Transport Area Working Group
Internet-Draft
Intended status: Historic
Expires: January 17, 2014

B. Briscoe, Ed.
A. Jacquet
BT
T. Moncaster
Moncaster.com
A. Smith
BT

July 16, 2013

Re-ECN: A Framework for adding Congestion Accountability to TCP/IP draft-briscoe-conex-re-ecn-motiv-02

Abstract

This document describes a framework for using a new protocol called re-ECN (re-inserted explicit congestion notification), which can be deployed incrementally around unmodified routers. Re-ECN allows accurate congestion monitoring throughout the network thus enabling the upstream party at any trust boundary in the internetwork to be held responsible for the congestion they cause, or allow to be caused. So, networks can introduce straightforward accountability for congestion and policing mechanisms for incoming traffic from end-customers or from neighbouring network domains. As well as giving the motivation for re-ECN this document also gives examples of mechanisms that can use the protocol to ensure data sources respond correctly to congestion. And it describes example mechanisms that ensure the dominant selfish strategy of both network domains and end-points will be to use the protocol honestly.

Note concerning Intended Status: If this draft were ever published as an RFC it would probably have historic status. There is limited space in the IP header, so re-ECN had to compromise by requiring the receiver to be ECN-enabled otherwise the sender could not use re-ECN. Re-ECN was a precursor to chartering of the IETF's Congestion Exposure (ConEx) working group, but during chartering there were still too few ECN receivers enabled, therefore it was decided to pursue other compromises in order to fit a similar capability into the IP header.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-

Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 17, 2014.

Table of Contents

1. Introduction 4
1.1. Motivation 4
1.2. Re-ECN Protocol in Brief 5
1.3. The Re-ECN Framework 6
1.4. Solving Hard Problems 7
1.5. The Rest of this Document 8
2. Requirements notation 8
3. Motivation 9
3.1. Policing Congestion Response 9
3.1.1. The Policing Problem 9
3.1.2. The Case Against Bottleneck Policing 10
4. Re-ECN Incentive Framework 11
4.1. Revealing Congestion Along the Path 11
4.1.1. Positive and Negative Flows 13
4.2. Incentive Framework Overview 13
4.3. Egress Dropper 17
4.4. Ingress Policing 19
4.5. Inter-domain Policing 21
4.6. Inter-domain Fail-safes 24
4.7. The Case against Classic Feedback 25
4.8. Simulations 26
5. Other Applications of Re-ECN 26

5.1. DDoS Mitigation 26
5.2. End-to-end QoS 28
5.3. Traffic Engineering 28
5.4. Inter-Provider Service Monitoring 28
6. Limitations 28
7. Incremental Deployment 29
7.1. Incremental Deployment Features 29
7.2. Incremental Deployment Incentives 30
8. Architectural Rationale 35
9. Related Work 38
9.1. Policing Rate Response to Congestion 38
9.2. Congestion Notification Integrity 38
9.3. Identifying Upstream and Downstream Congestion 39
10. Security Considerations 40
11. IANA Considerations 40
12. Conclusions 40
13. Acknowledgements 40
14. Comments Solicited 40
15. References 40
15.1. Normative References 40
15.2. Informative References 41
Appendix A. Example Egress Dropper Algorithm 43
Appendix B. Policer Designs to ensure Congestion Responsiveness 44
B.1. Per-user Policing 44
B.2. Per-flow Rate Policing 45
Appendix C. Downstream Congestion Metering Algorithms 47
C.1. Bulk Downstream Congestion Metering Algorithm 48
C.2. Inflation Factor for Persistently Negative Flows 48
Appendix D. Re-TTL 49
Appendix E. Argument for holding back the ECN nonce 50

Authors' Statement: Status (to be removed by the RFC Editor)

Although the re-ECN protocol is intended to make a simple but far-reaching change to the Internet architecture, the most immediate priority for the authors is to delay any move of the ECN nonce to Proposed Standard status. The argument for this position is developed in Appendix E.

1. Introduction

This document aims to:

- Describe the motivation for wanting to introduce re-ECN;
- Provide a very brief description of the protocol;
- The framework within which the protocol sits;
- To show how a number of hard problems become much easier to solve once re-ECN is available in IP.

This introduction starts with a run through of these 4 points.

1.1. Motivation

Re-ECN is proposed as a means of allowing accurate monitoring of congestion throughout the Internet. The current Internet relies on the vast majority of end-systems running TCP and reacting to detected congestion by reducing their sending rates. Thus congestion control is conducted by the collaboration of the majority of end-systems.

In this situation it is possible for applications that are unresponsive to congestion to take whatever share of bottleneck resources they want from responsive flows, the responsive flows reduce their sending rate in face of congestion and effectively get out of the way of unresponsive flows. An increasing proportion of such applications could lead to congestion collapse being more common [RFC3714]. Each network has no visibility of whole path congestion and can only respond to congestion on a local basis.

Using re-ECN will allow any point along a path to calculate congestion both upstream and downstream of that point. As a consequence of this policing of congestion /could/ be carried out in the network if end-systems fail to do so. Re-ECN enables flows and users to be policed and for policing to happen at network ingress and at network borders.

1.2. Re-ECN Protocol in Brief

In re-ECN each sender makes a prediction of the congestion that each flow will cause and signals that prediction within the IP headers of that flow. The prediction is based on, but not limited to, feedback received from the receiver. Sending a prediction of the congestion gives network equipment a view of the congestion downstream and upstream.

In order to explain this mechanism we introduce the notion of IP packets carrying different, notional values dependent on the state of their header flags:

- Negative - are those marked by queues when incipient congestion is detected. This is exactly the same as ECN [RFC3168];

- Positive - are sent by the sender in proportion to the number of bytes in packets that have been marked negative according to feedback received from the receiver;

- Cautious - are sent whenever the sender cannot be sure of the correct amount of positive bytes to inject into the network for example, at the start of a flow to indicate that feedback has not been established;

- Cancelled - packets sent by the sender as positive that get marked as negative by queues in the network due to incipient congestion;

- Neutral - normal IP packets but show queues that they can be marked negative.

A flow starts to transmit packets. No feedback has been established so a number of cautious packets are sent (see the protocol definition [Re-TCP] for an analysis of how many cautious packets should be sent at flow start). The rest are sent as neutral.

The packets traverse a congested queue. A fraction are marked negative as an indication of incipient congestion.

The packets are received by the receiver. The receiver feeds back to the sender a count of the number of packets that have been marked negative. This feedback can be provided either by the transport (e.g. TCP) or by higher-layer control messages.

The sender receives the feedback and then sends a number of positive packets in proportion to the bytes represented by packets that have been marked negative. It is important to note that congestion is revealed by the fraction of marked packets rather than a field in the

IP header. This is due to the limited code points available and includes use of the last unallocated bit (sometimes called the evil bit [RFC3514]). Full details of the code points used is given in [Re-TCP]. This lack of codepoints is, however, the case with IPv4. ECN is similarly restricted.

The number of bytes inside the negative packets and positive packets should therefore be approximately equal at the termination point of the flow. To put it another way, the balance of negative and positive should be zero.

1.3. The Re-ECN Framework

The introducion of the protocol enables 3 things:

- Gives a view of whole path congestion;
- Enables policing of flows;
- It allows networks to monitor the flow of congestion across their borders.

At any point in the network a device can calculate the upstream congestion by calculating the fraction of bytes in negative packets to total packets. This it could do using ECN by calculating the fraction of packets marked Congestion Experienced.

Using re-ECN a device in the network can calculate downstream congestion by subtracting the fraction of negative packets from the fraction of positive packets.

A user can be restricted to only causing a certain amount of congestion. A Policer could be introduced at the ingress of a network that counts the number of positive packets being sent and limits the sender if that sender ties to transmit more positive packets than their allowance.

A user could deliberately ignore some or all of the feedback and transmit packets with a zero or much lower proportion of positive packets than negative packets. To solve this a Dropper is proposed. This would be placed at the egress of a network. If the number of negative packets exceeds the number of positive packets then the flow could be dropped or some other sanction enacted.

Policers and droppers could be used between networks in order to police bulk traffic. A whole network harbouring users causing congestion in downstream networks can be held responsible or policed by its downstream neighbour.

1.4. Solving Hard Problems

We have already shown that by making flows declare the level of congestion they are causing that they can be policed, more specifically these are the kind of problems that can be solved:

- mitigating distributed denial of service (DDoS);
- simplifying differentiation of quality of service (QoS);
- policing compliance to congestion control;
- inter-provider service monitoring;
- etc.

Uniquely, re-ECN manages to enable solutions to these problems without unduly stifling innovative new ways to use the Internet. This was a hard balance to strike, given it could be argued that DDoS is an innovative way to use the Internet. The most valuable insight was to allow each network to choose the level of constraint it wishes to impose. Also re-ECN has been carefully designed so that networks that choose to use it conservatively can protect themselves against the congestion caused in their network by users on other networks with more liberal policies.

For instance, some network owners want to block applications like voice and video unless their network is compensated for the extra share of bottleneck bandwidth taken. These real-time applications tend to be unresponsive when congestion arises. Whereas elastic TCP-based applications back away quickly, ending up taking a much smaller share of congested capacity for themselves. Other network owners want to invest in large amounts of capacity and make their gains from simplicity of operation and economies of scale.

While we have designed re-ECN so that networks can choose to deploy stringent policing, this does not imply we advocate that every network should introduce tight controls on those that cause congestion. Re-ECN has been specifically designed to allow different networks to choose how conservative or liberal they wish to be with respect to policing congestion. But those that choose to be conservative can protect themselves from the excesses that liberal networks allow their users.

Re-ECN allows the more conservative networks to police out flows that have not asked to be unresponsive to congestion---not because they are voice or video---just because they don't respond to congestion. But it also allows other networks to choose not to police.

Crucially, when flows from liberal networks cross into a conservative network, re-ECN enables the conservative network to apply penalties to its neighbouring networks for the congestion they allow to be caused. And these penalties can be applied to bulk data, without regard to flows.

Then, if unresponsive applications become so dominant that some of the more liberal networks experience congestion collapse [RFC3714], they can change their minds and use re-ECN to apply tighter controls in order to bring congestion back under control.

Re-ECN reduces the need for complex network equipment to perform these functions.

1.5. The Rest of this Document

This document is structured as follows. First the motivation for the new protocol is given (Section 3) followed by the incentive framework that is possible with the protocol Section 4. Section 5 then describes other important applications re-ECN, such as policing DDoS, QoS and congestion control. Although these applications do not require standardisation themselves, they are described in a fair degree of detail in order to explain how re-ECN can be used. Given re-ECN proposes to use the last undefined bit in the IPv4 header, we felt it necessary to outline the potential that re-ECN could release in return for being given that bit.

Deployment issues discussed throughout the document are brought together in Section 7, which is followed by a brief section explaining the somewhat subtle rationale for the design from an architectural perspective (Section 8). We end by describing related work (Section 9), listing security considerations (Section 10) and finally drawing conclusions (Section 12).

2. Requirements notation

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].

This document first specifies a protocol, then describes a framework that creates the right incentives to ensure compliance to the protocol. This could cause confusion because the second part of the document considers many cases where malicious nodes may not comply with the protocol. When such contingencies are described, if any of the above keywords are not capitalised, that is deliberate. So, for instance, the following two apparently contradictory sentences would be perfectly consistent: i) x MUST do this; ii) x may not do this.

3. Motivation

3.1. Policing Congestion Response

3.1.1. The Policing Problem

The current Internet architecture trusts hosts to respond voluntarily to congestion. Limited evidence shows that the large majority of end-points on the Internet comply with a TCP-friendly response to congestion. But telephony (and increasingly video) services over the best effort Internet are attracting the interest of major commercial operations. Most of these applications do not respond to congestion at all. Those that can switch to lower rate codecs.

Of course, the Internet is intended to support many different application behaviours. But the problem is that this freedom can be exercised irresponsibly. The greater problem is that we will never be able to agree on where the boundary is between responsible and irresponsible. Therefore re-ECN is designed to allow different networks to set their own view of the limit to irresponsibility, and to allow networks that choose a more conservative limit to push back against congestion caused in more liberal networks.

As an example of the impossibility of setting a standard for fairness, mandating TCP-friendliness would set the bar too high for unresponsive streaming media, but still some would say the bar was too low [relax-fairness]. Even though all known peer-to-peer filesharing applications are TCP-compatible, they can cause a disproportionate amount of congestion, simply by using multiple flows and by transferring data continuously relative to other short-lived sessions. On the other hand, if we swung the other way and set the bar low enough to allow streaming media to be unresponsive, we would also allow denial of service attacks, which are typically unresponsive to congestion and consist of multiple continuous flows.

Applications that need (or choose) to be unresponsive to congestion can effectively take (some would say steal) whatever share of bottleneck resources they want from responsive flows. Whether or not such free-riding is common, inability to prevent it increases the risk of poor returns for investors in network infrastructure, leading to under-investment. An increasing proportion of unresponsive or free-riding demand coupled with persistent under-supply is a broken economic cycle. Therefore, if the current, largely co-operative consensus continues to erode, congestion collapse could become more common in more areas of the Internet [RFC3714].

While we have designed re-ECN so that networks can choose to deploy stringent policing, this does not imply we advocate that every

network should introduce tight controls on those that cause congestion. Re-ECN has been specifically designed to allow different networks to choose how conservative or liberal they wish to be with respect to policing congestion. But those that choose to be conservative can protect themselves from the excesses that liberal networks allow their users.

3.1.2. The Case Against Bottleneck Policing

The state of the art in rate policing is the bottleneck policer, which is intended to be deployed at any forwarding resource that may become congested. Its aim is to detect flows that cause significantly more local congestion than others. Although operators might solve their immediate problems by deploying bottleneck policers, we are concerned that widespread deployment would make it extremely hard to evolve new application behaviours. We believe the IETF should offer re-ECN as the preferred protocol on which to base solutions to the policing problems of operators, because it would not harm evolvability and, frankly, it would be far more effective (see later for why).

Approaches like [XCHOKe] & [pBox] are nice approaches for rate policing traffic without the benefit of whole path information (such as could be provided by re-ECN). But they must be deployed at bottlenecks in order to work. Unfortunately, a large proportion of traffic traverses at least two bottlenecks (in two access networks), particularly with the current traffic mix where peer-to-peer file-sharing is prevalent. If ECN were deployed, we believe it would be likely that these bottleneck policers would be adapted to combine ECN congestion marking from the upstream path with local congestion knowledge. But then the only useful placement for such policers would be close to the egress of the internetwork.

But then, if these bottleneck policers were widely deployed (which would require them to be more effective than they are now), the Internet would find itself with one universal rate adaptation policy (probably TCP-friendliness) embedded throughout the network. Given TCP's congestion control algorithm is already known to be hitting its scalability limits and new algorithms are being developed for high-speed congestion control, embedding TCP policing into the Internet would make evolution to new algorithms extremely painful. If a source wanted to use a different algorithm, it would have to first discover then negotiate with all the policers on its path, particularly those in the far access network. The IETF has already traveled that path with the Intserv architecture and found it constrains scalability [RFC2208].

Anyway, if bottleneck policers were ever widely deployed, they would

be likely to be bypassed by determined attackers. They inherently have to police fairness per flow or per source-destination pair. Therefore they can easily be circumvented either by opening multiple flows (by varying the end-point port number); or by spoofing the source address but arranging with the receiver to hide the true return address at a higher layer.

4. Re-ECN Incentive Framework

The aim is to create an incentive environment that ensures optimal sharing of capacity despite everyone acting selfishly (including lying and cheating). Of course, the mechanisms put in place for this can lie dormant wherever co-operation is the norm.

4.1. Revealing Congestion Along the Path

Throughout this document we focus on path congestion. But some forms of fairness, particularly TCP's, also depend on round trip time. If TCP-fairness is required, we also propose to measure downstream path delay using re-feedback. We give a simple outline of how this could work in Appendix D. However, we do not expect this to be necessary, as researchers tend to agree that only congestion control dynamics need to depend on RTT, not the rate that the algorithm would converge on after a period of stability.

Recall that re-ECN can be used to measure path congestion at any point on the path. End-systems know the whole path congestion. The receiver knows this by the ratio of negative packets to all other packets it observes. The sender knows this same information via the feedback.

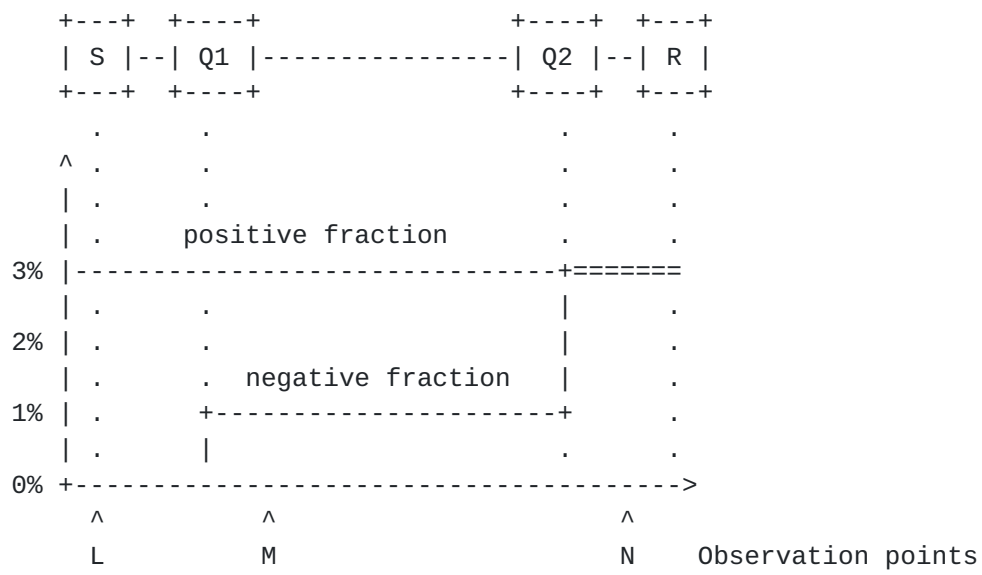

Figure 1: A 2-Queue Example (Imprecise)

Figure 1 uses a simple network to illustrate how re-ECN allows queues to measure downstream congestion. The receiver counts negative packets as being 3% of all received packets. This fraction is fed back to the sender. The sender sets 3% of its packets to be positive to match this. This fraction of positive packets can be observed along the path. This is shown by the horizontal line at 3% in the figure. The negative fraction is shown by the stepped line which rises to meet the positive fraction line with steps at at each queue where packets are marked negative. Two queues are shown (Q1 and Q2) that are currently congested. Each time packets pass through a fraction are marked red; 1% at Q1 and 2% at Q2). The approximate downstream congestion can be measured at the observation points shown along the path by subtracting the negative fraction from the positive fraction, as shown in the table below. [Re-TCP] [ref other document] derives these approximations from a precise analysis).

Observation point	Approx downstream congestion
L	3% - 0% = 3%
M	3% - 1% = 2%
N	3% - 3% = 0%

Table 1: Downstream Congestion Measured at Example Observation Points

All along the path, whole-path congestion remains unchanged so it can be used as a reference against which to compare upstream congestion.

The difference predicts downstream congestion for the rest of the path. Therefore, measuring the fractions of negative and positive packets at any point in the Internet will reveal upstream, downstream and whole path congestion.

Note: to be absolutely clear these fractions are averages that would result from the behaviour of the protocol handler mechanically sending positive packets in direct response to incoming feedback---we are not saying any protocol handler has to work with these average fractions directly.

4.1.1. Positive and Negative Flows

In section Section 1.2 we introduced the notion of IP packets having different values (negative, positive, cautious, cancelled and neutral). So positive and cautious packets have a value of +1, negative -1, and cancelled and neutral have zero value.

In the rest of this document we will loosely talk of positive or negative flows. A negative flow is one where more negative bytes than positive bytes arrive at the reciever. Likewise positive flows are where more positive bytes arrive than negative bytes. Both of these indicate that the wrong amount of positive bytes have been sent.

4.2. Incentive Framework Overview

Figure 2 sketches the incentive framework that we will describe piece by piece throughout this section. We will do a first pass in overview, then return to each piece in detail. We re-use the earlier example of how downstream congestion is derived by subtracting upstream congestion from path congestion (Figure 1) but depict multiple trust boundaries to turn it into an internetwork. For clarity, only downstream congestion is shown (the difference between the two earlier plots). The graph displays downstream path congestion seen in a typical flow as it traverses an example path from sender S to receiver R, across networks N1, N2 & N3. Everyone is shown using re-ECN correctly, but we intend to show why everyone would /choose/ to use it correctly, and honestly.

Three main types of self-interest can be identified:

- Users want to transmit data across the network as fast as possible, paying as little as possible for the privilege. In this respect, there is no distinction between senders and receivers, but we must be wary of potential malice by one on the other;

- Network operators want to maximise revenues from the resources they invest in. They compete amongst themselves for the custom of users.

- Attackers (whether users or networks) want to use any opportunity to subvert the new re-ECN system for their own gain or to damage the service of their victims, whether targeted or random.

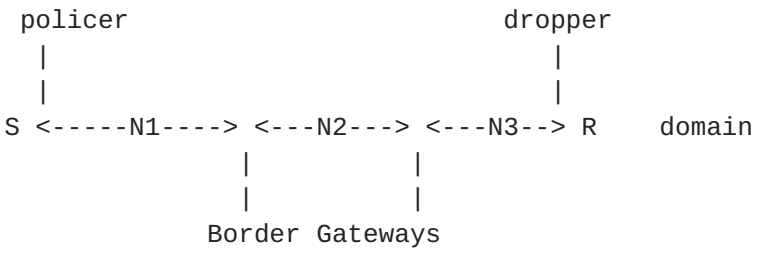

Figure 2: Incentive Framework

Source congestion control: We want to ensure that the sender will throttle its rate as downstream congestion increases. Whatever the agreed congestion response (whether TCP-compatible or some enhanced QoS), to some extent it will always be against the sender's interest to comply.

Ingress policing: But it is in all the network operators' interests to encourage fair congestion response, so that their investments are employed to satisfy the most valuable demand. The re-ECN protocol ensures packets carry the necessary information about their own expected downstream congestion so that N1 can deploy a policer at its ingress to check that S1 is complying with whatever congestion control it should be using (Section 4.4). If N1 is extremely conservative it could police each flow, but it is likely to just police the bulk amount of congestion each customer causes without regard to flows, or if it is extremely liberal it need not police congestion control at all. Whatever, it is always preferable to police traffic at the very first ingress into an internetwork, before non-compliant traffic can cause any damage.

Edge egress dropper: If the policer ensures the source has less right to a high rate the higher it declares downstream congestion, the source has a clear incentive to understate downstream congestion. But, if flows of packets are understated when they enter the internetwork, they will have become negative by the time they leave. So, we introduce a dropper at the last network

egress, which drops packets in flows that persistently declare negative downstream congestion (see Section 4.3 for details).

Inter-domain traffic policing: But next we must ask, if congestion arises downstream (say in N3), what is the ingress network's (N1's) incentive to police its customers' response? If N1 turns a blind eye, its own customers benefit while other networks suffer. This is why all inter-domain QoS architectures (e.g. Intserv, Diffserv) police traffic each time it crosses a trust boundary. We have already shown that re-ECN gives a trustworthy measure of the expected downstream congestion that a flow will cause by subtracting negative volume from positive at any intermediate point on a path. N3 (say) can use this measure to police all the responses to congestion of all the sources beyond its upstream neighbour (N2), but in bulk with one very simple passive mechanism, rather than per flow, as we will now explain.

Emulating policing with inter-domain congestion penalties: Between high-speed networks, we would rather avoid per-flow policing, and we would rather avoid holding back traffic while it is policed. Instead, once re-ECN has arranged headers to carry downstream congestion honestly, N2 can contract to pay N3 penalties in proportion to a single bulk count of the congestion metrics crossing their mutual trust boundary (Section 4.5). In this way, N3 puts pressure on N2 to suppress downstream congestion, for every flow passing through the border interface, even though they will all start and end in different places, and even though they may all be allowed different responses to congestion. The figure depicts this downward pressure on N2 by the solid downward arrow at the egress of N2. Then N2 has an incentive either to police the congestion response of its own ingress traffic (from N1) or to emulate policing by applying penalties to N1 in turn on the basis of congestion counted at their mutual boundary. In this recursive way, the incentives for each flow to respond correctly to congestion trace back with each flow precisely to each source, despite the mechanism not recognising flows (see Section 5.2).

Inter-domain congestion charging diversity: Any two networks are free to agree any of a range of penalty regimes between themselves but they would only provide the right incentives if they were within the following reasonable constraints. N2 should expect to have to pay penalties to N3 where penalties monotonically increase with the volume of congestion and negative penalties are not allowed. For instance, they may agree an SLA with tiered congestion thresholds, where higher penalties apply the higher the threshold that is broken. But the most obvious (and useful) form of penalty is where N3 levies a charge on N2 proportional to the volume of downstream congestion N2 dumps into N3. In the

explanation that follows, we assume this specific variant of volume charging between networks - charging proportionate to the volume of congestion.

We must make clear that we are not advocating that everyone should use this form of contract. We are well aware that the IETF tries to avoid standardising technology that depends on a particular business model. And we strongly share this desire to encourage diversity. But our aim is merely to show that border policing can at least work with this one model, then we can assume that operators might experiment with the metric in other models (see Section 4.5 for examples). Of course, operators are free to complement this usage element of their charges with traditional capacity charging, and we expect they will as predicted by economics.

No congestion charging to users: Bulk congestion penalties at trust boundaries are passive and extremely simple, and lose none of their per-packet precision from one boundary to the next (unlike Diffserv all-address traffic conditioning agreements, which dissipate their effectiveness across long topologies). But at any trust boundary, there is no imperative to use congestion charging. Traditional traffic policing can be used, if the complexity and cost is preferred. In particular, at the boundary with end customers (e.g. between S and N1), traffic policing will most likely be more appropriate. Policer complexity is less of a concern at the edge of the network. And end-customers are known to be highly averse to the unpredictability of congestion charging.

NOTE WELL: This document neither advocates nor requires congestion charging for end customers and advocates but does not require inter-domain congestion charging.

Competitive discipline of inter-domain traffic engineering: With inter-domain congestion charging, a domain seems to have a perverse incentive to fake congestion; N2's profit depends on the difference between congestion at its ingress (its revenue) and at its egress (its cost). So, overstating internal congestion seems to increase profit. However, smart border routing [Smart_rtg] by N1 will bias its routing towards the least cost routes. So, N2 risks losing all its revenue to competitive routes if it overstates congestion (see Section 5.3). In other words, if N2 is the least congested route, its ability to raise excess profits is limited by the congestion on the next least congested route.

Closing the loop: All the above elements conspire to trap everyone between two opposing pressures, ensuring the downstream congestion metric arrives at the destination neither above nor below zero. So, we have arrived back where we started in our argument. The ingress edge network can rely on downstream congestion declared in the packet headers presented by the sender. So it can police the sender's congestion response accordingly.

Evolvability of congestion control: We have seen that re-ECN enables policing at the very first ingress. We have also seen that, as flows continue on their path through further networks downstream, re-ECN removes the need for further per-domain ingress policing of all the different congestion responses allowed to each different flow. This is why the evolvability of re-ECN policing is so superior to bottleneck policing or to any policing of different QoS for different flows. Even if all access networks choose to conservatively police congestion per flow, each will want to compete with the others to allow new responses to congestion for new types of application. With re-ECN, each can introduce new controls independently, without coordinating with other networks and without having to standardise anything. But, as we have just seen, by making inter-domain penalties proportionate to bulk downtream congestion, downstream networks can be agnostic to the specific congestion response for each flow, but they can still apply more penalty the more liberal the ingress access network has been in the response to congestion it allowed for each flow.

We now take a second pass over the incentive framework, filling in the detail.

4.3. Egress Dropper

As traffic leaves the last network before the receiver (domain N3 in Figure 2), the fraction of positive octets in a flow should match the fraction of negative octets introduced by congestion marking (red packets), leaving a balance of zero. If it is less (a negative flow), it implies that the source is understating path congestion (which will reduce the penalties that N2 owes N3).

If flows are positive, N3 need take no action---this simply means its upstream neighbour is paying more penalties than it needs to, and the source is going slower than it needs to. But, to protect itself against persistently negative flows, N3 will need to install a dropper at its egress. Appendix A gives a suggested algorithm for this dropper. There is no intention that the dropper algorithm needs to be standardised, it is merely provided to show that an efficient, robust algorithm is possible. But whatever algorithm is used must meet the criteria below:

- It SHOULD introduce minimal false positives for honest flows;

- It SHOULD quickly detect and sanction dishonest flows (minimal false negatives);

- It SHOULD be invulnerable to state exhaustion attacks from malicious sources. For instance, if the dropper uses flow-state, it should not be possible for a source to send numerous packets, each with a different flow ID, to force the dropper to exhaust its memory capacity (rationale for SHOULD: Continuously sending keep-alive packets might be perfectly reasonable behaviour, so we can't distinguish a deliberate attack from reasonable levels of such behaviour. Therefore it is strictly impossible to be invulnerable to such an attack);

- It MUST introduce sufficient loss in goodput so that malicious sources cannot play off losses in the egress dropper against higher allowed throughput. Salvatori [CLoop_pol] describes this attack, which involves the source understating path congestion then inserting forward error correction (FEC) packets to compensate expected losses;

- It MUST NOT be vulnerable to `identity whitewashing', where a transport can label a flow with a new ID more cheaply than paying the cost of continuing to use its current ID.

Note that the dropper operates on flows but we would like it not to require per-flow state. This is why we have been careful to ensure that all flows MUST start with a cautious packet. If a flow does not start with a cautious packet, a dropper is likely to treat it unfavourably. This risk makes it worth sending a cautious packet at the start of a flow, even though there is a cost to the sender of doing so (positive `worth'). Indeed, with cautious packets, the rate at which a sender can generate new flows can be limited (Appendix B). In this respect, cautious packets work like Handley's state set-up bit [Steps_DoS].

Appendix A also gives an example dropper implementation that aggregates flow state. Dropper algorithms will often maintain a moving average across flows of the fraction of positive packets. When maintaining an average across flows, a dropper SHOULD only allow flows into the average if they start with a cautious packet, but it SHOULD NOT include cautious packets in the positive packet average. A sender sends cautious packets when it does not have the benefit of feedback from the receiver. So, counting cautious packets would be likely to make the average unnecessarily positive, providing headroom (or should we say footroom?) for dishonest (negative) traffic.

If the dropper detects a persistently negative flow, it SHOULD drop sufficient negative and neutral packets to force the flow to not be negative. Drops SHOULD be focused on just sufficient packets in misbehaving flows to remove the negative bias while doing minimal extra harm.

4.4. Ingress Policing

Access operators who wish to limit the congeston that a sender is able to cause can deploy policers at the very first ingress to the internetwork. Re-ECN has been designed to avoid the need for bottleneck policing so that we can avoid a future where a single rate adaptation policy is embedded throughout the network. Instead, re-ECN allows the particular rate adaptation policy to be solely agreed bilaterally between the sender and its ingress access provider ([ref other document] discusses possible ways to signal between them), which allows congestion control to be policed, but maintains its evolvability, requiring only a single, local box to be updated.

Appendix B gives examples of per-user policing algorithms. But there is no implication that these algorithms are to be standardised, or that they are ideal. The ingress rate policer is the part of the re-ECN incentive framework that is intended to be the most flexible. Once endpoint protocol handlers for re-ECN and egress droppers are in place, operators can choose exactly which congestion response they want to police, and whether they want to do it per user, per flow or not at all.

The re-ECN protocol allows these ingress policers to easily perform bulk per-user policing (Appendix B.1). This is likely to provide sufficient incentive to the user to correctly respond to congestion without needing the policing function to be overly complex. If an access operator chose they could use per-flow policing according to the widely adopted TCP rate adaptation (Appendix B.2) or other alternatives, however this would introduce extra complexity to the system.

If a per-flow rate policer is used, it should use path (not downstream) congestion as the relevant metric, which is represented by the fraction of octets in packets with positive (positive and cautious packets) and cancelled packets. Of course, re-ECN provides all the information a policer needs directly in the packets being policed. So, even policing TCP's AIMD algorithm is relatively straightforward (Appendix B.2).

Note that we have included cancelled packets in the measure of path congestion. cancelled packets arise when the sender sends a positive packet in response to feedback, but then this positive packet just

happens to be congestion marked itself. One would not normally expect many cancelled packets at the first ingress because one would not normally expect much congestion marking to have been necessary that soon in the path. However, a home network or campus network may well sit between the sending endpoint and the ingress policer, so some congestion may occur upstream of the policer. And if congestion does occur upstream, some cancelled packets should be visible, and should be taken into account in the measure of path congestion.

But a much more important reason for including cancelled packets in the measure of path congestion at an ingress policer is that a sender might otherwise subvert the protocol by sending cancelled packets instead of neutral packets. Like neutral, cancelled packets are worth zero, so the sender knows they won't be counted against any quota it might have been allowed. But unlike neutral packets, cancelled packets are immune to congestion marking, because they have already been congestion marked. So, it is both correct and useful that cancelled packets should be included in a policer's measure of path congestion, as this removes the incentive the sender would otherwise have to mark more packets as cancelled than it should.

An ingress policer should also ensure that flows are not already negative when they enter the access network. As with cancelled packets, the presence of negative packets will typically be unusual. Therefore it will be easy to detect negative flows at the ingress by just detecting negative packets then monitoring the flow they belong to.

Of course, even if the sender does operate its own network, it may arrange not to congestion mark traffic. Whether the sender does this or not is of no concern to anyone else except the sender. Such a sender will not be policed against its own network's contribution to congestion, but the only resulting problem would be overload in the sender's own network.

Finally, we must not forget that an easy way to circumvent re-ECN's defences is for the source to turn off re-ECN support, by setting the Not-RECT codepoint, implying RFC3168 compliant traffic. Therefore an ingress policer should put a general rate-limit on Not-RECT traffic, which SHOULD be lax during early, patchy deployment, but will have to become stricter as deployment widens. Similarly, flows starting without a cautious packet can be confined by a strict rate-limit used for the remainder of flows that haven't proved they are well-behaved by starting correctly (therefore they need not consume any flow state---they are just confined to the `misbehaving' bin if they carry an unrecognised flow ID).

4.5. Inter-domain Policing

One of the main design goals of re-ECN is for border security mechanisms to be as simple as possible, otherwise they will become the pinch-points that limit scalability of the whole internetwork. We want to avoid per-flow processing at borders and to keep to passive mechanisms that can monitor traffic in parallel to forwarding, rather than having to filter traffic inline---in series with forwarding. Such passive, off-line mechanisms are essential for future high-speed all-optical border interconnection where packets cannot be buffered while they are checked for policy compliance.

So far, we have been able to keep the border mechanisms simple, despite having had to harden them against some subtle attacks on the re-ECN design. The mechanisms are still passive and avoid per-flow processing.

The basic accounting mechanism at each border interface simply involves accumulating the volume of packets with positive worth (positive and cautious packets), and subtracting the volume of those with negative worth (red packets). Even though this mechanism takes no regard of flows, over an accounting period (say a month) this subtraction will account for the downstream congestion caused by all the flows traversing the interface, wherever they come from, and wherever they go to. The two networks can agree to use this metric however they wish to determine some congestion-related penalty against the upstream network. Although the algorithm could hardly be simpler, it is spelled out using pseudo-code in Appendix C.1.

Various attempts to subvert the re-ECN design have been made. In all cases their root cause is persistently negative flows. But, after describing these attacks we will show that we don't actually have to get rid of all persistently negative flows in order to thwart the attacks.

In honest flows, downstream congestion is measured as positive minus negative volume. So if all flows are honest (i.e. not persistently negative), adding all positive volume and all negative volume without regard to flows will give an aggregate measure of downstream congestion. But such simple aggregation is only possible if no flows are persistently negative. Unless persistently negative flows are completely removed, they will reduce the aggregate measure of congestion. The aggregate may still be positive overall, but not as positive as it would have been had the negative flows been removed.

In Section 4.3 we discussed how to sanction traffic to remove, or at least to identify, persistently negative flows. But, even if the sanction for negative traffic is to discard it, unless it is

discarded at the exact point it goes negative, it will wrongly subtract from aggregate downstream congestion, at least at any borders it crosses after it has gone negative but before it is discarded.

We rely on sanctions to deter dishonest understatement of congestion. But even the ultimate sanction of discard can only be effective if the sender is bothered about the data getting through to its destination. A number of attacks have been identified where a sender gains from sending dummy traffic or it can attack someone or something using dummy traffic even though it isn't communicating any information to anyone:

- A host can send traffic with no positive packets towards its intended destination, aiming to transmit as much traffic as any dropper will allow [Bauer06]. It may add forward error correction (FEC) to repair as much drop as it experiences.

- A host can send dummy traffic into the network with no positive packets and with no intention of communicating with anyone, but merely to cause higher levels of congestion for others who do want to communicate (DoS). So, to ride over the extra congestion, everyone else has to spend more of whatever rights to cause congestion they have been allowed.

- A network can simply create its own dummy traffic to congest another network, perhaps causing it to lose business at no cost to the attacking network. This is a form of denial of service perpetrated by one network on another. The preferential drop measures in [ref other document] provide crude protection against such attacks, but we are not overly worried about more accurate prevention measures, because it is already possible for networks to DoS other networks on the general Internet, but they generally don't because of the grave consequences of being found out. We are only concerned if re-ECN increases the motivation for such an attack, as in the next example.

- A network can just generate negative traffic and send it over its border with a neighbour to reduce the overall penalties that it should pay to that neighbour. It could even initialise the TTL so it expired shortly after entering the neighbouring network, reducing the chance of detection further downstream. This attack need not be motivated by a desire to deny service and indeed need not cause denial of service. A network's main motivator would most likely be to reduce the penalties it pays to a neighbour. But, the prospect of financial gain might tempt the network into mounting a DoS attack on the other network as well, given the gain would offset some of the risk of being detected.

The first step towards a solution to all these problems with negative flows is to be able to estimate the contribution they make to downstream congestion at a border and to correct the measure accordingly. Although ideally we want to remove negative flows themselves, perhaps surprisingly, the most effective first step is to cancel out the polluting effect negative flows have on the measure of downstream congestion at a border. It is more important to get an unbiased estimate of their effect, than to try to remove them all. A suggested algorithm to give an unbiased estimate of the contribution from negative flows to the downstream congestion measure is given in Appendix C.2.

Although making an accurate assessment of the contribution from negative flows may not be easy, just the single step of neutralising their polluting effect on congestion metrics removes all the gains networks could otherwise make from mounting dummy traffic attacks on each other. This puts all networks on the same side (only with respect to negative flows of course), rather than being pitched against each other. The network where this flow goes negative as well as all the networks downstream lose out from not being reimbursed for any congestion this flow causes. So they all have an interest in getting rid of these negative flows. Networks forwarding a flow before it goes negative aren't strictly on the same side, but they are disinterested bystanders---they don't care that the flow goes negative downstream, but at least they can't actively gain from making it go negative. The problem becomes localised so that once a flow goes negative, all the networks from where it happens and beyond downstream each have a small problem, each can detect it has a problem and each can get rid of the problem if it chooses to. But negative flows can no longer be used for any new attacks.

Once an unbiased estimate of the effect of negative flows can be made, the problem reduces to detecting and preferably removing flows that have gone negative as soon as possible. But importantly, complete eradication of negative flows is no longer critical---best endeavours will be sufficient.

For instance, let us consider the case where a source sends traffic with no positive packets at all, hoping to at least get as much traffic delivered as network-based droppers will allow. The flow is likely to go at least slightly negative in the first network on the path (N1 if we use the example network layout in Figure 2). If all networks use the algorithm in Appendix C.2 to inflate penalties at their border with an upstream network, they will remove the effect of negative flows. So, for instance, N2 will not be paying a penalty to N1 for this flow. Further, because the flow contributes no positive packets at all, a dropper at the egress will completely remove it.

The remaining problem is that every network is carrying a flow that is causing congestion to others but not being held to account for the congestion it is causing. Whenever the fail-safe border algorithm (Section 4.6) or the border algorithm to compensate for negative flows (Appendix C.2) detects a negative flow, it can instantiate a focused dropper for that flow locally. It may be some time before the flow is detected, but the more strongly negative the flow is, the more quickly it will be detected by the fail-safe algorithm. But, in the meantime, it will not be distorting border incentives. Until it is detected, if it contributes to drop anywhere, its packets will tend to be dropped before others if queues use the preferential drop rules in [ref other document], which discriminate against non-positive packets. All networks below the point where a flow goes negative (N1, N2 and N3 in this case) have an incentive to remove this flow, but the queue where it first goes negative (in N1) can of course remove the problem for everyone downstream.

In the case of DDoS attacks, Section 5.1 describes how re-ECN mitigates their force.

4.6. Inter-domain Fail-safes

The mechanisms described so far create incentives for rational network operators to behave. That is, one operator aims to make another behave responsibly by applying penalties and expects a rational response (i.e. one that trades off costs against benefits). It is usually reasonable to assume that other network operators will behave rationally (policy routing can avoid those that might not). But this approach does not protect against the misconfigurations and accidents of other operators.

Therefore, we propose the following two mechanisms at a network's borders to provide "defence in depth". Both are similar:

Highly positive flows: A small sample of positive packets should be picked randomly as they cross a border interface. Then subsequent packets matching the same source and destination address and DSCP should be monitored. If the fraction of positive packets is well above a threshold (to be determined by operational practice), a management alarm SHOULD be raised, and the flow MAY be automatically subject to focused drop.

Persistently negative flows: A small sample of congestion marked (red) packets should be picked randomly as they cross a border interface. Then subsequent packets matching the same source and destination address and DSCP should be monitored. If the balance of positive packets minus negative packets (measured in bytes) is persistently negative, a management alarm SHOULD be raised, and

the flow MAY be automatically subject to focused drop.

Both these mechanisms rely on the fact that highly positive (or negative) flows will appear more quickly in the sample by selecting randomly solely from positive (or negative) packets.

4.7. The Case against Classic Feedback

A system that produces an optimal outcome as a result of everyone's selfish actions is extremely powerful. Especially one that enables evolvability of congestion control. But why do we have to change to re-ECN to achieve it? Can't classic congestion feedback (as used already by standard ECN) be arranged to provide similar incentives and similar evolvability? Superficially it can. Kelly's seminal work showed how we can allow everyone the freedom to evolve whatever congestion control behaviour is in their application's best interest but still optimise the whole system of networks and users by placing a price on congestion to ensure responsible use of this freedom [Evol_cc]). Kelly used ECN with its classic congestion feedback model as the mechanism to convey congestion price information. The mechanism could be thought of as volume charging; except only the volume of packets marked with congestion experienced (CE) was counted.

However, below we explain why relying on classic feedback /required/ congestion charging to be used, while re-ECN achieves the same powerful outcome (given it is built on Kelly's foundations), but does not /require/ congestion charging. In brief, the problem with classic feedback is that the incentives have to trace the indirect path back to the sender---the long way round the feedback loop. For example, if classic feedback were used in Figure 2, N2 would have had to influence N1 via all of N3, R & S rather than directly.

Inability to agree what is happening downstream: In order to police its upstream neighbour's congestion response, the neighbours should be able to agree on the congestion to be responded to. Whatever the feedback regime, as packets change hands at each trust boundary, any path metrics they carry are verifiable by both neighbours. But, with a classic path metric, they can only agree on the /upstream/ path congestion.

Inaccessible back-channel: The network needs a whole-path congestion metric if it wants to control the source. Classically, whole path congestion emerges at the destination, to be fed back from receiver to sender in a back-channel. But, in any data network, back-channels need not be visible to relays, as they are essentially communications between the end-points. They may be encrypted, asymmetrically routed or simply omitted, so no network

element can reliably intercept them. The congestion charging literature solves this problem by charging the receiver and assuming this will cause the receiver to refer the charges to the sender. But, of course, this creates unintended side-effects...

`Receiver pays' unacceptable: In connectionless datagram networks, receivers and receiving networks cannot prevent reception from malicious senders, so `receiver pays' opens them to `denial of funds' attacks.

End-user congestion charging unacceptable in many societies: Even if 'denial of funds' were not a problem, we know that end-users are highly averse to the unpredictability of congestion charging and anyway, we want to avoid restricting network operators to just one retail tariff. But with classic feedback only an upstream metric is available, so we cannot avoid having to wrap the `receiver pays' money flow around the feedback loop, necessarily forcing end-users to be subjected to congestion charging.

To summarise so far, with classic feedback, policing congestion response without losing evolvability /requires/ congestion charging of end-users and a `receiver pays' model, whereas, with re-ECN, it is still possible to influence incentives using congestion charging but using the safer `sender pays' model. However, congestion charging is only likely to be appropriate between domains. So, without losing evolvability, re-ECN enables technical policing mechanisms that are more appropriate for end users than congestion pricing.

4.8. Simulations

Simulations of policer and dropper performance done for the multi-bit version of re-feedback have been included in section 5 "Dropper Performance" of [Re-fb]. Simulations of policer and dropper for the re-ECN version described in this document are work in progress.

5. Other Applications of Re-ECN

5.1. DDoS Mitigation

A flooding attack is inherently about congestion of a resource. Because re-ECN ensures the sources causing network congestion experience the cost of their own actions, it acts as a first line of defence against DDoS. As load focuses on a victim, upstream queues grow, requiring honest sources to pre-load packets with a higher fraction of positive packets. Once downstream queues are so congested that they are dropping traffic, they will be marking to negative the traffic they do forward 100%. Honest sources will therefore be sending positive packets 100% (and therefore being

severely rate-limited at the ingress).

Senders under malicious control can either do the same as honest sources, and be rate-limited at ingress, or they can understate congestion by sending more neutral RECT packets than they should. If sources understate congestion (i.e. do not re-echo sufficient positive packets) and the preferential drop ranking is implemented on queues ([ref othe document]), these queues will preserve positive traffic until last. So, the neutral traffic from malicious sources will all be automatically dropped first. Either way, the malicious sources cannot send more than honest sources.

Further, hosts under malicious control will tend to be re-used for many different attacks. They will therefore build up a long term history of causing congestion. Therefore, as long as the population of potentially compromisable hosts around the Internet is limited, the per-user policing algorithms in Appendix B.1 will gradually throttle down zombies and other launchpads for attacks. Therefore, widespread deployment of re-ECN could considerably dampen the force of DDoS. Certainly, zombie armies could hold their fire for long enough to be able to build up enough credit in the per-user policers to launch an attack. But they would then still be limited to no more throughput than other, honest users.

Inter-domain traffic policing (see Section 4.5)ensures that any network that harbours compromised `zombie' hosts will have to bear the cost of the congestion caused by traffic from zombies in downstream networks. Such networks will be incentivised to deploy per-user policers that rate-limit hosts that are unresponsive to congestion so they can only send very slowly into congested paths. As well as protecting other networks, the extremely poor performance at any sign of congestion will incentivise the zombie's owner to clean it up. However, the host should behave normally when using uncongested paths.

Uniquely, re-ECN handles DDoS traffic without relying on the validity of identifiers in packets. Certainly the egress dropper relies on uniqueness of flow identifiers, but not their validity. So if a source spoofs another address, re-ECN works just as well, as long as the attacker cannot imitate all the flow identifiers of another active flow passing through the same dropper (see Section 6). Similarly, the ingress policer relies on uniqueness of flow IDs, not their validity. Because a new flow will only be allowed any rate at all if it starts with a cautious packet, and the more cautious packets there are starting new flows, the more they will be limited. Essentially a re-ECN policer limits the bulk of all congestion entering the network through a physical interface; limiting the congestion caused by each flow is merely an optional extra.

5.2. End-to-end QoS

{ToDo: (Section 3.3.2 of [Re-fb] entitled `Edge QoS' gives an outline of the text that will be added here).}

5.3. Traffic Engineering

Classic feedback makes congestion-based traffic engineering inefficient too. Network N3 can see which of its two alternative upstream networks N2 and N3 are less congested. But it is N1 that makes the routing decision. This is why current traffic engineering requires a continuous message stream from congestion monitors to the routing controller. And even then the monitors can only be trusted for /intra-/domain traffic engineering. The trustworthiness of re-ECN enables /inter-/domain traffic engineering without messaging overhead. {ToDo: Elaborate}

5.4. Inter-Provider Service Monitoring

{ToDo: }

6. Limitations

{ToDo:See also: slide of limitations}

The known limitations of the re-ECN approach are:

- We still cannot defend against the attack described in Section 10 where a malicious source sends negative traffic through the same egress dropper as another flow and imitates its flow identifiers, allowing a malicious source to cause an innocent flow to experience heavy drop.

- Re-feedback for TTL (re-TTL) would also be desirable at the same time as re-ECN. Unfortunately this requires a further standards action for the mechanisms briefly described in Appendix D

- Traffic must be ECN-capable for re-ECN to be effective. The only defence against malicious users who turn off ECN capbility is that networks are expected to rate limit Not-ECT traffic and to apply higher drop preference to it during congestion. Although these are blunt instruments, they at least represent a feasible scenario for the future Internet where Not-ECT traffic co-exists with re-ECN traffic, but as a severely hobbled under-class. We recommend (Section 7.1) that while accommodating a smooth initial transition to re-ECN, policing policies should gradually be tightened to rate limit Not-ECT traffic more strictly in the longer term.

- When checking whether a flow is balancing positive packets with negative packets (measured in bytes), re-ECN can only account for congestion marking, not drops. So, whenever a sender experiences drop, it does not have to re-echo the congestion event by sending positive packet(s). Nonetheless, it is hardly any advantage to be able to send faster than other flows only if your traffic is dropped and the other traffic isn't.

- We are considering the issue of whether it would be useful to truncate rather than drop packets that appear to be malicious, so that the feedback loop is not broken but useful data can be removed.

{ToDo: Monopolies over Routes}

7. Incremental Deployment

7.1. Incremental Deployment Features

The design of the re-ECN protocol started from the fact that the current ECN marking behaviour of queues was sufficient and that re-feedback could be introduced around these queues by changing the sender behaviour but not the routers. Otherwise, if we had required routers to be changed, the chance of encountering a path that had every router upgraded would be vanishly small during early deployment, giving no incentive to start deployment. Also, as there is no new forwarding behaviour, routers and hosts do not have to signal or negotiate anything.

However, networks that choose to protect themselves using re-ECN do have to add new security functions at their trust boundaries with others. They distinguish legacy traffic by its ECN field. Traffic from Not-ECT transports is distinguishable by its Not-ECT marking . Traffic from RFC3168 compliant ECN transports is distinguished from re-ECN by which of ECT(0) or ECT(1) is used. We chose to use ECT(1) for re-ECN traffic deliberately. Existing ECN sources set ECT(0) on either 50% (the nonce) or 100% (the default) of packets, whereas re-ECN does not use ECT(0) at all. We can use this distinguishing feature of RFC3168 compliant ECN traffic to separate it out for different treatment at the various border security functions: egress dropping, ingress policing and border policing.

The general principle we adopt is that an egress dropper will not drop any legacy traffic, but ingress and border policers will limit the bulk rate of legacy traffic (Not-ECT, ECT(0) and those amrked with the unused codepoint as defined in [Re-TCP]) that can enter each network. Then, during early re-ECN deployment, operators can set very permissive (or non-existent) rate-limits on legacy traffic, but

once re-ECN implementations are generally available, legacy traffic can be rate-limited increasingly harshly. Ultimately, an operator might choose to block all legacy traffic entering its network, or at least only allow through a trickle.

Then, as the limits are set more strictly, the more RFC3168 ECN sources will gain by upgrading to re-ECN. Thus, towards the end of the voluntary incremental deployment period, RFC3168 compliant transports can be given progressively stronger encouragement to upgrade.

7.2. Incremental Deployment Incentives

It would only be worth standardising the re-ECN protocol if there existed a coherent story for how it might be incrementally deployed. In order for it to have a chance of deployment, everyone who needs to act must have a strong incentive to act, and the incentives must arise in the order that deployment would have to happen. Re-ECN works around unmodified ECN routers, but we can't just discuss why and how re-ECN deployment might build on ECN deployment, because there is precious little to build on in the first place. Instead, we aim to show that re-ECN deployment could carry ECN with it. We focus on commercial deployment incentives, although some of the arguments apply equally to academic or government sectors.

ECN deployment:

ECN is largely implemented in commercial routers, but generally not as a supported feature, and it has largely not been deployed by commercial network operators. ECN has been implemented in most Unix-based operating systems for some time. Microsoft first implemented ECN in Windows Vista, but it is only on by default for the server end of a TCP connection. Unfortunately the client end had to be turned off by default, because a non-zero ECN field triggers a bug in a legacy home gateway which makes it crash. For detailed deployment status, see [ECN-Deploy]. We believe the reason ECN deployment has not happened is twofold:

* ECN requires changes to both routers and hosts. If someone wanted to sell the improvement that ECN offers, they would have to co-ordinate deployment of their product with others. An ECN server only gives any improvement on an ECN network. An ECN network only gives any improvement if used by ECN devices. Deployment that requires co-ordination adds cost and delay and tends to dilute any competitive advantage that might be gained.

* ECN `only' gives a performance improvement. Making a product a bit faster (whether the product is a device or a network),

isn't usually a sufficient selling point to be worth the cost of co-ordinating across the industry to deploy it. Network operators tend to avoid re-configuring a working network unless launching a new product.

ECN and Re-ECN for Edge-to-edge Assured QoS:

We believe the proposal to provide assured QoS sessions using a form of ECN called pre-congestion notification (PCN) [RFC5559] is most likely to break the deadlock in ECN deployment first. It only requires edge-to-edge deployment so it does not require endpoint support. It can be deployed in a single network, then grow incrementally to interconnected networks. And it provides a different `product' (internetworked assured QoS), rather than merely making an existing product a bit faster.

Not only could this assured QoS application kick-start ECN deployment, it could also carry re-ECN deployment with it; because re-ECN can enable the assured QoS region to expand to a large internetwork where neighbouring networks do not trust each other. [Re-PCN] argues that re-ECN security should be built in to the QoS system from the start, explaining why and how.

If ECN and re-ECN were deployed edge-to-edge for assured QoS, operators would gain valuable experience. They would also clear away many technical obstacles such as firewall configurations that block all but the RFC3168 settings of the ECN field and the RE flag.

ECN in Access Networks:

The next obstacle to ECN deployment would be extension to access and backhaul networks, where considerable link layer differences makes implementation non-trivial, particularly on congested wireless links. ECN and re-ECN work fine during partial deployment, but they will not be very useful if the most congested elements in networks are the last to support them. Access network support is one of the weakest parts of this deployment story. All we can hope is that, once the benefits of ECN are better understood by operators, they will push for the necessary link layer implementations as deployment proceeds.

Policing Unresponsive Flows:

Re-ECN allows a network to offer differentiated quality of service as explained in Section 5.2. But we do not believe this will motivate initial deployment of re-ECN, because the industry is already set on alternative ways of doing QoS. Despite being much

more complicated and expensive, the alternative approaches are here and now.

But re-ECN is critical to QoS deployment in another respect. It can be used to prevent applications from taking whatever bandwidth they choose without asking.

Currently, applications that remain resolute in their lack of response to congestion are rewarded by other TCP applications. In other words, TCP is naively friendly, in that it reduces its rate in response to congestion whether it is competing with friends (other TCPs) or with enemies (unresponsive applications).

Therefore, those network owners that want to sell QoS will be keen to ensure that their users can't help themselves to QoS for free. Given the very large revenues at stake, we believe effective policing of congestion response will become highly sought after by network owners.

But this does not necessarily argue for re-ECN deployment. Network owners might choose to deploy bottleneck policers rather than re-ECN-based policing. However, under Related Work (Section 9) we argue that bottleneck policers are inherently vulnerable to circumvention.

Therefore we believe there will be a strong demand from network owners for re-ECN deployment so they can police flows that do not ask to be unresponsive to congestion, in order to protect their revenues from flows that do ask (QoS). In particular, we suspect that the operators of cellular networks will want to prevent VoIP and video applications being used freely on their networks as a more open market develops in GPRS and 3G devices.

Initial deployments are likely to be isolated to single cellular networks. Cellular operators would first place requirements on device manufacturers to include re-ECN in the standards for mobile devices. In parallel, they would put out tenders for ingress and egress policers. Then, after a while they would start to tighten rate limits on Not-ECT traffic from non-standard devices and they would start policing whatever non-accredited applications people might install on mobile devices with re-ECN support in the operating system. This would force even independent mobile device manufacturers to provide re-ECN support. Early standardisation across the cellular operators is likely, including interconnection agreements with penalties for excess downstream congestion.

We suspect some fixed broadband networks (whether cable or DSL) would follow a similar path. However, we also believe that larger parts of the fixed Internet would not choose to police on a per-flow basis. Some might choose to police congestion on a per-user basis in order to manage heavy peer-to-peer file-sharing, but it seems likely that a sizeable majority would not deploy any form of policing.

This hybrid situation begs the question, "How does re-ECN work for networks that choose to using policing if they connect with others that don't?" Traffic from non-ECN capable sources will arrive from other networks and cause congestion within the policed, ECN-capable networks. So networks that chose to police congestion would rate-limit Not-ECT traffic throughout their network, particularly at their borders. They would probably also set higher usage prices in their interconnection contracts for incoming Not-ECT and Not-RECT traffic. We assume that interconnection contracts between networks in the same tier will include congestion penalties before contracts with provider backbones do.

A hybrid situation could remain for all time. As was explained in the introduction, we believe in healthy competition between policing and not policing, with no imperative to convert the whole world to the religion of policing. Networks that chose not to deploy egress droppers would leave themselves open to being congested by senders in other networks. But that would be their choice.

The important aspect of the egress dropper though is that it most protects the network that deploys it. If a network does not deploy an egress dropper, sources sending into it from other networks will be able to understate the congestion they are causing. Whereas, if a network deploys an egress dropper, it can know how much congestion other networks are dumping into it, and apply penalties or charges accordingly. So, whether or not a network polices its own sources at ingress, it is in its interests to deploy an egress dropper.

Host support:

In the above deployment scenario, host operating system support for re-ECN came about through the cellular operators demanding it in device standards (i.e. 3GPP). Of course, increasingly, mobile devices are being built to support multiple wireless technologies. So, if re-ECN were stipulated for cellular devices, it would automatically appear in those devices connected to the wireless fringes of fixed networks if they coupled cellular with WiFi or

Bluetooth technology, for instance. Also, once implemented in the operating system of one mobile device, it would tend to be found in other devices using the same family of operating system.

Therefore, whether or not a fixed network deployed ECN, or deployed re-ECN policers and droppers, many of its hosts might well be using re-ECN over it. Indeed, they would be at an advantage when communicating with hosts across re-ECN policed networks that rate limited Not-RECT traffic.

Other possible scenarios:

The above is thankfully not the only plausible scenario we can think of. One of the many clubs of operators that meet regularly around the world might decide to act together to persuade a major operating system manufacturer to implement re-ECN. And they may agree between them on an interconnection model that includes congestion penalties.

Re-ECN provides an interesting opportunity for device manufacturers as well as network operators. Policers can be configured loosely when first deployed. Then as re-ECN take-up increases, they can be tightened up, so that a network with re-ECN deployed can gradually squeeze down the service provided to RFC3168 compliant devices that have not upgraded to re-ECN. Many device vendors rely on replacement sales. And operating system companies rely heavily on new release sales. Also support services would like to be able to force stragglers to upgrade. So, the ability to throttle service to RFC3168 compliant operating systems is quite valuable.

Also, policing unresponsive sources may not be the only or even the first application that drives deployment. It may be policing causes of heavy congestion (e.g. peer-to-peer file-sharing). Or it may be mitigation of denial of service. Or we may be wrong in thinking simpler QoS will not be the initial motivation for re-ECN deployment. Indeed, the combined pressure for all these may be the motivator, but it seems optimistic to expect such a level of joined-up thinking from today's communications industry. We believe a single application alone must be a sufficient motivator.

In short, everyone gains from adding accountability to TCP/IP, except the selfish or malicious. So, deployment incentives tend to be strong.

8. Architectural Rationale

In the Internet's technical community, the danger of not responding to congestion is well-understood, as well as its attendant risk of congestion collapse [RFC3714]. However, one side of the Internet's commercial community considers that the very essence of IP is to provide open access to the internetwork for all applications. They see congestion as a symptom of over-conservative investment, and rely on revising application designs to find novel ways to keep applications working despite congestion. They argue that the Internet was never intended to be solely for TCP-friendly applications. Meanwhile, another side of the Internet's commercial community believes that it is worthwhile providing a network for novel applications only if it has sufficient capacity, which can happen only if a greater share of application revenues can be /assured/ for the infrastructure provider. Otherwise the major investments required would carry too much risk and wouldn't happen.

The lesson articulated in [Tussle] is that we shouldn't embed our view on these arguments into the Internet at design time. Instead we should design the Internet so that the outcome of these arguments can get decided at run-time. Re-ECN is designed in that spirit. Once the protocol is available, different network operators can choose how liberal they want to be in holding people accountable for the congestion they cause. Some might boldly invest in capacity and not police its use at all, hoping that novel applications will result. Others might use re-ECN for fine-grained flow policing, expecting to make money selling vertically integrated services. Yet others might sit somewhere half-way, perhaps doing coarse, per-user policing. All might change their minds later. But re-ECN always allows them to interconnect so that the careful ones can protect themselves from the liberal ones.

The incentive-based approach used for re-ECN is based on Gibbens and Kelly's arguments [Evol_cc] on allowing endpoints the freedom to evolve new congestion control algorithms for new applications. They ensured responsible behaviour despite everyone's self-interest by applying pricing to ECN marking, and Kelly had proved stability and optimality in an earlier paper.

Re-ECN keeps all the underlying economic incentives, but rearranges the feedback. The idea is to allow a network operator (if it chooses) to deploy engineering mechanisms like policers at the front of the network which can be designed to behave /as if/ they are responding to congestion prices. Rather than having to subject users to congestion pricing, networks can then use more traditional charging regimes (or novel ones). But the engineering can constrain the overall amount of congestion a user can cause. This provides a

buffer against completely outrageous congestion control, but still makes it easy for novel applications to evolve if they need different congestion control to the norms. It also allows novel charging regimes to evolve.

Despite being achieved with a relatively minor protocol change, re-ECN is an architectural change. Previously, Internet congestion could only be controlled by the data sender, because it was the only one both in a position to control the load and in a position to see information on congestion. Re-ECN levels the playing field. It recognises that the network also has a role to play in moderating (policing) congestion control. But policing is only truly effective at the first ingress into an internetwork, whereas path congestion was previously only visible at the last egress. So, re-ECN democratises congestion information. Then the choice over who actually controls congestion can be made at run-time, not design time---a bit like an aircraft with dual controls. And different operators can make different choices. We believe non-architectural approaches to this problem are unlikely to offer more than partial solutions (see Section 9).

Importantly, re-ECN does not require assumptions about specific congestion responses to be embedded in any network elements, except at the first ingress to the internetwork if that level of control is desired by the ingress operator. But such tight policing will be a matter of agreement between the source and its access network operator. The ingress operator need not police congestion response at flow granularity; it can simply hold a source responsible for the aggregate congestion it causes, perhaps keeping it within a monthly congestion quota. Or if the ingress network trusts the source, it can do nothing.

Therefore, the aim of the re-ECN protocol is NOT solely to police TCP-friendliness. Re-ECN preserves IP as a generic network layer for all sorts of responses to congestion, for all sorts of transports. Re-ECN merely ensures truthful downstream congestion information is available in the network layer for all sorts of accountability applications.

The end to end design principle does not say that all functions should be moved out of the lower layers---only those functions that are not generic to all higher layers. Re-ECN adds a function to the network layer that is generic, but was omitted: accountability for causing congestion. Accountability is not something that an end-user can provide to themselves. We believe re-ECN adds no more than is sufficient to hold each flow accountable, even if it consists of a single datagram.

"Accountability" implies being able to identify who is responsible for causing congestion. However, at the network layer it would NOT be useful to identify the cause of congestion by adding individual or organisational identity information, NOR by using source IP addresses. Rather than bringing identity information to the point of congestion, we bring downstream congestion information to the point where the cause can be most easily identified and dealt with. That is, at any trust boundary congestion can be associated with the physically connected upstream neighbour that is directly responsible for causing it (whether intentionally or not). A trust boundary interface is exactly the place to police or throttle in order to directly mitigate congestion, rather than having to trace the (ir)responsible party in order to shut them down.

Some considered that ECN itself was a layering violation. The reasoning went that the interface to a layer should provide a service to the higher layer and hide how the lower layer does it. However, ECN reveals the state of the network layer and below to the transport layer. A more positive way to describe ECN is that it is like the return value of a function call to the network layer. It explicitly returns the status of the request to deliver a packet, by returning a value representing the current risk that a packet will not be served. Re-ECN has similar semantics, except the transport layer must try to guess the return value, then it can use the actual return value from the network layer to modify the next guess.

The guiding principle behind all the discussion in Section 4.5 on Policing is that any gain from subverting the protocol should be precisely neutralised, rather than punished. If a gain is punished to a greater extent than is sufficient to neutralise it, it will most likely open up a new vulnerability, where the amplifying effect of the punishment mechanism can be turned on others.

For instance, if possible, flows should be removed as soon as they go negative, but we do NOT RECOMMEND any attempts to discard such flows further upstream while they are still positive. Such over-zealous push-back is unnecessary and potentially dangerous. These flows have paid their `fare' up to the point they go negative, so there is no harm in delivering them that far. If someone downstream asks for a flow to be dropped as near to the source as possible, because they say it is going to become negative later, an upstream node cannot test the truth of this assertion. Rather than have to authenticate such messages, re-ECN has been designed so that flows can be dropped solely based on locally measurable evidence. A message hinting that a flow should be watched closely to test for negativity is fine. But not a message that claims that a positive flow will go negative later, so it should be dropped. .

9. Related Work

{Due to lack of time, this section is incomplete. The reader is referred to the Related Work section of [Re-fb] for a brief selection of related ideas.}

9.1. Policing Rate Response to Congestion

ATM network elements send congestion back-pressure messages [ITU-T.I.371] along each connection, duplicating any end to end feedback because they don't trust it. On the other hand, re-ECN ensures information in forwarded packets can be used for congestion management without requiring a connection-oriented architecture and re-using the overhead of fields that are already set aside for end to end congestion control (and routing loop detection in the case of re-TTL in Appendix D).

We borrowed ideas from policers in the literature [pBox],[XCHOKe], AFD etc. for our rate equation policer. However, without the benefit of re-ECN they don't police the correct rate for the condition of their path. They detect unusually high /absolute/ rates, but only while the policer itself is congested, because they work by detecting prevalent flows in the discards from the local RED queue. These policers must sit at every potential bottleneck, whereas our policer need only be located at each ingress to the internetwork. As Floyd & Fall explain [pBox], the limitation of their approach is that a high sending rate might be perfectly legitimate, if the rest of the path is uncongested or the round trip time is short. Commercially available rate policers cap the rate of any one flow. Or they enforce monthly volume caps in an attempt to control high volume file-sharing. They limit the value a customer derives. They might also limit the congestion customers can cause, but only as an accidental side-effect. They actually punish traffic that fills troughs as much as traffic that causes peaks in utilisation. In practice network operators need to be able to allocate service by cost during congestion, and by value at other times.

9.2. Congestion Notification Integrity

The choice of two ECT code-points in the ECN field [RFC3168] permitted future flexibility, optionally allowing the sender to encode the experimental ECN nonce [RFC3540] in the packet stream. This mechanism has since been included in the specifications of DCCP [RFC4340].

The ECN nonce is an elegant scheme that allows the sender to detect if someone in the feedback loop - the receiver especially - tries to claim no congestion was experienced when in fact congestion led to

packet drops or ECN marks. For each packet it sends, the sender chooses between the two ECT codepoints in a pseudo-random sequence. Then, whenever the network marks a packet with CE, if the receiver wants to deny congestion happened, she has to guess which ECT codepoint was overwritten. She has only a 50:50 chance of being correct each time she denies a congestion mark or a drop, which ultimately will give her away.

The purpose of a network-layer nonce should primarily be protection of the network, while a transport-layer nonce would be better used to protect the sender from cheating receivers. Now, the assumption behind the ECN nonce is that a sender will want to detect whether a receiver is suppressing congestion feedback. This is only true if the sender's interests are aligned with the network's, or with the community of users as a whole. This may be true for certain large senders, who are under close scrutiny and have a reputation to maintain. But we have to deal with a more hostile world, where traffic may be dominated by peer-to-peer transfers, rather than downloads from a few popular sites. Often the `natural' self-interest of a sender is not aligned with the interests of other users. It often wishes to transfer data quickly to the receiver as much as the receiver wants the data quickly.

In contrast, the re-ECN protocol enables policing of an agreed rate-response to congestion (e.g. TCP-friendliness) at the sender's interface with the internetwork. It also ensures downstream networks can police their upstream neighbours, to encourage them to police their users in turn. But most importantly, it requires the sender to declare path congestion to the network and it can remove traffic at the egress if this declaration is dishonest. So it can police correctly, irrespective of whether the receiver tries to suppress congestion feedback or whether the sender ignores genuine congestion feedback. Therefore the re-ECN protocol addresses a much wider range of cheating problems, which includes the one addressed by the ECN nonce.

9.3. Identifying Upstream and Downstream Congestion

Purple [Purple] proposes that queues should use the CWR flag in the TCP header of ECN-capable flows to work out path congestion and therefore downstream congestion in a similar way to re-ECN. However, because CWR is in the transport layer, it is not always visible to network layer routers and policers. Purple's motivation was to improve AQM, not policing. But, of course, nodes trying to avoid a policer would not be expected to allow CWR to be visible.

10. Security Considerations

{ToDo: enrich this section}{ToDo: Describe attacks by networks on flows (and by spoofing sources).} {ToDo: Re-ECN & DNS servers}

Nearly the whole of this document concerns security.

11. IANA Considerations

This memo includes no request to IANA.

12. Conclusions

{ToDo:}

13. Acknowledgements

Sebastien Cazalet and Andrea Soppera contributed to the idea of re-feedback. All the following have given helpful comments: Andrea Soppera, David Songhurst, Peter Hovell, Louise Burness, Phil Eardley, Steve Rudkin, Marc Wennink, Fabrice Saffre, Cefn Hoile, Steve Wright, John Davey, Martin Koyabe, Carla Di Cairano-Gilfedder, Alexandru Murgu, Nigel Geffen, Pete Willis, John Adams (BT), Sally Floyd (ICIR), Joe Babiarz, Kwok Ho-Chan (Nortel), Stephen Hailes, Mark Handley (who developed the attack with cancelled packets), Adam Greenhalgh (who developed the attack on DNS) (UCL), Jon Crowcroft (Uni Cam), David Clark, Bill Lehr, Sharon Gillett, Steve Bauer (who complemented our own dummy traffic attacks with others), Liz Maida (MIT), and comments from participants in the CRN/CFP Broadband and DoS-resistant Internet working groups.A special thank you to Alessandro Salvatori for coming up with fiendish attacks on re-ECN.

14. Comments Solicited

Comments and questions are encouraged and very welcome. They can be addressed to the IETF Transport Area working group's mailing list <tsvwg@ietf.org>, and/or to the authors.

15. References

15.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, September 2001.

15.2. Informative References

[Bauer06] Bauer, S., Faratin, P., and R. Beverly, "Assessing the assumptions underlying mechanism design for the Internet", Proc. Workshop on the Economics of Networked Systems (NetEcon06) , June 2006, <http://www.cs.duke.edu/nicl/netecon06/papers/ne06-assessing.pdf>.

[CLoop_pol] Salvatori, A., "Closed Loop Traffic Policing", Politecnico Torino and Institut Eurecom Masters Thesis , September 2005.

[ECN-Deploy] Floyd, S., "ECN (Explicit Congestion Notification) in TCP/IP; Implementation and Deployment of ECN", Web-page , May 2004, <http://www.icir.org/floyd/ecn.html#implementations>.

[Evol_cc] Gibbens, R. and F. Kelly, "Resource pricing and the evolution of congestion control", Automatica 35(12)1969--1985, December 1999, <http://www.statslab.cam.ac.uk/~frank/evol.html>.

[ITU-T.I.371] ITU-T, "Traffic Control and Congestion Control in B-ISDN", ITU-T Rec. I.371 (03/04), March 2004.

[Jiang02] Jiang, H. and D. Dovrolis, "The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm", ACM SIGCOMM CCR 32(3)75-88, July 2002, <http://doi.acm.org/10.1145/571697.571725>.

[Mathis97] Mathis, M., Semke, J., Mahdavi, J., and T. Ott, "The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm", ACM SIGCOMM CCR 27(3)67--82, July 1997, <http://doi.acm.org/10.1145/263932.264023>.

[Purple] Pletka, R., Waldvogel, M., and S. Mannal, "PURPLE: Predictive Active Queue Management Utilizing Congestion Information", Proc. Local Computer Networks (LCN 2003) , October 2003.

[RFC2208] Mankin, A., Baker, F., Braden, B., Bradner, S., O'Dell, M., Romanow, A., Weinrib, A., and L. Zhang, "Resource ReSerVation Protocol (RSVP) Version 1 Applicability Statement Some Guidelines on Deployment", RFC 2208, September 1997.

[RFC3514] Bellovin, S., "The Security Flag in the IPv4 Header", RFC 3514, April 2003.

[RFC3540] Spring, N., Wetherall, D., and D. Ely, "Robust Explicit Congestion Notification (ECN) Signaling with Nonces", RFC 3540, June 2003.

[RFC3714] Floyd, S. and J. Kempf, "IAB Concerns Regarding Congestion Control for Voice Traffic in the Internet", RFC 3714, March 2004.

[RFC4340] Kohler, E., Handley, M., and S. Floyd, "Datagram Congestion Control Protocol (DCCP)", RFC 4340, March 2006.

[RFC4341] Floyd, S. and E. Kohler, "Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 2: TCP-like Congestion Control", RFC 4341, March 2006.

[RFC4342] Floyd, S., Kohler, E., and J. Padhye, "Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 3: TCP-Friendly Rate Control (TFRC)", RFC 4342, March 2006.

[RFC5559] Eardley, P., "Pre-Congestion Notification (PCN) Architecture", RFC 5559, June 2009.

[Re-PCN] Briscoe, B., "Emulating Border Flow Policing using Re-PCN on Bulk Data", draft-briscoe-re-pcn-border-cheat-03 (work in progress), October 2009.

[Re-TCP] Briscoe, B., Jacquet, A., Moncaster, T., and A. Smith, "Re-ECN: Adding Accountability for Causing Congestion to TCP/IP", draft-briscoe-conex-re-ecn-tcp-02 (work in progress), July 2013.

[Re-fb] Briscoe, B., Jacquet, A., Di Cairano-Gilfedder, C., Salvatori, A., Soppera, A., and M. Koyabe, "Policing Congestion Response in an Internetwork Using Re-Feedback", ACM SIGCOMM CCR 35(4)277--288, August 2005, <http://www.acm.org/sigs/sigcomm/sigcomm2005/techprog.html#session8>.

[Savage99] Savage, S., Cardwell, N., Wetherall, D., and T. Anderson, "TCP congestion control with a

misbehaving receiver", ACM SIGCOMM CCR 29(5), October 1999, <http://citeseer.ist.psu.edu/savage99tcp.html>.

[Smart_rtg] Goldenberg, D., Qiu, L., Xie, H., Yang, Y., and Y. Zhang, "Optimizing Cost and Performance for Multihoming", ACM SIGCOMM CCR 34(4)79--92, October 2004, <http://citeseer.ist.psu.edu/698472.html>.

[Steps_DoS] Handley, M. and A. Greenhalgh, "Steps towards a DoS-resistant Internet Architecture", Proc. ACM SIGCOMM workshop on Future directions in network architecture (FDNA'04) pp 49--56, August 2004.

[Tussle] Clark, D., Sollins, K., Wroclawski, J., and R. Braden, "Tussle in Cyberspace: Defining Tomorrow's Internet", ACM SIGCOMM CCR 32(4)347--356, October 2002, <http://www.acm.org/sigcomm/sigcomm2002/papers/tussle.pdf>.

[XCHOKe] Chhabra, P., Chuig, S., Goel, A., John, A., Kumar, A., Saran, H., and R. Shorey, "XCHOKe: Malicious Source Control for Congestion Avoidance at Internet Gateways", Proceedings of IEEE International Conference on Network Protocols (ICNP-02) , November 2002, <http://www.cc.gatech.edu/~akumar/xchoke.pdf>.

[pBox] Floyd, S. and K. Fall, "Promoting the Use of End-to-End Congestion Control in the Internet", IEEE/ACM Transactions on Networking 7(4) 458--472, August 1999, <http://www.aciri.org/floyd/end2end-paper.html>.

[relax-fairness] Briscoe, B., Moncaster, T., and L. Burness, "Problem Statement: Transport Protocols Don't Have To Do Fairness", draft-briscoe-tsvwg-relax-fairness-01 (work in progress), July 2008.

Appendix A. Example Egress Dropper Algorithm

{ToDo: Write up the basic algorithm with flow state, then the aggregated one.}

Appendix B. Policer Designs to ensure Congestion Responsiveness

B.1. Per-user Policing

User policing requires a policer on the ingress interface of the access router associated with the user. At that point, the traffic of the user hasn't diverged on different routes yet; nor has it mixed with traffic from other sources.

In order to ensure that a user doesn't generate more congestion in the network than her due share, a modified bulk token-bucket is maintained with the following parameter:

- b_0 the initial token level
- r the filling rate
- b_max the bucket depth

The same token bucket algorithm is used as in many areas of networking, but how it is used is very different:

- all traffic from a user over the lifetime of their subscription is policed in the same token bucket.
- only positive and cancelled packets (positive, cautious and cancelled) consume tokens

Such a policer will allow network operators to throttle the contribution of their users to network congestion. This will require the appropriate contractual terms to be in place between operators and users. For instance: a condition for a user to subscribe to a given network service may be that she should not cause more than a volume C_user of congestion over a reference period T_user, although she may carry forward up to N_user times her allowance at the end of each period. These terms directly set the parameter of the user policer:

- b_0 = C_user
- r = C_user/T_user
- b_max = b_0 * (N_user +1)

Besides the congestion budget policer above, another user policer may be necessary to further rate-limit cautious packets, if they are to be marked rather than dropped (see discussion in [ref other document].). Rate-limiting cautious packets will prevent high bursts

of new flow arrivals, which is a very useful feature in DoS prevention. A condition to subscribe to a given network service would have to be that a user should not generate more than C_cautious cautious packets, over a reference period T_cautious, with no option to carry forward any of the allowance at the end of each period. These terms directly set the parameters of the cautious packet policer:

- b_0 = C_cautious
- r = C_cautious/T_cautious
- b_max = b_0

T_cautious should be a much shorter period than T_user: for instance T_cautious could be in the order of minutes while T_user could be in order of weeks.

B.2. Per-flow Rate Policing

Whilst we believe that simple per-user policing would be sufficient to ensure senders comply with congestion control, some operators may wish to police the rate response of each flow to congestion as well. Although we do not believe this will be neceesary, we include this section to show how one could perform per-flow policing using enforcement of TCP-fairness as an example. Per-flow policing aims to enforce congestion responsiveness on the shortest information timescale on a network path: packet roundtrips.

This again requires that the appropriate terms be agreed between a network operator and its users, where a congestion responsiveness policy might be required for the use of a given network service (perhaps unless the user specifically requests otherwise).

As an example, we describe below how a rate adaptation policer can be designed when the applicable rate adaptation policy is TCP-compliance. In that context, the average throughput of a flow will be expected to be bounded by the value of the TCP throughput during congestion avoidance, given in Mathis' formula [Mathis97]

$$x_{TCP} = k * s / (T * \sqrt{m})$$

where:

- x_TCP is the throughput of the TCP flow in packets per second,
- k is a constant upper-bounded by sqrt(3/2),

- s is the average packet size of the flow,
- T is the roundtrip time of the flow,
- m is the congestion level experienced by the flow.

We define the marking period N=1/m which represents the average number of packets between two positive or cancelled packets. Mathis' formula can be re-written as:

$$x\_TCP = k*s*sqrt(N)/T$$

We can then get the average inter-mark time in a compliant TCP flow, dt_TCP, by solving (x_TCP/s)*dt_TCP = N which gives

$$dt\_TCP = sqrt(N)*T/k$$

We rely on this equation for the design of a rate-adaptation policer as a variation of a token bucket. In that case a policer has to be set up for each policed flow. This may be triggered by cautious packets, with the remainder of flows being all rate limited together if they do not start with a cautious packet.

Where maintaining per flow state is not a problem, for instance on some access routers, systematic per-flow policing may be considered. Should per-flow state be more constrained, rate adaptation policing could be limited to a random sample of flows exhibiting positive or cancelled packets.

As in the case of user policing, only positive or cancelled packets will consume tokens, however the amount of tokens consumed will depend on the congestion signal.

When a new rate adaptation policer is set up for flow j, the following state is created:

- a token bucket b_j of depth b_max starting at level b_0
- a timestamp t_j = timenow()
- a counter N_j = 0
- a roundtrip estimate T_j
- a filling rate r

When the policing node forwards a packet of flow j with no positive packets:

- . the counter is incremented: N_j += 1

When the policing node forwards a packet of flow j carrying a negative packet:

- the counter is incremented: N_j += 1
- the token level is adjusted: b_j += r*(timenow()-t_j) - sqrt(N_j)* T_j/k
- the counter is reset: N_j = 0
- the timer is reset: t_j = timenow()

An implementation example will be given in a later draft that avoids having to extract the square root.

Analysis: For a TCP flow, for r= 1 token/sec, on average,

r*(timenow()-t_j)-sqrt(N_j)* T_j/k = dt_TCP - sqrt(N)*T/k = 0

This means that the token level will fluctuate around its initial level. The depth b_max of the bucket sets the timescale on which the rate adaptation policy is performed while the filling rate r sets the trade-off between responsiveness and robustness:

- the higher b_max, the longer it will take to catch greedy flows
- the higher r, the fewer false positives (greedy verdict on compliant flows) but the more false negatives (compliant verdict on greedy flows)

This rate adaptation policer requires the availability of a roundtrip estimate which may be obtained for instance from the application of re-feedback to the downstream delay Appendix D or passive estimation [Jiang02].

When the bucket of a policer located at the access router (whether it is a per-user policer or a per-flow policer) becomes empty, the access router SHOULD drop at least all packets causing the token level to become negative. The network operator MAY take further sanctions if the token level of the per-flow policers associated with a user becomes negative.

Appendix C. Downstream Congestion Metering Algorithms

C.1. Bulk Downstream Congestion Metering Algorithm

To meter the bulk amount of downstream congestion in traffic crossing an inter-domain border an algorithm is needed that accumulates the size of positive packets and subtracts the size of negative packets. We maintain two counters:

V_b: accumulated congestion volume

B: total data volume (in case it is needed)

A suitable pseudo-code algorithm for a border router is as follows:

```
=====================================================================
V_b = 0
B   = 0
for each Re-ECN-capable packet {
    b = readLength(packet)      /* set b to packet size          */
    B += b                      /* accumulate total volume       */
    if readEECN(packet) == (positive || cautious {
        V_b += b                /* increment...                  */
    } elseif readEECN(packet) == negative {
        V_b -= b                /* ...or decrement V_b...        */
    }                           /*...depending on EECN field     */
}
=====================================================================
```

At the end of an accounting period this counter V_b represents the congestion volume that penalties could be applied to, as described in Section 4.5.

For instance, accumulated volume of congestion through a border interface over a month might be V_b = 5PB (petabyte = 10^15 byte). This might have resulted from an average downstream congestion level of 1% on an accumulated total data volume of B = 500PB.

{ToDo: Include algorithm for precise downstream congestion.}

C.2. Inflation Factor for Persistently Negative Flows

The following process is suggested to complement the simple algorithm above in order to protect against the various attacks from persistently negative flows described in Section 4.5. As explained in that section, the most important and first step is to estimate the contribution of persistently negative flows to the bulk volume of downstream pre-congestion and to inflate this bulk volume as if these flows weren't there. The process below has been designed to give an unbiased estimate, but it may be possible to define other processes

that achieve similar ends.

While the above simple metering algorithm is counting the bulk of traffic over an accounting period, the meter should also select a subset of the whole flow ID space that is small enough to be able to realistically measure but large enough to give a realistic sample. Many different samples of different subsets of the ID space should be taken at different times during the accounting period, preferably covering the whole ID space. During each sample, the meter should count the volume of positive packets and subtract the volume of negative, maintaining a separate account for each flow in the sample. It should run a lot longer than the large majority of flows, to avoid a bias from missing the starts and ends of flows, which tend to be positive and negative respectively.

Once the accounting period finishes, the meter should calculate the total of the accounts V_{bI} for the subset of flows I in the sample, and the total of the accounts V_{fI} excluding flows with a negative account from the subset I. Then the weighted mean of all these samples should be taken a_S = sum_{forall I} V_{fI} / sum_{forall I} V_{bI}.

If V_b is the result of the bulk accounting algorithm over the accounting period (Appendix C.1) it can be inflated by this factor a_S to get a good unbiased estimate of the volume of downstream congestion over the accounting period a_S.V_b, without being polluted by the effect of persistently negative flows.

Appendix D. Re-TTL

This Appendix gives an overview of a proposal to be able to overload the TTL field in the IP header to monitor downstream propagation delay. This is included to show that it would be possible to take account of RTT if it was deemed desirable.

Delay re-feedback can be achieved by overloading the TTL field, without changing IP or router TTL processing. A target value for TTL at the destination would need standardising, say 16. If the path hop count increased by more than 16 during a routing change, it would temporarily be mistaken for a routing loop, so this target would need to be chosen to exceed typical hop count increases. The TCP wire protocol and handlers would need modifying to feed back the destination TTL and initialise it. It would be necessary to standardise the unit of TTL in terms of real time (as was the original intent in the early days of the Internet).

In the longer term, precision could be improved if routers decremented TTL to represent exact propagation delay to the next

router. That is, for a router to decrement TTL by, say, 1.8 time units it would alternate the decrement of every packet between 1 & 2 at a ratio of 1:4. Although this might sometimes require a seemingly dangerous null decrement, a packet in a loop would still decrement to zero after 255 time units on average. As more routers were upgraded to this more accurate TTL decrement, path delay estimates would become increasingly accurate despite the presence of some RFC3168 compliant routers that continued to always decrement the TTL by 1.

Appendix E. Argument for holding back the ECN nonce

The ECN nonce is a mechanism that allows a /sending/ transport to detect if drop or ECN marking at a congested router has been suppressed by a node somewhere in the feedback loop---another router or the receiver.

Space for the ECN nonce was set aside in [RFC3168] (currently proposed standard) while the full nonce mechanism is specified in [RFC3540] (currently experimental). The specifications for [RFC4340] (currently proposed standard) requires that "Each DCCP sender SHOULD set ECN Nonces on its packets...". It also mandates as a requirement for all CCID profiles that "Any newly defined acknowledgement mechanism MUST include a way to transmit ECN Nonce Echoes back to the sender.", therefore:

- The CCID profile for TCP-like Congestion Control [RFC4341] (currently proposed standard) says "The sender will use the ECN Nonce for data packets, and the receiver will echo those nonces in its Ack Vectors."

- The CCID profile for TCP-Friendly Rate Control (TFRC) [RFC4342] recommends that "The sender [use] Loss Intervals options' ECN Nonce Echoes (and possibly any Ack Vectors' ECN Nonce Echoes) to probabilistically verify that the receiver is correctly reporting all dropped or marked packets."

The primary function of the ECN nonce is to protect the integrity of the information about congestion: ECN marks and packet drops. However, when the nonce is used to protect the integrity of information about packet drops, rather than ECN marks, a transport layer nonce will always be sufficient (because a drop loses the transport header as well as the ECN field in the network header), which would avoid using scarce IP header codepoint space. Similarly, a transport layer nonce would protect against a receiver sending early acknowledgements [Savage99].

If the ECN nonce reveals integrity problems with the information about congestion, the sending transport can use that knowledge for

two functions:

- to protect its own resources, by allocating them in proportion to the rates that each network path can sustain, based on congestion control,

- and to protect congested routers in the network, by slowing down drastically its connection to the destination with corrupt congestion information.

If the sending transport chooses to act in the interests of congested routers, it can reduce its rate if it detects some malicious party in the feedback loop may be suppressing ECN feedback. But it would only be useful to congested routers when /all/ senders using them are trusted to act in interest of the congested routers.

In the end, the only essential use of a network layer nonce is when sending transports (e.g. large servers) want to allocate their /own/ resources in proportion to the rates that each network path can sustain, based on congestion control. In that case, the nonce allows senders to be assured that they aren't being duped into giving more of their own resources to a particular flow. And if congestion suppression is detected, the sending transport can rate limit the offending connection to protect its own resources. Certainly, this is a useful function, but the IETF should carefully decide whether such a single, very specific case warrants IP header space.

In contrast, Re-ECN allows all routers to fully protect themselves from such attacks, without having to trust anyone - senders, receivers, neighbouring networks. Re-ECN is therefore proposed in preference to the ECN nonce on the basis that it addresses the generic problem of accountability for congestion of a network's resources at the IP layer.

Delaying the ECN nonce is justified because the applicability of the ECN nonce seems too limited for it to consume a two-bit codepoint in the IP header. It therefore seems prudent to give time for an alternative way to be found to do the one function the nonce is essential for.

Moreover, while we have re-designed the Re-ECN codepoints so that they do not prevent the ECN nonce progressing, the same is not true the other way round. If the ECN nonce started to see some deployment (perhaps because it was blessed with proposed standard status), incremental deployment of Re-ECN would effectively be impossible, because Re-ECN marking fractions at inter-domain borders would be polluted by unknown levels of nonce traffic.

The authors are aware that Re-ECN must prove it has the potential it claims if it is to displace the nonce. Therefore, every effort has been made to complete a comprehensive specification of Re-ECN so that its potential can be assessed. We therefore seek the opinion of the Internet community on whether the Re-ECN protocol is sufficiently useful to warrant standards action.

Authors' Addresses

Bob Briscoe (editor)
BT
B54/77, Adastral Park
Martlesham Heath
Ipswich IP5 3RE
UK

Phone: +44 1473 645196
EMail: bob.briscoe@bt.com
URI: http://bobbriscoe.net/

Arnaud Jacquet
BT
B54/70, Adastral Park
Martlesham Heath
Ipswich IP5 3RE
UK

Phone: +44 1473 647284
EMail: arnaud.jacquet@bt.com
URI:

Toby Moncaster
Moncaster.com
Dukes
Layer Marney
Colchester CO5 9UZ
UK

EMail: toby@moncaster.com

Alan Smith
BT
B54/76, Adastral Park
Martlesham Heath
Ipswich IP5 3RE
UK

Phone: +44 1473 640404
EMail: alan.p.smith@bt.com