

Internet Area WG  
Internet-Draft  
Intended status: Best Current Practice  
Expires: November 15, 2019

R. Bonica  
Juniper Networks  
F. Baker  
Unaffiliated  
G. Huston  
APNIC  
R. Hinden  
Check Point Software  
O. Troan  
Cisco  
F. Gont  
SI6 Networks  
May 14, 2019

**IP Fragmentation Considered Fragile**  
**draft-ietf-intarea-frag-fragile-10**

**Abstract**

This document describes IP fragmentation and explains how it introduces fragility to Internet communication.

This document also proposes alternatives to IP fragmentation and provides recommendations for developers and network operators.

**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 <https://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 November 15, 2019.

**Copyright Notice**

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

<a href="#">1.</a>	Introduction . . . . .	<a href="#">3</a>
<a href="#">1.1.</a>	IP-in-IP Tunnels . . . . .	<a href="#">3</a>
<a href="#">2.</a>	IP Fragmentation . . . . .	<a href="#">3</a>
<a href="#">2.1.</a>	Links, Paths, MTU and PMTU . . . . .	<a href="#">3</a>
<a href="#">2.2.</a>	Fragmentation Procedures . . . . .	<a href="#">5</a>
<a href="#">2.3.</a>	Upper-Layer Reliance on IP Fragmentation . . . . .	<a href="#">6</a>
<a href="#">3.</a>	Requirements Language . . . . .	<a href="#">7</a>
<a href="#">4.</a>	Increased Fragility . . . . .	<a href="#">7</a>
<a href="#">4.1.</a>	Policy-Based Routing . . . . .	<a href="#">7</a>
<a href="#">4.2.</a>	Network Address Translation (NAT) . . . . .	<a href="#">8</a>
<a href="#">4.3.</a>	Stateless Firewalls . . . . .	<a href="#">9</a>
<a href="#">4.4.</a>	Equal Cost Multipath, Link Aggregate Groups and Stateless Load-Balancers . . . . .	<a href="#">9</a>
<a href="#">4.5.</a>	IPv4 Reassembly Errors at High Data Rates . . . . .	<a href="#">10</a>
<a href="#">4.6.</a>	Security Vulnerabilities . . . . .	<a href="#">11</a>
<a href="#">4.7.</a>	PMTU Blackholing Due to ICMP Loss . . . . .	<a href="#">12</a>
<a href="#">4.7.1.</a>	Transient Loss . . . . .	<a href="#">12</a>
<a href="#">4.7.2.</a>	Incorrect Implementation of Security Policy . . . . .	<a href="#">13</a>
<a href="#">4.7.3.</a>	Persistent Loss Caused By Anycast . . . . .	<a href="#">13</a>
<a href="#">4.7.4.</a>	Persistent Loss Caused By Unidirectional Routing . . . . .	<a href="#">14</a>
<a href="#">4.8.</a>	Blackholing Due To Filtering or Loss . . . . .	<a href="#">14</a>
<a href="#">5.</a>	Alternatives to IP Fragmentation . . . . .	<a href="#">15</a>
<a href="#">5.1.</a>	Transport Layer Solutions . . . . .	<a href="#">15</a>
<a href="#">5.2.</a>	Application Layer Solutions . . . . .	<a href="#">16</a>
<a href="#">6.</a>	Applications That Rely on IPv6 Fragmentation . . . . .	<a href="#">17</a>
<a href="#">6.1.</a>	Domain Name Service (DNS) . . . . .	<a href="#">17</a>
<a href="#">6.2.</a>	Open Shortest Path First (OSPF) . . . . .	<a href="#">18</a>
<a href="#">6.3.</a>	Packet-in-Packet Encapsulations . . . . .	<a href="#">18</a>
<a href="#">6.4.</a>	UDP Applications Enhancing Performance . . . . .	<a href="#">18</a>
<a href="#">7.</a>	Recommendations . . . . .	<a href="#">18</a>
<a href="#">7.1.</a>	For Application and Protocol Developers . . . . .	<a href="#">18</a>
<a href="#">7.2.</a>	For System Developers . . . . .	<a href="#">19</a>
<a href="#">7.3.</a>	For Middle Box Developers . . . . .	<a href="#">19</a>
<a href="#">7.4.</a>	For ECMP, LAG and Load-Balancer Developers And Operators . . . . .	<a href="#">19</a>
<a href="#">7.5.</a>	For Network Operators . . . . .	<a href="#">20</a>
<a href="#">8.</a>	IANA Considerations . . . . .	<a href="#">20</a>



<a href="#">9.</a>	Security Considerations . . . . .	<a href="#">20</a>
<a href="#">10.</a>	Acknowledgements . . . . .	<a href="#">20</a>
<a href="#">11.</a>	References . . . . .	<a href="#">21</a>
<a href="#">11.1.</a>	Normative References . . . . .	<a href="#">21</a>
<a href="#">11.2.</a>	Informative References . . . . .	<a href="#">22</a>
<a href="#">Appendix A.</a>	Contributors' Address . . . . .	<a href="#">25</a>
	Authors' Addresses . . . . .	<a href="#">25</a>

## [1.](#) Introduction

Operational experience [[Kent](#)] [[Huston](#)] [[RFC7872](#)] reveals that IP fragmentation introduces fragility to Internet communication. This document describes IP fragmentation and explains how it introduces fragility to Internet communication. This document also proposes alternatives to IP fragmentation and provides recommendations for developers and network operators.

While this document identifies issues associated with IP fragmentation, it does not recommend deprecation. Some applications (see [Section 6](#)) require IP fragmentation. Furthermore, fragmentation is expected to work in domains where security and interoperability issues are addressed.

Rather than deprecating IP Fragmentation, this document recommends that upper-layer protocols address the problem of fragmentation at their layer, reducing their reliance on IP fragmentation to the greatest degree possible.

### [1.1.](#) IP-in-IP Tunnels

This document acknowledges that in some cases, packets must be fragmented within IP-in-IP tunnels [[I-D.ietf-intarea-tunnels](#)]. Therefore, this document makes no recommendations regarding IP-in-IP tunnels.

## [2.](#) IP Fragmentation

### [2.1.](#) Links, Paths, MTU and PMTU

An Internet path connects a source node to a destination node. A path can contain links and routers. If a path contains more than one link, the links are connected in series and a router connects each link to the next.

Internet paths are dynamic. Assume that the path from one node to another contains a set of links and routers. If a link fails, the path can also change so that it includes a different set of links and routers.



Each link is constrained by the number of bytes that it can convey in a single IP packet. This constraint is called the link Maximum Transmission Unit (MTU). IPv4 [[RFC0791](#)] requires every link to support a specified MTU (see NOTE 1). IPv6 [[RFC8200](#)] requires every link to support an MTU of 1280 bytes or greater. These are called the IPv4 and IPv6 minimum link MTU's.

Likewise, each Internet path is constrained by the number of bytes that it can convey in a IP single packet. This constraint is called the Path MTU (PMTU). For any given path, the PMTU is equal to the smallest of its link MTU's. Because Internet paths are dynamic, PMTU is also dynamic.

For reasons described below, source nodes estimate the PMTU between themselves and destination nodes. A source node can produce extremely conservative PMTU estimates in which:

- o The estimate for each IPv4 path is equal to the IPv4 minimum link MTU.
- o The estimate for each IPv6 path is equal to the IPv6 minimum link MTU.

While these conservative estimates are guaranteed to be less than or equal to the actual PMTU, they are likely to be much less than the actual PMTU. This may adversely affect upper-layer protocol performance.

By executing Path MTU Discovery (PMTUD) [[RFC1191](#)] [[RFC8201](#)] procedures, a source node can maintain a less conservative estimate of the PMTU between itself and a destination node. In PMTUD, the source node produces an initial PMTU estimate. This initial estimate is equal to the MTU of the first link along the path to the destination node. It can be greater than the actual PMTU.

Having produced an initial PMTU estimate, the source node sends non-fragmentable IP packets to the destination node (see NOTE 2). If one of these packets is larger than the actual PMTU, a downstream router will not be able to forward the packet through the next link along the path. Therefore, the downstream router drops the packet and sends an Internet Control Message Protocol (ICMP) [[RFC0792](#)] [[RFC4443](#)] Packet Too Big (PTB) message to the source node (see NOTE 3). The ICMP PTB message indicates the MTU of the link through which the packet could not be forwarded. The source node uses this information to refine its PMTU estimate.

PMTUD produces a running estimate of the PMTU between a source node and a destination node. Because PMTU is dynamic, at any given time,



the PMTU estimate can differ from the actual PMTU. In order to detect PMTU increases, PMTUD occasionally resets the PMTU estimate to its initial value and repeats the procedure described above.

Ideally, PMTUD operates as described above. However, in some scenarios, PMTUD fails. For example:

- o PMTUD relies on the network's ability to deliver ICMP PTB messages to the source node. If the network cannot deliver ICMP PTB messages to the source node, PMTUD fails.
- o PMTUD is susceptible to attack because ICMP messages are easily forged [[RFC5927](#)] and not authenticated by the receiver. Such attacks can cause PMTUD to produce unnecessarily conservative PMTU estimates.

NOTE 1: In IPv4, every host must be capable of receiving a packet whose length is equal to 576 bytes. However, the IPv4 minimum link MTU is not 576. [Section 3.2 of RFC 791](#) explicitly states that the IPv4 minimum link MTU is 68 bytes. But for practical purposes, many network operators consider the IPv4 minimum link MTU to be 576 bytes. So, for the purposes of this document, we assume that the IPv4 minimum path MTU is 576 bytes.

NOTE 2: A non-fragmentable packet can be fragmented at its source. However, it cannot be fragmented by a downstream node. An IPv4 packet whose DF-bit is set to zero is fragmentable. An IPv4 packet whose DF-bit is set to one is non-fragmentable. All IPv6 packets are also non-fragmentable.

NOTE 3:: The ICMP PTB message has two instantiations. In ICMPv4 [[RFC0792](#)], the ICMP PTB message is Destination Unreachable message with Code equal to (4) fragmentation needed and DF set. This message was augmented by [[RFC1191](#)] to indicate the MTU of the link through which the packet could not be forwarded. In ICMPv6 [[RFC4443](#)], the ICMP PTB message is a Packet Too Big Message with Code equal to (0). This message also indicates the MTU of the link through which the packet could not be forwarded.

## **2.2. Fragmentation Procedures**

When an upper-layer protocol submits data to the underlying IP module, and the resulting IP packet's length is greater than the PMTU, the packet is divided into fragments. Each fragment includes an IP header and a portion of the original packet.

[RFC0791] describes IPv4 fragmentation procedures. An IPv4 packet whose DF-bit is set to one can be fragmented by the source node, but





cannot be fragmented by a downstream router. An IPv4 packet whose DF-bit is set to zero can be fragmented by the source node or by a downstream router. When an IPv4 packet is fragmented, all IP options appear in the first fragment, but only options whose "copy" bit is set to one appear in subsequent fragments.

[RFC8200] describes IPv6 fragmentation procedures. An IPv6 packet can be fragmented at the source node only. When an IPv6 packet is fragmented, all extension headers appear in the first fragment, but only per-fragment headers appear in subsequent fragments. Per-fragment headers include the following:

- o The IPv6 header.
- o The Hop-by-hop Options header (if present)
- o The Destination Options header (if present and if it precedes a Routing header)
- o The Routing Header (if present)
- o The Fragment Header

In both IPv4 and IPv6, the upper-layer header appears in the first fragment only. It does not appear in subsequent fragments.

### **2.3. Upper-Layer Reliance on IP Fragmentation**

Upper-layer protocols can operate in the following modes:

- o Do not rely on IP fragmentation.
- o Rely on IP fragmentation by the source node only.
- o Rely on IP fragmentation by any node.

Upper-layer protocols running over IPv4 can operate in all of the above-mentioned modes. Upper-layer protocols running over IPv6 can operate in the first and second modes only.

Upper-layer protocols that operate in the first two modes (above) require access to the PMTU estimate. In order to fulfil this requirement, they can:

- o Estimate the PMTU to be equal to the IPv4 or IPv6 minimum link MTU.
- o Access the estimate that PMTUD produced.



- o Execute PMTUD procedures themselves.
- o Execute Packetization Layer PMTUD (PLPMTUD) [[RFC4821](#)] [[I-D.ietf-tsvwg-datagram-plpmtud](#)] procedures.

According to PLPMTUD procedures, the upper-layer protocol maintains a running PMTU estimate. It does so by sending probe packets of various sizes to its upper-layer peer and receiving acknowledgements. This strategy differs from PMTUD in that it relies on acknowledgement of received messages, as opposed to ICMP PTB messages concerning dropped messages. Therefore, PLPMTUD does not rely on the network's ability to deliver ICMP PTB messages to the source.

### **3. Requirements Language**

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [BCP 14](#) [[RFC2119](#)] [[RFC8174](#)] when, and only when, they appear in all capitals, as shown here.

### **4. Increased Fragility**

This section explains how IP fragmentation introduces fragility to Internet communication.

#### **4.1. Policy-Based Routing**

IP Fragmentation causes problems for routers that implement policy-based routing.

When a router receives a packet, it identifies the next-hop on route to the packet's destination and forwards the packet to that next-hop. In order to identify the next-hop, the router interrogates a local data structure called the Forwarding Information Base (FIB).

Normally, the FIB contains destination-based entries that map a destination prefix to a next-hop. Policy-based routing allows destination-based and policy-based entries to coexist in the same FIB. A policy-based FIB entry maps multiple fields, drawn from either the IP or transport-layer header, to a next-hop.



Entry	Type	Dest. Prefix	Next Hdr / Dest. Port	Next-Hop
1	Destination-based	2001:db8::1/128	Any / Any	2001:db8::2
2	Policy-based	2001:db8::1/128	TCP / 80	2001:db8::3

Table 1: Policy-Based Routing FIB

Assume that a router maintains the FIB in Table 1. The first FIB entry is destination-based. It maps the a destination prefix (2001:db8::1/128) to a next-hop (2001:db8::2). The second FIB entry is policy-based. It maps the same destination prefix (2001:db8::1/128) and a destination port ( TCP / 80 ) to a different next-hop (2001:db8::3). The second entry is more specific than the first.

When the router receives the first fragment of a packet that is destined for TCP port 80 on 2001:db8::1, it interrogates the FIB. Both FIB entries satisfy the query. The router selects the second FIB entry because it is more specific and forwards the packet to 2001:db8::3.

When the router receives the second fragment of the packet, it interrogates the FIB again. This time, only the first FIB entry satisfies the query, because the second fragment contains no indication that the packet is destined for TCP port 80. Therefore, the router selects the first FIB entry and forwards the packet to 2001:db8::2.

Policy-based routing is also known as filter-based-forwarding.

#### [4.2.](#) Network Address Translation (NAT)

IP fragmentation causes problems for Network Address Translation (NAT) devices. When a NAT device detects a new, outbound flow, it maps that flow's source port and IP address to another source port and IP address. Having created that mapping, the NAT device translates:

- o The Source IP Address and Source Port on each outbound packet.



- o The Destination IP Address and Destination Port on each inbound packet.

A+P [[RFC6346](#)] and Carrier Grade NAT (CGN) [[RFC6888](#)] are two common NAT strategies. In both approaches the NAT device must virtually reassemble fragmented packets in order to translate and forward each fragment. (See NOTE 1.)

Virtual reassembly in the network is problematic, because it is computationally expensive and because it is prone to attacks ([Section 4.6](#)).

NOTE 1: Virtual reassembly is a procedure in which a device reassembles a packet, forwards its fragments, and discards the reassembled copy. In A+P and CGN, virtual reassembly is required in order to correctly translate fragment addresses.

#### **[4.3.](#) Stateless Firewalls**

IP fragmentation causes problems for stateless firewalls whose rules include TCP and UDP ports. Because port information is not available in the trailing fragments the firewall is limited to the following options:

- o Accept all trailing fragments, possibly admitting certain classes of attack.
- o Block all trailing fragments, possibly blocking legitimate traffic.

Neither option is attractive.

#### **[4.4.](#) Equal Cost Multipath, Link Aggregate Groups and Stateless Load-Balancers**

IP fragmentation causes problems for Equal Cost Multipath (ECMP), Link Aggregate Groups (LAG) and other stateless load-balancing technologies. In order to assign a packet or packet fragment to a link, an intermediate node executes a hash (i.e., load-balancing) algorithm. The following paragraphs describe a commonly deployed hash algorithm.

If the packet or packet fragment contains a transport-layer header, the algorithm accepts the following 5-tuple as input:

- o IP Source Address.
- o IP Destination Address.





- o IPv4 Protocol or IPv6 Next Header.
- o transport-layer source port.
- o transport-layer destination port.

If the packet or packet fragment does not contain a transport-layer header, the algorithm accepts only the following 3-tuple as input:

- o IP Source Address.
- o IP Destination Address.
- o IPv4 Protocol or IPv6 Next Header.

Therefore, non-fragmented packets belonging to a flow can be assigned to one link while fragmented packets belonging to the same flow can be divided between that link and another. This can cause suboptimal load-balancing.

[RFC6438] offers a partial solution to this problem for IPv6 devices only. According to [\[RFC6438\]](#):

"At intermediate routers that perform load distribution, the hash algorithm used to determine the outgoing component-link in an ECMP and/or LAG toward the next hop MUST minimally include the 3-tuple {dest addr, source addr, flow label} and MAY also include the remaining components of the 5-tuple."

If the algorithm includes only the 3-tuple {dest addr, source addr, flow label}, it will assign all fragments belonging to a packet to the same link. (See [\[RFC6437\]](#) and [\[RFC7098\]](#)).

In order to avoid the problem described above, implementations SHOULD implement the recommendations provided in [Section 7.4](#) of this document.

#### **[4.5](#). IPv4 Reassembly Errors at High Data Rates**

IPv4 fragmentation is not sufficiently robust for use under some conditions in today's Internet. At high data rates, the 16-bit IP identification field is not large enough to prevent frequent incorrectly assembled IP fragments, and the TCP and UDP checksums are insufficient to prevent the resulting corrupted datagrams from being delivered to higher protocol layers. [\[RFC4963\]](#) describes some easily reproduced experiments demonstrating the problem, and discusses some of the operational implications of these observations.



These reassembly issues are not easily reproducible in IPv6 because the IPv6 identification field is 32 bits long.

#### **4.6. Security Vulnerabilities**

Security researchers have documented several attacks that exploit IP fragmentation. The following are examples:

- o Overlapping fragment attacks [[RFC1858](#)][RFC3128][[RFC5722](#)]
- o Resource exhaustion attacks (such as the Rose Attack)
- o Attacks based on predictable fragment identification values [[RFC7739](#)]
- o Evasion of Network Intrusion Detection Systems (NIDS) [[Ptacek1998](#)]

In the overlapping fragment attack, an attacker constructs a series of packet fragments. The first fragment contains an IP header, a transport-layer header, and some transport-layer payload. This fragment complies with local security policy and is allowed to pass through a stateless firewall. A second fragment, having a non-zero offset, overlaps with the first fragment. The second fragment also passes through the stateless firewall. When the packet is reassembled, the transport layer header from the first fragment is overwritten by data from the second fragment. The reassembled packet does not comply with local security policy. Had it traversed the firewall in one piece, the firewall would have rejected it.

A stateless firewall cannot protect against the overlapping fragment attack. However, destination nodes can protect against the overlapping fragment attack by implementing the procedures described in [RFC 1858](#), [RFC 3128](#) and [RFC 8200](#). These reassembly procedures detect the overlap and discard the packet.

The fragment reassembly algorithm is a stateful procedure in an otherwise stateless protocol. Therefore, it can be exploited by resource exhaustion attacks. An attacker can construct a series of fragmented packets, with one fragment missing from each packet so that the reassembly is impossible. Thus, this attack causes resource exhaustion on the destination node, possibly denying reassembly services to other flows. This type of attack can be mitigated by flushing fragment reassembly buffers when necessary, at the expense of possibly dropping legitimate fragments.

Each IP fragment contains an "Identification" field that destination nodes use to reassemble fragmented packets. Many implementations set the Identification field to a predictable value, thus making it easy



for an attacker to forge malicious IP fragments that would cause the reassembly procedure for legitimate packets to fail.

NIDS aims at identifying malicious activity by analyzing network traffic. Ambiguity in the possible result of the fragment reassembly process may allow an attacker to evade these systems. Many of these systems try to mitigate some of these evasion techniques (e.g. By computing all possible outcomes of the fragment reassembly process, at the expense of increased processing requirements).

#### **4.7. PMTU Blackholing Due to ICMP Loss**

As mentioned in [Section 2.3](#), upper-layer protocols can be configured to rely on PMTUD. Because PMTUD relies upon the network to deliver ICMP PTB messages, those protocols also rely on the networks to deliver ICMP PTB messages.

According to [\[RFC4890\]](#), ICMP PTB messages must not be filtered. However, ICMP PTB delivery is not reliable. It is subject to both transient and persistent loss.

Transient loss of ICMP PTB messages can cause transient PMTU black holes. When the conditions contributing to transient loss abate, the network regains its ability to deliver ICMP PTB messages and connectivity between the source and destination nodes is restored. [Section 4.7.1](#) of this document describes conditions that lead to transient loss of ICMP PTB messages.

Persistent loss of ICMP PTB messages can cause persistent black holes. [Section 4.7.2](#) and [Section 4.7.3](#) of this document describe conditions that lead to persistent loss of ICMP PTB messages.

The problem described in this section is specific to PMTUD. It does not occur when the upper-layer protocol obtains its PMTU estimate from PLPMTUD or from any other source.

##### **4.7.1. Transient Loss**

The following factors can contribute to transient loss of ICMP PTB messages:

- o Network congestion.
- o Packet corruption.
- o Transient routing loops.
- o ICMP rate limiting.



The effect of rate limiting may be severe, as [RFC 4443](#) recommends strict rate limiting of IPv6 traffic.

#### **4.7.2. Incorrect Implementation of Security Policy**

Incorrect implementation of security policy can cause persistent loss of ICMP PTB messages.

Assume that a Customer Premise Equipment (CPE) router implements the following zone-based security policy:

- o Allow any traffic to flow from the inside zone to the outside zone.
- o Do not allow any traffic to flow from the outside zone to the inside zone unless it is part of an existing flow (i.e., it was elicited by an outbound packet).

When a correct implementation of the above-mentioned security policy receives an ICMP PTB message, it examines the ICMP PTB payload in order to determine whether the original packet (i.e., the packet that elicited the ICMP PTB message) belonged to an existing flow. If the original packet belonged to an existing flow, the implementation allows the ICMP PTB to flow from the outside zone to the inside zone. If not, the implementation discards the ICMP PTB message.

When a incorrect implementation of the above-mentioned security policy receives an ICMP PTB message, it discards the packet because its source address is not associated with an existing flow.

The security policy described above is implemented incorrectly on many consumer CPE routers.

#### **4.7.3. Persistent Loss Caused By Anycast**

Anycast can cause persistent loss of ICMP PTB messages. Consider the example below:

A DNS client sends a request to an anycast address. The network routes that DNS request to the nearest instance of that anycast address (i.e., a DNS Server). The DNS server generates a response and sends it back to the DNS client. While the response does not exceed the DNS server's PMTU estimate, it does exceed the actual PMTU.

A downstream router drops the packet and sends an ICMP PTB message the packet's source (i.e., the anycast address). The network routes the ICMP PTB message to the anycast instance closest to the





downstream router. That anycast instance may not be the DNS server that originated the DNS response. It may be another DNS server with the same anycast address. The DNS server that originated the response may never receive the ICMP PTB message and may never update its PMTU estimate.

#### **4.7.4. Persistent Loss Caused By Unidirectional Routing**

Unidirectional routing can cause persistent loss of ICMP PTB messages. Consider the example below:

A source node sends a packet to a destination node. All intermediate nodes maintain a route to the destination node, but do not maintain a route to the source node. In this case, when an intermediate node encounters an MTU issue, it cannot send an ICMP PTB message to the source node.

#### **4.8. Blackholing Due To Filtering or Loss**

In [RFC 7872](#), researchers sampled Internet paths to determine whether they would convey packets that contain IPv6 extension headers. Sampled paths terminated at popular Internet sites (e.g., popular web, mail and DNS servers).

The study revealed that at least 28% of the sampled paths did not convey packets containing the IPv6 Fragment extension header. In most cases, fragments were dropped in the destination autonomous system. In other cases, the fragments were dropped in transit autonomous systems.

Another recent study [[Huston](#)] confirmed this finding. It reported that 37% of sampled endpoints used IPv6-capable DNS resolvers that were incapable of receiving a fragmented IPv6 response.

It is difficult to determine why network operators drop fragments. Possible causes follow:

- o Hardware inability to process fragmented packets.
- o Failure to change vendor defaults.
- o Unintentional misconfiguration.
- o Intentional configuration (e.g., network operators consciously chooses to drop IPv6 fragments in order to address the issues raised in [Section 4.1](#) through [Section 4.7](#), above.)



## **5. Alternatives to IP Fragmentation**

### **5.1. Transport Layer Solutions**

The Transport Control Protocol (TCP) [[RFC0793](#)]) can be operated in a mode that does not require IP fragmentation.

Applications submit a stream of data to TCP. TCP divides that stream of data into segments, with no segment exceeding the TCP Maximum Segment Size (MSS). Each segment is encapsulated in a TCP header and submitted to the underlying IP module. The underlying IP module prepends an IP header and forwards the resulting packet.

If the TCP MSS is sufficiently small, the underlying IP module never produces a packet whose length is greater than the actual PMTU. Therefore, IP fragmentation is not required.

TCP offers the following mechanisms for MSS management:

- o Manual configuration
- o PMTUD
- o PLPMTUD

Manual configuration is always applicable. If the MSS is configured to a sufficiently low value, the IP layer will never produce a packet whose length is greater than the protocol minimum link MTU. However, manual configuration prevents TCP from taking advantage of larger link MTU's.

Upper-layer protocols can implement PMTUD in order to discover and take advantage of larger path MTUs. However, as mentioned in [Section 2.1](#), PMTUD relies upon the network to deliver ICMP PTB messages. Therefore, PMTUD is applicable only in environments where the risk of ICMP PTB loss is acceptable.

By contrast, PLPMTUD does not rely upon the network's ability to deliver ICMP PTB messages. It utilises probe messages sent as TCP segments to determine if the probed PMTU can be successfully used across the network path. In PLPMTUD, probing is separated from congestion control, so that loss of a TCP probe segment does not cause a reduction of the congestion control window. [[RFC4821](#)] defines PLPMTUD procedures for TCP.

While TCP will never cause the underlying IP module to emit a packet that is larger than the PMTU estimate, it can cause the underlying IP module to emit a packet that is larger than the actual PMTU. If this



occurs, the packet is dropped, the PMTU estimate is updated, the segment is divided into smaller segments and each smaller segment is submitted to the underlying IP module.

The Datagram Congestion Control Protocol (DCCP) [[RFC4340](#)], the Stream Control Protocol (SCP) [[RFC4960](#)], and the Stream Control Transport Protocol (SCTP) [[RFC4960](#)] also can be operated in a mode that does not require IP fragmentation. They both accept data from an application and divide that data into segments, with no segment exceeding a maximum size. Both DCCP and SCP offer manual configuration, PMTUD and PLPMTUD as mechanisms for managing that maximum size. [[I-D.ietf-tsvwg-datagram-plpmtud](#)] proposes PLPMTUD procedures for DCCP and SCP.

Currently, User Data Protocol (UDP) [[RFC0768](#)] lacks a fragmentation mechanism of its own and relies on IP fragmentation. However, [[I-D.ietf-tsvwg-udp-options](#)] proposes a fragmentation mechanism for UDP.

## 5.2. Application Layer Solutions

[RFC8085] recognizes that IP fragmentation reduces the reliability of Internet communication. It also recognizes that UDP lacks a fragmentation mechanism of its own and relies on IP fragmentation. Therefore, [[RFC8085](#)] offers the following advice regarding applications the run over the UDP.

"An application SHOULD NOT send UDP datagrams that result in IP packets that exceed the Maximum Transmission Unit (MTU) along the path to the destination. Consequently, an application SHOULD either use the path MTU information provided by the IP layer or implement Path MTU Discovery (PMTUD) itself to determine whether the path to a destination will support its desired message size without fragmentation."

[RFC 8085](#) continues:

"Applications that do not follow the recommendation to do PMTU/PLPMTUD discovery SHOULD still avoid sending UDP datagrams that would result in IP packets that exceed the path MTU. Because the actual path MTU is unknown, such applications SHOULD fall back to sending messages that are shorter than the default effective MTU for sending (EMTU\_S in [[RFC1122](#)]). For IPv4, EMTU\_S is the smaller of 576 bytes and the first-hop MTU. For IPv6, EMTU\_S is 1280 bytes. The effective PMTU for a directly connected destination (with no routers on the path) is the configured interface MTU, which could be less than the maximum link payload size. Transmission of minimum-sized



UDP datagrams is inefficient over paths that support a larger PMTU, which is a second reason to implement PMTU discovery."

[RFC 8085](#) assumes that for IPv4, an EMTU\_S of 576 is sufficiently small, even though the IPv4 minimum link MTU is 68 bytes.

This advice applies equally to application that run directly over IP.

## **6. Applications That Rely on IPv6 Fragmentation**

The following applications rely on IPv6 fragmentation:

- o DNS [[RFC1035](#)]
- o OSPFv3 [[RFC2328](#)][[RFC5340](#)]
- o Packet-in-packet encapsulations

Each of these applications relies on IPv6 fragmentation to a varying degree. In some cases, that reliance is essential, and cannot be broken without fundamentally changing the protocol. In other cases, that reliance is incidental, and most implementations already take appropriate steps to avoid fragmentation.

This list is not comprehensive, and other protocols that rely on IP fragmentation may exist. They are not specifically considered in the context of this document.

### **6.1. Domain Name Service (DNS)**

DNS relies on UDP for efficiency, and the consequence is the use of IP fragmentation for large responses, as permitted by the DNS EDNS(0) options in the query. It is possible to mitigate the issue of fragmentation-based packet loss by having queries use smaller EDNS(0) UDP buffer sizes, or by having the DNS server limit the size of its UDP responses to some self-imposed maximum packet size that may be less than the preferred EDNS(0) UDP Buffer Size. In both cases, large responses are truncated in the DNS, signalling to the client to re-query using TCP to obtain the complete response. However, the operational issue of the partial level of support for DNS over TCP, particularly in the case where IPv6 transport is being used, becomes a limiting factor of the efficacy of this approach [[Damas](#)].

Larger DNS responses can normally be avoided by aggressively pruning the Additional section of DNS responses. One scenario where such pruning is ineffective is in the use of DNSSEC, where large key sizes act to increase the response size to certain DNS queries. There is no effective response to this situation within the DNS other than





using smaller cryptographic keys and adoption of DNSSEC administrative practices that attempt to keep DNS response as short as possible.

## **6.2. Open Shortest Path First (OSPF)**

OSPF implementations can emit messages large enough to cause fragmentation. However, in order to optimize performance, most OSPF implementations restrict their maximum message size to a value that will not cause fragmentation.

## **6.3. Packet-in-Packet Encapsulations**

In this document, packet-in-packet encapsulations include IP-in-IP [[RFC2003](#)], Generic Routing Encapsulation (GRE) [[RFC2784](#)], GRE-in-UDP [[RFC8086](#)] and Generic Packet Tunneling in IPv6 [[RFC2473](#)]. [[RFC4459](#)] describes fragmentation issues associated with all of the above-mentioned encapsulations.

The fragmentation strategy described for GRE in [[RFC7588](#)] has been deployed for all of the above-mentioned encapsulations. This strategy does not rely on IP fragmentation except in one corner case. (see [Section 3.3.2.2 of RFC 7588](#) and [Section 7.1 of RFC 2473](#)). [Section 3.3 of \[RFC7676\]](#) further describes this corner case.

See [[I-D.ietf-intarea-tunnels](#)] for further discussion.

## **6.4. UDP Applications Enhancing Performance**

Some UDP applications rely on IP fragmentation to achieve acceptable levels of performance. These applications use UDP datagram sizes that are larger than the path MTU so that more data can be conveyed between the application and the kernel in a single system call.

For example, the Licklider Transmission Protocol (LTP) [[RFC5326](#)] which is in current use on the International Space Station (ISS) uses UDP datagram sizes larger than the path MTU to achieve acceptable levels of performance even though this invokes IP fragmentation.

## **7. Recommendations**

### **7.1. For Application and Protocol Developers**

Developers SHOULD NOT develop new protocols or applications that rely on IP fragmentation. When a new protocol or application is deployed in an environment that does not fully support IP fragmentation, it SHOULD operate correctly, either in its default configuration or in a specified alternative configuration.



Developers MAY develop new protocols or applications that rely on IP fragmentation if the protocol or application is to be run only in environments where IP fragmentation is known to be supported.

Legacy protocols that depend upon IP fragmentation SHOULD be updated to break that dependency. However, in some cases, there may be no viable alternative to IP fragmentation (e.g., IPSEC tunnel mode, IP-in-IP encapsulation). In these cases, the protocol will continue to rely on IP fragmentation but should only be used in environments where IP fragmentation is known to be supported.

Protocols may be able to avoid IP fragmentation by using a sufficiently small MTU (e.g. The protocol minimum link MTU), disabling IP fragmentation, and ensuring that the transport protocol in use adapts its segment size to the MTU. Other protocols may deploy a sufficiently reliable PMTU discovery mechanism (e.g., PLMPTUD).

UDP applications SHOULD abide by the recommendations state in [Section 3.2 of \[RFC8085\]](#).

## **7.2. For System Developers**

Software libraries SHOULD include provision for PLPMTUD for each supported transport protocol.

## **7.3. For Middle Box Developers**

Middle boxes should process IP fragments in a manner that is consistent with [\[RFC0791\]](#) and [\[RFC8200\]](#). In many cases, middle boxes must maintain state in order to achieve this goal.

Price and performance considerations frequently motivate network operators to deploy stateless middle boxes. These stateless middle boxes may perform sub-optimally, process IP fragments in a manner that is not compliant with [RFC 791](#) or [RFC 8200](#), or even discard IP fragments completely. Such behaviors are NOT RECOMMENDED. If a middleboxes implements non-standard behavior with respect to IP fragmentation, then that behavior MUST be clearly documented.

## **7.4. For ECMP, LAG and Load-Balancer Developers And Operators**

In their default configuration, when the IPv6 Flow Label is not equal to zero, IPv6 devices that implement ECMP, LAG or other load-balancing technologies SHOULD accept only the following fields as input to their hash algorithm:

- o IP Source Address.



- o IP Destination Address.
- o Flow Label.

Operators SHOULD deploy these devices in their default configuration.

These recommendations are similar to those presented in [\[RFC6438\]](#) and [\[RFC7098\]](#). They differ in that they specify a default configuration.

### **[7.5.](#) For Network Operators**

Operators MUST ensure proper PMTUD operation in their network, including making sure the network generates PTB packets when dropping packets too large compared to outgoing interface MTU. However, implementations MAY rate limit ICMP messages as per [\[RFC1812\]](#) and [\[RFC4443\]](#).

As per [RFC 4890](#), network operators MUST NOT filter ICMPv6 PTB messages unless they are known to be forged or otherwise illegitimate. As stated in [Section 4.7](#), filtering ICMPv6 PTB packets causes PMTUD to fail. Many upper-layer protocols rely on PMTUD.

As per [RFC 8200](#), network operators MUST NOT deploy IPv6 links whose MTU is less than 1280 bytes.

Network operators SHOULD NOT filter IP fragments if they originated at a domain name server or are destined for a domain name server. This is because domain name services are critical to operation of the Internet.

## **[8.](#) IANA Considerations**

This document makes no request of IANA.

## **[9.](#) Security Considerations**

This document mitigates some of the security considerations associated with IP fragmentation by discouraging its use. It does not introduce any new security vulnerabilities, because it does not introduce any new alternatives to IP fragmentation. Instead, it recommends well-understood alternatives.

## **[10.](#) Acknowledgements**

Thanks to Mikael Abrahamsson, Brian Carpenter, Silambu Chelvan, Lorenzo Colitti, Gorry Fairhurst, Mike Heard, Tom Herbert, Tatuya Jinmei, Jen Linkova, Paolo Lucente, Manoj Nayak, Eric Nygren, Fred Templin and Joe Touch for their comments.



## **11. References**

### **11.1. Normative References**

- [I-D.ietf-tsvwg-datagram-plpmtud]  
Fairhurst, G., Jones, T., Tuexen, M., Ruengeler, I., and  
T. Voelker, "Packetization Layer Path MTU Discovery for  
Datagram Transports", [draft-ietf-tsvwg-datagram-plpmtud-07](#)  
(work in progress), February 2019.
- [RFC0768] Postel, J., "User Datagram Protocol", STD 6, [RFC 768](#),  
DOI 10.17487/RFC0768, August 1980,  
<<https://www.rfc-editor.org/info/rfc768>>.
- [RFC0791] Postel, J., "Internet Protocol", STD 5, [RFC 791](#),  
DOI 10.17487/RFC0791, September 1981,  
<<https://www.rfc-editor.org/info/rfc791>>.
- [RFC0792] Postel, J., "Internet Control Message Protocol", STD 5,  
[RFC 792](#), DOI 10.17487/RFC0792, September 1981,  
<<https://www.rfc-editor.org/info/rfc792>>.
- [RFC0793] Postel, J., "Transmission Control Protocol", STD 7,  
[RFC 793](#), DOI 10.17487/RFC0793, September 1981,  
<<https://www.rfc-editor.org/info/rfc793>>.
- [RFC1035] Mockapetris, P., "Domain names - implementation and  
specification", STD 13, [RFC 1035](#), DOI 10.17487/RFC1035,  
November 1987, <<https://www.rfc-editor.org/info/rfc1035>>.
- [RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", [RFC 1191](#),  
DOI 10.17487/RFC1191, November 1990,  
<<https://www.rfc-editor.org/info/rfc1191>>.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate  
Requirement Levels", [BCP 14](#), [RFC 2119](#),  
DOI 10.17487/RFC2119, March 1997,  
<<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet  
Control Message Protocol (ICMPv6) for the Internet  
Protocol Version 6 (IPv6) Specification", STD 89,  
[RFC 4443](#), DOI 10.17487/RFC4443, March 2006,  
<<https://www.rfc-editor.org/info/rfc4443>>.
- [RFC4821] Mathis, M. and J. Heffner, "Packetization Layer Path MTU  
Discovery", [RFC 4821](#), DOI 10.17487/RFC4821, March 2007,  
<<https://www.rfc-editor.org/info/rfc4821>>.





- [RFC6437] Amante, S., Carpenter, B., Jiang, S., and J. Rajahalme, "IPv6 Flow Label Specification", [RFC 6437](#), DOI 10.17487/RFC6437, November 2011, <<https://www.rfc-editor.org/info/rfc6437>>.
- [RFC6438] Carpenter, B. and S. Amante, "Using the IPv6 Flow Label for Equal Cost Multipath Routing and Link Aggregation in Tunnels", [RFC 6438](#), DOI 10.17487/RFC6438, November 2011, <<https://www.rfc-editor.org/info/rfc6438>>.
- [RFC8085] Eggert, L., Fairhurst, G., and G. Shepherd, "UDP Usage Guidelines", [BCP 145](#), [RFC 8085](#), DOI 10.17487/RFC8085, March 2017, <<https://www.rfc-editor.org/info/rfc8085>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in [RFC 2119](#) Key Words", [BCP 14](#), [RFC 8174](#), DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, [RFC 8200](#), DOI 10.17487/RFC8200, July 2017, <<https://www.rfc-editor.org/info/rfc8200>>.
- [RFC8201] McCann, J., Deering, S., Mogul, J., and R. Hinden, Ed., "Path MTU Discovery for IP version 6", STD 87, [RFC 8201](#), DOI 10.17487/RFC8201, July 2017, <<https://www.rfc-editor.org/info/rfc8201>>.

## **11.2. Informative References**

- [Damas] Damas, J. and G. Huston, "Measuring ATR", April 2018, <<http://www.potaroo.net/ispcol/2018-04/atr.html>>.
- [Huston] Huston, G., "IPv6, Large UDP Packets and the DNS (<http://www.potaroo.net/ispcol/2017-08/xtn-hdrs.html>)", August 2017.
- [I-D.ietf-intarea-tunnels] Touch, J. and M. Townsley, "IP Tunnels in the Internet Architecture", [draft-ietf-intarea-tunnels-09](#) (work in progress), July 2018.
- [I-D.ietf-tsvwg-udp-options] Touch, J., "Transport Options for UDP", [draft-ietf-tsvwg-udp-options-07](#) (work in progress), March 2019.



- [Kent] Kent, C. and J. Mogul, "'Fragmentation Considered Harmful", In Proc. SIGCOMM '87 Workshop on Frontiers in Computer Communications Technology, DOI 10.1145/55483.55524", August 1987, <<http://www.hp1.hp.com/techreports/Compaq-DEC/WRL-87-3.pdf>>.
- [Ptacek1998] Ptacek, T. and T. Newsham, "Insertion, Evasion and Denial of Service: Eluding Network Intrusion Detection", 1998, <<http://www.aciri.org/vern/Ptacek-Newsham-Evasion-98.ps>>.
- [RFC1122] Braden, R., Ed., "Requirements for Internet Hosts - Communication Layers", STD 3, [RFC 1122](#), DOI 10.17487/RFC1122, October 1989, <<https://www.rfc-editor.org/info/rfc1122>>.
- [RFC1812] Baker, F., Ed., "Requirements for IP Version 4 Routers", [RFC 1812](#), DOI 10.17487/RFC1812, June 1995, <<https://www.rfc-editor.org/info/rfc1812>>.
- [RFC1858] Ziemba, G., Reed, D., and P. Traina, "Security Considerations for IP Fragment Filtering", [RFC 1858](#), DOI 10.17487/RFC1858, October 1995, <<https://www.rfc-editor.org/info/rfc1858>>.
- [RFC2003] Perkins, C., "IP Encapsulation within IP", [RFC 2003](#), DOI 10.17487/RFC2003, October 1996, <<https://www.rfc-editor.org/info/rfc2003>>.
- [RFC2328] Moy, J., "OSPF Version 2", STD 54, [RFC 2328](#), DOI 10.17487/RFC2328, April 1998, <<https://www.rfc-editor.org/info/rfc2328>>.
- [RFC2473] Conta, A. and S. Deering, "Generic Packet Tunneling in IPv6 Specification", [RFC 2473](#), DOI 10.17487/RFC2473, December 1998, <<https://www.rfc-editor.org/info/rfc2473>>.
- [RFC2784] Farinacci, D., Li, T., Hanks, S., Meyer, D., and P. Traina, "Generic Routing Encapsulation (GRE)", [RFC 2784](#), DOI 10.17487/RFC2784, March 2000, <<https://www.rfc-editor.org/info/rfc2784>>.
- [RFC3128] Miller, I., "Protection Against a Variant of the Tiny Fragment Attack ([RFC 1858](#))", [RFC 3128](#), DOI 10.17487/RFC3128, June 2001, <<https://www.rfc-editor.org/info/rfc3128>>.



- [RFC4340] Kohler, E., Handley, M., and S. Floyd, "Datagram Congestion Control Protocol (DCCP)", [RFC 4340](#), DOI 10.17487/RFC4340, March 2006, <<https://www.rfc-editor.org/info/rfc4340>>.
- [RFC4459] Savola, P., "MTU and Fragmentation Issues with In-the-Network Tunneling", [RFC 4459](#), DOI 10.17487/RFC4459, April 2006, <<https://www.rfc-editor.org/info/rfc4459>>.
- [RFC4890] Davies, E. and J. Mohacsi, "Recommendations for Filtering ICMPv6 Messages in Firewalls", [RFC 4890](#), DOI 10.17487/RFC4890, May 2007, <<https://www.rfc-editor.org/info/rfc4890>>.
- [RFC4960] Stewart, R., Ed., "Stream Control Transmission Protocol", [RFC 4960](#), DOI 10.17487/RFC4960, September 2007, <<https://www.rfc-editor.org/info/rfc4960>>.
- [RFC4963] Heffner, J., Mathis, M., and B. Chandler, "IPv4 Reassembly Errors at High Data Rates", [RFC 4963](#), DOI 10.17487/RFC4963, July 2007, <<https://www.rfc-editor.org/info/rfc4963>>.
- [RFC5326] Ramadas, M., Burleigh, S., and S. Farrell, "Licklider Transmission Protocol - Specification", [RFC 5326](#), DOI 10.17487/RFC5326, September 2008, <<https://www.rfc-editor.org/info/rfc5326>>.
- [RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", [RFC 5340](#), DOI 10.17487/RFC5340, July 2008, <<https://www.rfc-editor.org/info/rfc5340>>.
- [RFC5722] Krishnan, S., "Handling of Overlapping IPv6 Fragments", [RFC 5722](#), DOI 10.17487/RFC5722, December 2009, <<https://www.rfc-editor.org/info/rfc5722>>.
- [RFC5927] Gont, F., "ICMP Attacks against TCP", [RFC 5927](#), DOI 10.17487/RFC5927, July 2010, <<https://www.rfc-editor.org/info/rfc5927>>.
- [RFC6346] Bush, R., Ed., "The Address plus Port (A+P) Approach to the IPv4 Address Shortage", [RFC 6346](#), DOI 10.17487/RFC6346, August 2011, <<https://www.rfc-editor.org/info/rfc6346>>.
- [RFC6864] Touch, J., "Updated Specification of the IPv4 ID Field", [RFC 6864](#), DOI 10.17487/RFC6864, February 2013, <<https://www.rfc-editor.org/info/rfc6864>>.



- [RFC6888] Perreault, S., Ed., Yamagata, I., Miyakawa, S., Nakagawa, A., and H. Ashida, "Common Requirements for Carrier-Grade NATs (CGNs)", [BCP 127](#), [RFC 6888](#), DOI 10.17487/RFC6888, April 2013, <<https://www.rfc-editor.org/info/rfc6888>>.
- [RFC7098] Carpenter, B., Jiang, S., and W. Tarreau, "Using the IPv6 Flow Label for Load Balancing in Server Farms", [RFC 7098](#), DOI 10.17487/RFC7098, January 2014, <<https://www.rfc-editor.org/info/rfc7098>>.
- [RFC7588] Bonica, R., Pignataro, C., and J. Touch, "A Widely Deployed Solution to the Generic Routing Encapsulation (GRE) Fragmentation Problem", [RFC 7588](#), DOI 10.17487/RFC7588, July 2015, <<https://www.rfc-editor.org/info/rfc7588>>.
- [RFC7676] Pignataro, C., Bonica, R., and S. Krishnan, "IPv6 Support for Generic Routing Encapsulation (GRE)", [RFC 7676](#), DOI 10.17487/RFC7676, October 2015, <<https://www.rfc-editor.org/info/rfc7676>>.
- [RFC7739] Gont, F., "Security Implications of Predictable Fragment Identification Values", [RFC 7739](#), DOI 10.17487/RFC7739, February 2016, <<https://www.rfc-editor.org/info/rfc7739>>.
- [RFC7872] Gont, F., Linkova, J., Chown, T., and W. Liu, "Observations on the Dropping of Packets with IPv6 Extension Headers in the Real World", [RFC 7872](#), DOI 10.17487/RFC7872, June 2016, <<https://www.rfc-editor.org/info/rfc7872>>.
- [RFC8086] Yong, L., Ed., Crabbe, E., Xu, X., and T. Herbert, "GRE-in-UDP Encapsulation", [RFC 8086](#), DOI 10.17487/RFC8086, March 2017, <<https://www.rfc-editor.org/info/rfc8086>>.

## [Appendix A](#). Contributors' Address

### Authors' Addresses

Ron Bonica  
Juniper Networks  
2251 Corporate Park Drive  
Herndon, Virginia 20171  
USA

Email: [rbonica@juniper.net](mailto:rbonica@juniper.net)





Fred Baker  
Unaffiliated  
Santa Barbara, California 93117  
USA

Email: FredBaker.IETF@gmail.com

Geoff Huston  
APNIC  
6 Cordelia St  
Brisbane, 4101 QLD  
Australia

Email: gih@apnic.net

Robert M. Hinden  
Check Point Software  
959 Skyway Road  
San Carlos, California 94070  
USA

Email: bob.hinden@gmail.com

Ole Troan  
Cisco  
Philip Pedersens vei 1  
N-1366 Lysaker  
Norway

Email: ot@cisco.com

Fernando Gont  
SI6 Networks  
Evaristo Carriego 2644  
Haedo, Provincia de Buenos Aires  
Argentina

Email: fgont@si6networks.com

