INTERNET-DRAFT
Intended Status: Standard
Expires: March 2020

T. Herbert
Intel

September 24, 2019

ICMPv6 errors for discarding packets due to processing limits
draft-ietf-6man-icmp-limits-06

Abstract

Network nodes may discard packets if they are unable to process protocol headers of packets due to processing constraints or limits. When such packets are dropped, the sender receives no indication so it cannot take action to address the cause of discarded packets. This specification defines several new ICMPv6 errors that can be sent by a node that discards packets because it is unable to process the protocol headers. A node that receives such an ICMPv6 error may be able to modify what it sends in future packets to avoid subsequent packet discards.

Table of Contents

1 Introduction

This document specifies several new ICMPv6 errors that can be sent when a node discards a packet due to it being unable to process the necessary protocol headers because of processing constraints or limits. New ICMPv6 code points are defined as an update to [RFC4443]. Five of the errors are specific to processing of extension headers; another error is used when the aggregate protocol headers in a packet exceed the processing limits of a node.

1.1 Extension header limits

In IPv6, optional internet-layer information is carried in one or more IPv6 Extension Headers [RFC8200]. Extension Headers are placed between the IPv6 header and the Upper-Layer Header in a packet. The term "Header Chain" refers collectively to the IPv6 header, Extension Headers, and Upper-Layer Headers occurring in a packet. Individual extension headers may have a maximum length of 2048 octets and must fit into a single packet. Destination Options and Hop-by-Hop Options contain a list of options in Type-length-value (TLV) format. Each option includes a length of the data field in octets: the minimum size of an option (non-pad type) is two octets and the maximum size is 257 octets. The number of options in an extension header is only limited by the length of the extension header and the Path MTU from the source to the destination. Options may be skipped over by a receiver if they are unknown and the Option Type indicates to skip (first two high order bits are 00).

Per [RFC8200], except for Hop by Hop options, extension headers are not examined or processed by intermediate nodes. Many intermediate nodes, however, do examine extension header for various purposes. For instance, a node may examine all extension headers to locate the transport header of a packet in order to implement transport layer filtering or to track connections to implement a stateful firewall.

Destination hosts are expected to process all extension headers and options in Hop-by-Hop and Destination Options.

Due to the variable lengths, high maximum lengths, or potential for Denial of Service attack of extension headers, many devices impose operational limits on extension headers in packets they process. [RFC7045] discusses the requirements of intermediate nodes that discard packets because of unrecognized extension headers. [RFC8504] discusses limits that may be applied to the number of options in Hop-by-Hop Options or Destination Options extension headers. Both intermediate nodes and end hosts may apply limits to extension header processing. When a limit is exceeded, the typical behavior is to silently discard the packet.

This specification defines four Parameter Problem codes and extends the applicably of an existing code that may be sent by a node that discards a packet due to processing limits of extension headers being exceeded. A source host that receives an ICMPv6 error may modify its use of extension headers in subsequent packets sent to the destination in order to avoid further occurrences of packets being discarded.

1.2 Aggregate header limits

Some hardware devices implement a parsing buffer of a fixed size to process packets. The parsing buffer is expected to contain all the headers (often up to a transport layer header for filtering) that a device needs to examine. If the aggregate length of headers in a packet exceeds the size of the parsing buffer, a device will either discard the packet or defer processing to a software slow path. In any case, no indication of a problem is sent back to the sender.

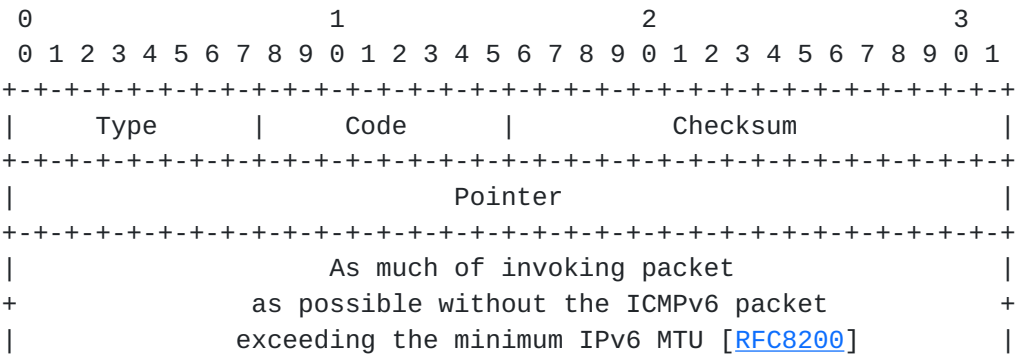
This document defines one code for ICMPv6 Destination Unreachable that is sent by a node that is unable to process the headers of a packet due to the aggregate size of the packet headers exceeding a processing limit. A source host that receives an ICMPv6 error can modify the headers used in subsequent packets to try to avoid further occurrences of packets being discarded or relegated to a slow path.

2 ICMPv6 errors for extension header limits

Four new codes are defined for the Parameter Problem type and applicability of one existing code is extended for ICMPv6 errors for extension header limits.

2.1 Format

The format of the ICMPv6 Parameter Problem message [RFC4443] for an extension header limit exceeded error is:



IPv6 Fields:

Destination Address
Copied from the Source Address field of the invoking packet.

ICMPv6 Fields:

Type
4 (Parameter Problem type)

Code (pertinent to this specification)
1 - Unrecognized Next Header type encountered
TBA - Extension header too big
TBA - Extension header chain too long
TBA - Too many options in extension header
TBA - Option too big

Pointer
Identifies the octet offset within the invoking packet where the problem occurred.

The pointer will point beyond the end of the ICMPv6 packet if the field having a problem is beyond what can fit in the maximum size of an ICMPv6 error message.

2.2 Unrecognized Next Header type encountered (code 1)

[RFC8200] specifies that a destination host should send an "unrecognized next header type" when a Next Header value is unrecognized in a packet. This document extends this to allow intermediate nodes to send this same error for a packet that is discarded because the node does not recognize a Next Header type.

This code SHOULD be sent by an intermediate node that discards a packet because it encounters a Next Header type that is unknown in its examination. The ICMPv6 Pointer field is set to the offset of the unrecognized next header value within the original packet.

Note that when the original sender receives the ICMPv6 error it can differentiate between the message being sent by a destination host, per [RFC4443], and an error sent by an intermediate host based on matching the source address of the ICMPv6 packet and the destination address of the packet in the ICMPv6 data.

2.3 Extension header too big (code TBA)

An ICMPv6 Parameter Problem with code for "extension header too big" SHOULD be sent when a node discards a packet because the size of an

extension header exceeds its processing limit. The ICMPv6 Pointer field is set to the offset of the first octet in the extension header that exceeds the limit.

2.4 Extension header chain too long (code TBA)

An ICMPv6 Parameter Problem with code for "extension header chain too long" SHOULD be sent when a node discards a packet with an extension header chain that exceeds its processing limits.

There are two different limits that might be applied: a limit on the total size in octets of the header chain, and a limit on the number of extension headers in the chain. This error code is used in both cases. In the case that the size limit is exceeded, the ICMPv6 Pointer is set to first octet beyond the limit. In the case that the number of extension headers is exceeded, the ICMPv6 Pointer is set to the offset of first octet of the first extension header that is beyond the limit.

2.5 Too many options in extension header (code TBA)

An ICMPv6 Parameter Problem with code for "too many options in extension header" SHOULD be sent when a node discards a packet with an extension header that has a number of options that exceed the processing limits of the node. This code is applicable for Destination options and Hop-by-Hop options. The ICMPv6 Pointer field is set to the first octet of the first option that exceeds the limit.

2.6 Option too big (code TBA)

An ICMPv6 Parameter Problem with code for "option too big" is sent in two different cases: when the length of an individual Hop-by-Hop or Destination option exceeds a limit, or when the length or number of consecutive Hop-by-Hop or Destination padding options exceeds a limit. In the case that the length of an option exceeds a processing limit, the ICMPv6 Pointer field is set to the offset of the first octet of the option that exceeds the limit. In the cases that the length or number of padding options exceeds a limit, the ICMPv6 Pointer field is set to the offset of first octet of the padding option that exceeds the limit.

Possible limits related to padding include:

* The number of consecutive PAD1 options in destination options or hop-by-hop options is limited to seven octets [RFC8504].

* The length of a PADN options in destination options or hop-by-hop options is limited seven octets [RFC8504].

* The aggregate length of a set of consecutive PAD1 or PADN options in destination options or hop-by-hop options is limited to seven octets.

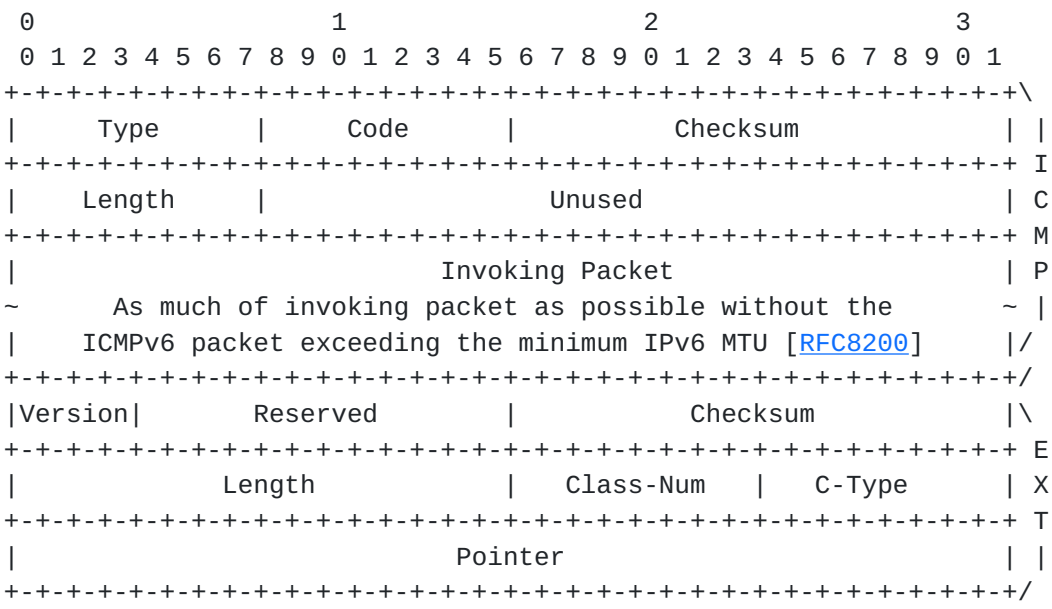
3 ICMPv6 error for aggregate header limits

One code is defined for Destination Unreachable type for aggregate header limits.

3.1 Format

The error for aggregate header limits employs a multi-part ICMPv6 message format as defined in [RFC4884]. An ICMP extension structure contains one ICMP extension object which contains a Pointer field.

The format of the ICMPv6 message for an aggregate header limit exceeded is:



IPv6 Fields:

Destination Address
Copied from the Source Address field of the invoking packet.

ICMPv6 Fields:

Type
1 - Destination Unreachable type

Code (pertinent to this specification)
TBA - Headers too long

Length
Length of the padded Invoking Packet measured in 64-bit words. The ICMP extension structure immediately follows the padded Invoking Packet.

Invoking Packet
Contains as much of invoking packet as possible without the ICMPv6 packet exceeding the minimum IPv6 MTU. The Invoking Packet MUST be zero padded to the nearest 64-bit boundary [RFC4884]. If the original invoking packet did not contain 128 octets, the Invoking Packet MUST be zero padded to 128 octets.

ICMP Extension Fields:

Version
2 - per [RFC4884]

Reserved
0

Checksum
The one's complement checksum of the ICMP extension [RFC4884]

Length
8 - length of the object header and Pointer field

Class-Num
TBA - Extended Information class

C-Type
TBA - Pointer sub-type

Pointer
Identifies the octet offset within the invoking packet where a limit was exceeded.

The pointer will point beyond the end of the Invoking Packet if the field exceeding the limit is beyond what can fit in the maximum size of an ICMPv6 error message with the ICMP extension.

3.2 Usage

An ICMPv6 Destination Unreachable error with code for "headers too long" SHOULD be sent when a node discards a packet because the aggregate length of headers in the packet exceeds the processing limits of the node. The Pointer in the extended ICMPv6 structure is set to the offset of the first octet that

exceeds the limit.

This error is sent in response to a node dropping a packet because the aggregate header chain exceeds the processing limits of a node. The aggregate header chain may be composed of protocol headers other than an IPv6 header and IPv6 extension headers. For instance, in the case of a node parsing a UDP encapsulation protocol, the encapsulating UDP header would be considered to be in the aggregate header chain.

As noted in section 4.1, the ICMPv6 Destination Unreachable error with code for "headers too long" has the lowest precedence of the ICMP errors discussed in this specification. If a packet contains an error corresponding to a Parameter Problem code then a node SHOULD send the Parameter Problem error instead of sending the ICMPv6 Destination Unreachable error with code for "headers too long".

4 Operation

Nodes that send or receive ICMPv6 errors due to header processing limits MUST comply with ICMPv6 processing as specified in [RFC4443].

4.1 Priority of reporting

More than one ICMPv6 error may be applicable to report for a packet. For instance, the number of extension headers in a packet might exceed a limit and the aggregate length of protocol headers might also exceed a limit. Only one ICMPv6 error SHOULD be sent for a packet, so a priority is defined to determine which error to report.

The RECOMMENDED reporting priority of ICMPv6 errors for processing limits is from highest to lowest priority:

1) Real error (existing codes)

2) "Unrecognized Next Header type" encountered by an intermediate node

3) "Extension header too big"

4) "Option too big" for length or number of consecutive padding options exceeding a limit

5) "Option too big" for the length of an option exceeding a limit

6) "Too many options in an extension header"

7) "Extension header chain too long" for number of extension headers exceeding a limit

8) "Extension header chain too long" for size of an extension header chain exceeding a limit

9) "Headers too long"

4.2 Host response

When a source host receives an ICMPv6 error for a processing limit being exceeded, it SHOULD verify the ICMPv6 error is valid and take appropriate action as suggested below.

The general validations for ICMP as described in [RFC4443] are applicable. The packet in the ICMP data SHOULD be validated to match the upper layer process or connection that generated the original packet. Other validation checks that are specific to the upper layers may be performed and are out of the scope of this specification.

The ICMPv6 error SHOULD be logged with sufficient detail for debugging packet loss. The details of the error, including the addresses and the offending extension header or data, should be retained. This, for instance, would be useful for debugging when a node is mis-configured and unexpectedly discarding packets, or when a new extension header is being deployed.

A host MAY modify its usage of protocol headers in subsequent packets to avoid repeated occurrences of the same error.

For ICMPv6 errors caused by extension header limits being exceeded:

* An error SHOULD be reported to an application if the application enabled extension headers for its traffic. In response, the application may terminate communications if extension headers are required, stop using extension headers in packets to the destination indicated by the ICMPv6 error, or attempt to modify its use of extension headers or headers to avoid further packet discards.

* A host system SHOULD take appropriate action if it is creating packets with extension headers on behalf of the application. If the offending extension header is not required for communication, the host may either stop sending it or otherwise modify its use in subsequent packets sent to the destination indicated in the ICMPv6 error.

5 Applicability and use cases

5.1 Nonconformant packet discard

The ICMP errors defined in this specification may be applicable to scenarios for which a node is dropping packets outside the auspices of any standard specification. For instance, an intermediate node might send a "Headers too long" code in the case that it drops a packet because it is unable to parse deep enough to extract transport layer information needed for packet filtering. Such behavior might be considered nonconformant (with respect to [RFC8200] for instance).

This specification does not advocate behaviors that might be considered nonconformant. However, packet discard does occur in real deployments and the intent of this specification is provide visibility as to why packets are being discarded. In the spirit that providing some reason is better than silent drop, this specification RECOMMENDS the sending of ICMP errors even in cases where a node might be discarding packets per a nonconformant behavior.

5.2 Reliability of ICMP

ICMP is fundamentally an unreliable protocol and in real deployment it may consistently fail over some paths. As with any other use of ICMP, it is assumed that the errors defined in this document are only best effort to be delivered. No protocol should be implemented that relies on reliable delivery of ICMP messages. If necessary, alternative or additional mechanisms may used to augment the processes used to to deduce the reason that packets are being discarded. Such alternative mechanisms are out of scope of this specification.

5.3 Processing limits

This section discusses the trends and motivations of processing limits that warrant ICMP errors.

5.3.1 Long headers and header chains

The trend towards longer and more complex headers and header chains needing to be processed by end nodes, as well as intermediate nodes, is driven by:

* Increasing prevalence of deep packet inspection in middleboxes. In particular, many intermediate nodes now parse network layer encapsulation protocols or transport layer protocols.

* Deployment of routing headers. For instance, [SRH] defines an

extension header format that includes a list of IPv6 addresses which may consume a considerable number of bytes.

* Development of In-situ OAM headers that allow a rich set of measurements to be gathered in the data path at the cost of additional header overhead which may be significant [IOAM].

* Other emerging use cases of Hop-by-Hop and Destination options.

5.3.2 At end hosts

End hosts may implement limits on processing extension headers as described in [RFC8504]. Host implementations are usually software stacks that typically don't have inherent processing limitations. Limits imposed by a software stack are more likely to be for denial of service mitigation or performance.

5.3.3 At intermediate nodes

Hardware devices that process packet headers may have limits as to how many headers or bytes of headers they can process. For instance, a middlebox hardware implementation might have a parsing buffer that contains some number of bytes of packet headers to process. Parsing buffers typically have a fixed size such as sixty-four, 128, or 256 bytes. In addition, hardware implementations (and some software implementations) often don't have loop constructs. Processing of a TLV list might be implemented as an unrolled loop so that the number of TLVs that can be processed is limited.

6 Security Considerations

The security considerations for ICMPv6 described in [RFC4443] are applicable. The ICMP errors described in this document MAY be filtered by firewalls in accordance with [RFC4890].

In some circumstances, the sending of ICMP errors might conceptually be exploited for denial of service attack or as a means to covertly deduce processing capabilities of nodes. As such, an implementation SHOULD allow configurable policy to withhold sending of the ICMP errors described in this specification in environments where security of ICMP errors is a concern.

7 IANA Considerations

7.1 Parameter Problem codes

IANA is requested to assign the following codes for ICMPv6 type 4 "Parameter Problem" [IANA-ICMPV6]:

* Extension header too big

* Extension header chain too long

* Too many options in extension header

* Option too big

7.2 Destination Unreachable codes

IANA is requested to assign the following code for ICMPv6 type 1 "Destination Unreachable" [IANA-ICMPV6]:

* Headers too long

7.3 ICMP Extension Object Classes and Class Sub-types

IANA is requested to assign the following Class value in the "ICMP Extension Object Classes and Class Sub-types" registry [IANA-ICMPEXT]:

* Extended information

IANA is requested to assign the following Sub-type within the aforementioned "Extended information" ICMP extension object class:

* Pointer

8 Acknowledgments

The author would like to thank Ron Bonica, Bob Hinden, Nick Hilliard, Michael Richardson, Mark Smith, Suresh Krishnan, and Ole Tran for their comments and suggestions that improved this document.

9 References

9.1 Normative References

[RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", RFC 4443, DOI 10.17487/RFC4443, March 2006, <http://www.rfc-editor.org/info/rfc4443>.

[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>.

[RFC7045] Carpenter, B. and S. Jiang, "Transmission and Processing of IPv6 Extension Headers", RFC 7045, DOI 10.17487/RFC7045, December 2013, <http://www.rfc-editor.org/info/rfc7045>.

[RFC4884] Bonica, R., Gan, D., Tappan, D., and C. Pignataro, "Extended ICMP to Support Multi-Part Messages", RFC 4884, DOI 10.17487/RFC4884, April 2007, <https://www.rfc-editor.org/info/rfc4884>.

[IANA-ICMPV6] "Internet Control Message Protocol version 6 (ICMPv6) Parameters", <https://www.iana.org/assignments/icmpv6-parameters/icmpv6-parameters.xhtml#icmpv6-parameters-codes-2>

[IANA-ICMPEXT] ICMP Extension Object Classes and Class Sub-types, <https://www.iana.org/assignments/icmp-parameters/icmp-parameters.xhtml#icmp-parameters-ext-classes>

9.2 Informative References

[RFC8504] Chown, T., Loughney, J., and T. Winters, "IPv6 Node Requirements", BCP 220, RFC 8504, DOI 10.17487/RFC8504, January 2019, <https://www.rfc-editor.org/info/rfc8504>.

[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>.

[SRH] Filsfils, C., Ed.. Dukes, D., Ed., Previdi, S., Leddy, J., Matsushima, S., Voyer, D., "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-21 (work in progress), August 2019. February

[IOAM] Bhandari, S., Brockners, F., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Kfir, A., Gafni, B., Lpaukhov, P., Spiegel, M., Krishnan, S., Asati, R., "In-situ OAM IPv6 Options", draft-ioametal-ippm-6man-ioam-ipv6-options-02, March 2018

Author's Address

Tom Herbert
Intel
Santa Clara, CA
USA

Email: tom@quantonium.net