Network Working Group
Internet-Draft
Expires: Febrary 29, 2008

I. van Beijnum
Consultant

August 29, 2007

IPv6 Extensions for Multi-MTU Subnets
draft-van-beijnum-multi-mtu-01

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on Febrary 28, 2008.

Copyright Notice

Copyright (C) The IETF Trust (2007).

Abstract

In the early days of the internet, many different link types with many different maximum packet sizes were in use. For point-to-point or point-to-multipoint links, there are still some other link types (PPP, ATM, Packet over SONET), but shared subnets are almost exclusively implemented as ethernets. Even though the relevant standards madate a 1500 octet maximum packet size for ethernet, more and more ethernet equipment is capable of handling packets bigger than 1500 octets. However, since this capability isn't standardized, it's seldom used today, despite the potential performance benefits of using larger

packets. This document specifies a mechanism to negotiate per-neighbor maximum packet sizes so that nodes on a shared subnet may use the maximum mutually supported packet size between them without being limited by nodes with smaller maximum sizes on the same subnet.

1 Introduction

Some protocols inherently generate small packets. Examples are VoIP, where it's necessary to send packets frequently before much data can be gathered to fill up the packet, and the DNS, where the queries are inherently small and the returned results also rarely fill up a full 1500-octet packet. However, most data that is transferred across the internet and private networks is at least several kilobytes in size (often much larger) and requires segmentation by TCP or another transport protocol. These types of data transfer can benefit from larger packets in several ways:

1. A higher data-to-header ratio makes for fewer overhead bytes

2. Fewer packets means fewer per-packet operations on the source and destination hosts

3. Fewer packets also means fewer per-packet operations in routers and middleboxes

4. TCP performance tends to increase with larger packet sizes

Even though today, the capability to use larger packets (often called jumbo frames) is present in a lot of ethernet hardware, this capability isn't used because IP assumes a common MTU size for all nodes connected to a link or subnet. In practice, this means that using a larger MTU requires manual configuration of the the non-standard MTU size on all hosts and routers and possibly on switches. Also, the MTU size for a subnet is limited to that of the least capable router, host or switch.

This document proposes to end this situation using several new options and messages:

1. An additional router advertisement MTU option to limit higher maximum packet sizes

2. A neighbor discovery option that allows nodes to inform their

neighbors of the maximum packet size they support

3. A neighbor discovery option for padding messages to make them suitable for probing a neighbor's MTU and link-layer MTU limitations

4. Padding for ARP messages to make them suitable for probing a neighbor's MTU and link-layer MTU limitations

2 Terminology

Local MTU:
The maximum packet size considered usable on an interface, based on the physical MTU, the MTU advertised by routers and administrative settings.

MTU:
Maximum Transmission Unit. This is the maximum IP packet size in octets supported on a link, towards a neighbor or towards a remote correspondent. In some cases, the term MRU (maximum receive unit) would be more appropriate, but for consistency, the term MTU is used throughout this document.

Neighbor MTU:
The maximum packet size that may be used towards a given on-link neighbor.

Node:
A host or router running IPv4 or IPv6.

Oversized packet:
A packet exceeding the size defined in the relevant IPv6-over-... or IP-over-... RFC.

Physical MTU:
The MTU reported by the driver for an interface when operating at a given link speed.

Probe:
An ARP or neighbor solicitation packet of a specific (oversized) size sent for the purpose of determining whether a neighbor can successfully receive packets of this size sent by the local node.

3 Disadvantages of larger packets

Although often desirable, the use of larger packets isn't universally advantageous for the following reasons:

1. Increased delay and jitter
2. Increased reliance on path MTU discovery
3. Increased packet loss through bit errors
4. Increased risk of undetected bit errors

3.1 Delay and jitter

An low-bandwidth links, the additional time it takes to transmit larger packets may lead to unacceptable delays. For instance, transmitting a 9000-octet packet takes 7.23 milliseconds at 10 Mbps, while transmitting a 1500-octet packet takes only 1.23 ms. Once transmission of a packet has started, additional traffic must wait for the transmission to finish, so a larger maximum packet size immediately leads to a higher worst-case head-of-line blocking delay, and as such, to a bigger difference between the best and worst cases (jitter). The increase in average delay depends on the number of packets that are buffered, the average packet size and the queuing strategy in use. Buffer sizes vary greatly, but assuming 40 buffers (not uncommon) leads to the following results:

Speed	500	1500	4500	9000	16384	65535
10 Mbps	17.22	49.21	145.22	289.22	525.50	2098.34
100 Mbps	1.72	4.92	14.52	28.92	52.55	209.83
1 Gbps	0.17	0.49	1.45	2.89	5.26	20.98
10 Gbps	0.02	0.05	0.15	0.29	0.52	2.01

In milliseconds and counting 38 additional octets of ethernet overhead.

If we assume that the delays involved with 1500-octet packets on 100 Mbps ethernet are acceptable for most, if not all, applications, then the conclusion must be that 9000-octet packets on 1 Gbps ethernet should also be acceptable. At 10 Gbps ethernet, much larger packet

sizes could be accommodated without adverse impact on delay-sensitive applications. Below 100 Mbps, larger packet sizes are probably not advisable.

3.2 Path MTU Discovery problems

PMTUD issues arise when routers can't fragment packets in transit because the DF bit is set or because the packet is IPv6, but the packet is too large to be forwarded over the next link, and the resulting "packet too big" ICMP messages from the router don't make it back to the sending host. This will typically happen when there is an MTU bottleneck somewhere in the middle of the path. If the MTU bottleneck is located at either end, the TCP MSS (maximum segment size) option makes sure that TCP packets conform to the limited MTU. PMTUD problems are of course possible with non-TCP protocols, but this is rare in practice.

Taking the delay and jitter issues to heart, maximum packet sizes should be larger for faster links. This means that in the majority of

cases, the MTU bottleneck will tend to be at one of the ends of a path, rather than somewhere in the middle.

A crucial difference between PMTUD problems that result from MTUs smaller than the standard 1500 octets and PMTUD problems that result from MTUs larger than the standard 1500 octets is that in the latter case, only a party that's actually using the non-standard MTU is affected. This puts potential problems and potential benefits in the same place so it's always possible to revert to a 1500-octet MTU if PMTUD problems can't be resolved otherwise.

Considering the above and the work that's going on in the IETF to resolve PMTUD issues as they exist today, means that increasing MTUs where desired doesn't involve undue risks.

3.3 Packet loss through bit errors

All transmission media are subject to bit errors. In many cases, a bit error leads to a CRC failure, after which the packet is lost. In other cases, packets are retransmitted a number of times, but if error conditions are severe, packets may still be lost because an error occurred at every try. Using larger packets means that the chance of a

packet being lost due to errors increases. And when a packet is lost, more data has to be retransmitted.

Both per-packet overhead and loss through errors reduce the amount of usable data transferred. The optimum tradeoff is reached when both types of loss are equal. If we make the simplifying assumption that the relationship between the bit error rate of a medium and the resulting number of lost packets is linear with packet size, the optimum packet size is computed as follows:

packet size = sqrt(overhead octets / bit error rate)

For IPv6 in ethernet framing, with 14 octets of ethernet header, 40 octets of IPv6 header, 20 octets of TCP header and 32 bits of ethernet CRC the total number of octets transmitted is 1538 while the useful data is 1440. (The preamble and inter frame gap are not relevant for error rate purposes.) 78 octets of overhead would result in a 1518-octet frame length for a bit error rate of 10^-5.3.

Note that the minimum BER for 1000BASE-T is 10^-10, which implies an optimum packet size of 312250 octets.

In practice, it's better to err on the side of smaller packets and lower packet loss to avoid triggering TCP congestion mechanisms. However, it's obvious that current maximum packet sizes are far below the optimum size with respect to optimum throughput.

3.4 Undetected bit errors

Nearly all link layers employ some kind of checksum to detect bit errors so that packets with errors can be discarded. In the case of ethernet, this is a frame check sequence in the form of a 32-bit CRC. The error detecting properties of the CRC are twofold: the minimum Hamming distance and the statistical unlikeliness of two packets resulting in the same CRC. Depending on the size of the packet, there is a minimum Hamming distance between two possible packets that result in the same CRC. For ethernet packets between 376 and 11454 octets long (including), the Hamming distance is 3 [CRC]. So all packets where transmission errors resulted in one or two flipped bits are detected. If 3 or more bits are flipped, most errors are caught because only in very few cases, the new bit pattern results in the same CRC as the old bit pattern. In theory, the chance of two

packets having the same CRC-32 is 1 in 2^32, but this assumes the CRC is as strong as it possibly could be.

It has been suggested that increasing packet lengths reduce the effectiveness of the CRC-32. For the statistical aspect of the CRC, this isn't true. Again, assuming a linear relationship between the likelihood of bit errors in a packet and the bit error rate, doubling the packet size means doubling the chance of a given number of bit errors in the packet. In turn, this doubles the chance of a packet with bit errors going undetected by the CRC. However, because the packet is twice as long, only half the number of packets is required to transmit any given amount of data. These aspects cancel each other out so the probability of a undetected errors occurring in any given data transfer doesn't vary with packet size when only considering the statistical properties of the CRC.

Obviously, choosing a packet size that leads to a reduced Hamming distance greatly increases the risk of undetected bit errors. However, even choosing a larger packet size with a Hamming distance of 3 leads to a reduction in error detection strength. The likelihood of a packet having enough bit errors to satisfy a given Hamming distance (packet error rate) and then generate the same CRC is:

PER = (packet length in bits * BER) ^ H / 2^32

The likelihood of a packet with enough bit errors to meet the Hamming distance and then generate an identical CRC in a transmission of a certain number of bits is:

TER = transmission length / packet length * PER

In other words:

TER = transmission length / (packet length ^ (H - 1) * BER ^ H) / 2^32

(Hence the irrelevance of the packet length for a Hamming distance of 1.)

For a 400 GB (approximately one hour) transmission over 1000BASE-T with a BER of 10^-10 and a 1518-octet ethernet frame length this means:

TER = 3.44*10^12 * 12144 ^ 2 * 10^-10 ^ 3 / 2^32 = 1.18*10^-19

For 11454-octet packets this becomes:

TER = 3.44*10^12 * 91632 ^ 2 * 10^-10 ^ 3 / 2^32 = 6.73*10^-18

Please note that this is 14 orders of magnitude better than the naive assumption of a Hamming distance of 1 suggests for standard 1518-octet ethernet frames:

TER = 3.44*10^12 * 12144 ^ 0 * 10^-10 ^ 1 / 2^32 = 9.73*10^-4

So the strength of the CRC, assuming a Hamming distance of 3, goes down with the square of the factor by which the packet length is increased. And it goes down with the third power of any increase of the bit error rate. However, this discussion is largely academic because of the assumption that bit errors happen in isolation. For instance, 1000BASE-T transmits two bits per symbol over four wire pairs, so bit errors are much more likely to (at least) happen in pairs rather than isolated.

Also, it should be possible to implement stronger frame check sequences for newer versions of ethernet. Unlike the packet length, the FCS is something switches can change when interconnecting different types of ethernet without harming interoperability.

3.5 Conclusion

Larger packets aren't universally desireable. The factors that factor into the decision to use larger packets include:

- A link's bit error rate
- The number of bits per symbol on a link and hence the likelihood of multiple bit errors in a single packet
- The strength of the Frame Check Sequence
- The link speed
- The number of buffers
- Queuing strategy

This means that choosing a good maximum packet size is, initially at least, the responsibility of hardware vendors. On top of that, robust

mechanisms must be available to operators to further limit maximum

packet sizes where appropriate.

4 The protocol mechanisms

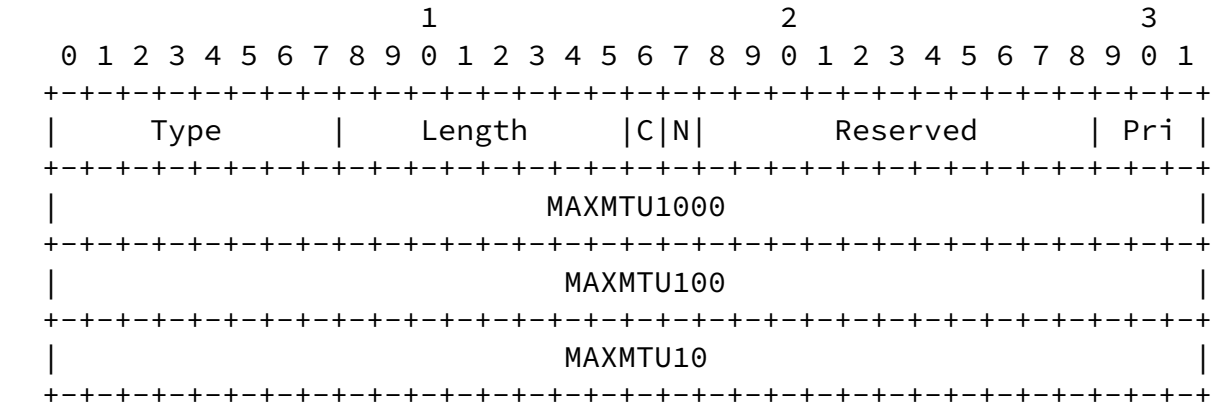
The basic idea is that nodes are free to negotiate larger MTUs with neighbors on a subnet. However, to avoid problems, probe packets are sent first before larger packets are used for actual traffic, and routers may inform hosts of MTU limitations that should be observed for three common ranges of link speeds. The rationale for having different MTU limitations for different link speeds is that it's common for devices operating at the link layer to support larger MTUs if they support and/or operate at higher link speeds. E.g., a LAN could consist of a gigabit ethernet switch with jumbo frame capabilities connected to a 10/100 Mbps ethernet switch which doesn't support jumbo frames. By limiting the use of oversized packets to nodes operating at 1000 Mbps, the 10/100 Mbps switch isn't exposed to oversized packets which would result in error conditions and use up unnecessary bandwidth. Additionally, it may be desireable to limit packet sizes at lower speeds even if a large MTU is supported for QoS purposes.

Additionally, routers send out two flags. One is intended to signal hosts to be conservative in the number of probes they transmit to avoid triggering undesired behavior by link-layer devices seeing a large number of out-of-spec packets. The other flag suppresses probing for compatibility with the existing practice where all nodes on a subnet are administratively configured with a non-standard MTU.

Probing consists of sending a large neighbor discovery or ARP packet to a neighbor. If the neighbor sends a reply, it managed to successfully receive the probe so the per-neighbor MTU for this neighbor can be set to the size of the probe packet and data packets of that size can now be sent.

4.1 The multi-MTU router advertisement option

Routers use this option to inform hosts on connected subnets about the maximum allowed MTU for three ranges of link speeds.



Type: TBD

Length:
1 or 2. A length of more than 2 indicates a future extension with additional fields and MUST NOT be treated as an error, the additional fields MUST be ignored.

C:
"Conservative" flag: when set, nodes should reduce the number of large packets sent by using a conservative timings and probing algorithms, if possible avoiding sending more than one unsuccessful probe per 60 seconds. When the flag is cleared, nodes may send send several oversized packets per second when probing.

N:
"No probe" flag: when set to 0, hosts MUST probe before using oversized packets towards a neighbor. When set to 1, hosts MUST NOT send probes and use the relevant MAXMTU field as their MTU. If MAXMTU is larger than the physical MTU, an error is logged.

Reserved: 0 on transmission, ignored on reception.

Pri:
Priority. Values have the following meaning:

000: Vendor default
001: Local override of 000
010: Site default
011: Local override of 010
100: Subnet default
101: Local override of 100
110: Per-node setting
111: Local override of 110

Vendors may only use priority 000 in default configurations. Site-wide administrative settings may only use 000 and 010.

Subnet-specific administrative settings may use 000, 010 or 110, but not 001, 011, 101 or 111.

MAXMTU1000:
The maximum packets size allowed on a link operating at a speed of 300 Mbps or more. Packets larger than this value SHOULD NOT be sent over the link in question. The MAXMTU1000 MUST be at least the MTU size specified in the relevant IPv6-over-... RFC. A value of 0 means that the MTU size is undefined and no maximum size is enforced for this link speed.

MAXMTU100:
The maximum packets size allowed on a link operating at a speed of 30 to 299 Mbps and links operating at an unknown speed if that speed can be 30 Mbps or higher. Packets larger than this value SHOULD NOT be sent over the link in question. The MAXMTU100 MUST be at least the MTU size specified in the relevant IPv6-over-... RFC. A value of 0 means that the MTU size is undefined and no maximum size is enforced for this link speed.

MAXMTU10:
The maximum packets size allowed on a link operating at a speed of less than 30 Mbps. Packets larger than this value SHOULD NOT be sent over the link in question. The MAXMTU10 MUST be at least the MTU size specified in the relevant IPv6-over-... RFC. A value of 0 means that the MTU size is undefined and no maximum size is enforced for this link speed.

When MAXMTU1000, MAXMTU100 and MAXMTU10 all contain the same value, it is allowed to omit MAXMTU100 and MAXMTU10 so the option has a length of 1 (8 octets) rather than 2 (16 octets). The receiver of the option should treat the shorter option the same as a full lenth option where the three MAXMTU fields all contain the value from MAXMTU1000.

Hosts are expected to recover the multi-MTU options from the router advertisements of at least the router they select as a default router, but it's encouraged (not required) to recover options from multiple

routers. The same option, or data constituting the same information, may be learned from other sources, such as local configuration and/or DHCPv6. Hosts SHOULD use the MAXMTU value relevant for the link speed the interface is currently operating at from the option or equivalent information with the largest priority value. If the relevant MAXMTU field is unspecified (zero) in the option or information with the highest priority, the field from the option or information with the next highest priority is considered, and so on. If no information is available because no option or equivalent is available, or the relevant MAXMTU field never has a

non-zero value, the host SHOULD use its physical MTU as the MAXMTU.

When a node's interface speed changes, it MAY reinitiate negotiation of per-neighbor MTUs, but it SHOULD remain prepared to receive packets of the maximum size indicated to neighbors previously.

Devices not acting as IPv6 routers that need to inform hosts on the local subnet of MTU limitations MAY send out a router advertisement with a Router Lifetime of 0 [RFC2461] and the pertinent information in a multi-MTU option.

4.2 Changes to the RA MTU option semantics

Hosts are currently supposed to ignore an MTU of more than 1500 in the MTU option in router advertisements on ethernet links [RFC2464]. This makes it impossible to use an MTU larger than 1500 octets for multicast packets. In order to lift this limitation, routers and hosts that implement multi-MTU subnets may advertise and accept, respectively, an MTU option with an MTU larger than 1500. Hosts should use the minimum of the MAXMTU for their link speed and the MTU in the RA MTU option for the transmission of multicast packets.

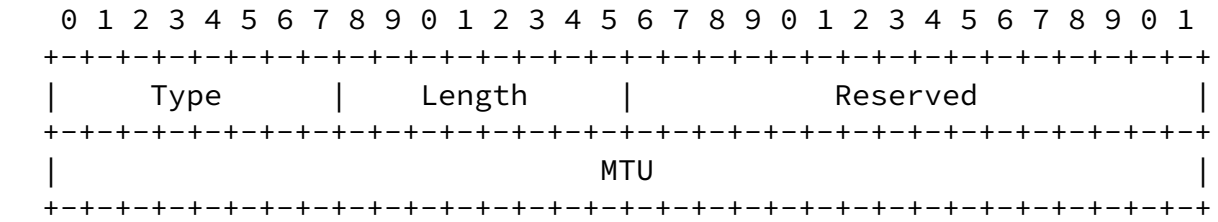
Note that advertising an MTU option larger than 1500 can only work on subnets where all the hosts implement multi-MTU subnets.

4.3 The IPv6 neighbor discovery MTU and padding options

A node that implements the multi-MTU subnet capability SHOULD include an MTU option in both neighbor solicitation and neighbor

advertisement messages [RFC2461]. A node MAY omit the option if the use of a larger MTU isn't desired at that time or if the MTU it would advertise is equal to or lower than the MTU that would otherwise be used. However, there is no requirement to omit the option depending on the value of the different MTU variables as the receiver must implement the logic required to determine which MTU to use anyway.

The format of the neighbor discovery MTU option is as follows:



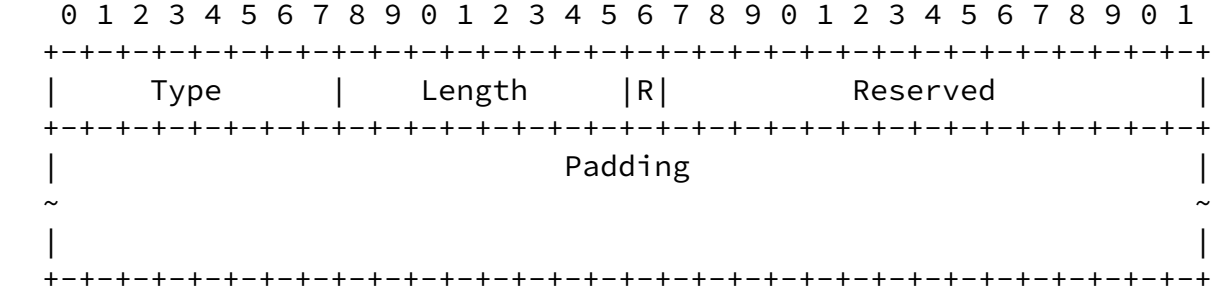
Type: TBD

Length: 1

Reserved: set to 0 on transmission, ignored on reception.

MTU:
The maximum packet size in octets that the node is prepared to receive. The minimum valid value is 1280.

The format of the neighbor discovery MTU option is as follows:



Type: TBD

Length: see below.

R: reply flag.

Reserved: set to 0 on transmission, ignored on reception.

Padding: 0 or more all-zero octects.

The MTU option is included in all neighbor advertisement and neighbor solicitation messages.

Reception of a neighbor solicitation or a neighbor advertisement triggers for a neighbor for which no per-neighbor MTU is known triggers, in addition to the normal response if it's a neighbor solicitation, the sending of an neighbor solicitation message wih the MTU and padding options in it. The size of this message is may vary between the IPv6-over-... size + 1 for the link and the minimum of the relevant MAXMTU, the physical MTU and the neighbor's MTU as advertised in the MTU option of the packet received. See below for considerations about the packet sizes to choose. The padding option is used to bring the neighbor solicitation message to this size. The padding option MUST be the last option in the packet.

There are two possible ways to determine the value of the length field:

1. Set it to 0. As the "length" field in options has a granularity of 8 octets and the behavior of nodes when they receive a neighbor solicitation packet which has a total length that doesn't match the length of the packet contents, an option length of 0 is used to make sure that hosts that don't understand the padding option will silently discard the packet.

2. If the intended packet length allows a valid value for the length field, the length field MAY be set to that value. The node MAY reduce the size of the intended packet to accommodate the requirement that the size field is a multiple of 8 octets. I.e., if the intended packet size is 4470 octets with 40 and 24 octets for the IPv4 and neighbor solicitation headers, respectively, the padding option would have to be 4406 octets long, which can't be expressed in the length field. The node may choose to use a packet size of 4464 instead, which results in a length field value of 550.

A neighbor solicitation message with the padding option is always sent in addition to a regular neighbor solicitation message, rather than in place of one.

When a node receives a neighbor solicitation message with the padding option, it stops evaluating options when it reaches the padding option and returns a regular neighbor advertisement message, which includes the MTU option with the R flag set to 1. Whenever the neighbor advertisement is not the result of receiving a neighbor solicitation with a padding option, the R flag is set to 0.

When a node receives a neighbor advertisement message, it must determine whether the message is in reaction to a locally sent neighbor solicitation with the padding option or not. If the MTU option is included in the message received, an R flag of 1 indicates that it is indeed a reply. In the absense of the MTU option the node must use heuristics relating to the timing of the messages it sent with and without the option, and the reception of the current message. If the message was a reply, the node sets the neighbor MTU to the size of the neighbor solicitation message that was replied to.

If no reply is received after some time, either the neighbor is incapable of receiving packets of the size that was used, or a device operating at the link layer was incapable for forwarding the frame. (Incidental packet loss is also a possibility.) In order to determine a workable MTU even in the presence of unknown limitations, a node may repeat sending a solicitation with the padding option. However, since presumably, some equipment may react badly to a large number of out-of-spec packets, it's important that

nodes adjust their behavior in the presence of the C (conservative) flag in router advertisements.

The above allows for two strategies in determining a neighbor's MTU: the node can depend on the presence of these mechanisms described in this document, including setting the padding option length field to 0, or it can try to interoperate with nodes that do have the capability of using larger packet sizes, but don't implement any of the mechanisms described. In that case, the padding option must conform to [RFC2461] and care must be taken to avoid overly aggressive probing of nodes that do not support larger

packets.

Nodes MUST support reception of both types of probes, but MAY be limited to generating only one type.

4.4 IPv4 ethernet jumbo ARP message

Due to lack of neighbor discovery, with IPv4, it's necessary to use ARP to probe for non-standard MTU capabilities. This is done by simply probing with an ARP packet padded to the desired size. If a reply comes back, the neighbor supports the probed MTU size.

4.5 Probe considerations

In cases where the neigbor's MTU was advertised in an MTU option, it makes sense to try with this size. If that probe fails or the neighbor's MTU is unknown, the best choice for a probe size would be the smallest possible non-standard MTU. This could be the IPv6-over-... RFC's MTU size + 1, or a slightly larger value that represents the first larger size that is actually useful, such as 1508 or 1520 for ethernet. Failure at this size wastes relatively little bandwidth and indicates that further probes are unnecessary. If this probe is successful, further choices for the probe size may be common MTU sizes such as 1508, 1530, 1536, 1546, 1998, 2000, 2018, 4464, 4470, 8092, 8192, 9000, 9176, 9180, 9216, 17976, 64000 and 65280 octets.

There is no requirement that a node tries a number of probes of different sizes; only that before oversized packets are sent, a reply for a probe of that size or larger MUST have been received from the neighbor in question, unless the N flag is set to 1. A simple strategy that would be appropriate when the C flag is set to 1, but may also be used otherwise, would be to initially send just one probe sized at the local MTU value, and if unsuccessful, only send a second probe when a probe from the neighbor is received. The second probe is made the same size as the neighbor's probe.

Probes MUST be sent as unicast.

4.6 Neighbor MTU garbage collection

The MTU size for a neighbor is garbage collected along with a

neighbor's link address in accordance with regular ARP and neighbor discovery timeouts. Additionally, a neighbor's MTU size is reset to unknown after dead neighbor detection declares a neighbor "dead".

5 References

5.1 Normative References

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

[RFC2461] Narten, T., Nordmark, E., and W. Simpson, "Neighbor Discovery for IP Version 6 (IPv6)", RFC 2461, December 1998.

[RFC2462] Thomson, S. and T. Narten, "IPv6 Stateless Address Autoconfiguration", RFC 2462, December 1998.

5.2 Informative References

[CRC] Jain, R., ""Error Characteristics of Fiber Distributed Data Interface (FDDI)", IEEE Transactions on Communications, August 1990.

6 Document and Author Information

This document expires February, 2008. The latest version will always be available at http://www.muada.com/drafts/. Please direct questions and comments to the ipv6 or int area mailinglists or directly to the author:

Iljitsch van Beijnum

Email: iljitsch@muada.com

Full Copyright Statement

Copyright (C) The IETF Trust (2007).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS

OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY, THE IETF TRUST AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).