setup SYN-ACK packet if they are also ECN-Capable, otherwise, acknowledge in a normal way, i.e. reply with non-ECN-setup SYN-ACK packet. $N_{01}$ is the number of the web servers which do reply to the negotiation with Not ECN-Capable initialization but do not reply to the negotiation with ECN-Capable initialization. $N_{00}$ is the opposite case, i.e. the number of web servers which do not reply to the negotiation with Not ECN-Capable initialization but do reply to the negotiation with ECN-Capable initialization. $\beta_{11}$ represents its percentage, which is calculated with $\beta_{11} = \frac{N_{11}}{T_{B0} + N_{11}} \times 100\%$. The breakdowns of $\beta_{11}$ by domains are shown in Fig. 8 and Fig. 9 for IPv4 and IPv6 web hosts, respectively. All the connectivity $\beta_{11}$ of domains .tw kr and .cn tend to be growing, and yet compared with other domains, they are still lower.

In Table 4, we should take notice of $N_{10}$ and its rate $\beta_{10}$ ($= \frac{N_{10}}{T_{B0}} \times 100\%$). The web servers with the ratio of $\beta_{10}$ do not reply to the ECN-Capable negotiation while
Table 4  Connectivity of ECN (Mar. 22, 2017)

| Symbols | ALL1M | JP | CN | PUH |
|---------|-------|----|----|-----|
|         | IPv4  | IPv6 | IPv4 | IPv6 | IPv4  | IPv6 | IPv4  | IPv6 |
| $N_{11}$ | 591139 | 52301 | 17330 | 298 | 8547  | 56 | 27633 | 1170 |
| $N_{10}$ | 1365  | 18  | 30  | 0  | 305   | 0 | 68   | 0   |
| $N_{01}$ | 405   | 22  | 0   | 1  | 69    | 0 | 17   | 0   |

| $\beta_{11}$ | 99.54% | 99.93% | 99.65% | 100.0% | 93.34% | 100.0% | 99.51% | 100.0% |
| $\beta_{10}$ | 0.23%  | 0.03%  | 0.17%  | 0.00%  | 3.33%  | 0.00%  | 0.24%  | 0.00%  |
| $\beta_{01}$ | 0.07%  | 0.04%  | 0.0%   | 0.34%  | 0.75%  | 0.0%   | 0.06%  | 0.0%   |

Fig. 8  Connectivity of ECN-capable hosts, IPv4

Fig. 9  Connectivity of ECN-capable hosts, IPv6

they do to non-ECN-Capable negotiation. $\beta_{10}$ can be considered as a sort of marginal risk. Here, “marginal risk” is defined as the risk which is not so serious ($\beta_{10}$ is quite low), but the users might as well be aware of it when they utilize ECN-Capable option with TCP at present. On the other hand, out of our expectations, only a very few servers ($N_{01}$) do not reply to non-ECN-setup SYN negotiation but reply to ECN-setup SYN negotiation. $\beta_{01}$ is definitely much less than $\beta_{10}$ for IPv4 hosts. It is explainable that $\beta_{10}$ occurs if one of the routers or security gateways on the path rejects the ECN-Capable negotiation. $\beta_{01}$ occurs very occasionally if the ECN-Capable connection and non-ECN-Capable connection pass through different paths, and the path for the latter is unreachable. $\beta_{10}$ broken down by domains are illustrated in Figs. 10 (IPv4 web hosts) and 11 (IPv6 web hosts) in detail. The same phenomena have been found that $\beta_{10}$ with domains cn kr and tw are relatively high. Especially, $\beta_{10}$ of domain cn is remarkably high for both cases of IPv4 and IPv6.

5.3 Variations of RTT and TTL

To clarify the influence of utilizing ECN option with TCP, we further investigated the variation of round trip
time (RTT) between the connections negotiated with ECN-Capable and Not ECN-Capable from the same web server. We only show the results for all of the domains (ALL1M) as a suitable sample. Let $D_{ECN}$ and $D_{NECN}$ be the RTT of the connections negotiated with ECN-Capable and Not ECN-Capable from the same web server, respectively. The average of $D_{NECN}$ ($D_{ECN}$) over all connections is around 219.79ms (227.34ms). The difference between $D_{ECN}$ and $D_{NECN}$ is calculated by $D_{\text{diff}} = D_{ECN} - D_{NECN}$. The number of $D_{\text{diff}} < 0$, $D_{\text{diff}} = 0$ and $D_{\text{diff}} > 0$ over all 643440 connections are 218714, 204526, and 241699, respectively. The average of $D_{\text{diff}}$ is about 7.55ms. Figure 12 illustrates the variations ($D_{\text{diff}}$) of all 643440 connections. The horizontal axis is the connection number in ascending order of $D_{\text{diff}}$. It is clear that the RTT of about 66.0% connections has increased when they negociate an ECN-Capable connection.

We also investigated the variation of time-to-live between the connection negotiated with ECN and the connection negotiated without ECN. Here, the “hop limit” of IPv6 is counted as “time to live” for simplicity. Let $T_{ECN}$ and $T_{NECN}$ be the values of TTL of the ECN-Capable connection and Not ECN-Capable connection, respectively.

Figure 13 shows the statistical distributions of TTL of IPv4 connections corresponding to ECN-Capable and to Not ECN-Capable. The vertical axis is the ratio of TTL, which is calculated by the number of the connections with the same TTL divided by the total number of the connections. The total number of the ECN-Capable and Not ECN-Capable connections are 442438 and 150471 (shown in Table 3), respectively. It is clear that the TTL’s distribution of ECN-Capable and Not ECN-Capable connections are quite different. The TTL’s distribution of ECN-Capable connections is concentrated in the range 35~60 while that of Not ECN-Capable connections has three peaks in the whole range. This difference probably originates from the operating systems of the web servers [24]. It is supposed that most ECN-Capable web servers are operating in OS Linux.

To show the change of TTL between ECN-Capable and Not ECN-Capable connections, let $T_{\text{diff}}$ be the difference between $T_{ECN}$ and $T_{NECN}$. $T_{\text{diff}} = T_{ECN} - T_{NECN}$. Figure 14 illustrates the distribution of the number of the connections which have the same $T_{\text{diff}}$ versus the difference of TTL for IPv4 connections. The accumulated number of $T_{\text{diff}} < 0$, $T_{\text{diff}} = 0$ and $T_{\text{diff}} > 0$ are 25853 (5.84%), 391151 (88.41%) and 25434 (5.75%), respectively. Furthermore, there are some cases in which the differences are extremely high. The number of $T_{\text{diff}} < -50$ and $T_{\text{diff}} > 50$ are 526 and 35, respectively.

The minimum and maximum of $T_{\text{diff}}$ are $-196$ and 196, respectively. We did not precisely determine the causes why the differences of those connections are so large, but it is thinkable that requests of ECN-Capable connection and Not ECN-Capable connection probably are branched into different concrete servers by the load balancer being installed at the front of the servers. It is likely that these servers are running in different operating systems.

Table 5 shows the differences of TTLs for all connections, where the total number is the sum of IPv4 and IPv6 connections. Obviously, TTLs of 90.07% connections have no changes between the ECN-Capable and Not ECN-Capable connections, and TTLs of 5.03% connections become lower and 4.90% connections larger.

To investigate the influence of the measuring points, measurements based on the same domain entries were concurrently made on June 15 2018 at each author’s college (PUH and National Institute of Technology, Ishikawa College (Ishikawa-NCT)). OS of the computers that we used is CentOS 6.9 with kernel-2.6.32-696.30.1. By analyzing the captured data packets at both sites, We found that there are almost no differences between the connectivity of ECN-capable hosts ($\beta_{11}$) and between the deployment rate of ECN ($\beta_{ECN}$). They are shown in the previous figures.

The difference of $\beta_{10}$ between IPv4 web hosts and IPv6

![Fig. 14 Difference of TTL with IPv4 web hosts (Mar. 22, 2017)](image)

| Table 5 | Difference of time to live |
|--------|---------------------------|
|        | Total No. of $L_{11}$ | $T_{\text{diff}} < 0$ | $T_{\text{diff}} > 0$ | $T_{\text{diff}} = 0$ |
| ALL1M  | 643440                   | 32390, 5.03%         | 31491, 4.90%         | 579559, 90.07%         |
web hosts is shown in Fig. 15 and Fig. 16 for the top domains, where blue bars and orange bars show the results obtained from the measurement at Ishikawa-NCT and at PUH, respectively. $\beta_{10}$ for IPv4 web hosts (Fig. 15) are almost the same. However, $\beta_{10}$ for IPv6 web hosts at Ishikawa-NCT are a little bit larger than those at PUH. This difference is thought to be due to the different topology of IPv6 network at both sides. The IPv6 network at PUH is connected to IPv6 service of Hurricane Electric by tunneling while IPv6 network at Ishikawa-NCT is connected to IPv6 service of National Institute of Informatics within dual stack [25].

Furthermore, similar to the results of the previous measurements, $\beta_{10}$ for domains cn, tw, kr are also relatively large.

6. Discussion

In this section, we discuss several attributes and findings acquired from above analysis in regards to ECN.

6.1 Deployment Ratio of ECN

According to the results measured on Mar. 22, 2017, Nov. 3, 2017 and on Jun. 15, 2018, it is obvious that the deployment ratio of ECN over the whole Internet is gradually increasing with time. The average ratio for IPv4 web servers change from 74.62% to 78.20% to 81.60% (Fig. 6, All1M) and from 94.1% to 94.9% to 92.46% for IPv6 web servers (Fig. 7, All1M). As of June 2018, it exceeded 81%. Therefore, in order to totally take the benefits of ECN, it is desirable for ISPs or network administrators to activate ECN in their network devices.

6.2 Marginal Risk

The connectivity of ECN-Capable connections is more than 99%, and $\beta_{10}$, as a marginal risk of an ECN-Capable connection, is almost less than 0.5%, which means that clients might fail to establish the connection at the rate of 0.005 when they initiate ECN-Capable connection. Considering that the route on the Internet may change frequently, we think that the marginal risk is negligible.

As far as the difference of RTT ($D_{\text{diff}}$) is concerned, RTT of about 66.0% ECN-Capable has increased. The cause of the change can be considered to be the additional time of processing ECN fields in TCP/IP packet header on the path. Compared to the average delay of $D_{\text{NECN}}$ (219.79ms), the average of $D_{\text{diff}}$ is around 7.55ms. Since somewhat of increase of RTT has no influence on the connectivity of ECN, we think that a little increase of RTT is acceptable.

6.3 Benefits for End Users

As the results empirically measured in Sect. 3, the total throughput efficiency of the network has been improved by using ECN with TCP/IP. ECN has been designed for avoiding the imminent congestion in the intermediate points, and consequently contributes to every connection the end users use as long as ECN prevails on the Internet.

6.4 State of ECN in Major Operating Systems

Deployment ratio of ECN at server side on the Internet was clarified and illustrated in the previous sections. It is also necessary to comprehend the state of ECN on the major operating systems. Table 6 shows the implementation of ECN in the current major operating systems, where “Passive” means that ECN is activated for inbound ECN-Capable connection but inactivated for outbound initial connection. Although it is true that the major OS vendors are aware of ECN and have implemented it in their products, ECN is almost inactivated by default settings. They just seem to leave the professional work (i.e. activate ECN) to end users, unfortunately, most of whom are not skilled at networking. By

| OS and Version       | ECN-Capable | Default setting |
|----------------------|-------------|-----------------|
| Windows Server 2008  | Yes         | Off             |
| Windows Server 2012, and higher | Yes | On              |
| Windows Vista 7, and higher | Yes | Off             |
| Linux, kernel 2.4.20, and higher | Yes | Passive        |
| FreeBSD 8.0, and higher | Yes | Off             |
| NetBSD 4.0, and higher | Yes | Off             |
| Mac OS X 10.5, 10.6  | Yes         | Off             |
| Mac OS X 10.11, and higher | Yes | On              |
| Apple iOS 9, and higher | Yes | On              |
| Solaris 11, and higher | Yes | Passive        |

By
Considering the deployment ratio, marginal risk and benefits of ECN, we highly recommend that ISPs and developers of operating systems should ship their products with ECN enabled by default settings.

7. Conclusion

In this paper, we investigated the deployment of ECN on the Internet by measuring and analyzing the status of the connections with ECN-Capable and Not ECN-Capable. Based on the consideration that the measurement would be influenced by mediate routers and end hosts, we used the data set published by Alexa and the data set from access logs at the author’s college as the targeted web servers to be tested.

To clarify the deployment gap corresponding to different domains, we analyzed the deployment rate within some major domains. We found that the rates of deployment are a bit different among the domains. The deployments vary with the different domains (Figs. 6 and 7). Furthermore, compared with other domains, deployment rates for domains jp, tw, cn and kr are relatively low.

Finally, we compared the variations of RTT and TTL between ECN-Capable and Not ECN-Capable connections. The RTTs of about two thirds of ECN-Capable connections become longer.

Based on the above analysis and discussion, we hope that the findings will incentivize ISPs, developers of the operating systems and vendors of networking equipments to ship their products (or configure their network systems) with ECN enabled by initial settings.

In this paper, we analyzed the features concerning ECN, such as the deployment ratio, variations of RTT (TTL) between ECN-Capable and Not ECN-Capable connections and etc. However, it is not clear yet what are the causes which lead to the difference. We will take these as future issues.

Acknowledgments

This work was partially supported by ISPS KAKENHI, Grant-in-Aid for Scientific Research (16K02886). The authors would like to thank the editor and anonymous reviewers for their helpful comments and suggestions, which improved this paper.

References

[1] A. Afanasyev, N. Tilley, P. Reiner, and L. Kleinrock, “Host-to-Host Congestion Control for TCP,” IEEE Communication Surveys and Tutorial, vol.12, no.3, pp.304–342, 2010.
[2] G. Hasegawa and M. Murata, “Research Trends on TCP Congestion Control Mechanisms,” IEICE Trans. Commun., vol.J94-B, no.5, pp.663–672, 2011.
[3] L.S. Brakmo and L.L. Peterson, “TCP Vegas: End to End Congestion Avoidance on a Global Internet,” IEEE J. Sel. Areas Commun., vol.13, no.8, pp.1465–1480, Oct. 1995.
[4] K. Ramakrishnan, S. Floyd, and D. Black, “The Addition of Explicit Congestion Notification (ECN) to IP,” RFC 3168, Sept. 2001.
[5] S. Floyd, “Inappropriate TCP Resets Considered Harmful,” RFC 3360, 2002.
[6] S. Floyd, “TCP and Explicit Congestion Notification,” ACM Computer Communication Review, vol.24, no.5, pp.8–23, Oct. 1994.
[7] J. Hadi Salim and U. Ahmed, “Performance Evaluation of Explicit Congestion Notification (ECN) in IP Networks,” RFC 2884, July 2000.
[8] A. Medina, M. Allman, and S. Floyd, “Measuring Interactions Between Transport Protocols and Middleboxes,” Internet Measurement Conference 2004.
[9] S. Bauer, R. Beverly, and A. Berger, “Measuring the State of ECN Readiness in Servers, Clients, and Routers,” Proc. 11th ACM SIGCOMM/USENIX Internet Measurement Conference (IMC 2011), pp.171–180, Berlin, DE, Nov. 2011.
[10] M. Küehlewind, S. Neumer, and B. Trammell, “On the State of ECN and TCP Options on the Internet,” Proc. 14th International Conference on Passive and Active Measurement, vol.7799, pp.135–144, 2013.
[11] B. Trammell, M. Küehlewind, D. Boppart, I. Learmonth, G. Fairhurst, and R. Scheffenegger, “Enabling Internet-Wide Deployment of Explicit Congestion Notification,” Proc. 16th International Conference, PAM 2015, New York, NY, USA, vol.8995, pp.193–205, March 2015.
[12] C.-X. Chen, “Measuring the State of ECN on the Internet,” Proc. IEEE ICTC2017, pp.173–178, Jeju, Korea, Oct. 2017.
[13] M. Küehlewind, R. Scheffenegger, and B. Briscoe, “Problem Statement and Requirements for a More Accurate ECN Feedback,” IETF; March 9, 2015.
[14] B. Briscoe, M. Küehlewind, and R. Scheffenegger, “More Accurate ECN Feedback in TCP,” IETF; March 5, 2018.
[15] https://github.com/britram/ecnsniper. (Accessed Nov. 19, 2018)
[16] http://s3.amazonaws.com/alexa-static/top-1m.csv.zip.
[17] R. Lübke, P. Büschel, D. Schuster, and A. Schill, “Measuring Accuracy and Performance of Network Emulators,” Proc. IEEE International conference, BlackSeaCom, pp.63–65, May 2014 (DOI: 10.1109/BlackSeaCom.2014.6849005).
[18] Iperf, available at https://sourceforge.net/projects/iperf/.
[19] “List of Internet top-level domains,” https://en.wikipedia.org/wiki/List_of_Internet_top-level_domains.
[20] “https://www.INTERNETSOCIETY.ORG/Resources/doc/2017/state-of-ipv6-deployment-2017/”
[21] “https://www.nic.ad.jp/ipv6/ftp6/8/data."
[22] H. Siritena, A. Haider, and V. Sreeram, “Control-Theoretic Design of RED AQM for Scalable TCP in High Speed Networks,” Proc. ICC-2006, vol.1, pp.177–182, 2006.
[23] A. Boudi and M. Loudini, “ERN: Explicit rate notification AQM for TCP-IP networks,” Proc. 2016 IEEE 13th International Conference on Networking, Sensing, and Control (ICNSC), pp.1–6, 2016.
[24] https://subinsb.com/default-device-ttl-values/
[25] T. Fujino, S. Abe, et al., “Development of IPv6 registry system for SINET,” NII Journal no.7, Sept. 2003.
[26] F. Baker and G. Fairhurst, “IETF Recommendations Regarding Active Queue Management,” RFC 7567, July 2015.
[27] G. Fairhurst and M. Welzl, “The Benefits of Using Explicit Congestion Notification (ECN),” RFC 8087, March 2017.
Chun-Xiang Chen received B.E. degree from Sichuan University, Sichuan, China in 1986, M.E. and Ph.D. degrees from Kanazawa University and Osaka University, Japan, in 1991, 1994, respectively. From 1994 to 1996, he was an assistant professor in the Department of Computer Engineering at Hiroshima City University, Japan. From 1997 to 2005, he was an Associate Professor at Hiroshima Prefectural University. From 2006 to 2007, he was an Associate Professor at Prefectural University of Hiroshima. Since April 2008, he has been a Professor at Prefectural University of Hiroshima, Hiroshima, Japan. His current research interests include performance evaluation of communication networks, multiple-traffic control on multiple-channel systems and QoS on the Internet. He is a member of the IPS, IEICE of Japan and the IEEE Communications society.

Kenichi Nagaoka received his B.E. and M.E. degrees from Kanazawa University in 1990 and 1992, respectively. From 1992 to 2001, he was an assistant professor and from 2001 to 2013, was an instructor in the department of electronics and information engineering at National Institute of Technology, Ishikawa College. Since 2014, he has been an associate professor at National Institute of Technology, Ishikawa College. His current research interests include performance evaluation of communication networks, QoS on the Internet and educational technology. He is a member of IEICE of Japan and IEEE.