IPv6 maintenance Working Group (6man)
Internet-Draft
Updates: 2460, 6145 (if approved)
Intended status: Standards Track
Expires: October 29, 2015

F. Gont
SI6 Networks / UTN-FRH
W. Liu
Huawei Technologies
T. Anderson
Redpill Linpro
April 27, 2015

Deprecating the Generation of IPv6 Atomic Fragments
draft-ietf-6man-deprecate-atomfrag-generation-01

Abstract

The core IPv6 specification requires that when a host receives an ICMPv6 "Packet Too Big" message reporting a "Next-Hop MTU" smaller than 1280, the host includes a Fragment Header in all subsequent packets sent to that destination, without reducing the assumed Path-MTU. The simplicity with which ICMPv6 "Packet Too Big" messages can be forged, coupled with the widespread filtering of IPv6 fragments, results in an attack vector that can be leveraged for Denial of Service purposes. This document briefly discusses the aforementioned attack vector, and formally updates RFC2460 such that generation of IPv6 atomic fragments is deprecated, thus eliminating the aforementioned attack vector. Additionally, it formally updates RFC6145 such that the Stateless IP/ICMP Translation Algorithm (SIIT) does not rely on the generation of IPv6 atomic fragments, thus improving the robustness of the protocol.

Table of Contents

1. Introduction

[RFC2460] specifies the IPv6 fragmentation mechanism, which allows IPv6 packets to be fragmented into smaller pieces such that they fit in the Path-MTU to the intended destination(s).

Section 5 of [RFC2460] states that, when a host receives an ICMPv6 "Packet Too Big" message [RFC4443] advertising a "Next-Hop MTU" smaller than 1280 (the minimum IPv6 MTU), the host is not required to reduce the assumed Path-MTU, but must simply include a Fragment Header in all subsequent packets sent to that destination. The resulting packets will thus *not* be actually fragmented into several pieces, but rather just include a Fragment Header with both the "Fragment Offset" and the "M" flag set to 0 (we refer to these packets as "atomic fragments"). As required by [RFC6946], these

atomic fragments are essentially processed by the destination host as non-fragment traffic (since there are not really any fragments to be reassembled). IPv6/IPv4 translators will typically employ the Fragment Identification information found in the Fragment Header to select an appropriate Fragment Identification value for the resulting IPv4 fragments.

While atomic fragments might seem rather benign, there are scenarios in which the generation of IPv6 atomic fragments can introduce an attack vector that can be exploited for denial of service purposes. Since there are concrete security implications arising from the generation of IPv6 atomic fragments, and there is no real gain in generating IPv6 atomic fragments (as opposed to e.g. having IPv6/IPv4 translators generate a Fragment Identification value themselves), this document formally updates [RFC2460], forbidding the generation of IPv6 atomic fragments, such that the aforementioned attack vector is eliminated. Additionally, it formally updates [RFC6145] such that the Stateless IP/ICMP Translation Algorithm (SIIT) does not rely on the generation of IPv6 atomic fragments.

Section 3 describes some possible attack scenarios. Section 4 provides additional considerations regarding the usefulness of generating IPv6 atomic fragments. Section 5 formally updates RFC2460 such that this attack vector is eliminated. Section 6 formally updates RFC6145 such that it does not relies on the generation of IPv6 atomic fragments.

2. Terminology

IPv6 atomic fragments
: IPv6 packets that contain a Fragment Header with the Fragment Offset set to 0 and the M flag set to 0 (as defined by [RFC6946]).

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

3. Denial of Service (DoS) attack vector

Let us assume that Host A is communicating with Server B, and that, as a result of the widespread filtering of IPv6 packets with extension headers (including fragmentation) [I-D.gont-v6ops-ipv6-ehs-in-real-world], some intermediate node filters fragments between Host A and Server B. If an attacker sends a forged ICMPv6 "Packet Too Big" (PTB) error message to server B, reporting a Next-Hop MTU smaller than 1280, this will trigger the generation of IPv6 atomic fragments from that moment on (as required by [RFC2460]). When server B starts sending IPv6 atomic fragments

(in response to the received ICMPv6 PTB), these packets will be dropped, since we previously noted that packets with IPv6 EHs were being dropped between Host A and Server B. Thus, this situation will result in a Denial of Service (DoS) scenario.

Another possible scenario is that in which two BGP peers are employing IPv6 transport, and they implement ACLs to drop IPv6 fragments (to avoid control-plane attacks). If the aforementioned BGP peers drop IPv6 fragments but still honor received ICMPv6 Packet Too Big error messages, an attacker could easily attack the peering session by simply sending an ICMPv6 PTB message with a reported MTU smaller than 1280 bytes. Once the attack packet has been fired, it will be the aforementioned routers themselves the ones dropping their own traffic.

The aforementioned attack vector is exacerbated by the following factors:

- The attacker does not need to forge the IPv6 Source Address of his attack packets. Hence, deployment of simple BCP38 filters will not help as a counter-measure.

- Only the IPv6 addresses of the IPv6 packet embedded in the ICMPv6 payload need to be forged. While one could envision filtering devices enforcing BCP38-style filters on the ICMPv6 payload, the use of extension (by the attacker) could make this difficult, if at all possible.

- Many implementations fail to perform validation checks on the received ICMPv6 error messages, as recommended in Section 5.2 of [RFC4443] and documented in [RFC5927]. It should be noted that in some cases, such as when an ICMPv6 error message has (supposedly) been elicited by a connection-less transport protocol (or some other connection-less protocol being encapsulated in IPv6), it may be virtually impossible to perform validation checks on the received ICMPv6 error messages. And, because of IPv6 extension headers, the ICMPv6 payload might not even contain any useful information on which to perform validation checks.

- Upon receipt of one of the aforementioned ICMPv6 "Packet Too Big" error messages, the Destination Cache [RFC4861] is usually updated to reflect that any subsequent packets to such destination should include a Fragment Header. This means that a single ICMPv6 "Packet Too Big" error message might affect multiple communication instances (e.g., TCP connections) with such destination.

- As noted in Section 4, SIIT [RFC6145] (including derivative protocols such as Stateful NAT64 [RFC6146]) is the only technology

which currently makes use of atomic fragments. Unfortunately, an IPv6 node cannot easily limit its exposure to the aforementioned attack vector by only generating IPv6 atomic fragments towards IPv4 destinations behind a stateless translator. This is due to the fact that Section 3.3 of RFC6052 [RFC6052] encourages operators to use a Network-Specific Prefix (NSP) that maps the IPv4 address space into IPv6. When an NSP is being used, IPv6 addresses representing IPv4 nodes (reached through a stateless translator) are indistinguishable from native IPv6 addresses.

4. Additional Considerations

Besides the security assessment provided in Section 3, it is interesting to evaluate the pros and cons of having an IPv6-to-IPv4 translating router rely on the generation of IPv6 atomic fragments.

Relying on the generation of IPv6 atomic fragments implies a reliance on:

1. ICMPv6 packets arriving from the translator to the IPv6 node

2. The ability of the nodes receiving ICMPv6 PTB messages reporting an MTU smaller than 1280 bytes to actually produce atomic fragments

3. Support for IPv6 fragmentation on the IPv6 side of the translator

Unfortunately,

- There exists a fair share of evidence of ICMPv6 Packet Too Big messages being dropped on the public Internet (for instance, that is one of the reasons for which PLPMTUD [RFC4821] was produced). Therefore, relying on such messages being successfully delivered will affect the robustness of the protocol that relies on them.

- A number of IPv6 implementations have been known to fail to generate IPv6 atomic fragments in response to ICMPv6 PTB messages reporting an MTU smaller than 1280 bytes (see Appendix A for a small survey). Additionally, results included in Section 6 of [RFC6145] note that 57% of the tested web servers failed to produce IPv6 atomic fragments in response to ICMPv6 PTB messages reporting an MTU smaller than 1280 bytes. Thus, any protocol relying on IPv6 atomic fragment generation for proper functioning will have interoperability problems with the aforementioned IPv6 stacks.

- IPv6 atomic fragment generation represents a case in which fragmented traffic is produced where otherwise it would not be

needed. Since there is widespread filtering of IPv6 fragments in the public Internet [I-D.gont-v6ops-ipv6-ehs-in-real-world], this would mean that the (unnecessary) use of IPv6 fragmentation might result, unnecessarily, in a Denial of Service situation even in legitimate cases.

Finally, we note that SIIT essentially employs the Fragment Header of IPv6 atomic fragments to signal the translator how to set the DF bit of IPv4 datagrams (the DF bit is cleared when the IPv6 packet contains a Fragment Header, and is otherwise set to 1 when the IPv6 packet does not contain an IPv6 Fragment Header). Additionally, the translator will employ the low-order 16-bits of the IPv6 Fragment Identification for setting the IPv4 Fragment Identification. At least in theory, this is expected to reduce the Fragment ID collision rate in the following specific scenario:

1. An IPv6 node communicates with an IPv4 node (through SIIT)

2. The IPv4 node is located behind an IPv4 link with an MTU < 1260

3. ECMP routing [RFC2992] with more than one translator are employed for e.g., redundancy purposes

In such a scenario, if each translator were to select the IPv4 Fragment Identification on its own (rather than selecting the IPv4 Fragment ID from the low-order 16-bits of the Fragment Identification of atomic fragments), this could possibly lead to IPv4 Fragment ID collisions. However, since a number of implementations set IPv6 Fragment ID according to the output of a Pseudo-Random Number Generator (PRNG) (see Appendix B of [I-D.ietf-6man-predictable-fragment-id]) and the translator only employs the low-order 16-bits of such value, it is very unlikely that relying on the Fragment ID of the IPv6 atomic fragment will result in a reduced Fragment ID collision rate (when compared to the case where the translator selects each IPv4 Fragment ID on its own).

Finally, we note that [RFC6145] is currently the only "consumer" of IPv6 atomic fragments, and it correctly and diligently notes (in Section 6) the possible interoperability problems of relying on IPv6 atomic fragments, proposing as a workaround something very similar to what we propose in Section 6. We believe that, by making the more robust behavior the default behavior of the "IP/ICMP Translation Algorithm", robustness is improved, and the corresponding code is simplified.

5. Updating RFC2460

The following text from Section 5 of [RFC2460]:

> "In response to an IPv6 packet that is sent to an IPv4 destination (i.e., a packet that undergoes translation from IPv6 to IPv4), the originating IPv6 node may receive an ICMP Packet Too Big message reporting a Next-Hop MTU less than 1280. In that case, the IPv6 node is not required to reduce the size of subsequent packets to less than 1280, but must include a Fragment header in those packets so that the IPv6-to-IPv4 translating router can obtain a suitable Identification value to use in resulting IPv4 fragments. Note that this means the payload may have to be reduced to 1232 octets (1280 minus 40 for the IPv6 header and 8 for the Fragment header), and smaller still if additional extension headers are used."

is formally replaced with:

> "An IPv6 node that receives an ICMPv6 Packet Too Big error message that reports a Next-Hop MTU smaller than 1280 bytes (the minimum IPv6 MTU) MUST NOT include a Fragment header in subsequent packets sent to the corresponding destination. That is, IPv6 nodes MUST NOT generate IPv6 atomic fragments."

6. Updating RFC6145

The following text from Section 4 (Translating from IPv4 to IPv6) of [RFC6145]:

---------- cut here ---------- cut here ----------
When the IPv4 sender does not set the DF bit, the translator SHOULD always include an IPv6 Fragment Header to indicate that the sender allows fragmentation. The translator MAY provide a configuration function that allows the translator not to include the Fragment Header for the non-fragmented IPv6 packets.

The rules in Section 4.1 ensure that when packets are fragmented, either by the sender or by IPv4 routers, the low-order 16 bits of the fragment identification are carried end-to-end, ensuring that packets are correctly reassembled. In addition, the rules in Section 4.1 use the presence of an IPv6 Fragment Header to indicate that the sender might not be using path MTU discovery (i.e., the packet should not have the DF flag set should it later be translated back to IPv4).
---------- cut here ---------- cut here ----------

is formally replaced with:

---------- cut here ---------- cut here ----------
The rules in Section 4.1 ensure that when packets are fragmented, either by the sender or by IPv4 routers, the low-order 16 bits of the fragment identification are carried end-to-end, ensuring that packets are correctly reassembled.
---------- cut here ---------- cut here ----------

The following text from Section 4.1 ("Translating IPv4 Headers into IPv6 Headers") of [RFC6145]:

---------- cut here ---------- cut here ----------
If there is a need to add a Fragment Header (the DF bit is not set or the packet is a fragment), the header fields are set as above with the following exceptions:
---------- cut here ---------- cut here ----------

is formally replaced with:

---------- cut here ---------- cut here ----------
If there is a need to add a Fragment Header (the packet is a fragment), the header fields are set as above with the following exceptions:
---------- cut here ---------- cut here ----------

The following text from Section 4.2 ("Translating ICMPv4 Headers into ICMPv6 Headers") of [RFC6145]:

---------- cut here ---------- cut here ----------
Code 4 (Fragmentation Needed and DF was Set): Translate to an ICMPv6 Packet Too Big message (Type 2) with Code set to 0. The MTU field MUST be adjusted for the difference between the IPv4 and IPv6 header sizes, i.e., minimum(advertised MTU+20, MTU_of_IPv6_nexthop, (MTU_of_IPv4_nexthop)+20). Note that if the IPv4 router set the MTU field to zero, i.e., the router does not implement [RFC1191], then the translator MUST use the plateau values specified in [RFC1191] to determine a likely path MTU and include that path MTU in the ICMPv6 packet. (Use the greatest plateau value that is less than the returned Total Length field.)
---------- cut here ---------- cut here ----------

is formally replaced with:

---------- cut here ---------- cut here ----------

Code 4 (Fragmentation Needed and DF was Set): Translate to an ICMPv6 Packet Too Big message (Type 2) with Code set to 0. The MTU field MUST be adjusted for the difference between the IPv4 and IPv6 header sizes, but MUST NOT be set to a value smaller than the minimum IPv6 MTU (1280 bytes). That is, it should be set to maximum(1280, minimum(advertised MTU+20, MTU_of_IPv6_nexthop, (MTU_of_IPv4_nexthop)+20)). Note that if the IPv4 router set the MTU field to zero, i.e., the router does not implement [RFC1191], then the translator MUST use the plateau values specified in [RFC1191] to determine a likely path MTU and include that path MTU in the ICMPv6 packet. (Use the greatest plateau value that is less than the returned Total Length field, but that is larger than or equal to 1280.)

---------- cut here ---------- cut here ----------

The following text from Section 5 ("Translating from IPv6 to IPv4") of [RFC6145]:

---------- cut here ---------- cut here ----------
There are some differences between IPv6 and IPv4 (in the areas of fragmentation and the minimum link MTU) that affect the translation. An IPv6 link has to have an MTU of 1280 bytes or greater. The corresponding limit for IPv4 is 68 bytes. Path MTU discovery across a translator relies on ICMP Packet Too Big messages being received and processed by IPv6 hosts, including an ICMP Packet Too Big that indicates the MTU is less than the IPv6 minimum MTU. This requirement is described in Section 5 of [RFC2460] (for IPv6's 1280-octet minimum MTU) and Section 5 of [RFC1883] (for IPv6's previous 576-octet minimum MTU).

In an environment where an ICMPv4 Packet Too Big message is translated to an ICMPv6 Packet Too Big message, and the ICMPv6 Packet Too Big message is successfully delivered to and correctly processed by the IPv6 hosts (e.g., a network owned/operated by the same entity that owns/operates the translator), the translator can rely on IPv6 hosts sending subsequent packets to the same IPv6 destination with IPv6 Fragment Headers. In such an environment, when the translator receives an IPv6 packet with a Fragment Header, the translator SHOULD generate the IPv4 packet with a cleared Don't Fragment bit, and with its identification value from the IPv6 Fragment Header, for all of the IPv6 fragments (MF=0 or MF=1).

In an environment where an ICMPv4 Packet Too Big message is filtered (by a network firewall or by the host itself) or not correctly processed by the IPv6 hosts, the IPv6 host will never generate an IPv6 packet with the IPv6 Fragment Header. In such an environment, the translator SHOULD set the IPv4 Don't Fragment bit. While setting the Don't Fragment bit may create PMTUD black holes [RFC2923] if there are IPv4 links smaller than 1260 octets, this is considered safer than causing IPv4 reassembly errors [RFC4963].
---------- cut here ---------- cut here ----------

is formally replaced with:

---------- cut here ---------- cut here ----------
There are some differences between IPv6 and IPv4 (in the areas of fragmentation and the minimum link MTU) that affect the translation. An IPv6 link has to have an MTU of 1280 bytes or greater. The corresponding limit for IPv4 is 68 bytes. Path MTU discovery across a translator relies on ICMP Packet Too Big messages being received and processed by IPv6 hosts.

The difference in the minimum MTUs of IPv4 and IPv6 is accommodated as follows:

- When translating an ICMPv4 "Fragmentation Needed" packet, the indicated MTU in the resulting ICMPv6 "Packet Too Big" will never be set to a value lower than 1280. This ensures that the IPv6 nodes will never have to encounter or handle Path MTU values lower than the minimum IPv6 link MTU of 1280. See Section 4.2.

- When the resulting IPv4 packet is smaller than or equal to 1260 bytes, the translator MUST send the packet with a cleared Don't Fragment bit. Otherwise, the packet MUST be sent with the Don't Fragment bit set. See Section 5.1.

This approach allows Path MTU Discovery to operate end-to-end for paths whose MTU are not smaller than minimum IPv6 MTU of 1280 (which corresponds to MTU of 1260 in the IPv4 domain). On paths that have IPv4 links with MTU < 1260, the IPv4 router(s) connected to those links will fragment the packets in accordance with Section 2.3 of [RFC0791].
---------- cut here ---------- cut here ----------

The following text from Section 5.1 ("Translating IPv6 Headers into IPv4 Headers") of [RFC6145]:

---------- cut here ---------- cut here ----------
Identification: All zero. In order to avoid black holes caused by ICMPv4 filtering or non-[RFC2460]-compatible IPv6 hosts (a workaround is discussed in Section 6), the translator MAY provide a function to generate the identification value if the packet size is greater than 88 bytes and less than or equal to 1280 bytes. The translator SHOULD provide a method for operators to enable or disable this function.

Flags: The More Fragments flag is set to zero. The Don't Fragment (DF) flag is set to one. In order to avoid black holes caused by ICMPv4 filtering or non-[RFC2460]-compatible IPv6 hosts (a workaround is discussed in Section 6), the translator MAY provide a function as follows. If the packet size is greater than 88 bytes and less than or equal to 1280 bytes, it sets the DF flag to zero; otherwise, it sets the DF flag to one. The translator SHOULD provide a method for operators to enable or disable this function.
---------- cut here ---------- cut here ----------

is formally replaced with:

---------- cut here ---------- cut here ----------
Identification: Set according to a Fragment Identification generator at the translator.

Flags: The More Fragments flag is set to zero. The Don't Fragment (DF) flag is set as follows: If the size of the translated IPv4 packet is less than or equal to 1260 bytes, it is set to zero; otherwise, it is set to one.
---------- cut here ---------- cut here ----------

The following text from Section 5.1.1 ("IPv6 Fragment Processing") of [RFC6145]:

---------- cut here ---------- cut here ----------
If a translated packet with DF set to 1 will be larger than the MTU of the next-hop interface, then the translator MUST drop the packet and send the ICMPv6 Packet Too Big (Type 2, Code 0) error message to the IPv6 host with an adjusted MTU in the ICMPv6 message.
---------- cut here ---------- cut here ----------

is formally replaced with:

---------- cut here ---------- cut here ----------
If an IPv6 packet that is smaller than or equal to 1280 bytes results (after translation) in an IPv4 packet that is larger than the MTU of the next-hop interface, then the translator MUST perform IPv4 fragmentation on that packet such that it can be transferred over the constricting link.
---------- cut here ---------- cut here ----------

Finally, the following text from 6 ("Special Considerations for ICMPv6 Packet Too Big") of [RFC6145]:

---------- cut here ---------- cut here ----------
Two recent studies analyzed the behavior of IPv6-capable web servers on the Internet and found that approximately 95% responded as expected to an IPv6 Packet Too Big that indicated MTU = 1280, but only 43% responded as expected to an IPv6 Packet Too Big that indicated an MTU < 1280. It is believed that firewalls violating Section 4.3.1 of [RFC4890] are at fault. Both failures (the 5% wrong response when MTU = 1280 and the 57% wrong response when MTU < 1280) will cause PMTUD black holes [RFC2923]. Unfortunately, the translator cannot improve the failure rate of the first case (MTU = 1280), but the translator can improve the failure rate of the second case (MTU < 1280). There are two approaches to resolving the problem with sending ICMPv6 messages indicating an MTU < 1280. It SHOULD be possible to configure a translator for either of the two approaches.

The first approach is to constrain the deployment of the IPv6/IPv4 translator by observing that four of the scenarios intended for stateless IPv6/IPv4 translators do not have IPv6 hosts on the Internet (Scenarios 1, 2, 5, and 6 described in [RFC6144], which refer to "An IPv6 network"). In these scenarios, IPv6 hosts, IPv6-host-based firewalls, and IPv6 network firewalls can be administered in compliance with Section 4.3.1 of [RFC4890] and therefore avoid the problem witnessed with IPv6 hosts on the Internet.

The second approach is necessary if the translator has IPv6 hosts, IPv6-host-based firewalls, or IPv6 network firewalls that do not (or cannot) comply with Section 5 of [RFC2460] -- such as IPv6 hosts on the Internet. This approach requires the translator to do the following:

1. In the IPv4-to-IPv6 direction: if the MTU value of ICMPv4 Packet Too Big (PTB) messages is less than 1280, change it to 1280. This is intended to cause the IPv6 host and IPv6 firewall to process the ICMP PTB message and generate subsequent packets to this destination with an IPv6 Fragment Header.

   Note: Based on recent studies, this is effective for 95% of IPv6

hosts on the Internet.

2. In the IPv6-to-IPv4 direction:

   A. If there is a Fragment Header in the IPv6 packet, the last 16 bits of its value MUST be used for the IPv4 identification value.

   B. If there is no Fragment Header in the IPv6 packet:

      a. If the packet is less than or equal to 1280 bytes:

         - The translator SHOULD set DF to 0 and generate an IPv4 identification value.

         - To avoid the problems described in [RFC4963], it is RECOMMENDED that the translator maintain 3-tuple state for generating the IPv4 identification value.

      b. If the packet is greater than 1280 bytes, the translator SHOULD set the IPv4 DF bit to 1.

---------- cut here ---------- cut here ----------

is formally replaced with:

---------- cut here ---------- cut here ----------
A number of studies (see e.g.) indicate that it not unusual for networks to drop ICMPv6 Packet Too Big error messages. Such packet drops will result in PMTUD blackholes [RFC2923], which can only be overcome with PLPMTUD [RFC4821].
---------- cut here ---------- cut here ----------

7. IANA Considerations

There are no IANA registries within this document. The RFC-Editor can remove this section before publication of this document as an RFC.

8. Security Considerations

This document describes a Denial of Service (DoS) attack vector that leverages the widespread filtering of IPv6 fragments in the public Internet by means of ICMPv6 PTB error messages. Additionally, it formally updates [RFC2460] such that this attack vector is eliminated, and also formally updated [RFC6145] such that it does not rely on IPv6 atomic fragments.

9. Acknowledgements

The authors would like to thank (in alphabetical order) Alberto Leiva, Bob Briscoe, Brian Carpenter, Tatuya Jinmei, Jeroen Massar, and Erik Nordmark, for providing valuable comments on earlier versions of this document.

Fernando Gont would like to thank Jan Zorz and Go6 Lab <http://go6lab.si/> for providing access to systems and networks that were employed to produce some of tests that resulted in the publication of this document. Additionally, he would like to thank SixXS <https://www.sixxs.net> for providing IPv6 connectivity.

10. References

10.1. Normative References

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

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

[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.

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

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

[RFC6145] Li, X., Bao, C., and F. Baker, "IP/ICMP Translation Algorithm", RFC 6145, April 2011.

10.2. Informative References

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

[RFC2992] Hopps, C., "Analysis of an Equal-Cost Multi-Path Algorithm", RFC 2992, November 2000.

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

[RFC6052] Bao, C., Huitema, C., Bagnulo, M., Boucadair, M., and X. Li, "IPv6 Addressing of IPv4/IPv6 Translators", RFC 6052, October 2010.

[RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, April 2011.

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

[I-D.ietf-6man-predictable-fragment-id] Gont, F., "Security Implications of Predictable Fragment Identification Values", draft-ietf-6man-predictable-fragment-id-05 (work in progress), April 2015.

[I-D.gont-v6ops-ipv6-ehs-in-real-world] Gont, F., Linkova, J., Chown, T., and W. Will, "Observations on IPv6 EH Filtering in the Real World", draft-gont-v6ops-ipv6-ehs-in-real-world-02 (work in progress), March 2015.

[Morbitzer] Morbitzer, M., "TCP Idle Scans in IPv6", Master's Thesis. Thesis number: 670. Department of Computing Science, Radboud University Nijmegen. August 2013, <https://www.ru.nl/publish/pages/578936/m_morbitzer_masterthesis.pdf>.

Appendix A. Small Survey of OSes that Fail to Produce IPv6 Atomic Fragments

[This section will probably be removed from this document before it is published as an RFC].

This section includes a non-exhaustive list of operating systems that *fail* to produce IPv6 atomic fragments. It is based on the results published in [RFC6946] and [Morbitzer].

The following Operating Systems fail to generate IPv6 atomic fragments in response to ICMPv6 PTB messages that report an MTU smaller than 1280 bytes:

- FreeBSD 8.0
- Linux kernel 2.6.32
- Linux kernel 3.2

- Mac OS X 10.6.7

- NetBSD 5.1

Authors' Addresses

Fernando Gont
SI6 Networks / UTN-FRH
Evaristo Carriego 2644
Haedo, Provincia de Buenos Aires 1706
Argentina

Phone: +54 11 4650 8472
Email: fgont@si6networks.com
URI: http://www.si6networks.com

Will(Shucheng) Liu
Huawei Technologies
Bantian, Longgang District
Shenzhen 518129
P.R. China

Email: liushucheng@huawei.com

Tore Anderson
Redpill Linpro
Vitaminveien 1A
Oslo 0485
Norway

Phone: +47 959 31 212
Email: tore@redpill-linpro.com
URI: http://www.redpill-linpro.com