Network Working Group F. Templin, Ed.
Internet-Draft Boeing Research & Technology
Intended status: Informational June 06, 2012
Expires: December 8, 2012

Generic Tunnel MTU Determination
draft-generic-v6ops-tunmtu-02.txt

Abstract

The Maximum Transmission Unit (MTU) for popular IP-within-IP tunnels is currently recommended to be set to 1500 (or less) minus the length of the encapsulation headers when static MTU determination is used. This requires the tunnel ingress to either fragment any IP packet larger than the MTU or drop the packet and return an ICMP Packet Too Big (PTB) message. Concerns for operational issues with Path MTU Discovery (PMTUD) point to the possibility of MTU-related black holes when a packet is dropped due to an MTU restriction. The current "Internet cell size" is therefore stuck at 1500 bytes (i.e., the minimum MTU configured by the vast majority of links in the Internet), but the desired end state is full accommodation of MTU diversity. This document therefore presents a method to boost the tunnel MTU to larger values.

Table of Contents

1. Introduction

The Maximum Transmission Unit (MTU) for popular IP-within-IP tunnels is currently recommended to be set to 1500 (or less) minus the length of the encapsulation headers when static MTU determination is used. This requires the tunnel ingress to either fragment any IP packet larger than the MTU or drop the packet and return an ICMP Packet Too Big (PTB) message [RFC0791][RFC2460]. Concerns for operational issues with Path MTU Discovery (PMTUD) [RFC1191][RFC1981] point to the possibility of MTU-related black holes when a packet is dropped due to an MTU restriction. The current "Internet cell size" is therefore stuck at 1500 bytes (i.e., the minimum MTU configured by the vast majority of links in the Internet), but the desired end state is full accommodation of MTU diversity. This document therefore presents a method to boost the tunnel MTU to larger values.

Pushing the tunnel MTU to 1500 bytes or beyond is met with the challenge that the addition of encapsulation headers would cause an inner IP packet that is slightly less than 1500 bytes to appear as a 1501 byte or larger outer IP packet on the wire. This can result in the packet being either fragmented or dropped by a router that connects to a 1500 byte link. Using the approach outlined in this document, the tunnel ingress avoids this issue by performing IP fragmentation on the inner packet before encapsulating each fragment in outer headers. The approach is outlined in the following sections.

2. Problem Statement

When an IP tunnel configures a smaller MTU than 1500 bytes, packets that are small enough to traverse earlier links in the path toward the final destination may be dropped at the tunnel ingress with a PTB message returned to the original source. However, operational experience has shown that the PTB messages can be lost in the network due to filtering in which case the source does not receive notification of the loss. It is therefore highly desirable that the tunnel configure an MTU of at least 1500 bytes, even though encapsulation would cause the tunneled packet to be larger than 1500 bytes.

One possibility is to use IP fragmentation of the outer IP layer protocol so that inner packets up to 1500 bytes are delivered even if the tunnel encapsulation causes the outer packet to be larger than 1500 bytes. However, fragmentation has been shown to be dangerous at high data rates due to the Identification field wrapping while reassemblies are still active [RFC4963]. Also, if outer IP fragmentation were used the tunnel egress would need to reassemble

which can be an onerous burden when the egress is located on a router. The tunnel ingress further has no assurance that the egress can reassemble packets larger than 1500 bytes.

A second possibility is to enable PMTUD on the outer packet. However, the PTB messages that may result could either be lost on the return path to the tunnel ingress or may not contain enough information for translation into an inner packet PTB for delivery to the original source. Still another possibility is for the tunnel ingress to maintain state about MTU sizes for various tunnel egresses, but this becomes unwieldy when the number of egresses is large.

In short, PMTUD is a mess and new approaches are needed.

3. Tunnel MTU

Section 3.2 of [RFC4213] presents both static and dynamic MTU determination algorithms. Similar algorithms appear in other tunneling mechanisms. These algorithms have been shown to be problematic in many instances, as discussed in Section 2. This document therefore proposes a generic MTU determination method suitable for all tunnel types via the following algorithm:

```
   1. set "HLEN" to the length of the encapsulation headers.
   2. set the tunnel ingress MTU to "infinity", where "infinity"
      is defined as ((2^32 - 1) - HLEN) for tunnels over IPv6
      and ((2^16 - 1) - HLEN) for tunnels over IPv4.
   3. for IP packets to be admitted into the tunnel:
      a) if the packet is 1501 or more:
         - if the packet is an atomic packet (*) admit it
           into the tunnel if it is no larger than the MTU
           of the underlying interface; otherwise, drop the
           packet and return a PTB message.
         - if the packet is not an atomic packet, break it
           into N pieces (where each piece is a random length
           between 500-1000 bytes) and admit each piece into
           the tunnel.
      b) if the packet is between 1281 - 1500:
         - break the packet into 2 pieces (where each piece
           is a random length between 500-1000 bytes) and
           admit each piece into the tunnel.
      c) if the packet is 1280 or less:
         - admit the packet into the tunnel
  4. the IP destination gets to reassemble if necessary
```

(*) An "atomic packet" is an IPv6 packet that does not contain a fragment header, or an IPv4 packet with (DF=1 && MF=0 && Offset=0) [I-D.ietf-intarea-ipv4-id-update].

In the above algorithm, clause 3 a) requires that atomic packets not be subject to fragmentation within the tunnel. Instead, the tunnel ingress should process any PTB messages returned by the tunnel and translate them into a corresponding PTB message to return to the original source. In clauses 3 b) and 3 c), fragmentation within the tunnel must be permitted, however the fragment size chosen for inner fragmentation before encapsulation reduces the likelihood that tunnel fragmentation will occur following encapsulation.

4. Inner Packet Fragmentation and Identification

For non-atomic inner IP packets, clause 3 b) in the algorithm in Section 3 performs inner fragmentation using the Identification value already present in the packet. The tunnel ingress then admits each fragment into the tunnel unconditionally, since it is the original source (and not the tunnel) that asserts the uniqueness of the packet's Identification value. For atomic inner IP packets, clause 3 b) in the algorithm in Section 3 ignores the requirement that routers in the network must not fragment atomic packets. The rest of this section discusses considerations for fragmentation of atomic IP packets.

The tunnel ingress maintains a randomly-initialized and arithmetically-increasing Identification value as either a per-tunnel or per-destination variable. For IPv6 atomic packets, the use of inner fragmentation requires that the tunnel ingress insert an IPv6 fragment header on each fragment. For IPv4 atomic packets, the tunnel ingress must rewrite the value in the packet header Identification field. In both cases, we observe that the Identification field provides sufficient protection against accidental reassembly of fragments from different IP packets given careful operational considerations.

Specifically, the tunnel ingress must ensure that there will be no IP fragments alive in the system with duplicate Identification values. Since [RFC2460] specifies that the maximum time a node may retain an incomplete fragmented packet is 60 seconds, this means that the tunnel ingress must not allow the Identification values to be repeated within this timeframe. The tunnel ingress can therefore calculate a maximum data rate for admission of fragmented packets into the tunnel.

For IPv4, to avoid Identification value duplication the tunnel

ingress must admit no more than (2^16 / 60) = 1092 IPv4 packets requiring fragmentation into the tunnel per second. In the worst case, consider that each packet is 1281 bytes (i.e., 10248 bits) in length. The tunnel ingress can then calculate the maximum data rate as (1092 * 10248) = 11190816 bits/sec, or approximately 11 Mbps. It is therefore essential that the tunnel ingress set a rate limit to no more than 11 Mbps for those atomic IPv4 packets that will require fragmentation. This restriction can be relaxed if the tunnel ingress maintains a per-destination Identification value instead of a single Identification value for all destinations.

For IPv6, to avoid Identification value duplication the tunnel ingress must admit no more than (2^32 / 60) = 71582788 IPv6 packets requiring fragmentation into the tunnel per second. In the worst case, consider that each packet is 1281 bytes (i.e., 10248 bits) in length. The tunnel ingress can then calculate the maximum data rate as (71582788 * 10248) = 733580411424 bits/sec, or approximately 733 Gbps. It is therefore essential that the tunnel ingress set a rate limit to no more than 733 Gbps for those atomic IPv6 packets that will require fragmentation. This restriction can be relaxed if the tunnel ingress maintains a per-destination Identification value instead of a single Identification value for all destinations.

Note that a possible conflict exists when a source host emits both atomic and non-atomic packets. In that case, there is a small possibility that the Identification values used by the source host in non-atomic packets will temporarily be in close correlation with those used by the tunnel ingress in atomic packets, where a "collision" may occur in the Identification values. Factors that mitigate such conflicts are the random assignment of the initial Identification value, random arrivals of atomic and non-atomic packets, the random length of the fragments used by the tunnel ingress (i.e., to cause a length mismatch for colliding reassemblies) and, in even rarer instance, the use of the Internet checksum following reassembly.

5. Applicability

This approach applies to common IPv6 transition mechanisms, including configured tunnels [RFC4213], 6to4 [RFC3056], ISATAP [RFC5214], DSMIP [RFC5555], 6rd [RFC5969], etc.

This same approach can further be applied to any variety of IP-within-IP tunnels, including GRE [RFC1701], IPv4-in-IPv4 [RFC2003], IPv6-in-IPv6 [RFC2473], IPv4-in-IPv6 [RFC6333], IPsec [RFC4301], Teredo [RFC4380], LISP [I-D.ietf-lisp], SEAL [I-D.templin-intarea-seal], etc.

6. IANA Considerations

There are no IANA considerations for this document.

7. Security Considerations

The security considerations for the various tunneling mechanisms apply also to this document.

8. Acknowledgments

This method was inspired through discussion on the IETF v6ops and NANOG mailing lists in the May/June 2012 timeframe.

9. References

9.1. Normative References

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

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

9.2. Informative References

[I-D.ietf-intarea-ipv4-id-update]
Touch, J., "Updated Specification of the IPv4 ID Field", draft-ietf-intarea-ipv4-id-update-05 (work in progress), May 2012.

[I-D.ietf-lisp]
Farinacci, D., Fuller, V., Meyer, D., and D. Lewis, "Locator/ID Separation Protocol (LISP)", draft-ietf-lisp-23 (work in progress), May 2012.

[I-D.templin-intarea-seal]
Templin, F., "The Subnetwork Encapsulation and Adaptation Layer (SEAL)", draft-templin-intarea-seal-42 (work in progress), December 2011.

[RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191, November 1990.

[RFC1701] Hanks, S., Li, T., Farinacci, D., and P. Traina, "Generic

Routing Encapsulation (GRE)", RFC 1701, October 1994.

[RFC1981] McCann, J., Deering, S., and J. Mogul, "Path MTU Discovery for IP version 6", RFC 1981, August 1996.

[RFC2003] Perkins, C., "IP Encapsulation within IP", RFC 2003, October 1996.

[RFC2473] Conta, A. and S. Deering, "Generic Packet Tunneling in IPv6 Specification", RFC 2473, December 1998.

[RFC3056] Carpenter, B. and K. Moore, "Connection of IPv6 Domains via IPv4 Clouds", RFC 3056, February 2001.

[RFC4213] Nordmark, E. and R. Gilligan, "Basic Transition Mechanisms for IPv6 Hosts and Routers", RFC 4213, October 2005.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC4380] Huitema, C., "Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs)", RFC 4380, February 2006.

[RFC4963] Heffner, J., Mathis, M., and B. Chandler, "IPv4 Reassembly Errors at High Data Rates", RFC 4963, July 2007.

[RFC5214] Templin, F., Gleeson, T., and D. Thaler, "Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)", RFC 5214, March 2008.

[RFC5555] Soliman, H., "Mobile IPv6 Support for Dual Stack Hosts and Routers", RFC 5555, June 2009.

[RFC5969] Townsley, W. and O. Troan, "IPv6 Rapid Deployment on IPv4 Infrastructures (6rd) -- Protocol Specification", RFC 5969, August 2010.

[RFC6333] Durand, A., Droms, R., Woodyatt, J., and Y. Lee, "Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion", RFC 6333, August 2011.

Author's Address

Fred L. Templin (editor)
Boeing Research & Technology
P.O. Box 3707
Seattle, WA 98124
USA

Email: fltemplin@acm.org