# Comparative study of RED, ECN and TCP Rate Control

Prasad Bagal, Shivkumar Kalyanaraman, Bob Packer [1]
Department of ECSE, Rensselaer Polytechnic Institute,
Troy NY 12180-3590
Ph: 518-276-8979; FAX: 518-276-2433
Email: bagal@cs.rpi.edu, shivkuma@ecse.rpi.edu, bob@packeteer.com

**Abstract**

TCP congestion control [9] is designed for network stability, robustness and opportunistic use of network buffer and bandwidth resources on an end-to-end per-connection basis. Upon detecting packet loss, TCP infers congestion and trades off per-user goodput for network stability. Specifically, TCP throughput can be approximated by a function which is inversely proportional to the round trip time, the timeout delays and the square root of loss probability [16].

While the use of packet loss as an indicator of congestion is a robust technique, packet loss itself has a profound effect on performance – especially in terms of the variance in goodput seen by individual connections. This "fairness" problem also results in what is commonly known as the "World Wide Wait" experienced by a majority of interactive Internet applications such as WWW or ftp. Another auxiliary problem in TCP congestion control is the lack of control over bottleneck queueing delay due to the end-to-end nature of control.

In this paper, we evaluate three proposed solutions for these problems - an improved drop scheme (RED), a bit-based explicit congestion notification scheme (ECN) and a scheme which explicitly and transparently controls TCP rate (Packeteer TCP rate control). Our studies indicate marked improvements in fairness as we move from RED through ECN to TCP rate control. All schemes control bottleneck queueing delay, but trade off other measures such as drop rate, utilization and fairness, with TCP rate control exhibiting the best performance in terms of all metrics. In terms of deployment flexibility, TCP rate control and RED allow widespread and immediate deployment because they are transparent to hosts (ECN is not because it requires TCP protocol modifications). The minimal state requirements and protocol transparency of RED allows it a large deployment space.

## 1 Introduction

TCP congestion control is designed for network stability, robustness and opportunistic use of network buffer and bandwidth resources on an end-to-end per-connection basis [9]. Using a robust technique to detect packet loss (timeout or triple-duplicate acks [21]), TCP infers congestion and trades off per-user goodput for network stability. Specifically, TCP throughput is known to be a function which is inversely proportional to the round trip time, the timeout delays and the square root of loss probability [16].

Besides stability and robustness, one could apply the following generic evaluation criteria to evaluate congestion control schemes. First, network operators evaluate performance by looking at a balance

[1] Bob Packer is with Packeteer, Inc.

between high utilization of bottleneck links, low average queue length and low packet drop rate. Second, users of the network evaluate performance (assuming infinite flows) based upon per-flow goodput as seen by the application (approximated by the combination of average and standard deviation of goodput). In particular, users desire average goodput to be as high as possible, and the standard deviation in goodput close to zero.

Based on these measures and the TCP throughput function [16], we observe that variance in parameters such as round trip times (RTT), drop probabilities or timeout delays leads to variance in per-flow goodput [10]. Since packet drop characteristics affect all three parameters (queueing delay component of RTT, drop probabilities and timeout occurrences), the problem of TCP performance optimization is greatly affected by drop characteristics.

In this paper, we evaluate the following network-based and end-to-end enhancements for addressing the issue of enhancing TCP performance:

- **Random Early Detection (RED)**: an active queue management technique [7, 3],
- **TCP-explicit congestion notification (ECN)**: which uses a one-bit explicit congestion notification instead of using packet drop as an implicit notification [19, 8], and
- **Packeteer's TCP Rate Control:** a network-based solution which controls the left and right edges of the TCP window, and shapes the TCP acknowledgment stream [15].

Our studies indicate marked improvements in fairness (as measured by the standard deviation in goodput and covariance) without significant tradeoffs in other metrics as we move from RED through ECN to TCP rate control. All schemes control bottleneck queueing delay, but trading off other measures such as drop rate, utilization and fairness, with TCP rate control showing the best performance. However for short transfers on low speed links, RED and ECN tradeoff fairness to achieve improvement in the total number of transfers as measured over a medium sized simulation time-window.

In terms of deployment, TCP rate control and RED offer the fastest route to immediate and widespread deployment. The minimal state requirements and protocol transparency of RED allow it a greater available space of deployment scenarios. TCP rate control has been shown in practice to be applicable in enterprise intra-nets, access points to servers/data centers, and in edges of service provider networks [15] (in general at edges nodes where microflows can be examined).

This paper is organized as follows. Section 2 discusses in more detail the issues regarding TCP performance, followed by Section 3 which gives a description of the solution alternatives considered. The performance analysis section (Section 4) is split into sections describing metrics, parameters and configurations (Sections 4.1, 4.2) and simulation results (Section 5). We summarize our observations and conclude in Section 6. Appendices A.1, A.2, A.3 examine performance results based upon additional parameter dimensions.

## 2 TCP Congestion Control Issues

The TCP congestion control protocol [9] builds upon reliability mechanisms in TCP such as windows, timeout, and acks to provide robust control. However, its dependence on packet loss for congestion detection leads to unfairness and little control over bottleneck queueing delays.

Specifically, until a packet is dropped TCP fills up bandwidth and buffer resources leading to large queues. One way to reduce queueing delay is by dropping packets. But naive drop schemes like drop-tail can lead to burst dropping of packets from all participating connections, which then timeout simultaneously. Such burst dropping leads to cycles of underutilization, window increase, followed by tail drop again. This is commonly known as "TCP synchronization" [7] - not a desirable tradeoff for the network operator.

So the question of optimizing queueing delay without affecting user metrics - average and variance in per-connection goodput - adversely is still open. Since drop characteristics affect these metrics, we carefully examine the effect of drop characteristics next. Also note that congestion control issues are exacerbated by non-uniform TCP/IP stack implementations and the use of different initial window and RTO parameters.

### 2.1 Packet Drop Characteristics and TCP performance

The effect of packet drop characteristics can be classified under three headings: cost of each packet drop, optimization/fairness issues, and scaling issues.

**Scaling issues due to packet drop costs:** Every packet drop has a non-negligible and non-linear cost for connections which experience it. This is partly because detecting drop robustly requires a timeout [21]. There are two components to this cost: delay cost and bandwidth/buffer overhead cost.

Delays occur in the form of retransmission and timeout (detection) delays. These delays translate into lower goodput on a per-source basis (a user metric – shows up as higher variance in goodput). The *delay component of packet drop cost increases in high speed links.* Recent experimental evidence suggests that a majority of retransmissions are due to timeout, and not due to fast retransmission [16, 1]. Further, the same studies suggest that *multiple successive timeouts* occur with a non-trivial frequency [16] leading to even greater delay-cost due to timer back off. These observations confirm the fact that packet drop today results in significant delay costs.

The second component of drop-cost is the overhead cost. This consists of bandwidth/buffer resources expended for transporting the dropped packet to the bottleneck and the additional resources expended for retransmission of packets. Our simulations indicate that in many cases with TCP Reno between 7-8 packets are retransmitted for every lost packet – partly a result of timeouts and the go-back-N retransmission strategy of TCP. Overhead cost manifests as lower aggregate goodput when compared to utilization (and aggregate throughput).

*The aggregate overhead component of packet-drop-cost increases with number of active flows at the bottleneck.* With the Internet expected to scale to a billion hosts, it is easy to imagine a large number of flows at core bottlenecks in the Internet. Under such conditions, a huge number of packet drops (distributed over sufficient number of sources) would be required to reduce congestion adequately.

Appendix A.2.1 shows that the overhead and delay costs appear particularly in smaller RTT/large bandwidth situations even for a moderate number of flows (100 flows). With larger RTT configurations (Appendix A.3.1) we see that aggregate goodput is low in both 10 and 100 flows-based simulations.

**Optimization Issues:** Though the use of packet drops as a congestion indicator is a robust technique, distribution of packet drops among active connections in a fair manner to affect goodput equally is a non-trivial optimization problem.

For example, we know that when a source experiences packet loss, it is an indication of a congestion event. But *the occurrence of congestion event at a bottleneck does not imply that at least one packet from all active flows are dropped*, or should be dropped. This ambiguity introduces unfairness because some flows may back off while others do not. Further, *multiple packet loss events experienced by a source do not imply multiple congestion events in the path* - packet losses may be correlated. It is known only that packet losses spaced by at least RTT can be considered independent [2].

Common TCP Reno-derived implementations further complicate this optimization problem because they do not filter loss-indications well enough to determine (a smaller set of) congestion events. For example, a burst drop of three or more packets with nearby sequence numbers would result in multiple window decreases followed by a timeout. Since only a subset of flows experience this phenomenon, the fairness/optimization problem is exacerbated. We observe these problems in sections 5.1.1, 5.2.1, A.2.1 and A.3.1. Given these types of issues in optimization of per-flow TCP goodput (and its variance across flows), it is worthwhile to consider a non-packet-drop based strategy as a primary method for TCP performance optimization. Packet-drop based control would be a robust backup method.

## 2.2 Summary of Issues

In summary, we see the need to reduce drop rate without losing control over congestion. Ideally, congestion control which is decoupled from packet loss and RTT (until loss occurs) would exhibit better fairness properties. These techniques could then also be leveraged to provide control over queueing delay. It is also important that solutions be immediately deployable and allow partial deployment. Network-based solutions have an advantage over end-to-end solutions in this regard. Finally the solution needs to exhibit good scalability properties.

The Internet Engineering Task Force (IETF) and independent researchers have proposed several improvements to TCP/IP-based control at the transport and network layers. The Random Early Detection (RED) buffer management strategy was designed to break the TCP synchronization problem, primarily through randomization and non-bursty early packet dropping (before the buffer was full). Proposed transport layer (end-to-end) enhancements include the fast retransmit and recovery (FRR) algorithms [22], and selective acknowledgments (SACK) [14]. The Explicit Congestion Notification (ECN, one bit explicit feedback) scheme [19] requires both end-to-end and network support. Network-based enhancements are aimed at improving fairness and throughput.

These include mechanisms like scheduling [5], improved packet discard policies [7, 6, 13, 12] and explicit rate control of TCP [20, 15].

The end-to-end proposals aim to provide better filtering to detect congestion events from loss events, improved retransmission, and attempt to reduce the occurrence of timeouts. Providing fairness and queueing delay control have required some form of network-based support. In this paper we shall study the fairness, queueing delay and loss control characteristics of three proposed schemes: RED, ECN and Packeteer's TCP rate control.

# 3 Solution Approaches Evaluated

This section briefly describes the solution approaches evaluated in this paper: *Random Early Detection* [7] (RED), *Explicit Congestion Notification* [19, 8](ECN), and Packeteer's *TCP Rate Control* [20, 15].

## 3.1 Random Early Detection (RED)

Random early detection is a active queue management technique proposed by Sally Floyd and Van Jacobson. Congestion is detected by monitoring the average queue size: if the average crosses a lower threshold (min_thresh), then congestion indications are given to connections based on a probabilistic function. The function aims for an average spacing between packet drops (to avoid burst drops) and uses randomization in the selection process. The probability that the router notifies a particular connection to reduce its window is roughly proportional to that connection's share of the bandwidth through the gateway. The RED gateway has no bias against bursty traffic and thus tackles the problem of global synchronization.

Congestion indications may be in the form of either by dropping a packet (RED) or by setting the ECN bit in the IP header [19] (ECN). When the average queue crosses a second threshold (max_thresh), all incoming packets are dropped.

A positive feature of RED is that it does not store per-flow state. The absence of per-flow state makes it immediately deployable at any queueing point in the Internet and it is amenable to partial deployment. Our simulations indicate that it keeps utilization high, queue length and drop rate relatively low in the steady state, providing a good tradeoff for the operator. It also solves the TCP synchronization problem, reduces bias in dropping through randomization and attempts to space drops apart.

However, when a large number of connections, or several connections are in the slow-start phase simultaneously, the RED upper threshold (max_thresh) might be hit leading to drop-tail like characteristics such as unfairness. Further, when there are a large number of connections, all of them may not get congestion indications – leading to unfairness. We have observed that this problem cannot be avoided by changing RED parameter settings since the parameters are not all independent.

### 3.2 Explicit Congestion Notification (ECN)

ECN is a solution that involves both end-systems and bottlenecks [19]. Bottlenecks may give a single bit indication for congestion instead dropping packets. In practice, bottlenecks may set bits to aid end-to-end congestion avoidance algorithms, but drop packets when congestion is not controlled. The recommended behavior is to set the congestion experienced bit in an ECN-Capable packet only if it would have otherwise dropped the packet. If the bit is already set, the packet transmitted as usual. The original idea of a bit-based feedback mechanism was proposed by Ramakrishnan and Jain in the Decbit scheme [18].

In TCP-ECN, the end system TCP implementation is upgraded to respond to network congestion indications in the form of the ECN bit. The response to an ECN-bit is essentially the same as the congestion control response to a *single* dropped packet, i.e, the source TCP is required to halve its congestion window. Further, the ECN-capable-TCP reacts to congestion indications (packet drops or ECNs) at most once per window of data (i.e. roughly at most once per round trip time). This avoids the TCP Reno problem of reacting multiple times to congestion indications within a single round trip time.

The advantage of the ECN technique is to decouple congestion indications from packet loss. The explicit indication removes any ambiguity as to whether the loss was caused due to reasons other than congestion. It promotes congestion avoidance and improves performance of short transfers which would be most adversely affected by even a single packet drop. However, since only a single bit is used, it is difficult to reduce the throughput dependence on round trip times – i.e., fairness issues remain. The most significant problem is the need for participation by both routers and end systems leading to possibly slow deployment. Currently ECN is in an experimental phase at the IETF.

### 3.3 TCP Rate Control

TCP rate control is a technique where the rate of a TCP flow is directly and explicitly controlled [20, 15]. Packeteer's rate control solution (product called "Packetshaper$^{TM}$") exploits the fact that **TCP's rate is determined by a) the rate of acknowledgments, b) the acknowledgment number and, c) the receiver maximum window size fields** (the last two variables determine the left and right edges of the TCP window).

Specifically, *given a constant window size, the TCP rate is equal to the rate of the "ack clock"* i.e. the stream of acknowledgments. Control of the ack rate also smoothes out burstiness in TCP transmission. Also, the packetshaper can tradeoff acknowledgment queues in the reverse direction (which can be optimized even further by compressing the information) for packet queues in the forward direction [20]. Observe that packet storage in queues can be considered to be maintainence of per-active-flow state. Based upon this definition, the packetshaper in fact reduces the amount of aggregate per-flow state (packets + acks). Further, since acks incur lesser transmission delays, ack shaping results in reduction in aggregate queueing delays experienced by the connection at the bottleneck.

The discussion above assumed a constant TCP window. But, the TCP window is not a constant - *the* *right edge of the window is controlled by the CWND (congestion window) variable and the receiver window field in TCP acks; and the left edge of the window is controlled by the acknowledgment number field in TCP acks.* The packetshaper therefore uses the two fields - the receiver window field and acknowledgment number field - to control the size of the window. The control of the window in addition to ack shaping allows packet shaper to control aggregate queueing delays as well as individual TCP throughput at a fine granularity.

In addition, when placed at key bottlenecks in the system, the packetshaper can *dynamically divide the available capacity among the contending flows in a fair manner using a combination of policy rules, a scheduling algorithm and a dynamic rate calculation algorithm.* Specifically, allocated capacity temporarily unused by a bottlenecked flow is fairly distributed to other contending flows to achieve max-min fairness [4, 20]. Observe that using this technique, packetshaper can rapidly and accurately adapt to dynamic capacity changes. We hypothesize based upon our current set of simulation results that RED and ECN would not have such response characteristics under conditions of highly variable available capacity.

Similar techniques for calculating rate allocations have been developed for the ATM Available Bit Rate (ABR) service and the reader is referred to [11], chapter 3 for a survey of these schemes. A striking feature in this class of schemes is that max-min fairness can be achieved even with FIFO queueing, given explicit rate feedback. *The calculated rate allocation is enforced by a) controlling the ack rate, and b) control of window edges after translation of rate into a window value.* Rate-to-window translation requires knowledge [20] of round trip times which can be estimated at the packetshaper.

The remarkable feature is that TCP rate can be completely controlled only by training the ack stream, and the solution is transparent to end systems (TCP) or routers. Due to patent and commercial reasons, we cannot reveal more details of the core algorithms, but the features mentioned above serve to differentiate it sufficiently from RED, ECN and scheduling schemes such as WFQ, or CBQ. The packetshaper product contains several other features to handle non-TCP flows, scheduling, other TCP grooming and behavior enforcement functions applicable to different speeds and handles implementation differences.

Packetshaper works best when the TCP packet and ack flow are accessible to it, and can perform the core functions even under limited asymmetry (eg: ack flow alone accessible). Since it requires reading and writing into TCP headers, these headers should be accessible (eg: not encrypted or authenticated). Though it maintains per-flow state, through information compression, the storage and state requirements scale very well (products supporting over 20,000 simultaneous flows are now in operation). Due to the first two of these constraints, the solution is more applicable to network edges rather than cores. In this deployment space, it is as easily deployable as RED and is transparent to hosts and routers. Note that ECN does not share the advantage of deployability since it requires changes to host TCP implementations.

## 4 Performance Analysis

The following sections present our performance analysis of the three solution alternatives: RED, ECN and TCP rate control. We use the high level model shown in Figure 1 to guide our evaluation.

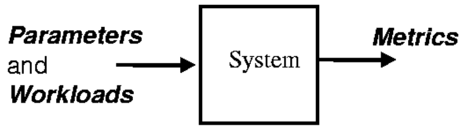


Figure 1: Performance Model

Specifically, we consider the system (of TCP flows in this case) as a black box to which is input a set of ***parameters*** and workloads (parameters include the choice of the scheme, configurations etc) and the output is a set of ***metrics*** which evaluate the tradeoff among various resource constraints in the system. The next two sections explain the choice of metrics and parameter dimensions explored in this evaluation.

## 4.1 Metrics

As mentioned briefly in the introduction section, we classify metrics into two major categories: user metrics and operator metrics. While the operator metrics are the same for all simulations, we use two sets of user metrics for simulations involving infinite transfers and finite (short) transfers.

**Operator Metrics:** The operators key resources are bandwidth and buffers. The operator is willing to tradeoff buffer resources to ensure high utilization of bandwidth. But high queueing delays or drop rates are undesired for supporting customers' interactive applications such as ftp or WWW. The operator metrics we consider are:

**Average link Utilization** : low link utilization, given adequate load is unacceptable.

**Average Queue Length** : Low average queue lengths imply lower average queueing delay experienced by participating connections. Prefer low queue lengths combined with high utilization.

**Maximum Queue Length** : Very high maximum queue lengths indicate high buffer requirements.

**Packet drop rate** : Packets dropped represent wasted bandwidth and buffer resources on upstream links. As discussed earlier packet drop also has a high cost for users, but is unavoidable during congestion. Infinite flows which are not rate controlled also require packet drops as congestion indicators.

**User Metrics for Infinite (long) Transfers:** The user is interested in his/her per-flow goodput for infinite flows (requires N metrics where N = number of flows). But for brevity, we use the average and standard deviation in goodput as metrics. The goodput excludes the retransmission rate and includes the effect of retransmission and timeout delays.

**Average (per-flow) Goodput** : ($\mu$ goodput) This quantity should be as high as possible. Ideally, for a single bottleneck, average goodput times number of sources should equal the product of utilization and bandwidth, where utilization is as close to 100% as possible.

**Standard deviation in (per-flow) goodput:** ($\sigma$ goodput) This quantity is a rough measure of fairness. As explained earlier, the throughput (and consequently the goodput) are affected by the drop probability, round trip times and timeout delays. Schemes which reduce the magnitude of the latter parameters or the dependency of throughput on these parameters would be more fair. Ideally, for a single bottleneck with infinite transfers, this metric should be close to zero.

**Covariance:** In our simulations, we also consider the covariance which ratio of standard deviation to average (goodput) and prefer a smaller ratio. The covariance is only used for convenience.

**User Metrics for Short Transfers:** The user is interested in per-transfer response time. As before, we use the following proxies for this quantity.

**Average time for a transfer (across all transfers):** Indicative per-transfer response time (along with the standard deviation)

**Standard deviation in time for transfer (across all transfers):** Indicative of per-transfer response time (along with with the mean).

Our simulation results are presented in a tabular format with user and operator metrics separated and significant results highlighted.

### 4.2 Parameters (factors) and configurations

Figure 2 gives the basic configuration template that we used in our simulations. It contains a single bottleneck shared by a set of unidirectional TCP flows. This simple template matches many Intra net layouts where the expensive WAN link or leased line is the key bottleneck in the system. It also allows evaluations involving a number of parameter dimensions. This evaluation does not consider configurations with multiple bottlenecks. The TCP sources and destination implemented *TCP Reno*. Sources and bottlenecks were appropriately modified as required for the three schemes evaluated.

The following are the parameter dimensions explored:

- Schemes: RED, ECN and Rate-control
- Infinite transfers and short transfers

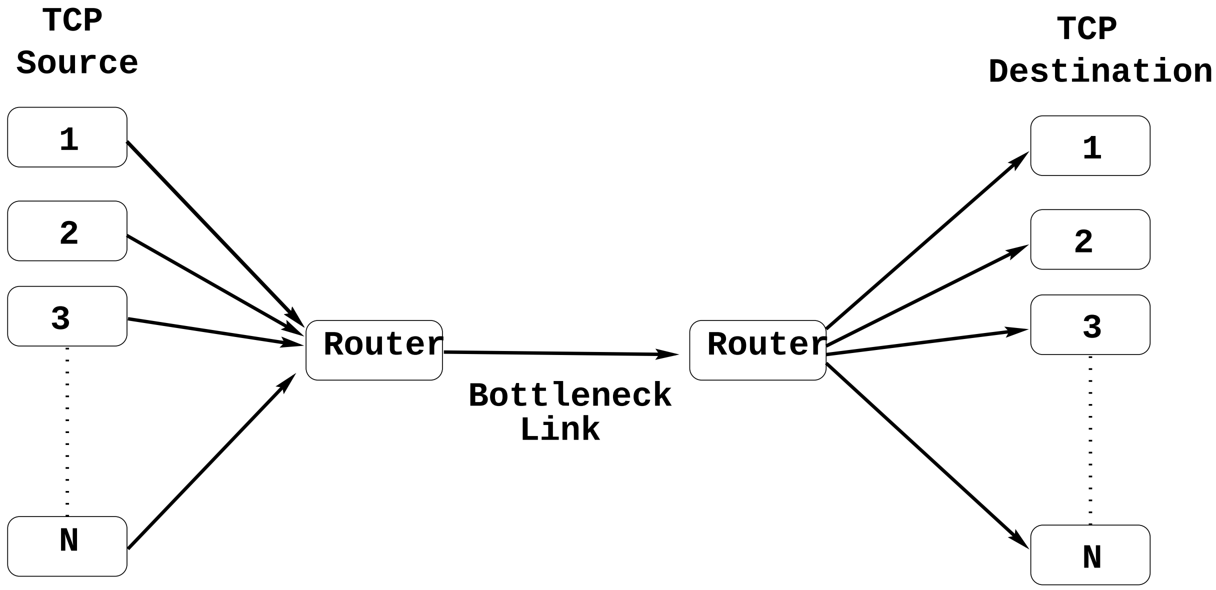


Figure 2: Configuration template used in Simulation

- Homogeneous and Heterogeneous RTTs
- Staggered or non-staggered connection start times
- Link speeds: 28 Kbps, 56 Kbps, 1.5 Mbps, 45 Mbps and 150 Mbps
- RTTs: one-way distance of 3 km, 30 km and 3000 km
- Number of TCP connections: 10, 100 and 500

We performed full factorial simulations [10] involving a subset of factors (i.e. exploring all combinations of these factors) and fractional sets of simulations involving other factors (eg: RED parameter values) for a total of over 2000 simulations. For brevity, we present only representative simulation results in this paper.

In all the simulations the links that connected the source/destination to its nearest router was 150 Mbps. For the heterogeneous RTT and staggered start-time simulations, the sources were grouped into four groups. In the staggered case, every flow in a given group started at the same time. The flows were staggered by 250 milliseconds.

These simulations do not explore multiple bottleneck cases where the schemes are implemented only in a limited subset of bottlenecks. Also we do not explore issues of variable available capacity which can occur when the queue management schemes are applied to one of many scheduler queues. We do not also implement packetshaper enhancements for low speed bottlenecks.

## 5 Simulation Results

This section reports simulation results using homogeneous RTT configuration, heterogeneous RTT configurations and with a workload of short transfers. These dimensions are sufficient to clearly differentiate between the performance of the three schemes. Additional parameter dimensions (staggered flow start times, shorter RTTs and different number of flows) are explored in the appendices, which also report significant performance issues.

### 5.1 Dimension: RED vs ECN vs TCP rate control

This section presents a preliminary look at the parameter dimension of schemes evaluated (RED vs ECN vs TCP rate control) while keeping most of other parameter dimensions (except link speeds) simple and constant. Specifically, we use 100 connections, one-way distance of 3000 km (corresponding to propagation delays of 15 ms one way), homogeneous RTT configuration. The link speeds vary from 28 Kbps to 150 Mbps.

Our primary observation in this section is that *good performance in terms of provider metrics does not imply good performance in terms of user metrics* (esp. in drop-based schemes such as RED). Due to the symmetric nature of the configuration, problems in ECN do not surface.

#### 5.1.1 RED (homogeneous RTTs)

In table 1 we observe the operator and user metrics when the bottleneck implements Random Early Detection. This simulation was conducted with 100 connections, one-way distance of 3000 km (corresponding to propagation delays of 15 ms one way), homogeneous RTT configuration. The link speeds vary from 28 Kbps to 150 Mbps.

| | *Operator Metrics* | | | *User Metrics* | | |
|---|---|---|---|---|---|---|
| **Speed** Mbps | **Util** Percent | **$\mu$ Q/Max Q** Pkts/Pkts | **Drop** Pkts | **$\mu$ Goodpt** Mbps | **$\sigma$ Goodpt** Mbps | **Covar** |
| 0.028 | 93.75 | 30.75/61 | 439 | 0.00082 | 0.00000 | 0.000 |
| 0.056 | 92.67 | 31.59/63 | 405 | 0.00098 | 0.00049 | 0.500 |
| 0.128 | 98.12 | 30.19/62 | 534 | 0.00206 | 0.00132 | 0.641 |
| 0.256 | 99.18 | 28.37/62 | 553 | 0.00349 | 0.00307 | 0.880 |
| 0.384 | 99.41 | 29.55/61 | 638 | 0.00487 | 0.00279 | 0.573 |
| 1.5 | 99.66 | 23.82/63 | 1002 | **0.01444** | **0.01558** | **1.078** |
| 10 | 99.50 | 18.59/63 | 2215 | 0.08652 | 0.05919 | 0.684 |
| 45 | 98.51 | 13.29/63 | 3818 | 0.37506 | 0.11369 | 0.303 |
| 150 | 96.57 | 9.86/63 | 3637 | 1.14209 | 0.35048 | 0.307 |

Table 1: RED; 100 sources; Homogeneous RTT; One-way distance = 3000 Km.

We immediately observe that RED satisfies operator metrics: high utilization and low queue lengths. A non-trivial number of packets were dropped in all cases which included burst drop instances. From the user perspective, the aggregate goodput (which is average goodput multiplied by 100, taken as a fraction of bottleneck link speed) degrades with increase in bottleneck link speed. Lower values of goodput indicate more retransmission overhead. The covariance and standard deviation in goodput is consistently high indicating unfairness. The highlighted entry shows that the highest covariance occurs for a 1.5 Mbps link. This is due to the effects of burst drop concentrated on a small subset of sources and its interaction with the TCP Reno implementation.

We believe that the use of a single canonical set of RED parameters may have introduced some of this sub optimality, but our investigation of the RED parameter space did not yield a single parameter combination which avoided burst drop. The simulation results presented above shows that RED has difficulty spreading out packet drops among even a moderate number of active connections.

### 5.1.2 ECN (homogeneous RTTs)

Table 2 shows the performance of ECN under the same set of conditions as the previous simulation set.

| | *Operator Metrics* | | | *User Metrics* | | |
|---|---|---|---|---|---|---|
| **Speed** Mbps | **Util** Percent | **$\mu$ Q/Max Q** Pkts/Pkts | **Drop** Pkts | **$\mu$ Goodpt** Mbps | **$\sigma$ Goodpt** Mbps | **Covar** |
| 0.028 | 98.25 | 208.60/441 | 0 | 0.00131 | 0.00075 | 0.573 |
| 0.056 | 98.90 | 233.01/487 | 0 | 0.00187 | 0.00079 | 0.421 |
| 0.128 | 99.33 | 275.04/542 | 0 | 0.00269 | 0.00057 | 0.212 |
| 0.256 | 99.57 | 292.08/502 | 0 | 0.00341 | 0.00081 | 0.237 |
| 0.384 | 99.68 | **327.63/557** | 0 | 0.00441 | 0.00083 | 0.187 |
| 1.5 | 99.90 | 238.64/295 | 0 | 0.01678 | 0.00072 | 0.043 |
| 10 | 99.98 | 228.57/264 | 0 | 0.10167 | 0.00081 | 0.008 |
| 45 | 99.99 | 103.01/193 | 0 | 0.44987 | 0.00061 | 0.001 |
| 150 | 96.53 | 38.21/174 | 0 | 1.40727 | 0.02705 | 0.019 |

Table 2: ECN; 100 sources; Homogeneous RTT; One-way distance = 3000 Km

With the introduction of ECN, we immediately remove the effects of packet drop and the large variance effects caused by it. The removal of packet drop effects are also partially due to the better filtering of congestion indications at the ECN-TCP source compared to TCP Reno. Except for long queueing delays experienced (which can be explained as a result of ECN marking on received packet at the tail of the queue rather than marking the packet in the front of the queue), all metrics show picture perfect performance. But as we shall see this performance is partly due to the homogeneous RTT configuration.

### 5.1.3 TCP Rate Control (homogeneous RTTs)

Table 3 shows the performance of TCP rate control under the same set of conditions as the previous two simulation sets.

| | *Operator Metrics* | | | *User Metrics* | | |
|---|---|---|---|---|---|---|
| **Speed** Mbps | **Util** Percent | **$\mu$ Q/Max Q** Pkts/Pkts | **Drop** Pkts | **$\mu$ Goodpt** Mbps | **$\sigma$ Goodpt** Mbps | **Covar** |
| 0.028 | 98.25 | 207.20/437 | 0 | 0.00128 | 0.00074 | 0.576 |
| 0.056 | 98.90 | 230.77/481 | 0 | 0.00183 | 0.00080 | 0.434 |
| 0.128 | 99.33 | 272.49/538 | 0 | 0.00269 | 0.00057 | 0.212 |
| 0.256 | 99.57 | 283.83/488 | 0 | 0.00337 | 0.00081 | 0.241 |
| 0.384 | 99.68 | 320.35/540 | 0 | 0.00428 | 0.00059 | 0.138 |
| 1.5 | 99.90 | 184.92/200 | 0 | 0.01656 | 0.00080 | 0.048 |
| 10 | 99.98 | 161.71/199 | 0 | 0.10130 | 0.00077 | 0.008 |
| 45 | 99.99 | 33.80/193 | 0 | **0.44981** | **0.00032** | **0.001** |
| 150 | 99.84 | 41.59/193 | 0 | 1.48251 | 0.02731 | 0.018 |

Table 3: TCP Rate Control; 100 sources; Homogeneous RTT; One-way distance = 3000 Km

Our model of TCP rate control did not include several features targeted for lower speed links. The result of these modeling limitations shows up as high queueing delays for slower speed bottlenecks. But, for medium and higher speed cases the performance is excellent in terms of all metrics.

## 5.2 Dimension: Homogeneous vs Heterogeneous RTTs

This section looks at performance of the three schemes when the round trip times are heterogeneous. These simulations are conducted with 10 sources (not 100 sources as used in the last three sets). As explained earlier, The flows were grouped into four sets and all flows from a given set had an one-way distances of 1002Km, 2000Km, 3000Km and 5000Km respectively. The link speeds vary from 28 Kbps to 150 Mbps.

This section clearly differentiates the benifits of explicit TCP rate control when compared to ECN and RED. Specifically, TCP rate control provides fairness (in terms of user metrics) even when RTTs are heterogeneous, without deteriorating any of the other metrics.

### 5.2.1 RED (heterogeneous RTTs)

Table 4 lists the performance metrics with RED in a heterogeneous RTT configuration.As explained in the introductory sections, TCP throughput is an inverse function of RTT as well as loss probability and timeout delays. The introduction of heterogeneous RTTs adds to the variance in per-flow goodput (and increases the co-variance) in TCP with or without packet loss. We expect the variance to be larger when both packet loss and RTT variance is present.

| | *Operator Metrics* | | | *User Metrics* | | |
|---|---|---|---|---|---|---|
| **Speed** Mbps | **Util** Percent | **$\mu$ Q/Max Q** Pkts/Pkts | **Drop** Pkts | **$\mu$ Goodpt** Mbps | **$\sigma$ Goodpt** Mbps | **Covar** |
| 0.028 | 97.22 | 30.08/46 | 11 | 0.00401 | 0.00113 | 0.281 |
| 0.056 | 98.55 | 28.24/46 | 52 | 0.00623 | 0.00111 | 0.179 |
| 0.128 | 99.36 | 19.81/45 | 68 | 0.01581 | 0.00615 | 0.389 |
| 0.256 | 99.68 | 19.29/45 | 89 | 0.02703 | 0.01463 | 0.541 |
| 0.384 | 99.16 | 18.30/45 | 114 | 0.03917 | 0.01327 | 0.339 |
| 1.5 | 99.63 | 16.97/44 | 242 | 0.14788 | 0.05646 | 0.382 |
| 10 | 99.10 | 13.96/42 | 743 | 0.91430 | 0.42644 | 0.466 |
| 45 | 93.33 | 10.00/47 | 951 | **2.65288** | 1.16895 | 0.441 |
| 150 | **79.88** | 7.79/50 | 1079 | **5.68535** | 2.83307 | 0.498 |

Table 4: RED; 10 sources; Heterogeneous RTTs

Interestingly, the utilization remains high in most cases (except the 150 Mbps case), but goodputs deteriorate sharply (aggregate goodput is less than 60% of bottleneck speed in the cases highlighted). Standard deviation in goodput and covariance remain high indicating unfairness in addition to lower average goodput. This further suggests that RED optimizes operator metrics, trading off user-metrics under these conditions.

We observed that staggering the flows (albeit in small groups as we have mentioned) does not solve this problem created by burst drops and heterogeneity in RTTs.

### 5.2.2 ECN (heterogeneous RTTs)

Table 5 shows the performance of ECN under similar conditions as the last simulation set (heterogeneous RTTs, 10 sources).

| | *Operator Metrics* | | | *User Metrics* | | |
|---|---|---|---|---|---|---|
| **Speed** Mbps | **Util** Percent | **$\mu$ Q/Max Q** Pkts/Pkts | **Drop** Pkts | **$\mu$ Goodpt** Mbps | **$\sigma$ Goodpt** Mbps | **Covar** |
| 0.028 | 97.22 | 33.02/56 | 0 | 0.00401 | 0.00113 | 0.281 |
| 0.056 | 98.57 | 41.61/69 | 0 | 0.00614 | 0.00123 | 0.200 |
| 0.128 | 99.37 | 62.00/108 | 0 | 0.01515 | 0.00303 | 0.200 |
| 0.256 | 99.68 | 62.46/86 | 0 | 0.02408 | 0.00659 | 0.273 |
| 0.384 | 99.79 | 42.64/70 | 0 | 0.03965 | 0.00549 | 0.138 |
| 1.5 | 99.95 | 35.74/79 | 0 | 0.15262 | 0.02161 | 0.142 |
| 10 | 99.99 | 26.60/68 | 0 | 1.00246 | 0.37674 | 0.376 |
| 45 | 100.00 | 19.32/69 | 0 | 4.50278 | 4.18975 | 0.930 |
| 150 | 100.00 | 16.50/76 | 0 | 15.00642 | **19.37648** | **1.291** |

Table 5: ECN; 10 sources; Heterogeneous RTTs

These simulations show that while ECN is capable of satisfying all the operator metrics in this heterogeneous RTT case, it is not capable of reducing the variance in goodput (unfairness). The covariance in fact increases as the link speeds increase. This points to the insufficiency of single-bit feedback in achieving fairness goals. The negative effects of packet drop are removed however. We also note the marked decrease in queue lengths with the decrease in the number of active connections.

### 5.2.3 TCP rate control (heterogeneous RTTs)

Table 6 shows the corresponding performance of TCP rate control under these circumstances.

| | *Operator Metrics* | | | *User Metrics* | | |
|---|---|---|---|---|---|---|
| **Speed** Mbps | **Util** Percent | $\mu$ **Q/Max Q** Pkts/Pkts | **Drop** Pkts | $\mu$ **Goodpt** Mbps | $\sigma$ **Goodpt** Mbps | **Covar** |
| 0.028 | 97.14 | 29.60/48 | 0 | 0.00328 | 0.00063 | 0.194 |
| 0.056 | 98.55 | 36.01/61 | 0 | 0.00500 | 0.00057 | 0.115 |
| 0.128 | 99.37 | 19.20/25 | 0 | 0.01286 | 0.00052 | 0.041 |
| 0.256 | 99.68 | 17.39/20 | 0 | 0.02711 | 0.00093 | 0.034 |
| 0.384 | 99.79 | 18.41/20 | 0 | 0.03981 | 0.00054 | 0.014 |
| 1.5 | 99.95 | 16.59/20 | 0 | 0.15106 | 0.01089 | 0.072 |
| 10 | 99.99 | 11.39/20 | 0 | 1.00115 | 0.44187 | 0.441 |
| 45 | 100.00 | 13.61/34 | 0 | 4.50401 | 1.01488 | 0.225 |
| 150 | 99.96 | 13.89/94 | 0 | **15.00257** | **1.37127** | **0.091** |

Table 6: TCP rate control; 10 sources; heterogeneous RTTs

We observe the superiority of the TCP rate control compared to ECN and RED in this configuration. The TCP sources achieve high fairness and goodput almost independent of the RTTs. Performance in terms of all metrics is excellent.

The reason TCP rate control avoids both the effects of heterogeneous RTTs and packet loss is because it uses explicit window control and hence does not conform to the Padhye formula for TCP throughput. This allows it to achieve fair allocations independent of RTT variance of participating flows. Though explicit control, packet drops are also avoided.

## 5.3 Dimension: Short transfers vs Long transfers

In this section we look at the performance of the schemes with respect to short file transfers. The same configuration template (Figure 2) is used in these simulations. Flows are grouped into four sets and the start time of every set is staggered by 250 ms. Every flow sends 10K bytes (10 packets) and closes the connection and transfer. After a pause of 250 ms, the same source would "re-open" the connection (with parameters set to initial values) and send another 10K bytes (10 packets) and so on.

Such short transfers last throughout the simulation time. The following statistics were collected from the experiment. As mentioned earlier, we use a new set of user metrics for these simulations – the the average transfer time and the standard deviation of the transfer time measured over all transfers. Together, these metrics are indicative of the response time of individual transfers. We ignore slower speeds because the simulation time (10 s) was not sufficient for enough transfers.

Though Web transfers are typically short, this model is not an accurate model of the world wide web (WWW). A similar model has been used in the past [1]. We also assume that the transfer size parameter in these simulations is fixed at 10KB. Paxson [17] observes that observed median transfer sizes ranges anywhere from 2KB to 5KB, and that the median varies across data sets.

Our primary observation in this configuration is that the tradeoff in fairness by RED and ECN in low speed bottleneck configurations leads to larger aggregate number of completed transfers compared to TCP rate control. At higher bottleneck speeds, there is no difference in performance.

### 5.3.1 RED (Short transfers)

| | *Operator Metrics* | | | *User Metrics* | | |
|---|---|---|---|---|---|---|
| **Speed** Mbps | **Util** Percent | $\mu$ **Q/Max Q** Pkts/Pkts | $\mu$ T. time Milli Secs | $\sigma$ **T. time** Milli Secs | **Covar** | **# Trans** |
| 0.256 | 99.29 | 28.90/51 | 9527 | 929.2244 | 0.098 | 9 |
| 0.384 | 99.49 | 28.93/51 | 8852 | 2127.2276 | 0.240 | 27 |
| 1.5 | 99.39 | 25.20/49 | 3763 | 2631.0244 | 0.699 | 227 |
| 10 | 97.87 | 13.07/50 | 588 | **1077.7262** | 1.832 | **1457** |
| 45 | 80.98 | 3.05/44 | 143 | 394.4417 | 2.750 | 2513 |

Table 7: RED; 100 sources; Repeated Short Transfers

Table 7 tabulates the results for simulations with RED. We observe that the average transfer time reduces with increase in bandwidth. But the effective bandwidth consumed by an average transfer is much less than available bandwidth. This is partially due to the fact that most of the transfer time is spent in slow start, and in timeouts. The standard deviation of transfer time (and covariance) is large indicating that some transfers were able complete at the expense of other transfer (unfairness).

### 5.3.2 ECN (Short transfers)

Table 8 shows the simulation results of short transfers with ECN.

We note that the average response times in slower speed configurations are slightly worse than RED because, in spite of not having loss-triggered effects, the queueing delays add significantly to the response time. Otherwise ECN performance scaled well with increase in link speeds allowing a larger number of transfers and reduction in covariance and standard deviation of transfer time.

| | *Operator Metrics* | | | *User Metrics* | | |
|---|---|---|---|---|---|---|
| **Speed** Mbps | **Util** Percent | $\mu$ **Q/Max Q** Pkts/Pkts | $\mu$ T. time Milli Secs | $\sigma$ **T. time** Milli Secs | **Covar** | **# Trans** |
| 0.256 | 98.80 | 312.53/528 | 9625 | 279.5135 | 0.02904 | 3 |
| 0.384 | 99.16 | 331.81/547 | 9430 | 1336.5891 | 0.14174 | 8 |
| 1.5 | 99.57 | 197.25/249 | 4172 | 1585.0223 | 0.37991 | 164 |
| 10 | 99.84 | 117.24/189 | 537 | **109.2574** | 0.20328 | **1191** |
| 45 | 83.71 | 4.28/51 | 123 | 6.4239 | 0.05243 | 2600 |

Table 8: ECN; 100 sources; Repeated Short Transfers

### 5.3.3 TCP rate control (Short transfers)

The results for TCP rate control in this situation are shown in table 9.

| | *Operator Metrics* | | | *User Metrics* | | |
|---|---|---|---|---|---|---|
| **Speed** Mbps | **Util** Percent | $\mu$ **Q/Max Q** Pkts/Pkts | $\mu$ T. time Milli Secs | $\sigma$ **T. time** Milli Secs | **Covar** | **# Trans** Number |
| 0.256 | 98.80 | 283.89/470 | - | - | - | 0 |
| 0.384 | 99.16 | 289.10/485 | - | - | - | 0 |
| 1.5 | 99.35 | 202.13/309 | 3845 | 1375.3581 | 0.35770 | 186 |
| 10 | 99.74 | 151.67/244 | **538** | **108.9222** | **0.20251** | **1198** |
| 45 | 83.41 | 3.82/67 | 125 | 16.5318 | 0.13245 | 2600 |

Table 9: TCP rate control; 100 sources; Repeated Short Flows

Interestingly, at slower speeds, the TCP rate controller results in a smaller number of transfers completed. This is because, the rates of all contending transfers are equally reduced (for fairness) and each of them take longer. Also the simulations were run for 10s - a limited observation window - which was not long enough for these transfers to complete.

On the other hand the ECN and RED schemes traded off fairness (which they could not achieve) for increase in number of transfers completed. Specifically the congestion response by a subset of transfers gives an opportunity for other transfers to grab the available bandwidth at line speed (without congestion back off). However at higher speeds the transfer times of rate control are as good as ECN.

## 6 Summary and Conclusions

TCP throughput is known to be a function which is inversely proportional to the round trip time, the timeout delays and the square root of loss probability [16]. Variance in any of these component factors introduces variance in TCP throughput (and goodput) resulting in unfairness. Variance in more than one factor has a multiplicative effect on variance in throughput.

Packet drops (esp. burst drops) have a multiplier effect especially in configurations with heterogeneous RTTs. RED and ECN cannot control this variance even though ECN does not incur packet drops (in the best case). Appendices A.2 and A.3 show more instances of effect of packet loss and delays due to implicit control. The explicit TCP rate control which controls the window edges of TCP transparently and shapes the ack rate provides best possible fairness, to a great extent removing the dependency of throughput on factors such as drop probability, RTT and timeouts. We observed that fairness in low speed configurations can lead to a reduction in the number of transfers completed during a small observation window, even though links are optimally utilized.

While the simulations exhaustively explored several dimensions using a simple configuration template, it does not explore the effects of remote bottlenecks and does not implement special enhancements in TCP rate control for slow speed links. We also do not explore effects of variable bandwidth bottlenecks. TCP rate control and RED offer the fastest route to immediate deployment. But the minimal state requirements and protocol transparency of RED allow it a greater available space of deployment scenarios (core as well as edge routers).

## References


[1] Hari Balakrishnan, Venkata Padmanabhan, Srini Seshan, Mark Stemm and Randy H. Katz, "TCP Behavior of a Busy Internet Server: Analysis and Improvements," *Proceedings of IEEE Infocom*, San Francisco, CA, USA, March 1998. http://www.cs.berkeley.edu/ hari/papers/infocom98.ps.gz

[2] J. Bolot and A. Vega-Garcia, "Control Mechanisms for packet audio in the Internet," *Proceedings of IEEE Infocom'96,* 1996.

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

[4] Anna Charny "An Algorithm for Rate Allocation in a Packet-Switching Network with feedback", Masters thesis. MIT 1994

[5] A. Demers, S. Keshav, and S. Shenker, "Analysis and Simulation of a Fair Queueing Algorithm," *Internetworking: Research and Experience*, Vol. 1, 1990, pp. 3-26.

[6] W. Feng, D. Kandlur, D. Saha, K. Shin, "Techniques for Eliminating Packet Loss in Congested TCP/IP Networks," *U. Michigan CSE-TR-349-97,* November 1997.

[7] S. Floyd, and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," *IEEE/ACM Transactions on Networking*, Vol. 1, No. 4, August 1993, pp.397-413.

[8] Sally Floyd, "TCP and Explicit Congestion Notification", *ACM Computer Communication Review*, Vol. 24, No. 5, October 1994, pp. 10-23. ftp://ftp.ee.lbl.gov/papers/tcp_ecn.4.ps.Z

[9] V. Jacobson, "Congestion Avoidance and Control," *Proceedings of the SIGCOMM'88 Symposium*, pp. 314-32, August 1988.

[10] Raj Jain, "The Art of Computer Systems Performance Analysis," *John Wiley & Sons Inc.,* 1991.

[11] S. Kalyanaraman, "Traffic Management for the Available Bit Rate (ABR) Service in Asynchronous Transfer Mode (ATM) networks" *Ph.D. Dissertation*, Dept. of Computer and Information Sciences, The Ohio State University, August 1997.

[12] Vijay Kumar, T.V. Lakshman, D. Stiliadis, "Beyond Best Effort: Router Architectures for the differentiated services of tomorrow's Internet," *IEEE Communications Magazine*, Vol. 36, No. 5, May 1998, pp. 152-164.

[13] Dong Lin and Robert Morris, "Dynamics of Random Early Detection," *Proceedings of SIGCOMM'97*, August 1997.

[14] M. Mathis, J. Mahdavi, S. Floyd, A. Romanow, "TCP Selective Acknowledgement Options," *Internet RFC 2018*, October 1996.

[15] Packeteer Inc., White papers on TCP rate control, http://www.packeteer.com/tcprate/

[16] Jitendra Padhye, Victor Firoiu, Don Towsley, and Jim Kurose, "Modeling TCP Throughput: A Simple Model and its Empirical Validation," *Proceedings of SIGCOMM'98*, Vancouver, August 1998.

[17] Vern Paxson, Sally Floyd, " 'Why We Don't Know How To Simulate The Internet," *Proceedings of the 1997 Winter Simulation Conference*, December 1997. ftp://ftp.ee.lbl.gov/papers/wsc97.ps

[18] K.K. Ramakrishnan and R. Jain, "A Binary Feedback Scheme for Congestion Avoidance in Computer Networks with Connectionless Network Layer," *Proceedings of SIGCOMM'88*, August 1988, pp. 303-313.

[19] K.K. Ramakrishnan, S. Floyd, "A proposal to add Explicit Congestion Notification (ECN) to IPv6 and to TCP," *IETF Internet Draft*, November 1997, Available as http://ds.internic.net/internet-drafts/draft-kksjf-ecn-00.txt

[20] Ramakrishna Satyavolu, Ketan Duvedi, Shivkumar Kalyanaraman, "Explicit rate control of TCP applications," *Submitted at IWQoS'99,* 1999. http://www.ecse.rpi.edu/Homepages/shivkuma/research/papers/iwqos99-rate.ps

[21] W. Richard Stevens, "TCP/IP Illustrated, V. 1," *Addison-Wesley, Reading, MA*, 1995.

[22] Stevens, W. R., "TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms," *Internet RFC 2001*, January 1997.

# A Effect of Other Parameter Dimensions

The next few sections summarize the effect of other parameter dimensions on the performance of these three schemes. Specifically, we look at the effect of:

- Staggered Flow Start Times
- Round Trip Times (for homogeneous RTT cases)
- Number of Flows

In all these cases the results are comparable to the corresponding results in sections 5.1.1, 5.1.2, 5.1.3 respectively. The interesting results are in the sections A.2 (effect of RTTs) and A.3 (effect of number of flows) where the subsections describing RED results indicate more negative effects due to packet drops.

### A.1 Dimension: Staggered Flow Start Times

We examine staggered flow start times in our experiments because, in reality, flows do not start together. In our experiments, the flows were grouped into four sets and the starting time of the flows in each set was offset by 250 milliseconds. The whole array of link speeds and end-to-end delays were considered for these simulations.

Our primary observation is that RED performance improves significantly due to staggered flow start times, whereas the performance of ECN or TCP rate control does not change appreciably. This effect is because staggering introduces some interleaving among flows, and reduces the average overload seen at the bottleneck during congestion resulting in more benign drop behavior in RED. We also see interesting examples where a larger number of packets dropped could result in better overall performance.

#### A.1.1 RED (Staggered Flow Start Times)

Table 10 lists observations in a sample subset of simulations involving RED. The entries in this table can be compared with those in Table 1 (section 5.1.1). In comparison, though the number of packets dropped (column 4) increases in the staggered flows case (Table 10), we find that performance in terms of all other metrics improved considerably (specifically average goodput, variance in goodput, utilization and queue metrics show improvement). This is an interesting example where a larger number of packets dropped could result in better overall performance.

#### A.1.2 ECN (Staggered Flow Start Times)

Table 11 lists observations in a sample subset of simulations involving RED. The entries in this table can be compared with those in Table 2 (section 5.1.2). Since no packets are dropped in either case, the only significant effect of staggering is a slight increase in unfairness (columns 6 and 7) and in queue length metrics (column 3). But the positive effect is the slight increase in average goodput (column 5). So, flow staggering does not significantly affect ECN performance.

| | *Provider Metrics* | | | *User Metrics* | | |
|---|---|---|---|---|---|---|
| **Link Speed** Mbps | **Link Util** Percent | $\mu$ **Q/Max Q** Pkts/Pkts | **Dropped** Pkts | $\mu$ **Goodpt** Mbps | $\sigma$ **Goodpt** Mbps | **Covariance** |
| 0.028 | 95.87 | 28.91/53 | 453 | 0.0014 | 0.0008 | 0.609 |
| 0.056 | 97.65 | 31.03/55 | 508 | 0.0019 | 0.0009 | 0.458 |
| 0.128 | 98.77 | 30.70/53 | 568 | 0.0026 | 0.0021 | 0.824 |
| 0.256 | 99.29 | 28.14/51 | 597 | 0.0041 | 0.0029 | 0.711 |
| 0.384 | 99.49 | 28.02/52 | 619 | 0.0054 | 0.0038 | 0.708 |
| 1.5 | 99.76 | 27.06/51 | 1166 | 0.0168 | 0.0101 | 0.601 |
| 10 | 99.67 | 20.52/59 | 3012 | 0.1022 | 0.0338 | 0.331 |
| 45 | 98.76 | 14.14/53 | 4944 | 0.4319 | 0.0906 | 0.210 |
| 150 | 96.34 | 10.11/47 | 4942 | 1.3461 | 0.2176 | 0.162 |

Table 10: Staggered Start; RED; 100 sources; Homogeneous RTT; One-way distance = 3000 Km

| | *Provider Metrics* | | | *User Metrics* | | |
|---|---|---|---|---|---|---|
| **Link Speed** Mbps | **Link Util** Percent | $\mu$ **Q/Max Q** Pkts/Pkts | **Dropped** Pkts | $\mu$ **Goodpt** Mbps | $\sigma$ **Goodpt** Mbps | **Covariance** |
| 0.028 | 95.96 | 254.14/493 | 0 | 0.0014 | 0.0008 | 0.544 |
| 0.056 | 96.80 | 273.63/510 | 0 | 0.0019 | 0.0009 | 0.469 |
| 0.128 | 97.88 | 308.12/563 | 0 | 0.0031 | 0.0009 | 0.307 |
| 0.256 | 98.80 | 312.05/528 | 0 | 0.0037 | 0.0013 | 0.352 |
| 0.384 | 99.16 | 328.08/542 | 0 | 0.0048 | 0.0020 | 0.415 |
| 1.5 | 99.76 | 267.50/354 | 0 | 0.0162 | 0.0032 | 0.197 |
| 10 | 99.96 | 235.08/280 | 0 | 0.1055 | 0.0201 | 0.190 |
| 45 | 99.94 | 144.98/179 | 0 | 0.4630 | 0.0891 | 0.192 |
| 150 | 96.55 | 36.64/122 | 0 | 1.4396 | 0.1135 | 0.079 |

Table 11: Staggered Start; ECN; 100 sources; Homogeneous RTT; One-way distance = 3000 Km

#### A.1.3 TCP Rate Control (Staggered Flow Start Times)

Table 12 lists observations in a sample subset of simulations involving RED. The entries in this table can be compared with those in Table 3 (section 5.1.3). The 150 Mbps case was invalidated due to simulation issues. But overall, the effect of staggering is minimal (like with ECN). There is slight increase in unfairness (columns 6 and 7) and in queue length metrics (column 3). But the positive effect is the slight increase in average goodput (column 5). So, flow staggering does not significantly affect TCP Rate performance. The similarity between TCP rate control and ECN in this case suggests that staggering of sources may not have much effect on explicit congestion indication schemes, but may improve the performance of implicit drop-based schemes.

### A.2 Dimension: Effect of Round trip times

This section looks at performance when the one-way distance is reduced to 3 km (from 3000km). When the round trip time (RTT) is smaller, TCP gets acks quicker, and in turn increases its window

| | *Provider Metrics* | | | *User Metrics* | | |
|---|---|---|---|---|---|---|
| **Link Speed** Mbps | **Link Util** Percent | $\mu$ **Q/Max Q** Pkts/Pkts | **Dropped** Pkts | $\mu$ **Goodpt** Mbps | $\sigma$ **Goodpt** Mbps | **Covariance** |
| 0.028 | 95.96 | 254.14/493 | 0 | 0.0014 | 0.0008 | 0.544 |
| 0.056 | 96.80 | 272.08/507 | 0 | 0.0019 | 0.0009 | 0.453 |
| 0.128 | 97.88 | 296.40/531 | 0 | 0.0029 | 0.0006 | 0.206 |
| 0.256 | 98.80 | 283.89/470 | 0 | 0.0035 | 0.0010 | 0.284 |
| 0.384 | 99.16 | 289.10/485 | 0 | 0.0047 | 0.0008 | 0.175 |
| 1.5 | 99.76 | 185.58/208 | 0 | 0.0167 | 0.0018 | 0.105 |
| 10 | 99.96 | 154.34/164 | 0 | 0.1050 | 0.0047 | 0.045 |
| 45 | 99.74 | 32.31/73 | 0 | 0.4591 | 0.0128 | 0.028 |

Table 12: Staggered Start; Rate controller; 100 sources; Homogeneous RTT; One-way distance = 3000 Km

faster. This results in more severe overload conditions at the bottleneck resulting in possible burst dropping of packets. Response to congestion is also quicker, except in cases when there is a timeout and the effect of the timeout dominates the response time.

As expected our primary observation is that RED performance deteriorates in this situation especially when the bottleneck bandwidth is larger (leading to faster queue growth). To explore the dimension carefully, we also examine the effect of number of flows in this small RTT configuration. With RED, we observe an interesting dichotomy: the performance improves with smaller number of flows at higher speeds, but degrades with smaller number of flows at lower speeds. The former effect is because of the influence of delay and overhead costs of packet drop when we have high bandwidth bottleneck shared by a large number of flows (as observed in section 2). In a lower speed bottleneck, having more flows means that during a congestion event, not all flows are affected and the aggregate goodput is high. Also, during different congestion events, different flows are affected, leading to improved overall fairness.

As opposed to RED, ECN and TCP rate control performance improves in these situations (compared to larger RTT cases) because the effect of packet drops is absent, and the faster window increases results in greater aggregate throughput.

### A.2.1 RED (LAN Round Trip Time)

Table 13 lists observations in a sample subset of simulations involving RED. The entries in this table can be compared with those in Table 1 (section 5.1.1). We observe that the user metrics (average goodput, standard deviation in goodput and covariance) degrade sharply at higher speeds, whereas the provider metrics are virtually unchanged. In particular, observe that in the last row of the table, the aggregate average throughput is only 0.6852 Mbps *100 sources = 68.52 Mbps out of a maximum possible of 150 Mbps !

The explanation for this behavior is that with smaller RTTs, we have faster window increases, which lead to more episodes of burstiness at the bottleneck. RED does not allocate drops well across flows during burstiness in input traffic, especially when both the number of flows and the

| | *Provider Metrics* | | | *User Metrics* | | |
|---|---|---|---|---|---|---|
| **Link Speed** Mbps | **Link Util** Percent | $\mu$ **Q/Max Q** Pkts/Pkts | **Dropped** Pkts | $\mu$ **Goodpt** Mbps | $\sigma$ **Goodpt** Mbps | **Covariance** |
| 0.028 | 93.75 | 30.75/61 | 439 | 0.0008 | 0.0000 | 0.000 |
| 0.056 | 92.67 | 31.08/62 | 407 | 0.0010 | 0.0005 | 0.500 |
| 0.128 | 98.31 | 31.00/63 | 540 | 0.0021 | 0.0013 | 0.635 |
| 0.256 | 99.32 | 29.15/62 | 538 | 0.0034 | 0.0029 | 0.843 |
| 0.384 | 99.54 | 29.02/63 | 637 | 0.0048 | 0.0034 | 0.711 |
| 1.5 | 99.88 | 24.70/64 | 1077 | 0.0145 | 0.0149 | 1.028 |
| 10 | 99.77 | 16.41/64 | 2009 | 0.0787 | 0.1729 | 2.196 |
| 45 | 96.26 | 13.40/65 | 2493 | 0.1919 | 0.5232 | 2.727 |
| 150 | **95.87** | 9.10/64 | 3482 | **0.6852** | **2.0595** | **3.006** |

Table 13: Simultaneous start: RED; 100 sources; Homogeneous RTT; One-way distance = 3 Km

bottleneck bandwidth are high. This is a classic case of scalability problems with implicit drop-based congestion control. The TCP flows incur delay cost and high overhead cost due to timeouts and retransmission. As a result, the high utilization (for example in the last row) does not translate into high average goodput.

Given this poor performance of RED with 100 sources, we examine the same case with 10 sources where we expect the overhead cost to be reduced, but the delay costs (due to timeouts) to still persist. We find that the aggregate average goodput at higher speeds is significantly better compared to the above table (upto 45 Mbps), but the delay costs limit aggregate average goodput for the 150 Mbps case (last row). The number of packets dropped are significantly lower (at least by a factor of 2), but the queueing delays are almost the same indicating that RED has remarkable control over queueing delay. An interesting dichotomy however is that higher aggregate goodput is achieved in the 100 flows case (above table) for lower link speeds (all the way from 28 Kbps to 384 Kbps, with the crossover occuring at 1.5 Mbps). This indicates that finer-grained multiplexing (sharing) of low speed links due to an increased number of flows leads to higher aggregate goodput (though at the expense of a significant number of packet losses).

| | *Provider Metrics* | | | *User Metrics* | | |
|---|---|---|---|---|---|---|
| **Link Speed** Mbps | **Link Util** Percent | $\mu$ **Q/Max Q** Pkts/Pkts | **Dropped** Pkts | $\mu$ **Goodpt** Mbps | $\sigma$ **Goodpt** Mbps | **Covariance** |
| 0.028 | 97.37 | 27.86/45 | 12 | 0.0042 | 0.0011 | 0.270 |
| 0.056 | 98.61 | 28.07/48 | 50 | 0.0063 | 0.0012 | 0.184 |
| 0.128 | 99.38 | 18.63/46 | 59 | 0.0160 | 0.0045 | 0.282 |
| 0.256 | 99.37 | 19.56/44 | 92 | 0.0264 | 0.0071 | 0.269 |
| 0.384 | 99.58 | 18.27/45 | 111 | 0.0392 | 0.0104 | 0.265 |
| 1.5 | 99.76 | 17.37/47 | 289 | 0.1486 | 0.0326 | 0.220 |
| 10 | 99.76 | 14.69/46 | 891 | 0.9701 | 0.1767 | 0.182 |
| 45 | 99.81 | 11.90/46 | 1570 | 4.0189 | 0.8579 | 0.213 |
| 150 | 93.11 | 9.23/47 | 1923 | 9.5236 | 5.8298 | 0.612 |

Table 14: Simultaneous start: RED; 10 sources; Homogeneous RTT; One-way distance = 3 Km

### A.2.2 ECN (LAN Round Trip Time)

Tables 15 and 16 list observations with ECN in a LAN setting, with 100 and 10 sources respectively. We immediately observe that ECN does not suffer from any of the loss-related costs and loss-induced scalability problems of RED. In fact it leverages the faster response times/faster window increases in a LAN to achieve even better aggregate average goodput and utilization compared to the WAN cases (Table 2 in section 5.1.2). These results again validate our hypothesis that loss-based congestion control faces significant scalability issues (when the bandwidth and number of sources increase).

| | *Provider Metrics* | | | *User Metrics* | | |
|---|---|---|---|---|---|---|
| **Link Speed** Mbps | **Link Util** Percent | **$\mu$ Q/Max Q** Pkts/Pkts | **Dropped** Pkts | **$\mu$ Goodpt** Mbps | **$\sigma$ Goodpt** Mbps | **Covariance** |
| 0.028 | 98.25 | 208.95/442 | 0 | 0.0013 | 0.0008 | 0.573 |
| 0.056 | 98.90 | 233.37/488 | 0 | 0.0019 | 0.0008 | 0.421 |
| 0.128 | 99.44 | 275.31/543 | 0 | 0.0027 | 0.0006 | 0.212 |
| 0.256 | 99.70 | 293.29/504 | 0 | 0.0034 | 0.0008 | 0.235 |
| 0.384 | 99.80 | 328.49/559 | 0 | 0.0044 | 0.0008 | 0.187 |
| 1.5 | 99.95 | 243.40/300 | 0 | 0.0168 | 0.0007 | 0.042 |
| 10 | 99.99 | 264.28/300 | 0 | 0.1019 | 0.0008 | 0.007 |
| 45 | 100.00 | 267.00/300 | 0 | 0.4525 | 0.0008 | 0.002 |
| 150 | 100.00 | 265.90/298 | 0 | 1.5028 | 0.0010 | 0.001 |

Table 15: Simultaneous start: ECN; 100 sources; Homogeneous RTT; One-way distance = 3 Km

| | *Provider Metrics* | | | *User Metrics* | | |
|---|---|---|---|---|---|---|
| **Link Speed** Mbps | **Link Util** Percent | **$\mu$ Q/Max Q** Pkts/Pkts | **Dropped** Pkts | **$\mu$ Goodpt** Mbps | **$\sigma$ Goodpt** Mbps | **Covariance** |
| 0.028 | 97.37 | 31.46/57 | 0 | 0.0042 | 0.0011 | 0.270 |
| 0.056 | 98.61 | 41.36/70 | 0 | 0.0063 | 0.0012 | 0.184 |
| 0.128 | 99.38 | 63.47/111 | 0 | 0.0151 | 0.0027 | 0.181 |
| 0.256 | 99.68 | 67.47/90 | 0 | 0.0227 | 0.0034 | 0.150 |
| 0.384 | 99.79 | 40.57/70 | 0 | 0.0406 | 0.0064 | 0.158 |
| 1.5 | 99.95 | 36.01/73 | 0 | 0.1530 | 0.0201 | 0.131 |
| 10 | 99.99 | 37.05/75 | 0 | 1.0032 | 0.0021 | 0.002 |
| 45 | 100.00 | 35.64/74 | 0 | 4.5040 | 0.4028 | 0.089 |
| 150 | 100.00 | 34.67/77 | 0 | 15.0065 | 0.0054 | 0.000 |

Table 16: Simultaneous start: ECN; 10 sources; Homogeneous RTT; One-way distance = 3 Km

### A.2.3 TCP Rate Control (LAN Round Trip Time)

Tables 17 and 18 list observations with TCP Rate control in a LAN setting, with 100 and 10 sources respectively. Our observations are identical to those made in the last section with ECN. Like ECN, TCP Rate Control does not suffer from any of the loss-related costs and loss-induced scalability problems of RED. In fact it leverages the faster response times/faster window increases in a LAN to achieve even better aggregate average goodput and utilization compared to the WAN cases (Table 3 in section 5.1.3).

| | *Provider Metrics* | | | *User Metrics* | | |
|---|---|---|---|---|---|---|
| **Link Speed** Mbps | **Link Util** Percent | **$\mu$ Q/Max Q** Pkts/Pkts | **Dropped** Pkts | **$\mu$ Goodpt** Mbps | **$\sigma$ Goodpt** Mbps | **Covariance** |
| 0.028 | 98.25 | 206.87/436 | 0 | 0.0013 | 0.0007 | 0.576 |
| 0.056 | 98.90 | 231.12/482 | 0 | 0.0018 | 0.0008 | 0.434 |
| 0.128 | 99.44 | 271.36/536 | 0 | 0.0027 | 0.0006 | 0.224 |
| 0.256 | 99.70 | 284.76/488 | 0 | 0.0034 | 0.0008 | 0.239 |
| 0.384 | 99.80 | 320.55/538 | 0 | 0.0043 | 0.0006 | 0.138 |
| 1.5 | 99.95 | 184.09/200 | 0 | 0.0165 | 0.0007 | 0.041 |
| 10 | 99.99 | 197.40/200 | 0 | 0.1016 | 0.0008 | 0.008 |
| 45 | 100.00 | 197.63/199 | 0 | 0.4517 | 0.0006 | 0.001 |
| 150 | 100.00 | 195.35/197 | 0 | 1.5019 | 0.0009 | 0.001 |

Table 17: Simultaneous start: Rate controller; 100 sources; Homogeneous RTT; One-way distance = 3 Km

| | *Provider Metrics* | | | *User Metrics* | | |
|---|---|---|---|---|---|---|
| **Link Speed** Mbps | **Link Util** Percent | **$\mu$ Q/Max Q** Pkts/Pkts | **Dropped** Pkts | **$\mu$ Goodpt** Mbps | **$\sigma$ Goodpt** Mbps | **Covariance** |
| 0.028 | 97.30 | 27.73/46 | 0 | 0.0033 | 0.0006 | 0.194 |
| 0.056 | 98.59 | 34.54/60 | 0 | 0.0048 | 0.0007 | 0.141 |
| 0.128 | 99.38 | 17.19/22 | 0 | 0.0139 | 0.0004 | 0.026 |
| 0.256 | 99.68 | 17.20/20 | 0 | 0.0270 | 0.0009 | 0.032 |
| 0.384 | 99.79 | 18.60/20 | 0 | 0.0399 | 0.0007 | 0.018 |
| 1.5 | 99.95 | 18.91/20 | 0 | 0.1516 | 0.0000 | 0.000 |
| 10 | 99.99 | 18.98/20 | 0 | 1.0017 | 0.0007 | 0.000 |
| 45 | 100.00 | 18.00/19 | 0 | 4.5025 | 0.0008 | 0.000 |
| 150 | 100.00 | 15.38/17 | 0 | 15.0049 | 0.0337 | 0.002 |

Table 18: Simultaneous start: Rate controller; 10 sources; Homogeneous RTT; One-way distance = 3 Km

### A.3 Dimension: Number of Flows

In this final appendix we examine the effect of number of flows (in addition to observations made in appendix A.2). Specifically, revert to the 3000 km one-way distance configuration, but reduce the number of active flows to 10 (from 100).

In the previous section, we noted that the costs of packet drop in terms of delay and overhead were readily apparent in a small RTT configuration. In this section, we observe that the use of larger RTTs result in TCPs increasing their load (and spreading their windows) over a longer time cycle. As a result the instantaneous bottleneck overloads and rate of queue increases are less dramatic. However, the aggregate goodput in high speed bottleneck cases in RED is low in both cases (10 or

100 flows) because either the delay costs (in the 10 flows case) or the overhead costs (in the 100 flows case) of packet loss dominate.

The effect of smaller number of flows is also a sharply reduced packet drop rate (in RED) or queue lengths (in ECN and TCP rate control). Another interesting phenomenon we noticed with ECN is reduced aggregate goodput at higher speeds because of congestion control delays (not packet loss-induced delays) because the TCP rate is not explicitly controlled.

### A.3.1 RED (10 flows, WAN RTT)

Our primary observation in this case is that RED sees an order of magnitude lesser drops, but the other metrics are virtually unchanged (except for a slight reduction in aggregate goodput in RED) compared to the 100 flows case of table 1 (section 5.1.1).

| | *Provider Metrics* | | | *User Metrics* | | |
|---|---|---|---|---|---|---|
| **Link Speed** Mbps | **Link Util** Percent | $\mu$ **Q/Max Q** Pkts/Pkts | **Dropped** Pkts | $\mu$ **Goodpt** Mbps | $\sigma$ **Goodpt** Mbps | **Covariance** |
| 0.028 | 97.37 | 28.08/45 | 10 | 0.0040 | 0.0011 | 0.281 |
| 0.056 | 98.61 | 27.66/44 | 48 | 0.0061 | 0.0012 | 0.171 |
| 0.128 | 99.25 | 20.31/47 | 72 | 0.0157 | 0.0049 | 0.312 |
| 0.256 | 99.55 | 18.81/45 | 94 | 0.0264 | 0.0101 | 0.384 |
| 0.384 | 99.21 | 18.01/46 | 118 | 0.0394 | 0.0171 | 0.434 |
| 1.5 | 99.51 | 16.63/45 | 258 | 0.1482 | 0.0287 | 0.194 |
| 10 | 97.66 | 10.55/44 | 454 | 0.8897 | 0.1518 | 0.171 |
| 45 | 97.41 | 5.58/46 | 403 | 3.5012 | 0.2856 | 0.082 |
| 150 | 95.42 | 4.59/46 | 246 | 10.5019 | 1.5019 | 0.143 |

Table 19: Simultaneous start: RED; 10 sources; Homogeneous RTT; One-way distance = 3000 Km

The non-decrease in TCP goodput is attributed to the fact that TCP rate increases are spread over longer time-scales (i.e. longer RTTs) in WAN configuration. The aggregate goodput in both cases is significantly lower than the maximum possible (105 Mbps and 114 Mbps compared to 150 Mbps maximum possible, even though utilization is very high in both cases).

### A.3.2 ECN (10 flows, WAN RTT)

In ECN, we do not see packet-loss related effects seen in RED. However, a significant observation in Table 20 compared to the 100 flows case of table 2 (section 5.1.2) is that the aggregate goodput with 100 flows (previous table) is significantly higher, especially in higher speed configurations. This interesting phenomenon is due to the effect of delays caused by the implicit TCP congestion control policy (i.e. cutting window by half and doing linear increase) as opposed to explicitly setting the window to the optimum value.

The queue lengths are also significantly higher in the 100 flows configuration (compared to an increased drop rate in RED in a similar situation).

| | *Provider Metrics* | | | *User Metrics* | | |
|---|---|---|---|---|---|---|
| **Link Speed** Mbps | **Link Util** Percent | **$\mu$ Q/Max Q** Pkts/Pkts | **Dropped** Pkts | **$\mu$ Goodpt** Mbps | **$\sigma$ Goodpt** Mbps | **Covariance** |
| 0.028 | 97.37 | 30.95/55 | 0 | 0.0040 | 0.0011 | 0.281 |
| 0.056 | 98.61 | 40.91/68 | 0 | 0.0061 | 0.0012 | 0.200 |
| 0.128 | 99.25 | 63.90/107 | 0 | 0.0150 | 0.0028 | 0.188 |
| 0.256 | 99.55 | 69.00/91 | 0 | 0.0229 | 0.0036 | 0.158 |
| 0.384 | 99.67 | 44.60/76 | 0 | 0.0389 | 0.0043 | 0.109 |
| 1.5 | 99.90 | 33.17/71 | 0 | 0.1525 | 0.0055 | 0.036 |
| 10 | 99.97 | 18.11/97 | 0 | 0.9980 | 0.0371 | 0.037 |
| 45 | 97.55 | 7.08/71 | 0 | 3.8608 | 0.1688 | 0.044 |
| 150 | 95.21 | 4.92/76 | 0 | 11.2914 | 0.8976 | 0.079 |

Table 20: Simultaneous start: ECN; 10 sources; Homogeneous RTT; One-way distance = 3000 Km

### A.3.3 TCP Rate Control (10 flows, WAN RTT)

Table 21 shows TCP rate control's performance with 10 flows in a WAN setting which may be compared to the ECN tables of the previous section and the TCP rate control (100 flows) case of table 3 (section 5.1.3). TCP rate control like ECN does not suffer from packet-drop related effects (which are seen in RED). But, unlike ECN, it does not suffer from the congestion control induced delays of TCP (window reduction/linear increase) because it explicitly controls the size of the active TCP window.

| | *Provider Metrics* | | | *User Metrics* | | |
|---|---|---|---|---|---|---|
| **Link Speed** Mbps | **Link Util** Percent | **$\mu$ Q/Max Q** Pkts/Pkts | **Dropped** Pkts | **$\mu$ Goodpt** Mbps | **$\sigma$ Goodpt** Mbps | **Covariance** |
| 0.028 | 97.30 | 27.83/48 | 0 | 0.0033 | 0.0006 | 0.194 |
| 0.056 | 98.59 | 33.82/59 | 0 | 0.0048 | 0.0008 | 0.160 |
| 0.128 | 99.25 | 20.62/27 | 0 | 0.0120 | 0.0013 | 0.111 |
| 0.256 | 99.55 | 17.26/20 | 0 | 0.0270 | 0.0009 | 0.031 |
| 0.384 | 99.67 | 17.59/20 | 0 | 0.0398 | 0.0008 | 0.019 |
| 1.5 | 99.90 | 13.94/20 | 0 | 0.1511 | 0.0008 | 0.005 |
| 10 | 98.95 | 2.48/19 | 0 | 0.9756 | 0.0426 | 0.044 |
| 45 | 99.69 | 3.01/114 | 0 | 4.3794 | 0.0127 | 0.003 |
| 150 | 99.90 | 6.56/201 | 0 | 14.5222 | 0.0259 | 0.002 |

Table 21: Simultaneous start: Rate controller; 10 sources; Homogeneous RTT; One-way distance = 3000 Km