Network Working Group I. van Beijnum
Internet-Draft IMDEA Networks
Intended status: Experimental February 24, 2008
Expires: August 27, 2008

Extensions for Multi-MTU Subnets
draft-van-beijnum-multi-mtu-02

Status of this Memo

Copyright Notice

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 now almost exclusively implemented as ethernets. Even though the relevant standards mandate a 1500 byte maximum packet size for ethernet, more and more ethernet equipment is capable of handling packets bigger than 1500 bytes. 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 for advertising a non-standard maximum packet size on a subnet. It also specifies optional mechanisms 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.

Table of Contents

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-byte 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 increases 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 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.

In the future, when hosts support [RFC4821] in all relevant transport protocols, it will be possible to simply ignore MTU limitations by sending at the maximum locally supported size and determining the maximum packet size towards a correspondent from acknowledgements that come back for packets of different sizes. However, [RFC4821] must be implemented in every transport protocol, and there is a significant probability for failures if hosts implementing [RFC4821] interact with hosts that don't implement this mechanism but do use a larger than standard MTU.

This document provides for a set of mechanisms that allow the use of larger packets between nodes that support them which interacts well with both manually configured non-standard MTUs and expected future [RFC4821] operation with larger MTUs. This is done using several new options and messages:

1. An additional router advertisement Multi-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

Only support of the Multi-MTU option is required to conform to to this specification, the neighbor discovery options and jumbo ARP are optional.

2. Terminology

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

MTU: Maximum Transmission Unit. This is the maximum IP packet size in bytes 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.

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

Node: A host or router running IPv4 and/or IPv6.

Oversized packet: A packet exceeding the Standard MTU size.

PhysicalMTU: 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.

SafeMTU: Maximum packet size that is supported by all nodes an all link layer devices on a link.

StandardMTU: For IPv4: the MTU for a link type defined in the relevant IP-over-... RFC. For IPv6: the minimum of the MTU for a link type defined in the relevant IPv6-over-... RFC and the value of the MTU option in router advertisements.

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-byte packet takes 7.23 milliseconds at 10 Mbps, while transmitting a 1500-byte 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 thus, 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 between implementations, from only a few buffers in some switches and on low-speed interfaces on routers, to hundreds of megabytes of buffer space on 10 Gbps interfaces on some routers.

If we assume that the delays involved with 1500-byte packets on 100 Mbps ethernet are acceptable for most, if not all, applications, then the conclusion must be that 15000-byte packets on 1 Gbps ethernet should also be acceptable, as the delay is the same. At 10 Gbps ethernet, much larger packet sizes could be accommodated without adverse impact on delay-sensitive applications. At 100 Mbps, and certainly below that, 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. If there is a PMTUD black hole, 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 smallest MTU in the path. PMTUD problems are of course possible with non-TCP protocols, but this is rare in practice because non-TCP protocols are generally not capable of adjusting their packet size on the fly and therefore use more conservative packet sizes which won't trigger PMTUD issues.

Taking the delay and jitter issues to heart, maximum packet sizes should be larger for faster links and smaller for slower 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, as in today's internet, core of the network is quite fast, while users usually connect at lower speeds.

A crucial difference between PMTUD problems that result from MTUs smaller than the standard 1500 bytes and PMTUD problems that result from MTUs larger than the standard 1500 bytes is that in the latter case, only a party that's actually using the non-standard MTU is affected. This puts potential problems, the potential benefits and the ability to solve any resulting problems in the same place so it's always possible to revert to a 1500-byte 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 for reasonable bit error rates, the optimum packet size is computed as follows:

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

According to this, the optimum packet size is one or more orders of magnitude larger than what's commonly used today. For instance, the minimum BER for 1000BASE-T is 10^-10, which implies an optimum packet size of 312250 bytes with ethernet framing and IP overhead.

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. Assuming a strong frame check sequence algorithm, this suggests that there is a 1 in 2^32 chance that a packet with one or more bit errors in it has the same CRC as the original packet, so the bit errors go undetected and data is corrupted. However, according to [CRC] the CRC-32 that's used for FDDI and ethernet has the property that packets between 376 and 11454 bytes long (including) have a Hamming distance of 3. (Smaller packets have a larger Hamming distance, larger packets a smaller Hamming distance.) As a result, all errors where only a single bit is flipped or two bits are flipped, will be detected, because they can't result in the same CRC as the original packet. The probability of a packet having undetected bit errors can be approximated as follows for a 32-bit CRC:

PER = (PL * BER) ^ H / 2^32

Where PER is the packet error rate, BER is the bit error rate, PL is the packet length in bits and H is the Hamming distance. Another consideration is the impact of packet length on a multi-packet transmission of a given size. This would be:

TER = transmission length / PL * PER

So

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

Where TER is the transmission error rate.

In the case of the ethernet FCS and a Hamming distance of 3 for a large range of packet sizes, this means that the risk of undetected

errors goes up with the square of the packet length, but goes down with the third power of the bit error rate. This suggest that for a given acceptable risk of undetected errors, a maximum packet size can be calculated from the expected bit error rate. It also suggests that given the low BER rates mandated for gigabit ethernet, packet sizes of up to 11454 bytes should be acceptable.

Additionally, unlike properties such as the packet length, the frame check sequence can be made dependent on the physical media, so it should be possible to define a stronger FCS in future ethernet standards, or to negotiate a stronger FCS between two stations on a point-to-point ethernet link (i.e., a host and a switch or a router and a switch).

3.5. IEEE 802.3 compatibility

According to the IEEE 802.3 standard, the field following the ethernet addresses is a length field. However, [RFC0894] uses this field as a type field. Ambiguity is largely avoided by numbering type codes above 2048. The mechanisms described in this memo only apply to the standard [RFC0894] and [RFC2464] encapsulation of IPv4 and IPv6 in ethernet, not to possible encapsulations of IPv4 or IPv6 in IEEE 802.3/IEEE 802.2 frames, so there is no change to the current use of the ethernet length/type field.

3.6. Conclusion

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

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

This means that choosing a good maximum packet size is, initially at least, the responsibility of hardware builders. On top of that, robust mechanisms MUST be available to operators to further limit maximum packet sizes where appropriate.

4. The protocol mechanisms

The new Multi-MTU router advertisement option lets IPv6 routers (and, if desired, devices that aren't IPv6 routers) inform hosts of the maximum packet sizes they should use, based on the link bandwidth of the host and whether the host supports probing for support of oversized packets.

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.

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     | Pri |         Reserved        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             MAXMTU                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            SLOWMTU                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            SAFEMTU                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type: TBD

Length: 1

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. Per-node configuration may use all values.

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

MAXMTU: The absolute maximum packets size allowed on a link. Packets larger than this size MUST NOT be sent.

SLOWMTU: The maximum packet size nodes operating at a link speed below 600 Mbps (Mbps = 1000000 bps) may use.

SAFEMTU: The maximum packet size supported by all nodes on a link, packets of this size can be sent without probing.

4.2. General operation

Hosts MUST 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.

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.

Routers and other systems generating router advertisements with a Multi-MTU option SHOULD NOT advertise a MAXMTU, SLOWMTU or SAFEMTU lower than the MTU defined in the relevant IP-over-... or IPv6-over-... RFC.

DISCUSSION: Is it appropriate that IPv4 and IPv6 use the same MTU?

4.3. Determining the InterfaceMTU

If the node supports probing and there is positive knowledge that the interface is currently operating at is at least 600 Mbps, the InterfaceMTU is set as follows:

InterfaceMTU = max(StandardMTU, min(MAXMTU, PhysicalMTU))

If the node supports probing and the interface is operating at a speed below 600 Mbps, or the interface speed is unknown, the InterfaceMTU is set as follows:

InterfaceMTU = max(StandardMTU, min(SLOWMTU, PhysicalMTU))

If the node doesn't support probing and there is positive knowledge that the interface is currently operating at is at least 600 Mbps, the InterfaceMTU is set as follows:

InterfaceMTU = max(StandardMTU, min(MAXMTU, SAFEMTU, PhysicalMTU))

If none of the above rules apply, the InterfaceMTU is set as follows:

InterfaceMTU = max(StandardMTU, min(SLOWMTU, SAFEMTU, PhysicalMTU))

If InterfaceMTU is smaller than SAFEMTU, an error SHOULD be logged but operation SHOULD continue.

4.4. Changes to the RA MTU option semantics

If in addition to a Multi-MTU option, there is also an MTU option in a router advertisement, hosts MUST ignore the MTU option and use the value of the SAFEMTU field in the Multi-MTU option as the default MTU size on the interface. However, it may be necessary to incorporate special case logic to allow for the use of larger packets than what the interface-wide MTU value that is set accordingly suggest. For instance, if a node supports explicit probing as outlined below, or [RFC4821] probing for some transport protocols, the transport protocols in question may need to be aware of the possibility of using packets larger than the SAFEMTU. For example TCP should probably advertise a maximum segment size based on the InterfaceMTU rather than on the SAFEMTU in the MSS option.

4.5. The IPv6 neighbor discovery MTU option

In order to be able to use the largest packet sizes under the widest range of circumstances, nodes SHOULD include a new MTU option in both neighbor solicitation and neighbor advertisement messages [RFC2461].

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

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     |R|T|  Transport flags  |  Res  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              MTU                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type: TBD

Length: 1

R: Reply flag. Set to 1 when the neighbor discovery packet is sent in reply to a neighbor discovery packet containing a padding option, otherwise set to 0.

T: TCP-MSS-override flag. If set to 1, the MTU field MAY overwrite the maximum segment size that was advertised earlier, in the TCP MSS option. (Note that the MSS option advertises a value that doesn't include IP overhead; the MTU field is the size of an entire IP packet, including the IP header.) If set to 0, the TCP MSS option MUST be honored even if it's smaller than the NeighborMTU.

Transport flags: Reserved for use with other transport protocols in the same way as the T flag. Set to 0 on transmission, ignored when receiving.

Res: Set to 0 on transmission, ignored on reception.

MTU: If the R flag is 0: the maximum packet size in bytes that the node would like to receive. The minimum valid value is 1280. However, the node MUST be prepared to receive packets up to the SAFEMTU size. If the R flag is 1: the minimum of the maximum packet size that the node would like to receive (as with R=0) and the size of the packet that this packet is a reply to.

4.6. The IPv6 neighbor discovery padding option

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

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     |           Reserved            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Padding                            |
~                                                               ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type: TBD

Length: see below.

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

Padding: 0 or more all-zero octets.

4.7. Use of the MTU and padding options

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

Reception of a neighbor solicitation or a neighbor advertisement 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 with the MTU and padding options in it. The size of this message is may vary between the IPv6 StandardMTU size + 1 for the link and the minimum of the local 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 in the padding option:

1. Set it to 0. Since the option is in fact larger than 0, this means that nodes that don't implement the option will silently discard the packet. Setting the length to 0 makes it possible to have packets with the padding option that aren't a multiple of 8 bytes long.

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 bytes. I.e., if the intended packet size is 4470 bytes with 40 and 24 bytes for the IPv4 and neighbor solicitation headers, respectively, the padding option would have to be 4406 bytes 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. This of course means that subsequent data packets MUST be no larger than 4464 bytes.

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. If the message was a reply, the node sets the NeighborMTU to the size of the MTU field in the received neighbor discovery packet.

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 limit the number of oversized packets to destinations that aren't yet known to be capable of receiving them. An upper limit would be to allow only 5 unacknowledged oversized packets per 300 second period.

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

4.8. 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.

MAXMTU, SLOWMTU and SAFEMTU parameters advertised by IPv6 routers MUST also be taken into account when probing and generating oversized IPv4 packets.

4.9. Probe considerations

In cases where the neighbor'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 StandardMTU + 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 bytes. A useful heuristic would be to monitor all Multi-MTU options advertised, regardless of their priority, and use the values in those options as candidates for the largest supported packet size.

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 before packets larger than SAFEMTU are sent. A simple strategy that would be to initially send just one probe sized at the InterfaceMTU size, 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.

4.10. 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. IANA considerations

IANA is requested to assign a router advertisement option number and two neighbor discovery options. In addition, IANA is requested to start a registry for the transport flags. There are 10 flags, numbered 18 to 27. Each flag may be assigned to a transport protocol that communicates a maximum segment size in-band. See the discussion of the T flag in section Section 4.5.

6. Security considerations

Generating false router advertisements and neighbor discovery packets with large MTUs may lead to a denial-of-serve condition, just like the advertisement of other false link parameters.

7. References

7.1. Normative References

[RFC0894] Hornig, C., "Standard for the transmission of IP datagrams over Ethernet networks", STD 41, RFC 894, April 1984.

[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.

[RFC2464] Crawford, M., "Transmission of IPv6 Packets over Ethernet Networks", RFC 2464, December 1998.

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

7.2. Informative References

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

Appendix A. Document and discussion information

The latest version of this document will always be available at http://www.muada.com/drafts/. Please direct questions and comments to the int-area mailinglist or directly to the author.

Author's Address

Iljitsch van Beijnum
IMDEA Networks
Avda. del Mar Mediterraneo, 22
Leganes, Madrid 28918
Spain

Email: iljitsch@muada.com

Full Copyright Statement

Intellectual Property

Acknowledgment

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