Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 27 April 2023

G. Fairhurst
University of Aberdeen

24 October 2022

# Guidelines for Internet Congestion Control at Endpoints
## draft-fairhurst-tsvwg-cc-07

## Abstract

When published as an RFC, this document provides guidance on the design of methods to avoid congestion collapse and how an endpoint needs to react to incipient congestion. The IETF provides recommendations and requirements on this topic that is distributed across many documents in the RFC series. This document therefore gathers and consolidates these recommendations. Based on these, and Internet engineering experience, the document provides best current practice for the design of new congestion control methods in Internet protocols.

When published, the document will update or replace the Best Current Practice in BCP 41, which currently includes "Congestion Control Principles" provided in RFC2914.

Table of Contents

# 1. Introduction

The IETF has specified Internet transports (e.g., TCP [RFC9293], UDP [RFC0768], UDP-Lite [RFC3828], SCTP [RFC4960], and DCCP [RFC4340]) as well as protocols layered on top of these transports (e.g., RTP [RFC3550], QUIC [RFC9000] [RFC9002], SCTP/UDP [RFC6951], DCCP/UDP [RFC6773]) and transports that work directly over the IP network layer. These transports are implemented in endpoints (either Internet hosts or routers acting as endpoints), and are designed to detect and react to network congestion. TCP was the first transport to provide this, although the TCP specifications found in RFC 793 predates the inclusion of congestion control and did not contain any discussion of using or managing a congestion window. RFC 9293 [RFC9293] seek to address this.

Internet transports need to react to avoid congestion that impacts other flows sharing a path. The Requirements for Internet Hosts [RFC1122] formally mandates that endpoints perform congestion control. "Because congestion control is critical to the stable operation of the Internet, applications and other protocols that choose to use UDP as an Internet transport must employ mechanisms to prevent congestion collapse and to establish some degree of fairness with concurrent flows [RFC2914].

The popularity of the Internet has led to a proliferation in the number of TCP implementations [RFC2914]. A variety of non-TCP transports have also being deployed. Some transport implementations fail to use standardised congestion avoidance mechanisms correctly because of poor implementation [RFC2525]. However, this is not the only reason for not using standard methods. Some transports have chosen mechanisms that are not presently standardised, or have adopted approaches to their design that differ from present standards. Guidance is needed therefore not only for future standardisation, but to ensure safe and appropriate evolution of transports that have not presently been submitted for standardisation.

Experience has shown that successful protocols developed in a specific context or for a particular application tend to also become used in a wider range of contexts. Therefore, IETF specifications by default target deployment on the general Internet, or need to be defined for use only within a controlled environment.

## 1.1. Incipient and Persistent Congestion

Paths through the Internet can experience congestion (loss or delay) that is a result of excess load at a bottleneck(s) along the path. Two levels of congestion are differentiated in this guidance:

* Incipient congestion is a consequential side effect of the statistical multiplexing of packet flows. There will be time where packets need to be buffered or dropped at the bottleneck(s) on the path, and flows need to react when they encounter this congestion to reduce their contribution to the load.

* Persistent congestion occurs when the pattern of arriving traffic results in over consumption of the path resources. Typically this results in packet loss. The effects of persistent congestion might impact the flow that induces congestion, but could also impact other flows, e.g., starving them of resources; or further reducing the efficiency of the path (e.g., congestion collapse).

### 1.2. Avoiding the effects of Persistent Congestion

Early RFCs recognised that a significant pathology can arise when a poorly designed transport creates significant congestion. This can result in severe service degradation or "Internet meltdown". This phenomenon was first observed during the early growth phase of the Internet in the mid 1980s [RFC0896] [RFC0970]. It is technically called "Congestion Collapse". [RFC2914] notes that informally, "congestion collapse occurs when an increase in the network load results in a decrease in the useful work done by the network." The problem of congetsion collapse was largely due to TCP connections unnecessarily retransmitting packets that were either in transit or had already been received at the receiver. This is a stable condition that can result in throughput that is a small fraction of normal [RFC0896].

A second form of congestion collapse occurs due to undelivered packets, where Section 5 of [RFC2914] notes: "Congestion collapse from undelivered packets arises when bandwidth is wasted by delivering packets through the network that are dropped before reaching their ultimate destination. Different scenarios can result in different degrees of congestion collapse, in terms of the fraction of the congested links' bandwidth used for productive work. The danger of congestion collapse from undelivered packets is due primarily to the increasing deployment of open-loop applications not using end-to-end congestion control. Even more destructive would be best-effort applications that increase their sending rate in response to an increased packet drop rate (e.g., automatically using an increased level of FEC (Forward Error Correction))."

The problems of congestion collapse have generally been corrected by improvements to timer and congestion control mechanisms, that were implemented in modern implementations of TCP [Jac88]. . Transports need to be specifically designed with measures to avoid starving other flows of capacity (e.g., [RFC7567]). Section 3 discusses

Fairness, stating "The equitable sharing of bandwidth among flows depends on the fact that all flows are running compatible congestion control algorithms". Section 3.1 describes preventing congestion collapse. [RFC2309] also discussed the dangers of congestion-unresponsive flows, and states that "all UDP-based streaming applications should incorporate effective congestion avoidance mechanisms." [RFC7567] and [RFC8085] both reaffirm this, encouraging development of methods to prevent starvation.

### 1.3. Mitigating the effects of Incipient Congestion

Incipient congestion can also result in normal operation of the Internet. Buffering (an increase in latency) or congestion loss (discard of a packet) arises when the traffic arriving at a link or network exceeds the resources available. Loss can also occur for other reasons, but it is usually not possible for an endpoint to reliably disambiguate the cause of packet loss (e.g., loss could be due to link corruption, receiver overrun, etc. [RFC3819]). A network device typically uses a drop-tail policy to drop excess IP packets when its queue(s) becomes full. This use of buffers can also be managed using Active Queue Management (AQM) [RFC7567], which can be combined with Explicit Congestion Notification (ECN) signalling.

Network devices can be configured to isolate the queuing of packets for different flows, or aggregates of flows, and thereby assist in reducing the impact of flow multiplexing on other flows (e.g., flow scheduling and AQM [RFC7567]). This could include methods seeking to equally distribute resources between sharing flows, but this is explicitly not a requirement for a network device [Flow-Rate-Fairness]. Endpoints can not rely on the presence and correct configuration of these methods, and therefore even when a path is expected to support such methods, also need to employ methods that work end-to-end.

In some case, Internet transports can also reserve capacity at routers or on the links/paths being used. This can assist CC in controlled environments, but most uses across an Internet path are unable to rely upon prior reservation of capacity along the path they use. In the absence of such a reservation, endpoints are unable to determine a safe rate at which to start or continue their transmission. The use of an Internet path therefore requires a combination of end-to-end transport mechanisms to detect and then respond to changes in the capacity that it discovers is available across the network path.

Section 3.3 of [RFC2914] notes: "In addition to the prevention of congestion collapse and concerns about fairness, a third reason for a flow to use end-to-end congestion control can be to optimize its own

performance regarding throughput, delay, and loss. In some circumstances, for example in environments with high statistical multiplexing, the delay and loss rate experienced by a flow are largely independent of its own sending rate. However, in environments with lower levels of statistical multiplexing or with per-flow scheduling, the delay and loss rate experienced by a flow is in part a function of the flow's own sending rate. Thus, a flow can use end-to-end congestion control to limit the delay or loss experienced by its own packets. We would note, however, that in an environment like the current best-effort Internet, concerns regarding congestion collapse and fairness with competing flows limit the range of congestion control behaviors available to a flow."

### 1.4. Current Challenges

Recommendations and requirements on congestion control are distributed across many documents in the RFC series. This document therefore gathers and consolidates these recommendations. These, and Internet engineering experience are used as a basis for the best current practice in the design of congestion control methods for Internet protocols.

The standardization of congestion control in new transports can avoid a congestion control "arms race" among competing protocols [RFC2914]. That is, avoid designs of transports that could compete for Internet resource in a way that significantly reduces the ability of other flows to use the Internet.

The general recommendation in the UDP Guidelines [RFC8085] is that applications SHOULD leverage existing congestion control techniques, such as those defined for TCP [RFC5681], TCP-Friendly Rate Control (TFRC) [RFC5348], SCTP [RFC4960], and other IETF-defined transports. This is because there are many trade offs and details that can have a serious impact on the performance of congestion control for the application they support and other traffic that seeks to share the resources along the path over which they communicate.

There are several reasons to think that things may have changed since the original best current practice was published: At one time, it was common that the serialisation delay of a packet at the bottleneck formed a large proportion of the round time of a path, motivating a need for conservative loss recovery. This is not often the case for today's higher capacity links. This general increase in the link speed often means that for many users, current traffic often does not normally experience persistent congestion.

There also have been changes over time in the way that protocol mechanisms are deployed in Internet endpoints:

On the one hand, techniques have evolved that now allow incremental deployment and testing of new methods. This can enable more rapid development of methods to detect and react to incipient congestion. This allows new mechanisms can be tested to ensure that 95%, 99%, etc of users see benefit in the networks they use. there has been considerable progress in developing new loss recovery and congestion responses that have been evaluated in this way.

On the other hand, the Internet continues to be heterogenous, some endpoints experience very different network path characteristics and some endpoints generate very different patterns of traffic. The IETF seeks to avoid congestion collapse, and also avoid prejudicing the performance (e.g., throughput, latency) experienced when the Internet is shared. The equitable or reasonable share of the bottleneck capacity is often judged using a fairness metric.

The focus of the present document is upon unicast point-to-point transports, this includes migration from using one path to another path. Some recommendations [RFC5783] and requirements in this document apply to point-to-multipoint transports (e.g., multicast), however this topic extends beyond the current document's scope. [RFC2914] provides additional guidance on the use of multicast.

## 2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

The path between endpoints (sometimes called "Internet Hosts" for IPv4 and called "source nodes" and "destination nodes" in IPv6) consists of the endpoint protocol stack at the sender and the receiver (which together implement the transport service), and a succession of links and network devices (routers or middleboxes) that provide connectivity across a network path. The set of network devices forming the path is not usually fixed, and it should generally be assumed that this set can change over arbitrary lengths of time.

[RFC5783] defines congestion control as "the feedback-based adjustment of the rate at which data is sent into the network. Congestion control is an indispensable set of principles and mechanisms for maintaining the stability of the Internet." [RFC5783] also provides an informational snapshot taken by the IRTF's Internet Congestion Control Research Group (ICCRG) from October 2008.

The text draws on language used in the specifications of TCP and other IETF transports. For example, a protocol timer is generally needed to detect persistent congestion, and this document uses the term Retransmission Timeout (RTO) to refer to the operation of this timer. Similarly, the document refers to a congestion window (cwnd) as the variable that controls the rate of transmission by the congestion controller. The use of these terms does not imply that endpoints need to implement functions in the way that TCP currently does. Each new transport needs to make its own design decisions about how to meet the recommendations and requirements for congestion control.

Other terminology is directly copied from the cited RFCs.

## 3. Author's Note on Additional Material

This section captures plans for work in progress. Topics not yet considered:

* Updated thinking related to Hystart.

* Updated thinking related to the challenges and merits of management with encryption.

* Update for latest TCPM developments.

* Update for latest QUIC/TSVAREA discussions relating to CC.

* How do operators understand that traffic is behaving reasonably?

* How can the IETF ensure safe (and efficient) congestion control?

This section will be removed in a future version. It will not be a part of the final document.

## 4. Requirements from the RFC Series

### 4.1. The need to React to Congestion

This includes:

* Endpoints MUST perform congestion control [RFC1122] and SHOULD leverage existing congestion control techniques [RFC8085].

* If an application or protocol chooses not to use a congestion-controlled transport protocol, it SHOULD control the rate at which it sends datagrams to a destination host, in order to fulfil the requirements of [RFC2914], as stated in [RFC8085].

* Transports SHOULD control the aggregate traffic they send on a path [RFC8085]. They ought not to use multiple congestion-controlled flows between the same endpoints to gain a performance advantage. An endpoint can become aware of congestion by various means (including, delay variation, timeout, ECN, packet loss). A signal that indicates congestion on the end-to-end network path, SHOULD result in a congestion control reaction by the transport that reduces the current rate of the sending endpoint [RFC8087]).

* Although network devices can be configured to reduce the impact of flow multiplexing on other flows, endpoints MUST NOT rely solely on the presence and correct configuration of these methods, except when they are constrained to operate in a controlled environment.

* A transport that does not target Internet deployment need to be constrained to only operate in a controlled environment (e.g., see Section 3.6 of [RFC8085]) and provide appropriate mechanisms to prevent this traffic from accidentally leaving the controlled environment [RFC8084].

## 4.2. Tolerance to a Diversity of Path Characteristics

* Path Change: The detection of congestion and the resulting reduction MUST NOT solely depend upon reception of a signal from the remote endpoint, because congestion indications could themselves be lost under persistent congestion. The only way to surely confirm that a sending endpoint has successfully communicated with a remote endpoint is to utilise a timer (see Section 5.2.3) to detect a lack of response that could result from a change in the path or the path characteristics (usually called the RTO). Congestion controllers that are unable to react after one (or at most a few) Round Trip Times (RTTs) after receiving a congestion indication should observe the guidance in section 3.3 of the UDP Guidelines [RFC8085].

## 4.3. Robustness: Protection of Protocol Mechanisms

An endpoint needs to provide protection from attacks on the traffic it generates, or attacks that seek to increase the capacity that is consumed (impacting other traffic that share a bottleneck).

The following guidance is provided on protection from attack:

* Off-Path Attack: A design MUST protect from off-path attack to the protocol [RFC8085] (i.e., an attack where the attacker is unable to observe the packets exchanged across the path). Such an attack on the congestion control can lead to a Denial of Service (DoS) vulnerability for the flow being controlled and/or other flows that share network resources along the path.

* On-Path Attack: A protocol can be designed to protect from on-path attacks (i.e., where the attacker can observe the packets exchanged across the path). Protecting from on path attacks can require more complexity and typically utilises encryption and/or authentication mechanisms (e.g., IPsec [RFC4301], QUIC [RFC9000]).

* Validation of Signals: Network signals and control messages (e.g., ICMP [RFC0792]) MUST be validated before they are used to protect from malicious abuse. This MUST at least include protection from off-path attack [RFC8085].

### 4.4. Current IETF Guidelines on Evaluation of Congestion Control

Congestion control is an evolving subject, responding to changes in protocol design, operation of applications using the network and understanding of the network operation under load. The IETF has provided guidance [RFC5033] for considering and evaluating alternate congestion control algorithms.

The IRTF has described a set of metrics and related trade-off between metrics that can be used to compare, contrast, and evaluate congestion control techniques [RFC5166]. [RFC5783] provides a snapshot of congestion-control research in 2008.

In contrast to fairness, a different approach is needed to analyse persistent congestion effects (the collateral impact on loss, starvation, collapse, etc). Such an analysis of the suitability of a new mechanism needs to consider the impact on the flows that have outliers in performance, (e.g., the last 5%, 1%) and specifically needs to understand how changes impact other flows sharing a bottleneck. For example, the flow performance often does not provide any indication that a new method could starve other applications that share the bottleneck capacity, or when patterns of packets (e.g., bursts) are sent that disrupt the packet timing needed by another application flow.

# 5. Principles of Congestion Control

This section summarises the principles for providing congestion control. The section seeks to differentiate mechanisms associated with preventing persistent congestion; reacting to incipient congestion and utilising additional path information.

## 5.1. Preventing Persistent Congestion

Principles include:

* Persistent congestion can result in congestion collapse, which MUST be aggressively avoided [RFC2914]. Endpoints that experience persistent congestion and have already exponentially reduced their congestion window to the restart window (e.g., one packet), MUST further reduce the rate if the RTO timer continues to expire. For example, TFRC [RFC5348] continues to reduce its sending rate under persistent congestion to one packet per RTT, and then exponentially backs-off the time between single packet transmissions if a congestion event continues to persist [RFC2914]. QUIC [RFC9002] does not directly specify a period, but does specify a probe to detect tail loss. The Tail Loss Probe (TLP) mechanism [RFC8985] determines that persisent congestion is experienced after a loss for a duration of 2 TLP probes plus the RTO.

### 5.1.1. Avoiding Congestion Collapse and Flow Starvation

Principles include:

* Transports MUST avoid inducing flow starvation to the other flows that share resources along the path they use.

* Endpoints MUST treat a loss of all feedback (e.g., expiry of a retransmission time out, RTO) as an indication of persistent congestion (i.e., an indication of potential congestion collapse).

* When an endpoint detects persistent congestion, it MUST reduce the maximum rate (e.g., reduce its congestion window). This normally involves the use of protocol timers to detect a lack of acknowledgment for transmitted data (Section 5.2.3).

* Network devices MAY provide mechanisms to mitigate the impact of congestion collapse by transport flows (e.g., priority forwarding of control information, and starvation detection), and SHOULD mitigate the impact of non-conforment and malicious flows [RFC7567]). These mechanisms complement, but do not replace, the endpoint congestion avoidance mechanisms.

#### 5.1.2. Robustness: Timers and Retransmission

A transport MUST adjust its timers to ensure exponential backoff each time persistent congestion is detected [RFC1122], until the path characteristics can again be confirmed.

Principles include:

* Protocol timers (e.g., for retransmission or to detect persistent congestion) need to be appropriately initialised.

* Maintaining the RTO: The RTO interval SHOULD be set based on recent RTT observations (including the RTT variance) (e.g., Section 3.1.1 of [RFC8085]).

* Measuring the Path RTT: Once an endpoint has started communicating with its peer, the RTT MUST be adjusted by measuring the actual path RTT. This adjustment MUST include adapting to the measured RTT variance (see equation 2.3 of [RFC6928]).

* RTO Expiry: Persistent lack of feedback (e.g., detected by an RTO timer expiry, or other means) MUST be treated as an indication of persistent congestion. A failure to receive any specific response within an RTO interval could potentially be a result of a RTT change, change of path, excessive loss, or even congestion collapse. If there is no response within the RTO interval, TCP collapses the congestion window to one segment [RFC5681]. Other transports MUST similarly respond when they detect loss of feedback. An endpoint needs to exponentially backoff the RTO interval [RFC8085] each time the RTO expires. That is, the RTO interval MUST be set to at least the RTO * 2 [RFC6298] [RFC8085].

* Maximum RTO:A maximum value MAY be placed on the RTO interval. This maximum limit to the RTO interval MUST NOT be less than 60 seconds [RFC6298].

* [[ Author Note: Check RTO-Consider. Note that this is the RTO backoff, not the recovery timer.]]

### 5.2. Reacting to Incipient Congestion

#### 5.2.1. Congestion Initialization

When a connection or flow to a new destination is first established, the endpoints have little information about the characteristics of the network path they will use.

* Flow Start: A new flow between two endpoints needs to initialise a congestion controller for the path it will use. It MUST NOT assume that capacity is available at the start of the flow, unless it uses a mechanism to explicitly reserve capacity. In the absence of a capacity signal, a flow can be expected to start slowly. The TCP slow-start algorithm is an accepted standard for flow startup [RFC5681], which uses the notion of an Initial Window (IW) [RFC3390], updated by [RFC6928]) to define the initial volume of data that can be sent on a path. This is not the smallest burst, or the smallest window, but it is considered a safe starting point for a path that is not suffering persistent congestion, and is applicable until feedback about the path is received. The initial sending rate needs to be viewed as tentative, until capacity is confirmed to be available.

* Cached State: A congestion controller MAY assume that the recently used capacity between a pair of endpoints is an indication of future capacity that might be available in the next RTT between the same endpoints. The congestion controller MUST reduce its rate if this is not subsequently confirmed to be true. [[Author note: we likely need to bound this reaction in time or size]].

* Initial RTO Interval: When a flow sends the first packet(s), it typically has no way to know the actual RTT of the path it will use. An initial value needs to be used to initialise the principal retransmission timer, which will be used to detect lack of responsiveness from the remote endpoint. In TCP, this is the starting value of the RTO. The selection of a safe initial value is a trade off that has important consequences on the overall Internet stability [RFC6928] [RFC8085]. In the absence of any knowledge about the latency of a path (including the initial value), the RTO MUST be conservatively set to no less than 1 second. Values shorter than 1 second can be problematic (see the appendix of [RFC6298]). (Note: Linux TCP has deployed a smaller initial RTO value).

* Initial RTO Expiry: If the RTO timer expires while awaiting completion of a connection setup, or handshake (e.g., the ACK of a SYN segment in the three-way handshake in TCP), and the implementation is using an RTO of less than 3 seconds, the local endpoint can resend the connection setup. [[Author note: It would be useful to discuss how the timer is managed to protect from multiple handshake failure]]. This RTO MUST then be re-initialized to increase it to 3 seconds when data transmission begins (i.e., after the handshake completes) [RFC6298] [RFC8085]. This conservative increase is necessary to avoid congestion collapse when many flows retransmit across a shared bottleneck with restricted capacity.

* Initial Measured RTO: Once an RTT measurement is available (e.g., through reception of an acknowledgement), the timeout value must be adjusted. This adjustment MUST take into account the RTT variance. For the first sample, this variance cannot be determined, and a local endpoint MUST therefore initialise the variance to RTT/2 (see equation 2.2 of [RFC6928] and related text for UDP in section 3.1.1 of [RFC8085]).

#### 5.2.2. Using Path Capacity

This section describes how a sender needs to regulate the maximum volume of data in flight over the interval of the current RTT, and how it manages the use of the capacity that it perceives is available, and reacts to incipient congestion.

* Transient Paths: Unless managed by a resource reservation protocol, path capacity information is transient. A sender that does not use capacity has no understanding whether previously used capacity remains available to use, or whether that capacity has disappeared (e.g., a change in the path that causes a flow to experience a smaller bottleneck, or when more traffic emerges that consumes previously available capacity resulting in a new bottleneck). For this reason, a transport that is limited by the volume of data available to send MUST NOT continue to grow its congestion window when the current congestion window is more than twice the volume of data acknowledged in the last RTT.

* Validating the congestion window: Standard TCP states that a TCP sender "SHOULD set the congestion window to no more than the Restart Window (R)" before beginning transmission, if the sender has not sent data in an interval that exceeds the current retransmission timeout, i.e., when an application becomes idle [RFC5681]. An experimental specification [RFC7661] permits TCP senders to tentatively maintain a congestion window that is larger than the path has supported in the last RTT when it is application-limited, provided that the endpoint appropriately and rapidly reduces the congestion window when potential congestion is detected. This mechanism is called Congestion Window Validation (CWV).

* Collateral Damage: Even in the absence of congestion, statistical multiplexing of flows can result in transient effects for flows sharing common resources. A sender therefore SHOULD avoid inducing excessive congestion to other flows (collateral damage that could result in flow starvation).

* Burst Mitigation: While a congestion controller ought to limit sending at the granularity of the current RTT, this can be insufficient to satisfy the goals of mitigating collateral damage. This requires moderating the burst rate of the sender to avoid significant periods where a flow(s) consume all buffer capacity at the path bottleneck, which would otherwise prevent other flows from gaining a reasonable share. Endpoints SHOULD provide mechanisms to regulate the bursts of transmission that the application/protocol sends to the network (section 3.1.6 of [RFC8085]). ACK-Clocking [RFC5681] can help mitigate bursts for protocols that receive continuous feedback of reception (such as TCP). Sender pacing can also mitigate this [RFC8085], (described in Section 4.6 of [RFC3449]), and has been recommended for TCP in conditions where ACK-Clocking is not effective, (e.g., [RFC3742], [RFC7661]). SCTP [RFC4960] defines a maximum burst length (Max.Burst) with a recommended value of 4 segments to limit the SCTP burst size.

#### 5.2.3. Loss Detection and Retransmission

This section describes mechanisms to detect and provide retransmission, and to protect the network in the absence of timely feedback.

* Loss Detection: Loss detection occurs after a sender determines there is no delivery confirmation within an expected period of time (e.g., by observing the time-ordering of the reception of ACKs, as in TCP DupACK) or by utilising a timer to detect loss (e.g., a transmission timer with a period less than the RTO, [RFC8085] [RFC8985]) or a combination of using a timer and ordering information to trigger retransmission of data.

* Retransmission: Retransmission of lost packets or messages is a common reliability mechanism. When loss is detected, the sender can choose to retransmit the lost data, ignore the loss, or send other data (e.g., [RFC8085] [RFC9002]), depending on the reliability model provided by the transport service. Any transmission consumes network capacity, therefore retransmissions MUST NOT increase the network load in response to congestion loss (which worsens that congestion) [RFC8085]. Any method that sends additional data following loss is therefore responsible for congestion control of the retransmissions (and any other packets sent, including FEC information) as well as the original traffic.

#### 5.2.4. Responding to Incipient Congestion

The safety and responsiveness of new proposals need to be evaluated [RFC5166]. In determining an appropriate congestion response to incipient congestion, designs could take into consideration the size of the packets that experience congestion [RFC4828].

* Congestion Response: An endpoint MUST promptly reduce the rate of transmission when it receive or detects an indication of congestion (e.g., loss) [RFC2914]. TCP Reno established a method that relies on multiplicative-decrease to halve the sending rate while congestion is detected. This response to congestion indications is considered sufficient for safe Internet operation, but other decrease factors have also been published in the RFC Series [I-D.ietf-tcpm-rfc8312bis].

* ECN Response: A congestion control design should provide the necessary mechanisms to support ECN [RFC3168] [RFC6679], as described in section 3.1.7 of [RFC8085]. ECN can help determine an appropriate congestion window to enable early indication of incipient congestion when it is supported by routers on the path [RFC7567]. An early detection of incipient congestion allows a different reaction to an explicit congestion signal compared to the reaction to a detected packet loss [RFC8311] [RFC8087]. Simple feedback of received Congestion Experienced (CE) marks [RFC3168], relies only on an indication that congestion has been experienced within the last RTT. This style of response is appropriate when a flow uses ECT(0) [RFC3168]. ABE included a modification to the reaction to ECN [RFC8511]. Further detail about the received CE-marking can be obtained by using more accurate receiver feedback (e.g., [I-D.ietf-tcpm-accurate-ecn] and extended RTP feedback). The more detailed feedback provides an opportunity for a finer-granularity of congestion response. The L4S architecture [I-D.ietf-tsvwg-l4s-arch] defines a change to the reaction for packets marked with ECT(1), building on the feedback provided by [I-D.ietf-tcpm-accurate-ecn] and a modified marking system that can provide early reaction to incipient congestion [I-D.ietf-tsvwg-aqm-dualq-coupled].

* [RFC8085] provides guidelines for a sender that does not, or is unable to, adapt the congestion window.

#### 5.2.5. Using More Capacity

In the phase where a sender is increasing the congestion window, it will transmit faster than the last confirmed safe rate. Such an increase above the last confirmed rate needs to be regarded as tentative and a sender needs to reduce its rate below the last confirmed safe rate when congestion is detected.

* In the absence of congestion, an endpoint MAY increase its congestion window and hence the sending rate. An increase should only occur when there is additional data available to send across the path (i.e., the sender will utilise the additional capacity in the next RTT). This helps manage incipient congestion.

* Increasing Congestion Window: A sender MUST NOT increase its rate for more than one RTT after congestion is detected.

* After detecting congestion: An endpoint MUST utilise a method that assures the sender will keep the rate below the previously confirmed safe rate for multiple RTT periods after an observed congestion event. In TCP, this is performed by using a linear increase from a slow start threshold that is re-initialised when congestion is experienced.

* Avoiding Overshoot: Overshoot of the congestion window beyond the point of congestion can significantly impact other flows sharing resources along a path, and can impact the performance of the flow itself. As endpoints experience more paths with a large Bandwidth Delay Product (BDP) and a wider range of potential path RTT, variability or changes in the path can significantly impact the appropriate dynamics for increasing a congestion window (see also burst mitigation, Section 5.2.2). Methods such as HyStart are designed to avoid overshoot [I-D.ietf-tcpm-hystartplusplus].

#### 5.2.6. Utilising Additional Path Information

An endpoint is permitted to cache path information. This could be used to inform parameter selection for a new or on-going flow. An endpoint might also utilise signals from the network to help determine how to regulate the traffic it sends.

Any information used to accelerate the growth of the congestion window MUST be viewed as tentative until the path capacity is confirmed by receiving a confirmation that actual traffic has been sent across the path. (i.e., the new flow needs to either use or loose the capacity that has been tentatively offered to it). A sender MUST reduce its rate if this capacity is not confirmed within the current RTO interval.

* Utilising Cached Path Information: A congestion controller that recently used a specific path could allow a flow to take-over the capacity that was previously consumed by another flow (e.g., in the last RTT) which it understands is using the same path and no will longer use the capacity it recently used. In TCP, this mechanism was called TCP Control Block (TCB) sharing and is now called TCP Control Block Interdependence, and is described in [RFC9040]. The capacity and other information can be used to suggest a faster initial sending rate.

* Receiving Network Signals: Mechanisms MUST NOT solely rely on transport messages or specific signalling messages to perform safely. (Section 5.2 of [RFC8085] describes use of ICMP messages). Mechanisms need to be designed to safely operate when path characteristics can change at any time. Transport mechanisms MUST be robust to potential loss of any signals. Loss or modification of packets can occur after a path changes, even when a signal was successfully first used by a flow, see Section 4.2).

* Utilising Network Signals: A mechanism that utilises signals originating in the network (e.g., RSVP, NSIS, Quick-Start, ECN), MUST assume that the set of network devices on the path can change. This motivates a design that uses soft-state for protocols that interact with signals originating from network devices [RFC9049] (e.g., ECN) and includes context-sensitive treatment of "soft" signals provided to the endpoint [RFC5164].

## 6. Acknowledgements

This document owes much to the insight offered by Sally Floyd, both at the time of writing of RFC2914 and her help and review in the many years that followed this.

Nicholas Kuhn helped develop the first draft of these guidelines. Tom Jones and Ana Custura reviewed the first version of this draft. The University of Aberdeen has received funding to support this work from the European Space Agency.

## 7. IANA Considerations

This memo includes no request to IANA.

RFC Editor Note: If there are no requirements for IANA, the section will be removed during conversion into an RFC by the RFC Editor.

## 8. Security Considerations

This document introduces no new security considerations. Each RFC listed in this document discusses the security considerations of the specification it contains. The security considerations for the use of transports are provided in the references section of the cited RFCs. Security guidance for applications using UDP is provided in the UDP Usage Guidelines [RFC8085].

Section 4.3 describes general requirements relating to the design of safe protocols and their protection from on and off path attack.

Section 5.2.6 follows current best practice to validate ICMP messages prior to use.

## 9. Normative References

[RFC1122] Braden, R., Ed., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, DOI 10.17487/RFC1122, October 1989, <https://www.rfc-editor.org/info/rfc1122>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC2914] Floyd, S., "Congestion Control Principles", BCP 41, RFC 2914, DOI 10.17487/RFC2914, September 2000, <https://www.rfc-editor.org/info/rfc2914>.

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, DOI 10.17487/RFC3168, September 2001, <https://www.rfc-editor.org/info/rfc3168>.

[RFC3390] Allman, M., Floyd, S., and C. Partridge, "Increasing TCP's Initial Window", RFC 3390, DOI 10.17487/RFC3390, October 2002, <https://www.rfc-editor.org/info/rfc3390>.

[RFC5348] Floyd, S., Handley, M., Padhye, J., and J. Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification", RFC 5348, DOI 10.17487/RFC5348, September 2008, <https://www.rfc-editor.org/info/rfc5348>.

[RFC5681] Allman, M., Paxson, V., and E. Blanton, "TCP Congestion Control", RFC 5681, DOI 10.17487/RFC5681, September 2009, <https://www.rfc-editor.org/info/rfc5681>.

[RFC6298] Paxson, V., Allman, M., Chu, J., and M. Sargent, "Computing TCP's Retransmission Timer", RFC 6298, DOI 10.17487/RFC6298, June 2011, <https://www.rfc-editor.org/info/rfc6298>.

[RFC7567] Baker, F., Ed. and G. Fairhurst, Ed., "IETF Recommendations Regarding Active Queue Management", BCP 197, RFC 7567, DOI 10.17487/RFC7567, July 2015, <https://www.rfc-editor.org/info/rfc7567>.

[RFC8085] Eggert, L., Fairhurst, G., and G. Shepherd, "UDP Usage Guidelines", BCP 145, RFC 8085, DOI 10.17487/RFC8085, March 2017, <https://www.rfc-editor.org/info/rfc8085>.

## 10. Informative References

[Flow-Rate-Fairness]
Briscoe, Bob., "Flow Rate Fairness: Dismantling a Religion, ACM Computer Communication Review 37(2):63-74", April 2007.

[I-D.ietf-tcpm-accurate-ecn]
Briscoe, B., Kühlewind, M., and R. Scheffenegger, "More Accurate ECN Feedback in TCP", Work in Progress, Internet-Draft, draft-ietf-tcpm-accurate-ecn-20, 25 July 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-tcpm-accurate-ecn-20>.

[I-D.ietf-tcpm-hystartplusplus]
Balasubramanian, P., Huang, Y., and M. Olson, "HyStart++: Modified Slow Start for TCP", Work in Progress, Internet-Draft, draft-ietf-tcpm-hystartplusplus-10, 3 October 2022, <https://www.ietf.org/archive/id/draft-ietf-tcpm-hystartplusplus-10.txt>.

[I-D.ietf-tcpm-rfc8312bis]
Xu, L., Ha, S., Rhee, I., Goel, V., and L. Eggert, "CUBIC for Fast and Long-Distance Networks", Work in Progress, Internet-Draft, draft-ietf-tcpm-rfc8312bis-13, 12 October 2022, <https://www.ietf.org/archive/id/draft-ietf-tcpm-rfc8312bis-13.txt>.

[I-D.ietf-tsvwg-aqm-dualq-coupled]
Schepper, K. D., Briscoe, B., and G. White, "DualQ Coupled AQMs for Low Latency, Low Loss and Scalable Throughput (L4S)", Work in Progress, Internet-Draft, draft-ietf-tsvwg-aqm-dualq-coupled-25, 29 August 2022, <https://www.ietf.org/archive/id/draft-ietf-tsvwg-aqm-dualq-coupled-25.txt>.

[I-D.ietf-tsvwg-l4s-arch]
Briscoe, B., Schepper, K. D., Bagnulo, M., and G. White, "Low Latency, Low Loss, Scalable Throughput (L4S) Internet Service: Architecture", Work in Progress, Internet-Draft, draft-ietf-tsvwg-l4s-arch-20, 29 August 2022, <https://www.ietf.org/archive/id/draft-ietf-tsvwg-l4s-arch-20.txt>.

[I-D.nishida-tcpm-standard-cc-analysis]
Nishida, Y., "Analysis for the Differences Between Standard Congestion Control Schemes", Work in Progress, Internet-Draft, draft-nishida-tcpm-standard-cc-analysis-00, 19 October 2022, <https://www.ietf.org/archive/id/draft-nishida-tcpm-standard-cc-analysis-00.txt>.

[Jac88] Jacobson, V., "Congestion Avoidance and Control", Computer Communication Review, vol. 18, no. 4, pp. 314-329 , August 1988, <ftp://ftp.ee.lbl.gov/papers/congavoid.ps.Z.>.

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, DOI 10.17487/RFC0768, August 1980, <https://www.rfc-editor.org/info/rfc768>.

[RFC0792] Postel, J., "Internet Control Message Protocol", STD 5, RFC 792, DOI 10.17487/RFC0792, September 1981, <https://www.rfc-editor.org/info/rfc792>.

[RFC0896] Nagle, J., "Congestion Control in IP/TCP Internetworks", RFC 896, DOI 10.17487/RFC0896, January 1984, <https://www.rfc-editor.org/info/rfc896>.

[RFC0970] Nagle, J., "On Packet Switches With Infinite Storage", RFC 970, DOI 10.17487/RFC0970, December 1985, <https://www.rfc-editor.org/info/rfc970>.

[RFC2309] Braden, B., Clark, D., Crowcroft, J., Davie, B., Deering, S., Estrin, D., Floyd, S., Jacobson, V., Minshall, G., Partridge, C., Peterson, L., Ramakrishnan, K., Shenker, S., Wroclawski, J., and L. Zhang, "Recommendations on Queue Management and Congestion Avoidance in the Internet", RFC 2309, DOI 10.17487/RFC2309, April 1998, <https://www.rfc-editor.org/info/rfc2309>.

[RFC2525] Paxson, V., Allman, M., Dawson, S., Fenner, W., Griner, J., Heavens, I., Lahey, K., Semke, J., and B. Volz, "Known TCP Implementation Problems", RFC 2525, DOI 10.17487/RFC2525, March 1999, <https://www.rfc-editor.org/info/rfc2525>.

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, DOI 10.17487/RFC2616, June 1999, <https://www.rfc-editor.org/info/rfc2616>.

[RFC3449] Balakrishnan, H., Padmanabhan, V., Fairhurst, G., and M. Sooriyabandara, "TCP Performance Implications of Network Path Asymmetry", BCP 69, RFC 3449, DOI 10.17487/RFC3449, December 2002, <https://www.rfc-editor.org/info/rfc3449>.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <https://www.rfc-editor.org/info/rfc3550>.

[RFC3742] Floyd, S., "Limited Slow-Start for TCP with Large Congestion Windows", RFC 3742, DOI 10.17487/RFC3742, March 2004, <https://www.rfc-editor.org/info/rfc3742>.

[RFC3819] Karn, P., Ed., Bormann, C., Fairhurst, G., Grossman, D., Ludwig, R., Mahdavi, J., Montenegro, G., Touch, J., and L. Wood, "Advice for Internet Subnetwork Designers", BCP 89, RFC 3819, DOI 10.17487/RFC3819, July 2004, <https://www.rfc-editor.org/info/rfc3819>.

[RFC3828] L-Larzon, A., Degermark, M., Pink, S., L-Jonsson, E., Ed., and G. Fairhurst, Ed., "The Lightweight User Datagram Protocol (UDP-Lite)", RFC 3828, DOI 10.17487/RFC3828, July 2004, <https://www.rfc-editor.org/info/rfc3828>.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <https://www.rfc-editor.org/info/rfc4301>.

[RFC4340] Kohler, E., Handley, M., and S. Floyd, "Datagram Congestion Control Protocol (DCCP)", RFC 4340, DOI 10.17487/RFC4340, March 2006, <https://www.rfc-editor.org/info/rfc4340>.

[RFC4828] Floyd, S. and E. Kohler, "TCP Friendly Rate Control (TFRC): The Small-Packet (SP) Variant", RFC 4828, DOI 10.17487/RFC4828, April 2007, <https://www.rfc-editor.org/info/rfc4828>.

[RFC4960] Stewart, R., Ed., "Stream Control Transmission Protocol", RFC 4960, DOI 10.17487/RFC4960, September 2007, <https://www.rfc-editor.org/info/rfc4960>.

[RFC5033] Floyd, S. and M. Allman, "Specifying New Congestion Control Algorithms", BCP 133, RFC 5033, DOI 10.17487/RFC5033, August 2007, <https://www.rfc-editor.org/info/rfc5033>.

[RFC5164] Melia, T., Ed., "Mobility Services Transport: Problem Statement", RFC 5164, DOI 10.17487/RFC5164, March 2008, <https://www.rfc-editor.org/info/rfc5164>.

[RFC5166] Floyd, S., Ed., "Metrics for the Evaluation of Congestion Control Mechanisms", RFC 5166, DOI 10.17487/RFC5166, March 2008, <https://www.rfc-editor.org/info/rfc5166>.

[RFC5783] Welzl, M. and W. Eddy, "Congestion Control in the RFC Series", RFC 5783, DOI 10.17487/RFC5783, February 2010, <https://www.rfc-editor.org/info/rfc5783>.

[RFC6363] Watson, M., Begen, A., and V. Roca, "Forward Error Correction (FEC) Framework", RFC 6363, DOI 10.17487/RFC6363, October 2011, <https://www.rfc-editor.org/info/rfc6363>.

[RFC6679] Westerlund, M., Johansson, I., Perkins, C., O'Hanlon, P., and K. Carlberg, "Explicit Congestion Notification (ECN) for RTP over UDP", RFC 6679, DOI 10.17487/RFC6679, August 2012, <https://www.rfc-editor.org/info/rfc6679>.

[RFC6773] Phelan, T., Fairhurst, G., and C. Perkins, "DCCP-UDP: A Datagram Congestion Control Protocol UDP Encapsulation for NAT Traversal", RFC 6773, DOI 10.17487/RFC6773, November 2012, <https://www.rfc-editor.org/info/rfc6773>.

[RFC6928] Chu, J., Dukkipati, N., Cheng, Y., and M. Mathis, "Increasing TCP's Initial Window", RFC 6928, DOI 10.17487/RFC6928, April 2013, <https://www.rfc-editor.org/info/rfc6928>.

[RFC6951] Tuexen, M. and R. Stewart, "UDP Encapsulation of Stream Control Transmission Protocol (SCTP) Packets for End-Host to End-Host Communication", RFC 6951, DOI 10.17487/RFC6951, May 2013, <https://www.rfc-editor.org/info/rfc6951>.

[RFC7661] Fairhurst, G., Sathiaseelan, A., and R. Secchi, "Updating TCP to Support Rate-Limited Traffic", RFC 7661, DOI 10.17487/RFC7661, October 2015, <https://www.rfc-editor.org/info/rfc7661>.

[RFC8084] Fairhurst, G., "Network Transport Circuit Breakers", BCP 208, RFC 8084, DOI 10.17487/RFC8084, March 2017, <https://www.rfc-editor.org/info/rfc8084>.

[RFC8087] Fairhurst, G. and M. Welzl, "The Benefits of Using Explicit Congestion Notification (ECN)", RFC 8087, DOI 10.17487/RFC8087, March 2017, <https://www.rfc-editor.org/info/rfc8087>.

[RFC8311] Black, D., "Relaxing Restrictions on Explicit Congestion Notification (ECN) Experimentation", RFC 8311, DOI 10.17487/RFC8311, January 2018, <https://www.rfc-editor.org/info/rfc8311>.

[RFC8511] Khademi, N., Welzl, M., Armitage, G., and G. Fairhurst, "TCP Alternative Backoff with ECN (ABE)", RFC 8511, DOI 10.17487/RFC8511, December 2018, <https://www.rfc-editor.org/info/rfc8511>.

[RFC8985] Cheng, Y., Cardwell, N., Dukkipati, N., and P. Jha, "The RACK-TLP Loss Detection Algorithm for TCP", RFC 8985, DOI 10.17487/RFC8985, February 2021, <https://www.rfc-editor.org/info/rfc8985>.

[RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/info/rfc9000>.

[RFC9002] Iyengar, J., Ed. and I. Swett, Ed., "QUIC Loss Detection and Congestion Control", RFC 9002, DOI 10.17487/RFC9002, May 2021, <https://www.rfc-editor.org/info/rfc9002>.

[RFC9040] Touch, J., Welzl, M., and S. Islam, "TCP Control Block Interdependence", RFC 9040, DOI 10.17487/RFC9040, July 2021, <https://www.rfc-editor.org/info/rfc9040>.

[RFC9049] Dawkins, S., Ed., "Path Aware Networking: Obstacles to Deployment (A Bestiary of Roads Not Taken)", RFC 9049, DOI 10.17487/RFC9049, June 2021, <https://www.rfc-editor.org/info/rfc9049>.

[RFC9293] Eddy, W., Ed., "Transmission Control Protocol (TCP)", STD 7, RFC 9293, DOI 10.17487/RFC9293, August 2022, <https://www.rfc-editor.org/info/rfc9293>.

## Appendix A. Internet Congestion Control

### A.1. Flow Multiplexing and Congestion

When a transport uses a path to send packets (i.e. a flow), this impacts any other Internet flows (possibly from or to other endpoints) that share the capacity of any common network device or link (i.e., are multiplexed) along the path. As with loss, latency can also be incurred for other reasons [RFC3819] (Quality of Service link scheduling, link radio resource management/bandwidth on demand, transient outages, link retransmission, and connection/resource setup below the IP layer, etc).

When choosing an appropriate sending rate, packet loss needs to be considered. Although losses are not always due to congestion, endpoint congestion control needs to conservatively react to loss as a potential signal of reduced available capacity and reduce the sending rate. Many designs place the responsibility of rate-adaption at the sender (source) endpoint, utilising feedback information provided by the remote endpoint (receiver). Congestion control can also be implemented by determining an appropriate rate limit at the receiver and using this limit to control the maximum transport rate (e.g., using methods such as [RFC5348] and [RFC4828]).

It is normal to observe some perturbation in latency and/or loss when flows shares a common network bottleneck with other traffic. This impact needs to be considered and Internet flows ought to implement appropriate safeguards to avoid inappropriate impact on other flows that share the resources along a path. Congestion control methods satisfy this requirement and therefore can help avoid congestion collapse.

"This raises the issue of the appropriate granularity of a 'flow', where we define a 'flow' as the level of granularity appropriate for the application of both fairness and congestion control. [RFC2309]

states: "There are a few 'natural' answers: 1) a TCP or UDP connection (source address/port, destination address/port); 2) a source/destination host pair; 3) a given source host or a given destination host. We would guess that the source/destination host pair gives the most appropriate granularity in many circumstances. The granularity of flows for congestion management is, at least in part, a policy question that needs to be addressed in the wider IETF community." [RFC2914]

Endpoints can send more than one flow. "The specific issue of a browser opening multiple connections to the same destination has been addressed by [RFC2616]. Section 8.1.4 states that "Clients that use persistent connections SHOULD limit the number of simultaneous connections that they maintain to a given server. A single-user client SHOULD NOT maintain more than 2 connections with any server or proxy." [RFC9040].

This suggests that there are opportunities for transport connections between the same endpoints (from the same or differing applications) might share some information, including their congestion control state, if they are known to share the same path. [RFC8085] adds "An application that forks multiple worker processes or otherwise uses multiple sockets to generate UDP datagrams SHOULD perform congestion control over the aggregate traffic."

In the absence of persistent congestion, an endpoint is permitted to increase its congestion window and hence the sending rate. An increase should only occur when there is additional data available to send across the path (i.e., the sender will utilise the additional capacity in the next RTT).

TCP Reno [RFC5681] defines an algorithm, known as the Additive-Increase/ Multiplicative-Decrease (AIMD) algorithm, which allows a sender to exponentially increase the congestion window each RTT from the initial window to the first detected congestion event. This is designed to allow new flows to rapidly acquire a suitable congestion window. Where the bandwidth delay product (BDP) is large, it can take many RTT periods to determine a suitable share of the path capacity. Such high BDP paths benefit from methods that more rapidly increase the congestion window, but in compensation these need to be designed to also react rapidly to any detected congestion (e.g., TCP Cubic [I-D.ietf-tcpm-rfc8312bis]).

### A.2. Adjusting the Rate

* The capacity available to a flow could be expressed as the number of bytes in flight, the sending rate or a limit on the number of unacknowledged segments. When determining the capacity used, all

data sent by a sender needs to be accounted, this includes any additional overhead or data generated by the transport. A transport performing congestion management will usually optimise performance for its application by avoiding excessive loss or delay and maintain a congestion window. In steady-state this congestion window reflects a safe limit to the sending rate that has not resulted in persistent congestion. A congestion controller for a flow that uses packet Forward Error Correction (FEC) encoding (e.g., [RFC6363]) needs to consider all additional overhead introduced by packet FEC when setting and managing its congestion window.

* One common model views the path between two endpoints as a "pipe". New packets enter the pipe at the sending endpoint, older ones leave the pipe at the receiving endpoint. Congestion and other forms of loss result in "leakage" from this pipe. Received data (leaving the network path at the remote endpoint) is usually acknowledged to the congestion controller.

* The rate that data leaves the pipe indicates the share of the capacity that has been utilised by the flow. If, on average (over an RTT), the sending rate equals the receiving rate, this indicates the path capacity. This capacity can be safely used again in the next RTT. If the average receiving rate is less than the sending rate, then the path is either queuing packets, the RTT/path has changed, or there is packet loss.

## Appendix B. Revision Notes

Note to RFC-Editor: please remove this entire section prior to publication.

Individual draft -00:

* Comments and corrections are welcome directly to the authors or via the IETF TSVWG, working group mailing list.

Individual draft -01:

* This update is proposed for initial WG comments.

* If there is interest in progressing this document, the next version will include more complete referencing to cited material.

Individual draft -02:

* Correction of typos.

Individual draft -00:

* Added section 1.1 with text on current BCP status with additional alignment and updates to RFC2914 on Congestion Control Principles (after question from M. Scharf).

* Edits to consolidate starvation text.

* Added text that multicast currently noting that this is out of scope.

* Revised sender-based CC text after comment from C. Perkins (Section 3.1,3.3 and other places).

* Added more about FEC after comment from C. Perkins.

* Added an explicit reference to RFC 5783 and updated this text (after question from M. Scharf).

* To avoid doubt, added a para about "Each new transport needs to make its own design decisions about how to meet the recommendations and requirements for congestion control."

* Updated references.

Individual draft -00:

* Correction of NiTs. Further clarifications.

* This draft does not attempt to address further alignment with draft-ietf-tcpm-rto-consider. This will form part of a future revision.

Individual draft -05:

* Moved intro to appendix and re-issued as a live draft.

* This draft does not attempt to address further alignment with draft-ietf-tcpm-rto-consider. This will form part of a future revision.

Individual draft -06:

* Reformat src for modern XML2RFC.

* Restructured draft around different types of congestion reaction.

Individual draft -07:

* Editorial pass with updated references.

* Restructured draft around different types of congestion reaction.

Author's Address

Godred Fairhurst
University of Aberdeen
School of Engineering
Fraser Noble Building
Aberdeen
AB24 3UE
United Kingdom
Email: gorry@erg.abdn.ac.uk