Transport Area Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 15, 2009

B. Briscoe
BT

July 14, 2008

Layered Encapsulation of Congestion Notification
draft-briscoe-tsvwg-ecn-tunnel-01

Status of this Memo

Abstract

This document redefines how the explicit congestion notification (ECN) field of the outer IP header of a tunnel should be constructed. It brings all IP in IP tunnels (v4 or v6) into line with the way IPsec tunnels now construct the ECN field. It includes a thorough analysis of the reasoning for this change and the implications. It also gives guidelines on the encapsulation of IP congestion notification by any outer header, whether encapsulated in an IP tunnel or in a lower layer header. Following these guidelines should help interworking, if the IETF or other standards bodies specify any new encapsulation of congestion notification.

Table of Contents

Changes from previous drafts (to be removed by the RFC Editor)

From -00 to -01:

* Related everything conceptually to the uniform and pipe models of RFC2983 on Diffserv Tunnels, and completely removed the dependence of tunnelling behaviour on the presence of any in-path load regulation by using the [1 - Before] [2 - Outer] function placement concepts from RFC2983.

* Added specifc cases where the existing standards limit new proposals.

* Added sub-structure to Introduction (Need for Rationalisation, Roadmap), added new Introductory subsection on "Scope" and improved clarity

* Added Design Guidelines for New Encapsulations of Congestion Notification

* Considerably clarified the Backward Compatibility section

* Considerably extended the Security Considerations section

* Summarised the primary rationale much better in the conclusions

* Added numerous extra acknowledgements

* Added Appendix A. "Why resetting CE on encapsulation harms PCN", Appendix B. "Contribution to Congestion across a Tunnel" and Appendix C. "Ideal Decapsulation Rules"

* Changed Appendix A "In-path Load Regulation" to "Non-Dependence of Tunnelling on In-path Load Regulation" and added sub-section on "Dependence of In-Path Load Regulation on Tunnelling"

1. Introduction

This document redefines how the explicit congestion notification (ECN) field [RFC3168] of the outer IP header of a tunnel should be constructed. It brings all IP in IP tunnels (v4 or v6) into line with the way IPsec tunnels [RFC4301] now construct the ECN field, ensuring that the outer header reveals any congestion experienced so far on the whole path, not just since the last tunnel ingress.

ECN allows a congested resource to notify the onset of congestion without having to drop packets, by explicitly marking a proportion of

packets with the congestion experienced (CE) codepoint. Because congestion is exhaustion of a physical resource, if the transport layer is to deal with congestion, congestion notification must propagate upwards; from the physical layer to the transport layer. The transport layer can directly detect loss of a packet (or frame) by a lower layer. But if a lower layer marks rather than drops a forward-travelling data packet (or frame) in order to notify incipient congestion, this marking has to be explicitly copied up the layers at every header decapsulation. So, at each decapsulation of an outer (lower layer) header a congestion marking has to be arranged to propagate into the forwarded (upper layer) header. It must continue upwards until it reaches the destination transport. Then typically the destination feeds this congestion notification back to the source transport. Given encapsulation by lower layer headers is functionally similar to tunnelling, it is necessary to arrange similar propagation of congestion notification up the layers. For instance, ECN and its propagation up the layers has recently been specified for MPLS [RFC5129].

As packets pass up the layers, current specifications of decapsulation behaviours are largely all consistent and correct. However, as packets pass down the layers, specifications of encapsulation behaviours are not consistent. This document is primarily aimed at rationalising encapsulation. (Nevertheless, Appendix C explains why the consistency of decapsulation solutions will not last for long and proposes a fix to decapsulation rules as well. The IETF can then discuss whether to rationalise decapsulation at the same time as encapsulation.)

1.1. The Need for Rationalisation

IPsec tunnel mode is a specific form of tunnelling that can hide the inner headers. Because the ECN field has to be mutable, it cannot be covered by IPsec encryption or authentication calculations. Therefore concern has been raised in the past that the ECN field could be used as a low bandwidth covert channel to communicate with someone on the unprotected public Internet even if an end-host is restricted to only communicate with the public Internet through an IPsec gateway. However, the updated version of IPsec [RFC4301] chose not to block this covert channel, deciding that the threat could be managed given the channel bandwidth is so limited (ECN is a 2-bit field).

An unfortunate sequence of standards actions leading up to this latest change in IPsec has left us with nearly the worst of all possible combinations of outcomes, despite the best endeavours of everyone concerned. The controversy has been over whether to reveal information about congestion experienced on the path upstream of the

tunnel ingress. Even though this has various uses if it is revealed in the outer header of a tunnel, when ECN was standardised[RFC3168] it was decided that all IP in IP tunnels should hide this upstream congestion simply to avoid the extra complexity of two different mechanisms for IPsec and non-IPsec tunnels. However, now that [RFC4301] IPsec tunnels deliberately no longer hide this information, we are left in the perverse position where non-IPsec tunnels still hide congestion information unnecessarily. This document is designed to correct that anomaly.

Specifically, RFC3168 says that, if a tunnel fully supports ECN (termed a 'full-functionality' ECN tunnel in [RFC3168]), the tunnel ingress must not copy a CE marking from the inner header into the outer header that it creates. Instead the tunnel ingress has to set the ECN field of the outer header to ECT(0) (i.e. codepoint 10). We term this 'resetting' a CE codepoint. However, RFC4301 reverses this, stating that the tunnel ingress must simply copy the ECN field from the inner to the outer header. The main purpose of this document is to carry the new behaviour of IPsec over to all IP in IP tunnels, so all tunnel ingress nodes consistently copy the ECN field.

Why does it matter if we have different ECN encapsulation behaviours for IPsec and non-IPsec tunnels? The general argument is that gratuitous inconsistency constrains the available design space and makes it harder to design networks and new protocols that work predictably.

Already complicated constraints have had to be added to a standards track congestion marking proposal. The section of the pre-congestion notification (PCN) architecture [I-D.ietf-pcn-architecture] on tunnelling says PCN works correctly in the presence of RFC4301 IPsec encapsulation (and RFC5129 MPLS encapsulation). However it doesn't work with RFC3168 IP in IP encapsulation (Appendix A explains why).

Section 3 assesses further security, control and management functions that cannot be achieved in each case (resetting vs copying CE markings). It finds that resetting CE makes life difficult in a number of directions, while copying CE harms nothing (other than opening a low bit-rate covert channel vulnerability which the Security Area deems is manageable).

1.2. Document Roadmap

Most of the document gives a thorough analysis of the knock-on effects of the apparently minor change to tunnel encapsulation. The reader may jump to Section 5 if only interested in standards actions impacting implementation. The whole document is organised as follows:

- S.5 of RFC3168 permits the Diffserv codepoint (DSCP)[RFC2474] to 'switch in' different behaviours for marking the ECN field, just as it switches in different per-hop behaviours (PHBs) for scheduling. Therefore we cannot only discuss the ECN protocol that RFC3168 gives as a default. We need to also give guidance for possible different marking schemes. Therefore in Section 3 we lay out the design constraints when tunnelling congestion notification.

- Then in Section 4 we resolve the tensions between these constraints to give general design principles and guidelines on how a tunnel should process congestion notification; principles that could apply to any marking behaviour for any PHB, not just the default in RFC3168. In particular, we examine the underlying principles behind whether CE should be reset or copied into the outer header at the ingress to a tunnel--or indeed at the ingress of any layered encapsulation of headers with congestion notification fields. We end this section with a bulleted list of more design guidelines for new encapsulations of congestion notification.

- Section 5 then uses precise standards terminology to confirm the rules for the default ECN tunnelling behaviour based on the above design principles.

- Extending the new IPsec tunnel ingress behaviour to all IP in IP tunnels requires consideration of backwards compatibility, which is covered in Section 6 and changes from earlier RFCs are brought together in Section 7.

- Finally, a number of security considerations are discussed and conclusions are drawn.

1.3. Scope

This document only concerns wire protocol processing at tunnel endpoints and makes no changes or recommendations concerning algorithms for congestion marking or congestion response.

This document specifies a common, default congestion encapsulation for any IP in IP tunnelling, based on that now specified for IPsec. It applies irrespective of whether IPv4 or IPv6 is used for either of the inner and outer headers. It applies to all PHBs, unless stated otherwise in the specification of a PHB. It is intended to be a good trade off between somewhat conflicting security, control and management requirements.

Nonetheless, if necessary, an alternate congestion encapsulation

behaviour can be introduced as part of the definition of an alternate congestion marking scheme used by a specific Diffserv PHB (see S.5 of [RFC3168] and [RFC4774]). When designing such new encapsulation schemes, the principles in Section 4 should be followed as closely as possible. There is no requirement for a PHB to state anything about ECN tunnelling behaviour if the default behaviour is sufficient.

Often lower layer resources (e.g. a point-to-point Ethernet link) are arranged to be protected by higher layer buffers, so instead of congestion occurring at the lower layer, it merely causes the queue from the higher layer to overflow. Such non-blocking link and physical layer technologies do not have to implement congestion notification, which can be introduced solely in the active queue management (AQM) from the IP layer. However, not all link layer technologies are always protected from congestion by buffers at higher layers (e.g. a subnetwork of Ethernet links and switches can congest internally). In these cases, when adding congestion notification to lower layers, we have to arrange for it to be explicitly copied up the layers, just as when IP is tunnelled in IP.

As well as guiding alternate IP in IP tunnelling schemes, the design guidelines of Section 4 are intended to be followed when IP packets are encapsulated by any connectionless datagram/packet/frame where the outer header is designed to support a congestion notification capability. [RFC5129] already deals with handling ECN for IP in MPLS and MPLS in MPLS, and S.9.3 of [RFC3168] lists IP encapsulated in L2TP [RFC2661], GRE [RFC1701] or PPTP [RFC2637] as possible examples where ECN may be added in future.

Of course, the IETF does not have standards authority over every link or tunnel protocol, so this document merely aims to define the interface between IP ECN and lower layer congestion notification. Then the IETF or the relevant standards body can be free to define the specifics of each lower layer scheme, but a common interface should ensure interworking across all technologies.

Note that just because there is forward congestion notification in a lower layer protocol, if the lower layer has its own feedback and load regulation, there is no need to propagate it up the layers. For instance, FECN (forward ECN) has been present in Frame Relay and EFCI (explicit forward congestion indication) in ATM [ITU-T.I.371] for a long time, but they have been used for internal management rather than being propagated to endpoint transports for them to control end-to-end congestion.

[RFC2983] is a comprehensive primer on differentiated services and tunnels. Given ECN raises similar issues to differentiated services when interacting with tunnels, useful concepts introduced in RFC2983

are used throughout, with brief recaps of the explanations where necessary.

2. Requirements Language

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 RFC 2119 [RFC2119].

3. Design Constraints

Tunnel processing of a congestion notification field has to meet congestion control needs without creating new information security vulnerabilities (if information security is required).

3.1. Security Constraints

Information security can be assured by using various end to end security solutions (including IPsec in transport mode [RFC4301]), but a commonly used scenario involves the need to communicate between two physically protected domains across the public Internet. In this case there are certain management advantages to using IPsec in tunnel mode solely across the publicly accessible part of the path. The path followed by a packet then crosses security 'domains'; the ones protected by physical or other means before and after the tunnel and the one protected by an IPsec tunnel across the otherwise unprotected domain. We will use the scenario in Figure 1 where endpoints 'A' and 'B' communicate through a tunnel with ingress 'I' and egress 'E' within physically protected edge domains across an unprotected internetwork where there may be 'men in the middle', M.

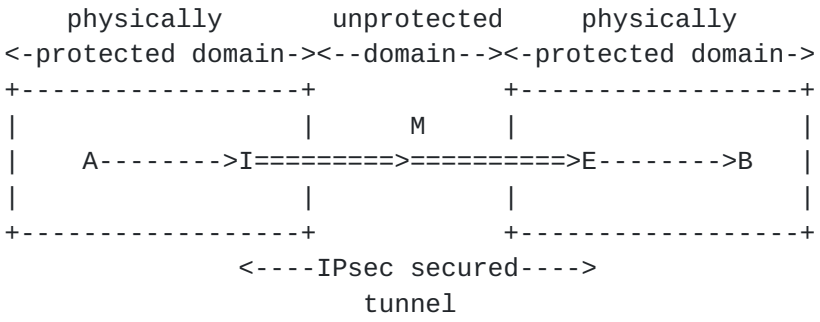

Figure 1: IPsec Tunnel Scenario

IPsec encryption is typically used to prevent 'M' seeing messages from 'A' to 'B'. IPsec authentication is used to prevent 'M' masquerading as the sender of messages from 'A' to 'B' or altering

their contents. But 'I' can also use IPsec tunnel mode to allow 'A' to communicate with 'B', but impose encryption to prevent 'A' leaking information to 'M'. Or 'E' can insist that 'I' uses tunnel mode authentication to prevent 'M' communicating information to 'B'. Mutable IP header fields such as the ECN field (as well as the TTL/ Hop Limit and DS fields) cannot be included in the cryptographic calculations of IPsec. Therefore, if 'I' copies these mutable fields into the outer header that is exposed across the tunnel it will have allowed a covert channel from 'A' to M that bypasses its encryption of the inner header. And if 'E' copies these fields from the outer header to the inner, even if it validates authentication from 'I', it will have allowed a covert channel from 'M' to 'B'.

ECN at the IP layer is designed to carry information about congestion from a congested resource towards downstream nodes. Typically a downstream transport might feed the information back somehow to the point upstream of the congestion that can regulate the load on the congested resource, but other actions are possible (see [RFC3168] S.6). In terms of the above unicast scenario, ECN is typically intended to create an information channel from 'M' to 'B', for 'B' to forward to 'A'. Therefore the goals of IPsec and ECN are mutually incompatible.

With respect to the DS or ECN fields, S.5.1.2 of RFC4301 says, "controls are provided to manage the bandwidth of this [covert] channel". Using the ECN processing rules of RFC4301, the channel bandwidth is two bits per datagram from 'A' to 'M' and one bit per datagram from 'M' to 'A' (because 'E' limits the combinations of the 2-bit ECN field that it will copy). In both cases the covert channel bandwidth is further reduced by noise from any real congestion marking. RFC4301 therefore implies that these covert channels are sufficiently limited to be considered a manageable threat. However, with respect to the larger (6b) DS field, the same section of RFC4301 says not copying is the default, but a configuration option can allow copying "to allow a local administrator to decide whether the covert channel provided by copying these bits outweighs the benefits of copying". Of course, an administrator considering copying of the DS field has to take into account that it could be concatenated with the ECN field giving an 8b per datagram covert channel.

Thus, for tunnelling the 6b Diffserv field two conceptual models have had to be defined so that administrators can trade off security against the needs of traffic conditioning [RFC2983]:

The uniform model: where the DIffserv field is preserved end-to-end by copying into the outer header on encapsulation and copying from the outer header on decapsulation.

The pipe model: where the outer header is independent of that in the inner header so it hides the Diffserv field of the inner header from any interaction with nodes along the tunnel.

However, for ECN, the new IPsec security architecture in RFC4301 only standardised one tunnelling model equivalent to the uniform model. It deemed that simplicity was more important than allowing administrators the option of a tiny increment in security especially given not copying congestion indications could seriously harm everyone's network service.

3.2. Control Constraints

Congestion control requires that any congestion notification marked into packets by a resource will be able to traverse a feedback loop back to a node capable of controlling the load on that resource. To be precise, rather than calling this node the data source, we will call it the Load Regulator. This will allow us to deal with exceptional cases where load is not regulated by the data source, but usually the two terms will be synonymous. Note the term "a node _capable of_ controlling the load" deliberately includes a source application that doesn't actually control the load but ought to (e.g. an application without congestion control that uses UDP).

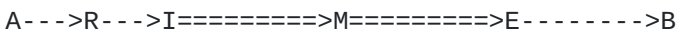

Figure 2: Simple Tunnel Scenario

We now consider a similar tunnelling scenario to the IPsec one just described, but without the different security domains so we can just focus on ensuring the control loop and management monitoring can work (Figure 2). If we want resources in the tunnel to be able to explicitly notify congestion and the feedback path is from 'B' to 'A', it will certainly be necessary for 'E' to copy any CE marking from the outer header to the inner header for onward transmission to 'B', otherwise congestion notification from resources like 'M' cannot be fed back to the Load Regulator ('A'). But it doesn't seem necessary for 'I' to copy CE markings from the inner to the outer header. For instance, if resource 'R' is congested, it can send congestion information to 'B' using the congestion field in the inner header without 'I' copying the congestion field into the outer header and 'E' copying it back to the inner header. 'E' can still write any additional congestion marking introduced across the tunnel into the congestion field of the inner header.

It might be useful for the tunnel egress to be able to tell whether

congestion occurred across a tunnel or upstream of it. If outer header congestion marking was reset by the tunnel ingress ('I'), at the end of a tunnel ('E') the outer headers would indicate congestion experienced across the tunnel ('I' to 'E'), while the inner header would indicate congestion upstream of 'I'. But similar information can be gleaned even if the tunnel ingress copies the inner to the outer headers. At the end of the tunnel ('E'), any packet with an _extra_ mark in the outer header relative to the inner header indicates congestion across the tunnel ('I' to 'E'), while the inner header would still indicate congestion upstream of ('I'). Appendix B gives a more precise method for inferring the congestion level introduced across a tunnel.

All this shows that 'E' can preserve the control loop irrespective of whether 'I' copies congestion notification into the outer header or resets it.

That is the situation for existing control arrangements but, because copying reveals more information, it would open up possibilities for better control system designs. For instance, Appendix A describes how resetting CE marking at a tunnel ingress confuses a proposed congestion marking scheme on the standards track. It ends up removing excessive amounts of traffic unnecessarily. Whereas copying CE markings at ingress leads to the correct control behaviour.

3.3. Management Constraints

As well as control, there are also management constraints. Specifically, a management system may monitor congestion markings in passing packets, perhaps at the border between networks as part of a service level agreement. For instance, monitors at the borders of autonomous systems may need to measure how much congestion has accumulated since the original source to determine between them how much of the congestion is contributed by each domain.

Therefore, when monitoring the middle of a path, it should be possible to establish how far back in the path congestion markings have accumulated from. In this document we term this the baseline of congestion marking (or the Congestion Baseline), i.e. the source of the layer that last reset (or created) the congestion notification field. Given some tunnels cross domain borders (e.g. consider M in Figure 2 is monitoring a border), it would therefore be desirable for 'I' to copy congestion accumulated so far into the outer headers exposed across the tunnel.

Appendix D discusses various scenarios where the Load Regulator lies in-path, not at the source host as we would typically expect. It concludes that a Congestion Baseline is determined by where the Load

Regulator function is, which should be identified in the transport layer, not by addresses in network layer headers. This applies whether the Load Regulator is at the source host or within the path. The appendix also discusses where a Load Regulator function should be located relative to a local encapsulation function.

4. Design Principles

The constraints from the three perspectives of security, control and management in Section 3 are somewhat in tension as to whether a tunnel ingress should copy congestion markings into the outer header it creates or reset them. From the control perspective either copying or resetting works for existing arrangements, but copying has more potential for simplifying control. From the management perspective copying is preferable. From the security perspective resetting is preferable but copying is now considered acceptable given the bandwidth of a 2-bit covert channel can be managed.

Therefore an outer encapsulating header capable of carrying congestion markings SHOULD reflect accumulated congestion since the last interface designed to regulate load (the Load Regulator). This implies congestion notification SHOULD be copied into the outer header of each new encapsulating header that supports it.

We have said that a tunnel ingress SHOULD (as opposed to MUST) copy incoming congestion notification into an outer encapsulating header that supports it. In the case of 2-bit ECN, the IETF security area has deemed the benefit always outweighs the risk. Therefore for 2-bit ECN we can and we will say 'MUST' (Section 5). But in this section where we are setting down general design principles, we leave it as a 'SHOULD'. This allows for future multi-bit congestion notification fields where the risk from the covert channel created by copying congestion notification might outweigh the congestion control benefit of copying.

The Load Regulator is the node to which congestion feedback should be returned by the next downstream node with a transport layer feedback function (typically but not always the data receiver). The Load Regulator is not always (or even typically) the same thing as the node identified by the source address of the outermost exposed header. In general the addressing of the outermost encapsulation header says nothing about the identifiers of either the upstream or the downstream transport layer functions. As long as the transport functions know each other's addresses, they don't have to be identified in the network layer or in any link layer. It was only a convenience that a TCP receiver assumed that the address of the source transport is the same as the network layer source address of

an IP packet it receives.

More generally, the return transport address for feedback could be identified solely in the transport layer protocol. For instance, a signalling protocol like RSVP [RFC2205] breaks up a path into transport layer hops and informs each hop of the address of its transport layer neighbour without any need to identify these hops in the network layer. RSVP can be arranged so that these transport layer hops are bigger than the underlying network layer hops. The host identity protocol (HIP) architecture [RFC4423] also supports the same principled separation (for mobility amongst other things), where the transport layer sender identifies its transport address for feedback to be sent to, using an identifier provided by a shim below the transport layer.

Keeping to this layering principle deliberately doesn't require a network layer packet header to reveal the origin address from where congestion notification accumulates (its Congestion Baseline). It is not necessary for the network and lower layers to know the address of the Load Regulator. Only the destination transport needs to know that. With forward congestion notification, the network and link layers only notify congestion forwards; they aren't involved in feeding it backwards. If they are (e.g. backward congestion notification (BCN) in Ethernet [IEEE802.1au] or EFCI in ATM [ITU-T.I.371]), that should be considered as a transport function added to the lower layer, which must sort out its own addressing. Indeed, this is one reason why ICMP source quench is now deprecated [RFC1254]; when congestion occurs within a tunnel it is complex (particularly in the case of IPsec tunnels) to return the ICMP messages beyond the tunnel ingress back to the Load Regulator.

Similarly, if a management system is monitoring congestion and needs to know the Congestion Baseline, the management system has to find this out from the transport; in general it cannot tell solely by looking at the network or link layer headers.

4.1. Design Guidelines for New Encapsulations of Congestion Notification

The following guidelines are for specifications of new schemes for encapsulating congestion notification (e.g. for specialised Diffserv PHBs in IP, or for lower layer technologies):

1. Congestion notification in outer headers SHOULD be relative to a Congestion Baseline at the node expected to regulate the load on the link in question (the Load Regulator). This implies incoming congestion notifications from the higher layer SHOULD be copied into encapsulating headers. This guideline is particularly

important where outer headers might cross trust boundaries, but less important otherwise.

2. Congestion notification MUST NOT simply be copied from outer headers to the forwarded header on decapsulation. The forwarded congestion notification field SHOULD be calculated from the inner and outer headers, taking account of the following, in the order given:

    1. If the inner header does not support congestion notification, or indicates that the transport does not support congestion notification, any explicit congestion notifications in the outer header will not be understood if propagated further, so if the only way to indicate congestion to onward nodes is to drop the packet, it MUST be dropped.

    2. If the outer header does not support explicit congestion notification, but the inner header does, the inner header SHOULD be forwarded unchanged.

    3. Congestion indications may be ranked by strength. For instance no congestion would be the weakest indication, with possibly increasing levels of congestion given increasingly stronger indications.

    4. Where the inner and outer headers carry indications of congestion of different strengths, the stronger indication SHOULD be forwarded in preference to the weaker. Obviously, if the strengths in both inner and outer are the same, the same strength should be forwarded.

    5. If the outer header carries a weaker indication of congestion than the inner, it MAY be appropriate to raise a warning, as this would be in illegal combination if Guideline Paragraph 1 had been followed.

3. Where framing boundaries are different between the two layers, congestion indications SHOULD be propagated on the basis that a congestion indication in a packet or frame applies to all the octets in the frame/packet. On average, a tunnel endpoint SHOULD approximately preserve the number of marked octets arriving and leaving. An algorithm for spreading congestion indications over multiple smaller `fragments' SHOULD propagate congestion indications as soon as they arrive, and SHOULD NOT hold them back for later frames.

4. Assumptions on incremental deployment MUST be stated.

Regarding incremental deployment, the Per-Domain ECT Checking of[RFC5129] is a good example to follow. In this example, header space in the lower layer protocol (MPLS) was extremely limited, so no ECN-capable transport codepoint was added to the MPLS header. Interior nodes in a domain were allowed to set explicit congestion indications without checking whether the frame was destined for a transport that would understand them. This was made safe by emphasising repeatedly that all the decapsulating edges of a whole domain had to be upgraded at once, so there would always be a check that the higher layer transport was ECN-capable on decapsulation. If the decapsulator discovered that the higher layer showed the transport would not understand ECN, it dropped the packet on behalf of the earlier congestion node (see Guideline Paragraph 2.1).

Note that such a deployment strategy that assumes a savvy operator was only appropriate because MPLS is targeted solely at professional operators. This strategy would not be appropriate for other link technologies (e.g. Ethernet) targeted at deployment by the general public.

5. Default ECN Tunnelling Rules

The following ECN tunnel processing rules are the default for a packet with any DSCP. If required, different ECN encapsulation rules MAY be defined as part of the definition of an appropriate Diffserv PHB using the guidelines in Section 4. However, the burden of handling exceptional PHBs in implementations of all affected tunnels and lower layer link protocols should not be underestimated.

A tunnel ingress compliant with this specification MUST copy the 2-bit ECN field of the arriving IP header into the outer encapsulating IP header, for all types of IP in IP tunnel. This encapsulation behaviour MUST only be used if the tunnel ingress is in `normal state'. A `compatibility state' with a different encapsulation behaviour is also specified in Section 6 for backward compatibility with legacy tunnel egresses that do not understand ECN.

To decapsulate the inner header at the tunnel egress, a compliant tunnel egress MUST set the outgoing ECN field to the codepoint at the intersection of the appropriate incoming inner header (row) and outer header (column) in Table 1.

Incoming Inner Header	+--Incoming Outer Header--- Not-ECT	ECT(0)	ECT(1)	CE
Not-ECT	Not-ECT	drop(!!!)	drop(!!!)	drop(!!!)
ECT(0)	ECT(0)	ECT(0)	ECT(0)	CE
ECT(1)	ECT(1)	ECT(1)	ECT(1)	CE
CE	CE	CE	CE (!!!)	CE

+-----Outgoing Header------

Table 1: IP in IP Decapsulation

The exclamation marks '(!!!)' in Table 1 indicate that this combination of inner and outer headers should not be possible if only legal transitions have taken place. So, the decapsulator should drop or mark the ECN field as the table specifies, but it MAY also raise an appropriate alarm. It MUST NOT raise an alarm so often that the illegal combinations would amplify into a flood of alarm messages.

6. Backward Compatibility

Note: in RFC3168, a tunnel was in one of two modes: limited functionality or full functionality. Rather than working with modes of the tunnel as a whole, this specification uses the term `state' to refer separately to the state of each tunnel end point, which is how implementations have to work.

If one end of an IPsec tunnel is compliant with [RFC4301], the other end can be guaranteed to also be [RFC4301] compliant (there could be corner cases where manual keying is used, but they will be ignored here). So there is no backward compatibility problem with IKEv2 RFC4301 IPsec tunnels. But once we extend our scope to any IP in IP tunnel, we have to cater for the possibility that a legacy tunnel egress may not know how to process an ECN field, so if ECN capable outer headers were sent towards a legacy (e.g. [RFC2003]) egress, it would most likely simply disregard the outer headers, dangerously discarding information about congestion experienced within the tunnel. ECN-capable traffic sources would not see any congestion feedback and instead continually ratchet up their share of the bandwidth without realising that cross-flows from other ECN sources were continually having to ratchet down.

To be compliant with this specification a tunnel ingress that does

not always know the ECN capability of its tunnel egress MUST implement a 'normal' state and a 'compatibility' state, and it MUST initiate each negotiated tunnel in the compatibility state.

However, a tunnel ingress can be compliant even if it only implements the 'normal state' of encapsulation behaviour, but only as long as it is designed or configured so that all possible tunnel egress nodes it will ever talk to will have full ECN functionality (RFC3168 full functionality mode, RFC4301 and this present specification). The `normal state' is that defined in Section 5 (i.e. header copying). Note that a [RFC4301] tunnel ingress that has used IKEv2 key management [RFC4306] can guarantee that its tunnel egress is also RFC4301-compliant and therefore need not further negotiate ECN capabilities.

Before switching to normal state, a compliant tunnel ingress that does not know the egress ECN capability MUST negotiate with the tunnel egress. If the egress says it is in full functionality state (or mode), the ingress puts itself into normal state. In normal state the ingress follows the encapsulation rule in Section 5 (i.e. header copying). If the egress says it is not in full-functionality state/mode or doesn't understand the question, the tunnel ingress MUST remain in compatibility state.

A tunnel ingress in compatibility state MUST set all outer headers to Not-ECT. This is the same per packet behaviour as the ingress end of RFC3168's limited functionality mode.

A tunnel ingress that only implements compatibility state is at least safe with the ECN behaviour of any egress it may encounter (any of RFC2003, RFC2401, either mode of RFC2481 and RFC3168's limited functionality mode). But an ingress cannot claim compliance with this specification simply by disabling ECN processing across the tunnel. A compliant tunnel ingress MUST at least implement `normal state' and, if it might be used with arbitrary tunnel egress nodes, it MUST also implement `compatibility state'.

A compliant tunnel egress on the other hand merely needs to implement the one behaviour in Section 5, which we term 'full-functionality' state, as it is the same as the egress end of the full-functionality mode of [RFC3168]. It is also the same as the [RFC4301] egress behaviour.

The decapsulation rules for the egress of the tunnel in Section 5 have been defined in such a way that congestion control will still work safely if any of the earlier versions of ECN processing are used unilaterally at the encapsulating ingress of the tunnel (any of RFC2003, RFC2401, either mode of RFC2481, either mode of RFC3168,

RFC4301 and this present specification). If a tunnel ingress tries to negotiate to use limited functionality mode or full functionality mode [RFC3168], a decapsulating tunnel egress compliant with this specification MUST agree to either request, as its behaviour will be the same in both cases.

For 'forward compatibility', a compliant tunnel egress SHOULD raise a warning about any requests to enter states or modes it doesn't recognise, but it can continue operating. If no ECN-related state or mode is requested, a compliant tunnel egress need not raise an error or warning as its egress behaviour is compatible with all the legacy ingress behaviours that don't negotiate capabilities.

Implementation note: if a compliant node is the ingress for multiple tunnels, a state setting will need to be stored for each tunnel ingress. However, if a node is the egress for multiple tunnels, none of the tunnels will need to store a state setting, because a compliant egress can only be in one state.

7. Changes from Earlier RFCs

The rule that a normal state tunnel ingress MUST copy any ECN field into the outer header is a change to the ingress behaviour of RFC3168, but it is the same as the rules for IPsec tunnels in RFC4301.

The rules for calculating the outgoing ECN field on decapsulation at a tunnel egress are in line with the full functionality mode of ECN in RFC3168 and with RFC4301, except that neither identified that an outer header of ECT(1) combined with an inner header of CE was an illegal combination.

The rules for how a tunnel establishes whether the egress has full functionality ECN capabilities are an update to RFC3168. For all the typical cases, RFC4301 is not updated by the ECN capability check in this specification, because a typical RFC4301 tunnel ingress will have already established that it is talking to an RFC4301 tunnel egress (e.g. if it uses IKEv2). However, there may be some corner cases (e.g. manual keying) where an RFC4301 tunnel ingress talks with an egress with limited functionality ECN handling. Strictly, for such corner cases, the requirement to use compatibility mode in this specification updates RFC4301.

The optional ECN Tunnel field in the IPsec security association database (SAD) and the optional ECN Tunnel Security Association Attribute defined in RFC3168 are no longer needed. The security association (SA) has no policy on ECN usage, because all RFC4301

tunnels now support ECN without any policy choice.

RFC3168 defines a (required) limited functionality mode and an (optional) full functionality mode for a tunnel, but RFC4301 doesn't need modes. In this specification only the ingress might need two states: a normal state (required) and a compatibility state (required in some scenarios, optional in others). The egress needs only full-functionality state which handles ECN the same as either mode of RFC3168 or RFC4301.

Additional changes to the RFC Index (to be removed by the RFC Editor):

In the RFC index, RFC3168 should be identified as an update to RFC2003 and RFC4301 should be identified as an update to RFC3168.

This specification updates RFC3168. It also suggests a minor optional warning and a corner-case change to RFC4301, but these don't really count as an update.

8. IANA Considerations

This memo includes no request to IANA.

9. Security Considerations

Section 3.1 discusses the security constraints imposed on ECN tunnel processing. The Design Principles of Section 4 trade-off between security (covert channels) and congestion monitoring & control. In fact, ensuring congestion markings are not lost is itself another aspect of security, because if we allowed congestion notification to be lost, any attempt to enforce a response to congestion would be much harder.

If alternate congestion notification semantics are defined for a certain PHB (e.g. the pre-congestion notification architecture [I-D.ietf-pcn-architecture]), the scope of the alternate semantics might typically be bounded by the limits of a Diffserv region or regions, as envisaged in [RFC4774]. The inner headers in tunnels crossing the boundary of such a Diffserv region but ending within the region can potentially leak the external congestion notification semantics into the region, or leak the internal semantics out of the region. [RFC2983] discusses the need for Diffserv traffic conditioning to be applied at these tunnel endpoints as if they are at the edge of the Diffserv region. Similar concerns apply to any processing or propagation of the ECN field at the edges of a Diffserv region with alternate ECN semantics. Such edge processing must also

be applied at the endpoints of tunnels with ends both inside and outside the domain. [I-D.ietf-pcn-architecture] gives specific advice on this for the PCN case, but other definitions of alternate semantics will need to discuss the specific security implications in their case.

With the rules as they stand in RFC3168 and RFC4301, a small part of the protection of the ECN nonce [RFC3540] is compromised. One reason two ECT codepoints were defined was to enable the data source to detect if a CE marking had been applied then subsequently removed. The source could detect this by weaving a pseudo-random sequence of ECT(0) and ECT(1) values into a stream of packets, which is termed an ECN nonce. By the decapsulation rules in RFC3168 and RFC4301, if the inner and outer headers carry contradictory ECT values only the inner header is preserved for onward forwarding. So if a CE marking added to the outer ECN field has been illegally (or accidentally) suppressed by a subsequent node in the tunnel, the decapsulator will revert the ECN field to its value before tampering, hiding all evidence of the crime from the onward feedback loop. To close this loophole, we could have specified that an outer header value of ECT should overwrite a contradictory ECT value in the inner header (for how, see the ideal decapsulation rules proposed in Appendix C). But currently we choose to keep the 'broken' behaviour defined in RFC3168 & RFC4301 for all the following reasons:

1. We wanted to avoid any changes to IPsec tunnelling behaviour;

2. Allowing ECT values in the outer header to override the inner header would have increased the bandwidth of the covert channel through the egress gateway from 1 to 1.5 bit per datagram, potentially threatening to upset the consensus established in the security area that says that the bandwidth of this covert channel can now be safely managed;

3. This loophole is only applicable in the corner case where the attacker is a network node downstream of a congested node in the same tunnel;

4. In tunnelling scenarios, the ECN nonce is already vulnerable to suppression by nodes downstream of a congested node in the same tunnel, if they can copy the ECT value in the inner header to the outer header (any node in the tunnel can do this if the inner header is not encrypted, and an IPsec tunnel egress can do it whether or not the tunnel is encrypted);

5. Although the 'broken' decapsulation behaviour removes evidence of congestion suppression from the onward feedback loop, the decapsulator itself can at least detect that congestion within

the tunnel has been suppressed;

6. The ECN nonce [RFC3540] currently has experimental status and there has been no evidence that anyone has implemented it beyond the author's prototype.

If a legacy security policy configures a legacy tunnel ingress to negotiate to turn off ECN processing, a compliant tunnel egress will agree to a request to turn off ECN processing but it will actually still copy CE markings from the outer to the forwarded header. Although the tunnel ingress 'I' in Figure 1 will set all ECN fields in outer headers to Not-ECT, 'M' could still toggle CE on and off to communicate covertly with 'B', because we have specified that 'E' only has one mode regardless of what mode it says it has negotiated. We could have specified that 'E' should have a limited functionality mode and check for such behaviour. But we decided not to add the extra complexity of two modes on a compliant tunnel egress merely to cater for a legacy security concern that is now considered manageable.

10. Conclusions

This document updates the ingress tunnelling encapsulation of RFC3168 ECN for all IP in IP tunnels to bring it into line with the new behaviour in the IPsec architecture of RFC4301.

At a tunnel egress, header decapsulation for the default ECN marking behaviour is broadly unchanged except that one exceptional case has been catered for. At the ingress, for all forms of IP in IP tunnel, encapsulation has been brought into line with the new IPsec rules in RFC4301 which copy rather than reset CE markings when creating outer headers.

This change to encapsulation has been motivated by analysis from the three perspectives of security, control and management. They are somewhat in tension as to whether a tunnel ingress should copy congestion markings into the outer header it creates or reset them. From the control perspective either copying or resetting works for existing arrangements, but copying has more potential for simplifying control and resetting breaks at least one proposal already on the standards track. From the management and monitoring perspective copying is preferable. From the network security perspective (theft of service etc) copying is preferable. From the information security perspective resetting is preferable, but the IETF Security Area now considers copying acceptable given the bandwidth of a 2-bit covert channel can be managed. Therefore there are no points against copying and a number against resetting CE on ingress.

The change ensures ECN processing in all IP in IP tunnels reflects this slightly more permissive attitude to revealing congestion information in the new IPsec architecture. Once all tunnelling of ECN works the same, ECN markings will have a defined meaning when measured at any point in a network. This new certainty will enable new uses of the ECN field that would otherwise be confounded by ambiguity.

Also, this document defines more generic principles to guide the design of alternate forms of tunnel processing of congestion notification, if required for specific Diffserv PHBs or for other lower layer encapsulating protocols that might support congestion notification in the future.

11. Acknowledgements

Thanks to David Black for explaining a better way to think about function placement and to Louise Burness for a better way to think about multilayer transports and networks, having read [Patterns_Arch]. Also thanks to Arnaud Jacquet for ideas behind the algorithms in Appendix B. Thanks to Bruce Davie, Toby Moncaster, Gorry Fairhurst, Sally Floyd, Alfred Hoenes and Gabriele Corliano for their thoughts and careful review comments.

12. Comments Solicited

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

13. References

13.1. Normative References

[RFC2003] Perkins, C., "IP Encapsulation within IP", RFC 2003, October 1996.

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

[RFC2474] Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, December 1998.

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

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

13.2. Informative References

[I-D.eardley-pcn-marking-behaviour]
Eardley, P., "Marking behaviour of PCN-nodes", draft-eardley-pcn-marking-behaviour-01 (work in progress), June 2008.

[I-D.ietf-pcn-architecture]
Eardley, P., "Pre-Congestion Notification Architecture", draft-ietf-pcn-architecture-03 (work in progress), February 2008.

[I-D.ietf-pwe3-congestion-frmwk]
Bryant, S., Davie, B., Martini, L., and E. Rosen, "Pseudowire Congestion Control Framework", draft-ietf-pwe3-congestion-frmwk-01 (work in progress), May 2008.

[I-D.moncaster-pcn-3-state-encoding]
Moncaster, T., Briscoe, B., and M. Menth, "A three state extended PCN encoding scheme", draft-moncaster-pcn-3-state-encoding-00 (work in progress), June 2008.

[IEEE802.1au]
IEEE, "IEEE Standard for Local and Metropolitan Area Networks--Virtual Bridged Local Area Networks - Amendment 10: Congestion Notification", 2008, <http://www.ieee802.org/1/pages/802.1au.html>.

(Work in Progress; Access Controlled link within page)

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

[PCNcharter]
IETF, "Congestion and Pre-Congestion Notification (pcn)", IETF w-g charter , Feb 2007, <http://www.ietf.org/html.charters/pcn-charter.html>.

[Patterns_Arch] Day, J., "Patterns in Network Architecture: A Return to Fundamentals", Pub: Prentice Hall ISBN-13: 9780132252423, Jan 2008.

[RFC1254] Mankin, A. and K. Ramakrishnan, "Gateway Congestion Control Survey", RFC 1254, August 1991.

[RFC1701] Hanks, S., Li, T., Farinacci, D., and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 1701, October 1994.

[RFC2205] Braden, B., Zhang, L., Berson, S., Herzog, S., and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, September 1997.

[RFC2637] Hamzeh, K., Pall, G., Verthein, W., Taarud, J., Little, W., and G. Zorn, "Point-to-Point Tunneling Protocol", RFC 2637, July 1999.

[RFC2661] Townsley, W., Valencia, A., Rubens, A., Pall, G., Zorn, G., and B. Palter, "Layer Two Tunneling Protocol "L2TP"", RFC 2661, August 1999.

[RFC2983] Black, D., "Differentiated Services and Tunnels", RFC 2983, October 2000.

[RFC3426] Floyd, S., "General Architectural and Policy Considerations", RFC 3426, November 2002.

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

[RFC4306] Kaufman, C., "Internet Key Exchange (IKEv2) Protocol", RFC 4306, December 2005.

[RFC4423] Moskowitz, R. and P. Nikander, "Host Identity Protocol (HIP) Architecture", RFC 4423, May 2006.

[RFC4774] Floyd, S., "Specifying Alternate Semantics for the Explicit Congestion Notification (ECN) Field", BCP 124, RFC 4774, November 2006.

[RFC5129] Davie, B., Briscoe, B., and J. Tay, "Explicit Congestion Marking in MPLS", RFC 5129, January 2008.

[Shayman] "Using ECN to Signal Congestion Within an MPLS Domain", 2000, <http://www.ee.umd.edu/~shayman/papers.d/

draft-shayman-mpls-ecn-00.txt>.

(Expired)

Appendix A. Why resetting CE on encapsulation harms PCN

Regarding encapsulation, the section of the PCN architecture [I-D.ietf-pcn-architecture] on tunnelling says that header copying (RFC4301) allows PCN to work correctly. However, resetting CE markings confuses PCN marking.

The specific issue here concerns PCN excess rate marking [I-D.eardley-pcn-marking-behaviour], i.e. the bulk marking of traffic that exceeds a configured threshold rate. One of the goals of excess rate marking is to enable the speedy removal of excess admission controlled traffic following re-routes caused by link failures or other disasters. This maintains a share of the capacity for competing admission controlled traffic and for traffic in lower priority classes. After failures, traffic re-routed onto remaining links can often stress multiple links along a path. Therefore, traffic can arrive at a link under stress with some proportion already marked for removal by a previous link. By design, marked traffic will be removed by the overall system in subsequent round trips. So when the excess rate marking algorithm decides how much traffic to mark for removal, it doesn't include traffic already marked for removal by another node upstream (the `Excess traffic meter function' of [I-D.eardley-pcn-marking-behaviour]).

However, if an RFC3168 tunnel ingress intervenes, it resets the ECN field in all the outer headers, hiding all the evidence of problems upstream. Thus, although excess rate marking works fine with RFC4301 IPsec tunnels, with RFC3168 tunnels it typically removes large volumes of traffic that it didn't need to remove at all.

Appendix B. Contribution to Congestion across a Tunnel

This specification mandates that a tunnel ingress determines the ECN field of each new outer tunnel header by copying the arriving header. If instead the outer ECN field were reset at a tunnel ingress (as it was for the full functionality mode of RFC3168), it would be possible for the tunnel egress to measure:

- congestion marking before the tunnel ingress (fraction of inner header markings, p_i);

- congestion marking across the tunnel (fraction of outer header markings, p_t);

- congestion marking after the tunnel egress (fraction of departing header markings, p_o).

Although the newly mandated copying behaviour at ingress gains the advantages described in the body of this specification, this one advantage of the resetting behaviour of RFC3168 seems to have been lost: on first impressions, it seems that the egress can no longer accurately measure congestion contributed along the tunnel (p_t). The egress could _estimate _the contribution along the tunnel by measure which packets carry only a mark in the outer header (not the inner). But this is not precisely the same as the congestion contributed along the tunnel; tunnel nodes may have tried to mark some packets that already had a marking in both the inner and outer header. Measuring only additional outer markings will miss these. Nonetheless, with the newly proposed scheme, a tunnel egress can derive a precise estimate of marking introduced across a tunnel (p_t) as follows.

The combined fraction of markings at the tunnel egress will be p_o = 1 - (1 - p_i)(1 - p_t). Explanation: this is (1 - the probability a departing packet is not marked), which is (1 - (prob not marked before tunnel)(prob not marked along tunnel)). Therefore, rearranging, the egress can infer the fraction of marks introduced across the tunnel as p_t = (p_o - p_i)/(1 - p_i). If arriving congestion is low (p_i <<1), then the approximation p_t ~ (p_o - p_i) should be good enough. This is the estimate we advised originally; i.e. measuring only the extra markings in the outer header that are not present in the inner header. If a better approximation is needed p_t ~ (p_o - p_i)(1 + p_i), which removes the division, but still assumes p_i<<1.

Using any of these formulae (including the precise one), it would be possible for a tunnel egress to calculate a moving average of the fraction of packets being marked by tunnel nodes, including those already marked in the inner header. Alternatively, it should even be possible for a tunnel egress to reverse engineer which packets would have been marked across the tunnel if CE was reset on ingress even if CE was actually copied on ingress.[[anchor3: Note from Bob: I've worked out an algorithm so the tunnel egress can reverse engineer marking as if CE was reset at the ingress even though CE was copied at the ingress. It typically consumes 2 cycles / pkt, occasionally 4 and very occasionally 8. {ToDo: On testing an implementation just now it still has a wrinkle in it, but with a little more development I believe it would work well. I'll write it into the next revision if I get it working.}]]

Appendix C. Ideal Decapsulation Rules

Compliance with this appendix is NOT REQUIRED for compliance with the present specification.

If the default ECN encapsulation behaviour does not offer suitable trade offs, procedures exist for associating a new behaviour with a new Diffserv PHB. However, it is unrealistic to expect vendors of all IPSec and all IP in IP tunnel endpoints to cater for the exceptional behaviour of PHB XXX. If all tunnels did require XXX-specific behaviour, the resulting patchy and error-prone deployment would probably cause XXX to suffer byzantine feature interactions with poorly implemented tunnels. The default rules for tunnel endpoints to handle both the Diffserv field and the ECN field should 'just work' when handling packets with an XXX Diffserv codepoint.

Given this specification requests a standards action to update the RFC3168 encapsulation behaviour, this appendix explores a further change to decapsulation that we ought to specify at the same time. If instead this further change is added later, it will add another set of backward compatibility combinations to the already complicated change history of ECN tunnelling.

Multi-level congestion notification is currently on the IETF's standards track agenda in the Congestion and Pre-Congestion Notification (PCN) working group. The PCN working group requires three congestion states (not marked and two levels of congestion marking) [I-D.ietf-pcn-architecture]. The aim is for the first level of marking to stop admitting new traffic and the second level to terminate sufficient existing flows to bring a network back to its operating point after a serious failure.

Although the ECN field gives sufficient codepoints for these three states, the PCN working group cannot use them in case any tunnel decapsulations occur within a PCN region. If a node in a tunnel sets the ECN field to ECT(0) or ECT(1), this change will be discarded by a tunnel egress compliant with RFC4301 and RFC3168. This can be seen in Table 1, where the ECT values in the outer header are ignored unless the inner header is the same. Effectively the ECT(0) and ECT(1) codepoints have to be treated as just one codepoint when they could otherwise have been used for their intended purpose of congestion notification. Instead, the PCN w-g has had to propose using extra Diffserv codepoint(s) to encode the extra states [I-D.moncaster-pcn-3-state-encoding], using up the rapidly exhausting DSCP space while leaving ECN codepoints unused.

Although this is currently most pressing for the PCN working group, the issue is more general. Under Security Considerations (Section 9)

it has already been explained that a data sender cannot use the experimental ECN nonce [RFC3540] to detect suppression of congestion notification along a tunnel.

More generally, the currently standardised tunnel decapsulation behaviour unnecessarily wastes a quarter of two bits (i.e. half a bit) in the IP (v4 & v6) header. As explained in Section 3.1, the original reason for not copying down outer ECT codepoints for onward forwarding was to limit the covert channel across a decapsulator to 1 bit per packet. However, now that the IETF Security Area has deemed that a 2-bit covert channel through an encapsulator is a manageable risk, the same should be true for a decapsulator.

Table 2 proposes a more ideal layered decapsulation behaviour. Note: this table is only to support discussion. It is not currently proposed for standards action. The only difference from Table 1 (that is proposed for standards action), is the swapping of the cells highlighted as *ECT(X)*.

	+--Incoming Outer Header---			
Incoming Inner Header	Not-ECT	ECT(0)	ECT(1)	CE
Not-ECT	Not-ECT	drop(!!!)	drop(!!!)	drop(!!!)
ECT(0)	ECT(0)	ECT(0)	*ECT(1)*	CE
ECT(1)	ECT(1)	*ECT(0)*	ECT(1)	CE
CE	CE	CE	CE (!!!)	CE
	+-----Outgoing Header------			

Table 2: Ideal IP in IP Decapsulation (currently NOT REQUIRED)

Note that, if this ideal proposal were taken up, extra backwards compatibility issues would have to be resolved.

Appendix D. Non-Dependence of Tunnelling on In-path Load Regulation

We have said that at any point in a network, the Congestion Baseline (where congestion notification starts from zero) should be the previous upstream Load Regulator. We have also said that the ingress of an IP in IP tunnel must copy congestion indications to the encapsulating outer headers it creates. If the Load Regulator is in-path rather than at the source, and also a tunnel ingress, these two requirements seem to be contradictory. A tunnel ingress must not

reset incoming congestion, but a Load Regulator must be the Congestion Baseline, implying it needs to reset incoming congestion.

In fact, the two requirements are not contradictory, because a Load Regulator and a tunnel ingress are functions within a node that occur in sequence on a stream of packets, not at the same point. Figure 3 is borrowed from [RFC2983] (which was making a similar point about the location of Diffserv traffic conditioning relative to the encapsulation function of a tunnel). An in-path Load Regulator can act on packets either at [1 - Before] encapsulation or at [2 - Outer] after encapsulation. Load Regulation does not ever need to be integrated with the [Encapsulate] function (but it can be for efficiency). Therefore we can still maintain that the [Encapsulate] function always copies CE into the outer header.

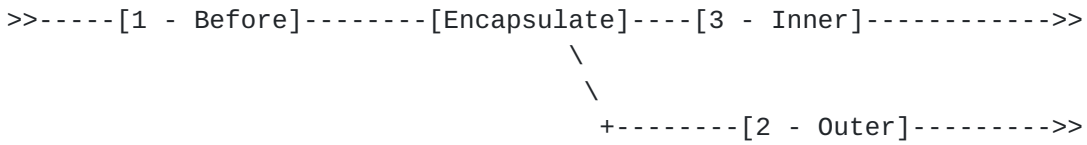

Figure 3: Placement of In-Path Load Regulator Relative to Tunnel Ingress

Then separately, if there is a Load Regulator at location [2 - Outer], it might reset CE to ECT(0), say. Then the Congestion Baseline for the lower layer (outer) will be [2 - Outer], while the Congestion Baseline of the inner layer will be unchanged. But how encapsulation works has nothing to do with whether a Load Regulator is present or where it is.

If on the other hand a Load Regulator resets CE at [1 - Before], the Congestion Baseline of both the inner and outer headers will be [1 - Before]. But again, encapsulation is independent of load regulation.

D.1. Dependence of In-Path Load Regulation on Tunnelling

Although encapsulation doesn't need to depend on in-path load regulation, the reverse is not true. The placement of an in-path Load Regulator must be carefully considered relative to encapsulation. Some examples are given in the following for guidance.

In the traditional Internet architecture one tends to think of the source host as the Load Regulator for a path. It is generally not desirable or practical for a node part way along the path to regulate the load. However, various reasonable proposals for in-path load

regulation have been made from time to time (e.g. fair queuing, traffic engineering, flow admission control). The IETF has recently chartered a working group to standardise admission control across a part of a path using pre-congestion notification (PCN) [PCNcharter]. This is of particular relevance here because it involves congestion notification with an in-path Load Regulator, it can involve tunnelling and it certainly involves encapsulation more generally.

We will use the more complex scenario in Figure 4 to tease out all the issues that arise when combining congestion notification and tunnelling with various possible in-path load regulation schemes. In this case 'I1' and 'E2' break up the path into three separate congestion control loops. The feedback for these loops is shown going right to left across the top of the figure. The 'V's are arrow heads representing the direction of feedback, not letters. But there are also two tunnels within the middle control loop: 'I1' to 'E1' and 'I2' to 'E2'. The two tunnels might be VPNs, perhaps over two MPLS core networks. M is a congestion monitoring point, perhaps between two border routers where the same tunnel continues unbroken across the border.

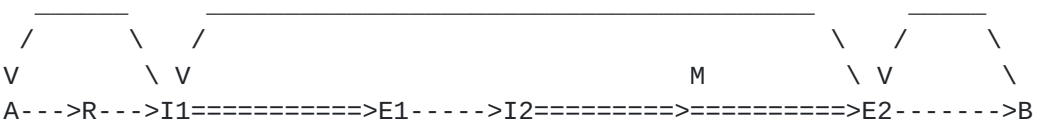

Figure 4: complex Tunnel Scenario

The question is, should the congestion markings in the outer exposed headers of a tunnel represent congestion only since the tunnel ingress or over the whole upstream path from the source of the inner header (whatever that may mean)? Or put another way, should 'I1' and 'I2' copy or reset CE markings?

Based on the design principles in Section 4, the answer is that the Congestion Baseline should be the nearest upstream interface designed to regulate traffic load--the Load Regulator. In Figure 4 'A', 'I1' or 'E2' are all Load Regulators. We have shown the feedback loops returning to each of these nodes so that they can regulate the load causing the congestion notification. So the Congestion Baseline exposed to M should be 'I1' (the Load Regulator), not 'I2'. Therefore I1 should reset any arriving CE markings. In this case, 'I1' knows the tunnel to 'E1' is unrelated to its load regulation function. So the load regulation function within 'I1' should be placed at [1 - Before] tunnel encapsulation within 'I1' (using the terminology of Figure 3). Then the Congestion Baseline all across the networks from 'I1' to 'E2' in both inner and outer headers will be 'I1'.

The following further examples illustrate how this answer might be applied:

- We argued in Appendix A that resetting CE on encapsulation could harm PCN excess rate marking, which marks excess traffic for removal in subsequent round trips. This marking relies on not marking packets if another node upstream has already marked them for removal. If there were a tunnel ingress between the two which reset CE markings, it would confuse the downstream node into marking far too much traffic for removal. So why do we say that 'I1' should reset CE, while a tunnel ingress shouldn't? The answer is that it is the Load Regulator function at 'I1' that is resetting CE, not the tunnel encapsulator. The Load Regulator needs to set itself as the Congestion Baseline, so the feedback it gets will only be about congestion on links it can relieve itself by regulating the load into them. When it resets CE markings, it knows that something else upstream will have dealt with the congestion notifications it removes, given it is part of an end-to-end admission control signalling loop. It therefore knows that previous hops will be covered by other Load Regulators. Meanwhile, the tunnel ingresses at both 'I1' and 'I2' should follow the new rule for any tunnel ingress and copy congestion marking into the outer tunnel header. The ingress at 'I1' will happen to copy headers that have already been reset just beforehand. But it doesn't need to know that.

- [Shayman] suggested feedback of ECN accumulated across an MPLS domain could cause the ingress to trigger re-routing to mitigate congestion. This case is more like the simple scenario of Figure 2, with a feedback loop across the MPLS domain ('E' back to 'I'). I is a Load Regulator because re-routing around congestion is a load regulation function. But in this case 'I' should only reset itself as the Congestion Baseline in outer headers, as it is not handling congestion outside its domain, so it must preserve the end-to-end congestion feedback loop for something else to handle (probably the data source). Therefore the Load Regulator within 'I' should be placed at [2 - Outer] to reset CE markings just after the tunnel ingress has copied them from arriving headers. Again, the tunnel encapsulation function at 'I' simply copies incoming headers, unaware that the load regulator will subsequently reset its outer headers.

- The PWE3 working group of the IETF is considering the problem of how and whether an aggregate edge-to-edge pseudo-wire emulation should respond to congestion [I-D.ietf-pwe3-congestion-frmwk]. Although the study is still at the requirements stage, some (controversial) solution proposals include in-path load regulation at the ingress to the tunnel that could lead to tunnel

arrangements with similar complexity to that of Figure 4.

These are not contrived scenarios--they could be a lot worse. For instance, a host may create a tunnel for IPsec which is placed inside a tunnel for Mobile IP over a remote part of its path. And around this all we may have MPLS labels being pushed and popped as packets pass across different core networks. Similarly, it is possible that subnets could be built from link technology (e.g. future Ethernet switches) so that link headers being added and removed could involve congestion notification in future Ethernet link headers with all the same issues as with IP in IP tunnels.

One reason we introduced the concept of a Load Regulator was to allow for in-path load regulation. In the traditional Internet architecture one tends to think of a host and a Load Regulator as synonymous, but when considering tunnelling, even the definition of a host is too fuzzy, whereas a Load Regulator is a clearly defined function. Similarly, the concept of innermost header is too fuzzy to be able to (wrongly) say that the source address of the innermost header should be the Congestion Baseline. Which is the innermost header when multiple encapsulations may be in use? Where do we stop? If we say the original source in the above IPsec-Mobile IP case is the host, how do we know it isn't tunnelling an encrypted packet stream on behalf of another host in a p2p network?

We have become used to thinking that only hosts regulate load. The end to end design principle advises that this is a good idea [RFC3426], but it also advises that it is solely a guiding principle intended to make the designer think very carefully before breaking it. We do have proposals where load regulation functions sit within a network path for good, if sometimes controversial, reasons, e.g. PCN edge admission control gateways [I-D.ietf-pcn-architecture] or traffic engineering functions at domain borders to re-route around congestion [Shayman]. Whether or not we want in-path load regulation, we have to work round the fact that it will not go away.

Author's Address

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

Phone: +44 1473 645196
Email: bob.briscoe@bt.com
URI: http://www.cs.ucl.ac.uk/staff/B.Briscoe/

Full Copyright Statement

Intellectual Property

Acknowledgment

This document was produced using xml2rfc v1.33 (of http://xml.resource.org/) from a source in RFC-2629 XML format.