Network Working Group
Internet-Draft
Obsoletes: rfc5320 (if approved)
Updates: rfc2460 (if approved)
Intended status: Standards Track
Expires: July 7, 2014

F. Templin, Ed.
Boeing Research & Technology

January 03, 2014

The Subnetwork Encapsulation and Adaptation Layer (SEAL) draft-templin-intarea-seal-68.txt

Abstract

This document specifies a Subnetwork Encapsulation and Adaptation Layer (SEAL). SEAL operates over virtual topologies configured over connected IP network routing regions bounded by encapsulating border nodes. These virtual topologies are manifested by tunnels that may span multiple IP and/or sub-IP layer forwarding hops, where they may incur packet duplication, packet reordering, source address spoofing and traversal of links with diverse Maximum Transmission Units (MTUs). SEAL addresses these issues through the encapsulation and messaging mechanisms specified in this document.

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 July 7, 2014.

Table of Contents

1. Introduction

As Internet technology and communication has grown and matured, many techniques have developed that use virtual topologies (manifested by tunnels of one form or another) over an actual network that supports the Internet Protocol (IP) [RFC0791][RFC2460]. Those virtual topologies have elements that appear as one network layer hop, but are actually multiple IP or sub-IP layer hops which comprise the "subnetwork" over which the tunnel operates.

The use of IP encapsulation (also known as "tunneling") has long been considered as the means for creating such virtual topologies (e.g., see [RFC2003][RFC2473]). Tunnels serve a wide variety of purposes, including mobility, security, routing control, traffic engineering, multihoming, etc., and will remain an integral part of the architecture moving forward. However, the encapsulation headers often include insufficiently provisioned per-packet identification values. IP encapsulation also allows an attacker to produce encapsulated packets with spoofed source addresses even if the source address in the encapsulating header cannot be spoofed. A denial-of-service vector that is not possible in non-tunneled subnetworks is therefore presented.

Additionally, the insertion of an outer IP header reduces the effective Maximum Transmission Unit (MTU) visible to the inner network layer. When IPv6 is used as the encapsulation protocol, original sources expect to be informed of the MTU limitation through IPv6 Path MTU discovery (PMTUD) [RFC1981]. When IPv4 is used, this reduced MTU can be accommodated through the use of IPv4 fragmentation, but unmitigated in-the-network fragmentation has been deemed harmful through operational experience and studies conducted over the course of many years [FRAG][FOLK][RFC4963]. Additionally, classical IPv4 PMTUD [RFC1191] has known operational issues that are exacerbated by in-the-network tunnels [RFC2923][RFC4459].

The following subsections present further details on the motivation and approach for addressing these issues.

1.1. Motivation

Before discussing the approach, it is necessary to first understand the problems. In both the Internet and private-use networks today, IP is ubiquitously deployed as the Layer 3 protocol. The primary functions of IP are to provide for routing, addressing, and a fragmentation and reassembly capability used to accommodate links with diverse MTUs. While it is well known that the IP address space is rapidly becoming depleted, there is also a growing awareness that other IP protocol limitations have already or may soon become

problematic.

First, the Internet historically provided no means for discerning whether the source addresses of IP packets are authentic. This shortcoming is being addressed more and more through the deployment of site border router ingress filters [RFC2827], however the use of encapsulation provides a vector for an attacker to circumvent filtering for the encapsulated packet even if filtering is correctly applied to the encapsulation header. Secondly, the IP header does not include a well-behaved identification value unless the source has included a fragment header for IPv6 or unless the source permits fragmentation for IPv4. These limitations preclude an efficient means for routers to detect duplicate packets and packets that have been re-ordered within the subnetwork. Additionally, recent studies have shown that the arrival of fragments at high data rates can cause denial-of-service (DoS) attacks on performance-sensitive networking gear, prompting some administrators to configure their equipment to drop fragments unconditionally [I-D.taylor-v6ops-fragdrop].

For IPv4 encapsulation, when fragmentation is permitted the header includes a 16-bit Identification field, meaning that at most 2^16 unique packets with the same (source, destination, protocol)-tuple can be active in the network at the same time [RFC6864]. (When middleboxes such as Network Address Translators (NATs) re-write the Identification field to random values, the number of unique packets is even further reduced.) Due to the escalating deployment of high-speed links, however, these numbers have become too small by several orders of magnitude for high data rate packet sources such as tunnel endpoints [RFC4963].

Furthermore, there are many well-known limitations pertaining to IPv4 fragmentation and reassembly - even to the point that it has been deemed "harmful" in both classic and modern-day studies (see above). In particular, IPv4 fragmentation raises issues ranging from minor annoyances (e.g., in-the-network router fragmentation [RFC1981]) to the potential for major integrity issues (e.g., mis-association of the fragments of multiple IP packets during reassembly [RFC4963]).

As a result of these perceived limitations, a fragmentation-avoiding technique for discovering the MTU of the forward path from a source to a destination node was devised through the deliberations of the Path MTU Discovery Working Group (MTUDWG) during the late 1980's through early 1990's which resulted in the publication of [RFC1191]. In this negative feedback-based method, the source node provides explicit instructions to routers in the path to discard the packet and return an ICMP error message if an MTU restriction is encountered. However, this approach has several serious shortcomings that lead to an overall "brittleness" [RFC2923].

In particular, site border routers in the Internet have been known to discard ICMP error messages coming from the outside world. This is due in large part to the fact that malicious spoofing of error messages in the Internet is trivial since there is no way to authenticate the source of the messages [RFC5927]. Furthermore, when a source node that requires ICMP error message feedback when a packet is dropped due to an MTU restriction does not receive the messages, a path MTU-related black hole occurs. This means that the source will continue to send packets that are too large and never receive an indication from the network that they are being discarded. This behavior has been confirmed through documented studies showing clear evidence of PMTUD failures for both IPv4 and IPv6 in the Internet today [TBIT][WAND][SIGCOMM][RIPE].

The issues with both IP fragmentation and this "classical" PMTUD method are exacerbated further when IP tunneling is used [RFC4459]. For example, a tunnel ingress may be required to forward encapsulated packets into the subnetwork on behalf of hundreds, thousands, or even more original sources. If the ITE allows IP fragmentation on the encapsulated packets, persistent fragmentation could lead to undetected data corruption due to Identification field wrapping and/or reassembly congestion at the tunnel egress. If the ingress instead uses classical IP PMTUD it must rely on ICMP error messages coming from the subnetwork that may be suspect, subject to loss due to filtering middleboxes, or insufficiently provisioned for translation into error messages to be returned to the original sources.

Although recent works have led to the development of a positive feedback-based end-to-end MTU determination scheme [RFC4821], they do not excuse tunnels from accounting for the encapsulation overhead they add to packets. Moreover, in current practice existing tunneling protocols mask the MTU issues by selecting a "lowest common denominator" MTU that may be much smaller than necessary for most paths and difficult to change at a later date. Therefore, a new approach to accommodate tunnels over links with diverse MTUs is necessary.

1.2. Approach

This document concerns subnetworks manifested through a virtual topology configured over a connected network routing region and bounded by encapsulating border nodes. Example connected network routing regions include Mobile Ad hoc Networks (MANETs), enterprise networks, aviation networks and the global public Internet itself. Subnetwork border nodes forward unicast and multicast packets over the virtual topology across multiple IP and/or sub-IP layer forwarding hops that may introduce packet duplication and/or traverse

links with diverse Maximum Transmission Units (MTUs).

This document introduces a Subnetwork Encapsulation and Adaptation Layer (SEAL) for tunneling inner network layer protocol packets over IP subnetworks that connect Ingress and Egress Tunnel Endpoints (ITEs/ETEs) of border nodes. It provides a modular specification designed to be tailored to specific associated tunneling protocols. (A transport-mode of operation is also possible but out of scope for this document.)

SEAL treats tunnels that traverse the subnetwork as ordinary links that must support network layer services. Moreover, SEAL provides dynamic mechanisms (including limited fragmentation and reassembly) to ensure a maximal path MTU over the tunnel. This is in contrast to static approaches which avoid MTU issues by selecting a lowest common denominator MTU value that may be overly conservative for the vast majority of tunnel paths and difficult to change even when larger MTUs become available.

1.3. Differences with RFC5320

This specification of SEAL is descended from an experimental independent RFC publication of the same name [RFC5320]. However, this specification introduces a number of fundamental differences from the earlier publication. This specification therefore obsoletes (i.e., and does not update) [RFC5320].

First, [RFC5320] forms a 32-bit Identification value by concatenating the 16-bit IPv4 Identification field with a 16-bit Identification "extension" field in the SEAL header. This means that [RFC5320] can only operate over IPv4 networks (since IPv6 headers do not include a 16-bit version number) and that the SEAL Identification value can be corrupted if the Identification in the outer IPv4 header is rewritten. In contrast, this specification includes a 32-bit Identification value that is independent of any identification fields found in the inner or outer IP headers, and is therefore compatible with any inner and outer IP protocol version combinations.

Additionally, the SEAL fragmentation and reassembly procedures defined in [RFC5320] differ significantly from those found in this specification. In particular, this specification defines an 13-bit Offset field that allows for finer-grained fragment sizes when SEAL fragmentation and reassembly is necessary. In contrast, [RFC5320] includes only a 3-bit Segment field and performs reassembly through concatenation of consecutive segments.

Finally, SEAL no longer uses the IPv4 fragmentation sensing method specified in [RFC5320] as well as in earlier versions of this

document. This departure is based on the fact that there is no way for the ITE or ETE to control the way in which middleboxes perform IPv4 fragmentation (e.g., largest fragment first, smallest fragment first, all fragments the same size, etc.). Moreover, there may be middleboxes in the path that reassemble IPv4 fragmented packets before delivering them to the ETE as the final destination. Use of IPv4 fragmentation sensing in the ETE also greatly complicated the specification and proved difficult to implement. Therefore, although the IPv4 fragmentation sensing method is conceptually elegant and natural, it is no longer included.

2. Terminology

The following terms are defined within the scope of this document:

subnetwork
: a virtual topology configured over a connected network routing region and bounded by encapsulating border nodes.

IP
: used to generically refer to either Internet Protocol (IP) version, i.e., IPv4 or IPv6.

Ingress Tunnel Endpoint (ITE)
: a portal over which an encapsulating border node (host or router) sends encapsulated packets into the subnetwork.

Egress Tunnel Endpoint (ETE)
: a portal over which an encapsulating border node (host or router) receives encapsulated packets from the subnetwork.

inner packet
: an unencapsulated network layer protocol packet (e.g., IPv4 [RFC0791], OSI/CLNP [RFC0994], IPv6 [RFC2460], etc.) before any outer encapsulations are added. Internet protocol numbers that identify inner packets are found in the IANA Internet Protocol registry [RFC3232]. SEAL protocol packets that incur an additional layer of SEAL encapsulation are also considered inner packets.

outer IP packet
: a packet resulting from adding an outer IP header (and possibly other outer headers) to a SEAL-encapsulated inner packet.

packet-in-error
the leading portion of an invoking data packet encapsulated in the body of an error control message (e.g., an ICMPv4 [RFC0792] error message, an ICMPv6 [RFC4443] error message, etc.).

Packet Too Big (PTB) message
a control plane message indicating an MTU restriction (e.g., an ICMPv6 "Packet Too Big" message [RFC4443], an ICMPv4 "Fragmentation Needed" message [RFC0792], etc.).

Don't Fragment (DF) bit
a bit that indicates whether the packet may be fragmented by the network. The DF bit is explicitly included in the IPv4 header [RFC0791] and may be set to '0' to allow fragmentation or '1' to disallow further in-network fragmentation. The bit is absent from the IPv6 header [RFC2460], but implicitly set to '1' because fragmentation can occur only at IPv6 sources.

3. Requirements

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]. When used in lower case (e.g., must, must not, etc.), these words MUST NOT be interpreted as described in [RFC2119], but are rather interpreted as they would be in common English.

4. Applicability Statement

SEAL was originally motivated by the specific case of subnetwork abstraction for Mobile Ad hoc Networks (MANETs), however the domain of applicability also extends to subnetwork abstractions over enterprise networks, mobile networks, aviation networks, ISP networks, SO/HO networks, the global public Internet itself, and any other connected network routing region.

SEAL provides a network sublayer used during encapsulation of an inner network layer packet within outer encapsulating headers. SEAL can also be used as a sublayer within a transport layer protocol data payload, where transport layer encapsulation is typically used for Network Address Translator (NAT) traversal as well as operation over subnetworks that give preferential treatment to certain "core" Internet protocols, e.g., TCP, UDP, etc. (However, note that TCP encapsulation may not be appropriate for all use cases; particularly those that require low delay and/or delay variance.) The SEAL header is processed in the same manner as for IPv6 extension headers, i.e.,

it is not part of the outer IP header but rather allows for the creation of an arbitrarily extensible chain of headers in the same way that IPv6 does.

To accommodate MTU diversity, the Ingress Tunnel Endpoint (ITE) may need to perform limited fragmentation which the Egress Tunnel Endpoint (ETE) reassembles. The ITE and ETE further engage in minimal path probing to determine when the path can be traversed without fragmentation. This allows the ITE to send whole packets instead of fragmented packets whenever possible.

In practice, SEAL is typically used as an encapsulation sublayer in conjunction with existing tunnel types such as IPsec [RFC4301] , GRE[RFC1701], IP-in-IPv6 [RFC2473], IP-in-IPv4 [RFC4213][RFC2003], etc. When used with existing tunnel types that insert mid-layer headers between the inner and outer IP headers (e.g., IPsec, GRE, etc.), the SEAL header is inserted between the mid-layer headers and outer IP header.

5. SEAL Specification

The following sections specify the operation of SEAL:

5.1. SEAL Tunnel Model

SEAL is an encapsulation sublayer used within point-to-point, point-to-multipoint, and non-broadcast, multiple access (NBMA) tunnels. SEAL can also be used with multicast-capable tunnels, but the path probing mechanisms specified in the following sections may not always be sufficient to determine an optimal MTU for a multicast group.

Each tunnel is configured over one or more underlying interfaces attached to subnetwork links, where each link represents a different subnetwork path. The tunnel connects an ITE to one or more ETE "neighbors" via encapsulation across an underlying subnetwork, where each tunnel neighbor relationship is maintained over one or more subnetwork paths. The tunnel neighbor relationship may be bidirectional, partially unidirectional or fully unidirectional.

A bidirectional tunnel neighbor relationship is one over which both tunnel endpoints can exchange both data and control messages. A partially unidirectional tunnel neighbor relationship allows the near end ITE to send data packets forward to the far end ETE, while the far end only returns control messages when necessary. Finally, a fully unidirectional mode of operation is one in which the near end ITE can receive neither data nor control messages from the far end ETE.

5.2. SEAL Model of Operation

SEAL-enabled ITEs encapsulate each inner packet in any ancillary tunnel protocol headers and trailers, a SEAL header, and any outer header encapsulations as shown in Figure 1:

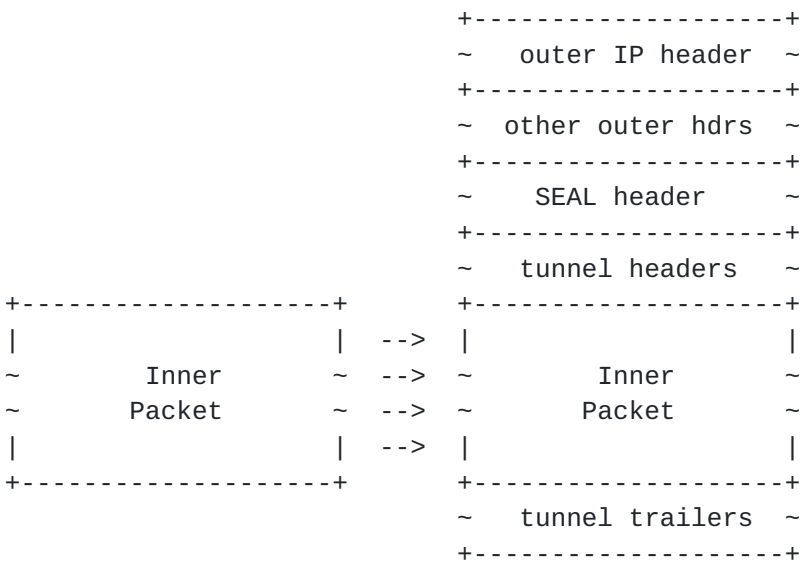

Figure 1: SEAL Encapsulation

The ITE inserts the SEAL header according to the specific tunneling protocol. For simple encapsulation of an inner network layer packet within an outer IP header, the ITE inserts the SEAL header following the outer IP header and before the inner packet as: IP/SEAL/{inner packet}.

For encapsulations over transports such as UDP, the ITE inserts the SEAL header following the outer transport layer header and before the inner packet, e.g., as IP/UDP/SEAL/{inner packet}. In that case, the UDP header is seen as an "other outer header" as depicted in Figure 1 and the outer IP and transport layer headers are together seen as the outer encapsulation headers. (Note that outer transport layer headers such as UDP must sometimes be included to ensure that SEAL packets will traverse the path to the ETE without loss due filtering middleboxes. The ETE MUST accept both IP/SEAL and IP/UDP/SEAL as equivalent packets so that the ITE can discontinue outer transport layer encapsulation if the path supports raw IP/SEAL encapsulation.)

For SEAL encapsulations that involve tunnel types that include ancillary tunnel headers (e.g., GRE, IPsec, etc.) the ITE inserts the SEAL header as a leading extension to the tunnel headers, i.e., the SEAL encapsulation appears as part of the same tunnel and not a separate tunnel. For example, for GRE the ITE inserts the SEAL

header as IP/SEAL/GRE/{inner packet}, and for IPsec the ITE inserts the SEAL header as IP/SEAL/IPsec-header/{inner packet}/IPsec-trailer. In such cases, SEAL considers the length of the inner packet only (i.e., and not the other tunnel headers and trailers) when performing its packet size calculations.

SEAL supports both "nested" tunneling and "re-encapsulating" tunneling. Nested tunneling occurs when a first tunnel is encapsulated within a second tunnel, which may then further be encapsulated within additional tunnels. Nested tunneling can be useful, and stands in contrast to "recursive" tunneling which is an anomalous condition incurred due to misconfiguration or a routing loop. Considerations for nested tunneling and avoiding recursive tunneling are discussed in Section 4 of [RFC2473] as well as in Section 9 of this document.

Re-encapsulating tunneling occurs when a packet arrives at a first ETE, which then acts as an ITE to re-encapsulate and forward the packet to a second ETE connected to the same subnetwork. In that case each ITE/ETE transition represents a segment of a bridged path between the ITE nearest the source and the ETE nearest the destination. Uses for re-encapsulating tunneling are discussed in [I-D.templin-aerolink]. Combinations of nested and re-encapsulating tunneling are also naturally supported by SEAL.

The SEAL ITE considers each underlying interface as the ingress attachment point to a separate subnetwork path to the ETE. The ITE therefore may experience different path MTUs on different subnetwork paths.

Finally, the SEAL ITE ensures that the inner network layer protocol will see a minimum MTU of 1500 bytes over each subnetwork path regardless of the outer network layer protocol version, i.e., even if a small amount of fragmentation and reassembly are necessary. This is to avoid path MTU "black holes" for the minimum MTU configured by the vast majority of links in the Internet.

5.3. SEAL Encapsulation Format

The SEAL header shares the same format and IP protocol number ('44') as the IPv6 Fragment Header specified in Section 4.5 of [RFC2460]. The SEAL header is differentiated from the IPv6 Fragment Header by defining bit number 30 as the "SEAL (S)" bit which is set to 1 when SEAL encapsulation is used and set to 0 for ordinary IPv6 fragmentation. SEAL therefore updates the IPv6 Fragment Header specification as shown in Figure 2:

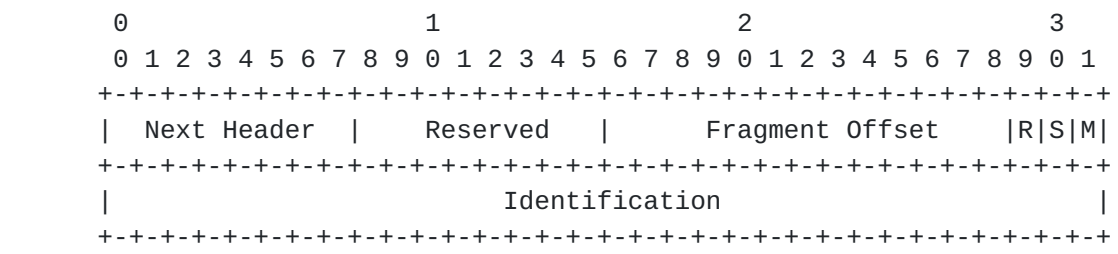

Figure 2: SEAL Encapsulation Format

5.4. ITE Specification

5.4.1. Tunnel MTU

The tunnel must present a stable MTU value to the inner network layer as the size for admission of inner packets into the tunnel. Since tunnels may support a large set of subnetwork paths that accept widely varying maximum packet sizes, however, a number of factors should be taken into consideration when selecting a tunnel MTU.

Due to the ubiquitous deployment of standard Ethernet and similar networking gear, the nominal Internet cell size has become 1500 bytes; this is the de facto size that end systems have come to expect will either be delivered by the network without loss due to an MTU restriction on the path or a suitable ICMP Packet Too Big (PTB) message returned. When large packets sent by end systems incur additional encapsulation at an ITE, however, they may be dropped silently within the tunnel since the network may not always deliver the necessary PTBs [RFC2923]. The ITE SHOULD therefore set a tunnel MTU of at least 1500 bytes and provide accommodations to ensure that packets up to that size are successfully conveyed to the ETE.

The inner network layer protocol consults the tunnel MTU when admitting a packet into the tunnel. For non-SEAL inner IPv4 packets with the IPv4 Don't Fragment (DF) bit cleared (i.e., DF==0), if the packet is larger than the tunnel MTU the inner IPv4 layer uses IPv4 fragmentation to break the packet into fragments no larger than the MTU. The ITE then admits each fragment into the tunnel as an independent packet.

For all other inner packets, the inner network layer admits the packet if it is no larger than the tunnel MTU; otherwise, it drops the packet and sends a PTB error message to the source with the MTU value set to the MTU. The message contains as much of the invoking packet as possible without the entire message exceeding the network layer minimum MTU size.

The ITE can alternatively set an indefinite tunnel MTU such that all

inner packets are admitted into the tunnel regardless of their size (practical maximums are 64KB for IPv4 and 4GB for IPv6 [RFC2675]). For ITEs that host applications that use the tunnel directly, this option must be carefully coordinated with protocol stack upper layers since some upper layer protocols (e.g., TCP) derive their packet sizing parameters from the MTU of the outgoing interface and as such may select too large an initial size. This is not a problem for upper layers that use conservative initial maximum segment size estimates and/or when the tunnel can reduce the upper layer's maximum segment size, e.g., by reducing the size advertised in the MSS option of outgoing TCP messages (sometimes known as "MSS clamping").

In light of the above considerations, the ITE SHOULD configure an indefinite MTU on *router* tunnels so that SEAL performs all subnetwork adaptation from within the tunnel as specified in the following sections. The ITE MAY instead set a smaller MTU on *host* tunnels; in that case, the RECOMMENDED MTU is the maximum of 1500 bytes and the smallest MTU among all of the underlying links minus the size of the encapsulation headers.

5.4.2. Tunnel Neighbor Soft State

The ITE maintains a number of soft state variables and constants.

The ITE maintains a per-ETE window of Identification values for the packets it sends to the ETE. The ITE increments the current Identification value monotonically (modulo 2^32) for each packet it sends.

For each subnetwork path, the ITE must also account for encapsulation header lengths. The ITE therefore maintains the per subnetwork path constant values "SHLEN" set to the length of the SEAL header, "THLEN" set to the length of the outer encapsulating transport layer headers (or 0 if outer transport layer encapsulation is not used), "IHLEN" set to the length of the outer IP layer header, and "HLEN" set to (SHLEN+THLEN+IHLEN). When calculating these lengths, the ITE must include the length of the uncompressed headers even if header compression is enabled. When SEAL is used in conjunction with tunnel types that insert additional headers/trailers such as GRE or IPsec, the length of the additional headers and trailers is also included in the HLEN calculation.

The ITE also sets a global constant value "MINMTU" to 1500 bytes and sets a per subnetwork path constant value 'FRAGMTU' to (1280-HLEN) bytes (where 1280 is the minimum path MTU for IPv6 [RFC2460]). The value 1280 is used regardless of the outer IP protocol version even though the practical minimum MTU for IPv4 is only 576 bytes [RFC1122] and the theoretical minimum MTU for IPv4 is only 68 bytes [RFC0791].

The value 1280 is applied also to IPv4 since IPv4 links with MTUs smaller than 1280 are presumably performance-constrained such that IPv4 fragmentation can be used to accommodate MTU underruns without risk of high data rate reassembly misassociations.

The ITE also sets a per subnetwork path variable "MAXMTU" to the maximum of MINMTU and the MTU of the underlying interface minus HLEN. The ITE thereafter adjusts MAXMTU based on any PTB messages it receives from the subnetwork, but does not reduce MAXMTU below MINMTU.

The ITE finally maintains a per subnetwork path boolean variable "DOFRAG", which is initially set to TRUE and may be reset to FALSE if the ITE discovers that the MTU on the path to the ETE is sufficient to accommodate packet sizes of MINMTU bytes or larger.

5.4.3. SEAL Layer Pre-Processing

The SEAL layer is logically positioned between the inner and outer network protocol layers, where the inner layer is seen as the (true) network layer and the outer layer is seen as the (virtual) data link layer. Each packet to be processed by the SEAL layer is either admitted into the tunnel by the inner network layer protocol as described in Section 5.4.1 or is undergoing re-encapsulation from within the tunnel. The SEAL layer sees the former class of packets as inner packets that include inner network and transport layer headers, and sees the latter class of packets as transitional SEAL packets that include the outer and SEAL layer headers that were inserted by the previous hop SEAL ITE. For these transitional packets, the SEAL layer re-encapsulates the packet with new outer and SEAL layer headers when it forwards the packet to the next hop SEAL ITE.

We now discuss the SEAL layer pre-processing actions for these two classes of packets.

5.4.3.1. Inner Packet Pre-Processing

For each IPv4 inner packet with DF==0 in the IP header, if the packet is larger than MINMTU bytes the ITE first uses standard IPv4 fragmentation to fragment the packet into N pieces of at most MINMTU bytes each. In this process, the ITE MUST additionally ensure that N is minimized, the first fragment is the largest fragment and no fragments are overlapping. The ITE then submits each fragment for SEAL encapsulation as specified in Section 5.4.4.

For all other inner packets, if the packet is no larger than MAXMTU the ITE submits it for SEAL encapsulation as specified in Section

5.4.4. Otherwise, the ITE discards the packet and sends a PTB message appropriate to the inner protocol version (subject to rate limiting) with the MTU field set to MAXMTU.

5.4.3.2. Transitional SEAL Packet Pre-Processing

For each transitional packet that is to be processed by the SEAL layer from within the tunnel, if the packet is larger than MAXMTU for the next hop subnetwork path the ITE discards the packet and sends a PTB message appropriate to the inner protocol version (subject to rate limiting) with the MTU field set to MAXMTU. Otherwise, the ITE sets aside the encapsulating SEAL and outer headers for later reference (see Section 5.4.5) and submits the inner packet for SEAL re-encapsulation as discussed in the following sections.

5.4.4. SEAL Encapsulation and Fragmentation

For each inner packet/fragment submitted for SEAL encapsulation, the ITE next encapsulates the packet in a SEAL header formatted as specified in Section 5.3. The ITE next sets S=1 and sets the Next Header field to the protocol number corresponding to the address family of the encapsulated inner packet. For example, the ITE sets the Next Header field to the value '4' for encapsulated IPv4 packets [RFC2003], '41' for encapsulated IPv6 packets [RFC2473][RFC4213], '47' for GRE [RFC1701], '80' for encapsulated OSI/CLNP packets [RFC1070], etc.

Next, if the inner packet is no larger than FRAGMTU, or if the inner packet is larger than MINMTU, or if the DOFRAG flag is FALSE, the ITE sets (M=0; Offset=0) and considers the packet an "atomic fragment" (see: [RFC6946]). Otherwise, the ITE fragments the inner packet using the fragmentation procedures specified in Section 4.5 of [RFC2460]. In this process, the ITE breaks the inner packet into two non-overlapping fragments, where the encapsulated SEAL packet containing the first fragment MUST be as large as possible without exceeding 1280 bytes (i.e., the IPv6 minimum MTU) and the encapsulated SEAL packet containing the second fragment MUST include the remainder of the inner packet. This ensures that the entire IP header (plus extensions) is likely to fit within the first fragment and that the number of fragments is minimized. The ITE then adds the outer encapsulating headers as specified in Section 5.4.5.

5.4.5. Outer Encapsulation

Following SEAL encapsulation and fragmentation, the ITE next encapsulates each fragment in the requisite outer transport (when necessary) and IP layer headers. When a transport layer header such as UDP or TCP is included, the ITE writes the port number for SEAL in

the transport destination service port field.

When UDP encapsulation is used, the ITE sets the UDP checksum field to zero for both IPv4 and IPv6 packets (see: [RFC6935][RFC6936]).

The ITE then sets the outer IP layer headers the same as specified for ordinary IP encapsulation (e.g., [RFC1070][RFC2003], [RFC2473], [RFC4213], etc.) except that for ordinary SEAL packets the ITE copies the "TTL/Hop Limit", "Type of Service/Traffic Class" and "Congestion Experienced" values in the inner network layer header into the corresponding fields in the outer IP header. For transitional SEAL packets undergoing re-encapsulation, the ITE instead copies the "TTL/Hop Limit", "Type of Service/Traffic Class" and "Congestion Experienced" values in the original outer IP header of the transitional packet into the corresponding fields in the new outer IP header of the packet to be forwarded (i.e., the values are transferred between outer headers and *not* copied from the inner network layer header).

The ITE also sets the IP protocol number to the appropriate value for the first protocol layer within the encapsulation (e.g., UDP, TCP, IPv6 Fragment Header, etc.). When IPv6 is used as the outer IP protocol, the ITE then sets the flow label value in the outer IPv6 header the same as described in [RFC6438]. When IPv4 is used as the outer IP protocol, if the encapsulated SEAL packet is no larger than 1280 bytes the ITE sets DF=0 in the IPv4 header to allow the packet to be fragmented if it encounters a restricting link; otherwise, the ITE sets DF=1 (for IPv6 subnetwork paths, the DF bit is absent but implicitly set to 1). The ITE finally sends each outer packet via the corresponding underlying subnetwork path.

5.4.6. Path MTU Probing and ETE Reachability Verification

When the ITE is actively sending packets over a subnetwork path to an ETE, it also sends explicit probes subject to rate limiting to test the path MTU. To generate a probe, the ITE creates an ICMPv6 Echo Request message [RFC4443] of length MINMTU bytes and encapsulates the message in a SEAL header and any other outer headers, i.e., with the length of the resulting SEAL packet being (MINMTU+HLEN) bytes. It then sets (Offset=0; S=1; M=0) in the SEAL header, and also sets DF=1 in the outer IP header when IPv4 is used. It finally writes the value '58' in the Next Header field of the SEAL header to indicate that the message is a SEAL-encapsulated ICMPv6 message.

The ITE sends such MINMTU probes to determine whether SEAL fragmentation is still necessary (see Section 5.4.4). In particular, if the ITE sends a probe and receives a SEAL-encapsulated ICMPv6 Echo Reply message probe reply (see: section 5.5.4), it SHOULD set DOFRAG

for this subnetwork path to FALSE. Note that the nominal probe size of MINMTU bytes is RECOMMENDED since probes slightly smaller than this size may be fragmented by the ITE of a nested tunnel further down the path. For example, a successful probe size of 1400 bytes does not guarantee that fragmentation is not occurring at the ITE of another tunnel nesting level. While this would not necessarily result in communication failure, it could yield poor performance not only for the other tunnel nesting levels but also for the ITE itself.

The ITE can also send smaller probes to determine whether the ETE is still reachable over this subnetwork path. The ITE prepares the probe as described above then sends the message to the ETE. If the ITE receives a probe reply, its upper layers can consider the message as a reachability indication. The ITE can also send larger probes to test for larger MTU sizes; however, SEAL considers probing for MTU sizes larger than MINMTU as an end-to-end consideration to be addressed by end systems (see: Section 7).

Finally, the ITE can also send probes to detect whether an outer transport layer header is no longer necessary to reach this ETE. For example, if the ITE sends its initial packets as IP/UDP/SEAL/*, it can send probes constructed as IP/SEAL/[probe] to determine whether the ETE is reachable without the use of UDP encapsulation. If so, the ITE should also send a new MINMTU probe since switching to a new encapsulation format may result in a path change.

While probing, the ITE processes ICMP messages as specified in Section 5.4.7.

5.4.7. Processing ICMP Messages

When the ITE sends SEAL packets, it may receive ICMP error messages [RFC0792][RFC4443] from a router on the path to the ETE. Each ICMP message includes an outer IP header, followed by an ICMP header, followed by a portion of the SEAL packet that generated the error (also known as the "packet-in-error"). Note that the ITE may receive an ICMP message from either an ordinary router on the path or from another ITE that is at the head end of a nested level of encapsulation. The ITE has no security associations with this nested ITE, hence it should consider the message the same as if it originated from an ordinary router.

The ITE should process ICMP Protocol/Port Unreachable messages as a hint that the ETE does not implement SEAL. The ITE can optionally ignore other ICMP messages that do not include sufficient information in the packet-in-error, or process them as a hint that the subnetwork path to the ETE may be failing. The ITE then discards these types of messages.

For other ICMP messages, the ITE SHOULD examine the SEAL data packet within the packet-in-error field. If the IP source and/or destination addresses are invalid, or if the value in the SEAL header Identification field (if present) is not within the window of packets the ITE has recently sent to this ETE, the ITE discards the message.

Next, if the received ICMP message is a PTB the ITE sets MAXMTU to the maximum of MINMTU and the MTU value in the message minus HLEN. If the MTU value in the message is smaller than (MINMTU+HLEN), the ITE also resets DOFRAG to TRUE and discards the message.

If the ICMP message was not discarded, the ITE transcribes it into a message appropriate for the inner protocol version (e.g., ICMPv4 for IPv4, ICMPv6 for IPv6, etc.) and forwards the transcribed message to the previous hop toward the inner source address.

5.4.8. Detecting Path MTU Changes

The ITE SHOULD periodically reset MAXMTU to the MTU of the underlying subnetwork interface to determine whether the subnetwork path MTU has increased. If the path still has a too-small MTU, the ITE will receive a PTB message that reports a smaller size.

5.5. ETE Specification

5.5.1. Reassembly Buffer Requirements

The ETE MUST configure a minimum SEAL reassembly buffer size of (MINMTU+HLEN) bytes for the reassembly of fragmented SEAL packets (see: Section 5.5.4). Note that the value "HLEN" may be variable and initially unknown to the ETE. It is therefore RECOMMENDED that the ETE configure a slightly larger SEAL reassembly buffer size of 2048 bytes (2KB).

When IPv4 is used as the outer layer of encapsulation, the ETE MUST also configure a minimum IPv4 reassembly buffer size of 1280 bytes.

5.5.2. Tunnel Neighbor Soft State

The ETE maintains a window of Identification values for the packets it has recently received from this ITE as well as a window of Identification values for the packets it has recently sent to this ITE.

5.5.3. IPv4-Layer Reassembly

The ETE reassembles fragmented IPv4 packets that are explicitly addressed to itself. For IPv4 fragments of SEAL packets, the ETE

SHOULD maintain conservative reassembly cache high- and low-water marks. When the size of the reassembly cache exceeds this high-water mark, the ETE SHOULD actively discard stale incomplete reassemblies (e.g., using an Active Queue Management (AQM) strategy) until the size falls below the low-water mark. The ETE SHOULD also actively discard any pending reassemblies that clearly have no opportunity for completion, e.g., when a considerable number of new fragments have arrived before a fragment that completes a pending reassembly arrives.

The ETE processes IPv4 fragments as specified in the normative references, i.e., it performs any necessary IPv4 reassembly then submits the packet to the appropriate upper layer protocol module. For SEAL packets, the ETE then performs SEAL decapsulation as specified in Section 5.5.4.

5.5.4. Decapsulation, SEAL-Layer Reassembly, and Re-Encapsulation

For each SEAL packet accepted for decapsulation, the ETE first examines the Identification field. If the Identification is not within the window of acceptable values for this ITE, the ETE silently discards the packet..

Next, if the SEAL header has (Offset!=0 || M=1) the ETE submits the packet for reassembly as specified for IPv6 reassembly in Section 4.5 of [RFC2460]. During the reassembly process, the ETE discards any fragments that are overlapping with respect to fragments that have already been received (see: [RFC5722]), and also discards any fragments that have M=1 in the SEAL header but do not contain an integer multiple of 8 bytes. The ETE further SHOULD manage the SEAL reassembly cache the same as described for the IPv4-Layer Reassembly cache in Section 5.5.3, i.e., it SHOULD perform an early discard for any pending reassemblies that have low probability of completion.

Next, if the (reassembled) packet is an ICMPv6 Echo Request probe message, the ETE prepares an ICMPv6 Echo Reply probe reply message to send back to the ITE. The ETE then encapsulates the probe reply as specified in Section 5.4.4 and fragments the message if necessary according to the DOFRAG flag (i.e., to ensure that the probe reply is delivered to the ITE). The ETE then sends the probe reply to the ITE and discards the probe. When the ITE receives the probe reply, it reassembles the message if necessary and processes it as specified in Section 5.4.6.

Finally, the ETE discards the outer headers of the (reassembled) packet and processes the inner packet according to the header type indicated in the SEAL Next Header field. If the next hop toward the inner destination address is via a different interface than the SEAL

packet arrived on, the ETE discards the SEAL and outer headers and delivers the inner packet either to the local host or to the next hop if the packet is not destined to the local host.

If the next hop is on the same tunnel the SEAL packet arrived on, however, the ETE submits the packet for SEAL re-encapsulation beginning with the specification in Section 5.4.3 above and without decrementing the value in the inner (TTL / Hop Limit) field.

6. Link Requirements

Subnetwork designers are expected to follow the recommendations in Section 2 of [RFC3819] when configuring link MTUs.

7. End System Requirements

End systems are encouraged to implement end-to-end MTU assurance (e.g., using Packetization Layer Path MTU Discovery (PLPMTUD) per [RFC4821]) even if the subnetwork is using SEAL.

When end systems use PLPMTUD, SEAL will ensure that the tunnel behaves as a link in the path that assures an MTU of at least 1500 bytes while still allowing end systems to discover larger MTUs. The PLPMTUD mechanism will therefore be able to function as designed in order to discover and utilize larger MTUs.

8. Router Requirements

Routers within the subnetwork are expected to observe the standard IP router requirements, including the implementation of IP fragmentation and reassembly as well as the generation of ICMP messages [RFC0792][RFC1122][RFC1812][RFC2460][RFC4443][RFC6434].

Note that, even when routers support existing requirements for the generation of ICMP messages, these messages are often filtered and discarded by middleboxes on the path to the original source of the message that triggered the ICMP. It is therefore not possible to assume delivery of ICMP messages even when routers are correctly implemented.

9. Multicast/Anycast Considerations

On multicast-capable tunnels, encapsulated packets sent by an ITE may be received by potentially many ETEs. In that case, the ITE can

still send unicast probe messages to receive probe replies from a specific ETE, or it can send multicast probe messages to receive replies from all ETEs in the multicast group that receive the probe. If the ITE were to send a multicast MINMTU probe message as described in Section 5.4.6, however, it would be unable to discern whether all ETEs received the probe unless it had some way of tracking the full constituency of the multicast group. For multicast ETE addresses, the ITE would therefore ordinarily set MAXMTU=MINMTU and DOFRAG=TRUE. But, the setting of these values may be situation-dependent and based on whether the ITE can tolerate packet loss to ETEs that may be reached by subnetwork paths having small MTUs.

For ETEs that configure an anycast address, if the ITE sends a MINMTU probe message it may receive a probe reply from a first ETE but then be re-routed to a second ETE. It is therefore necessary for the ITE to continue to send periodic probes (subject to rate limiting) as described in Section 5.4.6 so that any path oscillations between ETEs that configure the same anycast address will not result in a sustained path MTU black hole.

10. Compatibility Considerations

Since SEAL is based on the standard IPv6 fragment header, the ITE can implement the scheme independently of any ETE implementations. Therefore, if the ITE uses SEAL but the ETE does not the ITE can still send a MINMTU probe as specified in Section 5.4.6 but may receive an ordinary (i.e., non SEAL-encapsulated) probe reply. If so, it SHOULD reset DOFRAG to FALSE the same as if the ETE returned a SEAL-encapsulated probe reply.

In some cases, a non-SEAL ETE may not be able to reassemble fragmented SEAL packets up to (MINMTU+HLEN) bytes, since [RFC2460] only requires IPv6 nodes to reassemble packets up to 1500 bytes in length. To test for this condition, the ITE can create a MINMTU probe message, fragment the message into two pieces, then send both fragments to the ETE. If the ETE returns a probe reply, the ITE has assurance that the ETE is capable of reassembly. Otherwise, the ITE SHOULD reset MAXMTU for this subnetwork path to (MINMTU-HLEN) or even smaller if the ETE still cannot accept packets of this size.

11. Nested Encapsulation Considerations

SEAL supports nested tunneling - an example would be a recursive nesting of mobile networks, where the first network receives service from an ISP, the second network receives service from the first network, the third network receives service from the second network,

etc. Since it is imperative that such nesting not extend indefinitely, tunnels that use SEAL SHOULD honor the Encapsulation Limit option defined in [RFC2473].

12. Reliability Considerations

Although a tunnel may span an arbitrarily-large subnetwork expanse, the IP layer sees the tunnel as a simple link that supports the IP service model. Links with high bit error rates (BERs) (e.g., IEEE 802.11) use Automatic Repeat-ReQuest (ARQ) mechanisms [RFC3366] to increase packet delivery ratios, while links with much lower BERs typically omit such mechanisms. Since Tunnels may traverse arbitrarily-long paths over links of various types that are already either performing or omitting ARQ as appropriate, it would therefore be inefficient to require the tunnel endpoints to also perform ARQ.

13. Integrity Considerations

Fragmentation and reassembly schemes must consider packet-splicing errors, e.g., when two fragments from the same packet are concatenated incorrectly, when a fragment from packet X is reassembled with fragments from packet Y, etc. The primary sources of such errors include implementation bugs and wrapping ID fields.

In particular, the IPv4 16-bit ID field can wrap with only 64K packets with the same (src, dst, protocol)-tuple alive in the system at a given time [RFC4963]. When the IPv4 ID field is re-written by a middlebox such as a NAT or Firewall, ID field wrapping can occur with even fewer packets alive in the system.

Fortunately, SEAL includes a 32-bit ID field the same as for IPv6 fragmentation and also only employs SEAL fragmentation for packets up to 1500 bytes in length. SEAL also only allows IPv4 network fragmentation for packets up to 1280 bytes in length, but this size is small enough to fit within the MTU of modern high-speed IPv4 links without fragmentation. IPv4 links with smaller MTUs certainly exist, but typically support data rates that are slow enough to preclude high data rate reassembly misassociations errors; hence, a small amount of IPv4 fragmentation is deemed acceptable.

14. IANA Considerations

The IANA is requested to allocate a User Port number for "SEAL" in the 'port-numbers' registry. The Service Name is "SEAL", and the Transport Protocols are TCP and UDP. The Assignee is the IESG

(iesg@ietf.org) and the Contact is the IETF Chair (chair@ietf.org). The Description is "Subnetwork Encapsulation and Adaptation Layer (SEAL)", and the Reference is the RFC-to-be currently known as 'draft-templin-intarea-seal'.

15. Security Considerations

Neighbor relationships between the ITE and ETE should be secured in environments where authentication and/or confidentiality are a matter of concern. Securing mechanisms such as Secure Neighbor Discovery (SeND) [RFC3971] and IPsec [RFC4301] can be used for this purpose, however the tunnel neighbor relationship is managed by the tunnel protocols that ride over SEAL (as an encapsulation sublayer) rather than by SEAL itself.

Security issues that apply to tunneling in general are discussed in [RFC6169].

16. Related Work

Section 3.1.7 of [RFC2764] provides a high-level sketch for supporting large tunnel MTUs via a tunnel-layer fragmentation and reassembly capability to avoid IP layer fragmentation.

Section 3 of [RFC4459] describes inner and outer fragmentation at the tunnel endpoints as alternatives for accommodating the tunnel MTU.

Section 4 of [RFC2460] specifies a method for inserting and processing extension headers between the base IPv6 header and transport layer protocol data. The SEAL header is inserted and processed in exactly the same manner.

The concepts of path MTU determination through the report of fragmentation and extending the IPv4 Identification field were first proposed in deliberations of the TCP-IP mailing list and the Path MTU Discovery Working Group (MTUDWG) during the late 1980's and early 1990's. An historical analysis of the evolution of these concepts, as well as the development of the eventual PMTUD mechanism, appears in [RFC5320].

17. Implementation Status

An early implementation of the first revision of SEAL [RFC5320] is available at: http://isatap.com/seal.

An implementation of the current version of SEAL is available at: http://linkupnetworks.com/seal/sealv2-1.0.tgz.

18. Acknowledgments

The following individuals are acknowledged for helpful comments and suggestions: Jari Arkko, Fred Baker, Iljitsch van Beijnum, Oliver Bonaventure, Teco Boot, Bob Braden, Brian Carpenter, Steve Casner, Ian Chakeres, Noel Chiappa, Remi Denis-Courmont, Remi Despres, Ralph Droms, Aurnaud Ebalard, Gorry Fairhurst, Washam Fan, Dino Farinacci, Joel Halpern, Brian Haberman, Sam Hartman, John Heffner, Thomas Henderson, Bob Hinden, Christian Huitema, Eliot Lear, Darrel Lewis, Joe Macker, Matt Mathis, Erik Nordmark, Dan Romascanu, Dave Thaler, Joe Touch, Mark Townsley, Ole Troan, Margaret Wasserman, Magnus Westerlund, Robin Whittle, James Woodyatt, and members of the Boeing Research & Technology NST DC&NT group.

Discussions with colleagues following the publication of [RFC5320] have provided useful insights that have resulted in significant improvements to this, the Second Edition of SEAL. In particular, this work has been encouraged and supported by Boeing colleagues including Balaguruna Chidambaram, Jeff Holland, Cam Brodie, Yueli Yang, Wen Fang, Ed King, Mike Slane, Kent Shuey, Gen MacLean, and other members of the BR&T and BIT mobile networking teams.

This document received substantial review input from the IESG and IETF area directorates in the February 2013 timeframe. IESG members and IETF area directorate representatives who contributed helpful comments and suggestions are gratefully acknowledged. Discussions on the IETF IPv6 and Intarea mailing lists in the summer 2013 timeframe also stimulated several useful ideas.

Path MTU determination through the report of fragmentation was first proposed by Charles Lynn on the TCP-IP mailing list in 1987. Extending the IP identification field was first proposed by Steve Deering on the MTUDWG mailing list in 1989. Steve Deering also proposed the IPv6 minimum MTU of 1280 bytes on the IPng mailing list in 1997.

19. References

19.1. Normative References

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

[RFC0792] Postel, J., "Internet Control Message Protocol", STD 5, RFC 792, September 1981.

[RFC1122] Braden, R., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, October 1989.

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

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[RFC4443] Conta, A., Deering, S., and M. Gupta, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", RFC 4443, March 2006.

19.2. Informative References

[FOLK] Shannon, C., Moore, D., and k. claffy, "Beyond Folklore: Observations on Fragmented Traffic", December 2002.

[FRAG] Kent, C. and J. Mogul, "Fragmentation Considered Harmful", October 1987.

[I-D.taylor-v6ops-fragdrop] Jaeggli, J., Colitti, L., Kumari, W., Vyncke, E., Kaeo, M., and T. Taylor, "Why Operators Filter Fragments and What It Implies", draft-taylor-v6ops-fragdrop-01 (work in progress), June 2013.

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980.

[RFC0994] International Organization for Standardization (ISO) and American National Standards Institute (ANSI), "Final text of DIS 8473, Protocol for Providing the Connectionless-mode Network Service", RFC 994, March 1986.

[RFC1070] Hagens, R., Hall, N., and M. Rose, "Use of the Internet as a subnetwork for experimentation with the OSI network layer", RFC 1070, February 1989.

[RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191, November 1990.

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

[RFC1812] Baker, F., "Requirements for IP Version 4 Routers", RFC 1812, June 1995.

[RFC1981] McCann, J., Deering, S., and J. Mogul, "Path MTU Discovery for IP version 6", RFC 1981, August 1996.

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

[RFC2473] Conta, A. and S. Deering, "Generic Packet Tunneling in IPv6 Specification", RFC 2473, December 1998.

[RFC2675] Borman, D., Deering, S., and R. Hinden, "IPv6 Jumbograms", RFC 2675, August 1999.

[RFC2764] Gleeson, B., Heinanen, J., Lin, A., Armitage, G., and A. Malis, "A Framework for IP Based Virtual Private Networks", RFC 2764, February 2000.

[RFC2827] Ferguson, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", BCP 38, RFC 2827, May 2000.

[RFC2923] Lahey, K., "TCP Problems with Path MTU Discovery", RFC 2923, September 2000.

[RFC3232] Reynolds, J., "Assigned Numbers: RFC 1700 is Replaced by an On-line Database", RFC 3232, January 2002.

[RFC3366] Fairhurst, G. and L. Wood, "Advice to link designers on link Automatic Repeat reQuest (ARQ)", BCP 62, RFC 3366, August 2002.

[RFC3819] Karn, P., Bormann, C., Fairhurst, G., Grossman, D., Ludwig, R., Mahdavi, J., Montenegro, G., Touch, J., and L. Wood, "Advice for Internet Subnetwork Designers", BCP 89, RFC 3819, July 2004.

[RFC3971] Arkko, J., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)", RFC 3971, March 2005.

[RFC4213] Nordmark, E. and R. Gilligan, "Basic Transition Mechanisms for IPv6 Hosts and Routers", RFC 4213, October 2005.

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

[RFC4459] Savola, P., "MTU and Fragmentation Issues with In-the-

Network Tunneling", RFC 4459, April 2006.

[RFC4821] Mathis, M. and J. Heffner, "Packetization Layer Path MTU Discovery", RFC 4821, March 2007.

[RFC4963] Heffner, J., Mathis, M., and B. Chandler, "IPv4 Reassembly Errors at High Data Rates", RFC 4963, July 2007.

[RFC5320] Templin, F., "The Subnetwork Encapsulation and Adaptation Layer (SEAL)", RFC 5320, February 2010.

[RFC5722] Krishnan, S., "Handling of Overlapping IPv6 Fragments", RFC 5722, December 2009.

[RFC5927] Gont, F., "ICMP Attacks against TCP", RFC 5927, July 2010.

[RFC6169] Krishnan, S., Thaler, D., and J. Hoagland, "Security Concerns with IP Tunneling", RFC 6169, April 2011.

[RFC6434] Jankiewicz, E., Loughney, J., and T. Narten, "IPv6 Node Requirements", RFC 6434, December 2011.

[RFC6438] Carpenter, B. and S. Amante, "Using the IPv6 Flow Label for Equal Cost Multipath Routing and Link Aggregation in Tunnels", RFC 6438, November 2011.

[RFC6864] Touch, J., "Updated Specification of the IPv4 ID Field", RFC 6864, February 2013.

[RFC6935] Eubanks, M., Chimento, P., and M. Westerlund, "IPv6 and UDP Checksums for Tunneled Packets", RFC 6935, April 2013.

[RFC6936] Fairhurst, G. and M. Westerlund, "Applicability Statement for the Use of IPv6 UDP Datagrams with Zero Checksums", RFC 6936, April 2013.

[RFC6946] Gont, F., "Processing of IPv6 "Atomic" Fragments", RFC 6946, May 2013.

[RIPE] De Boer, M. and J. Bosma, "Discovering Path MTU Black Holes on the Internet using RIPE Atlas", July 2012.

[SIGCOMM] Luckie, M. and B. Stasiewicz, "Measuring Path MTU Discovery Behavior", November 2010.

[TBIT] Medina, A., Allman, M., and S. Floyd, "Measuring Interactions Between Transport Protocols and Middleboxes", October 2004.

[WAND] Luckie, M., Cho, K., and B. Owens, "Inferring and Debugging Path MTU Discovery Failures", October 2005.

Author's Address

Fred L. Templin (editor)
Boeing Research & Technology
P.O. Box 3707
Seattle, WA 98124
USA

Email: fltemplin@acm.org