Internet Engineering Task Force
INTERNET-DRAFT
draft-deering-ipv6-encap-addr-deletion-00.txt
November 14, 2001
Expires May 14, 2002

Steve Deering
Cisco
Brian Zill
Microsoft

Redundant Address Deletion when Encapsulating IPv6 in IPv6

Status of this Memo

Abstract

In some potentially common uses of IPv6-in-IPv6 encapsulation ("tunneling"), a node that is performing an encapsulation or decapsulation will also be the source or destination of the packet being encapsulated. That can result in the same IPv6 address appearing in both the outer (encapsulating) and inner (encapsulated) IPv6 headers. This document specifies a method for deleting such redundant addresses from an inner header when performing an encapsulation, and restoring those addresses when decapsulating, resulting in a 16-octet (128-bit) reduction in header overhead, per address deleted.

1. Introduction

Encapsulation of IP packets inside other IP packets (usually called

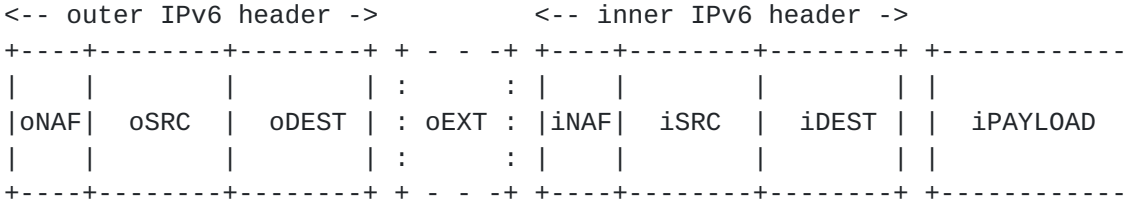
"tunneling") has been used to achieve a number of goals in the IPv4 Internet, and the same mechanism is likely also to be widely used in

in the IPv6 Internet. In some of the common uses of IP-in-IP encapsulation, such as mobile IP tunnels or so-called "virtual private network" (VPN) tunnels terminating on individual hosts, a node performing an encapsulation or decapsulation may also be the source or destination of the packet being encapsulated. In other words, a tunnel entrance/exit may coincide with one of the endpoints of the traffic being tunneled. When that is the case, the same IP address may appear in both the outer (encapsulating) IP header and the inner (encapsulated) IP header. In the case of IPv6, those addresses are 16 octets (128 bits) long -- a significant per-packet overhead -- and it would be desirable to avoid such duplication of information if possible. This document specifies a method for deleting such addresses from IPv6-in-IPv6 encapsulated packets.

2. Redundant Address Deletion/Restoration

At the point at which an IPv6 packet is being encapsulated in another IPv6 packet, the normal behavior is to take a packet of the following format:

and prepend one or more headers to produce a packet of the following format:



where:

NAF represents the non-address fields of an IPv6 header (I.e., the first 8 octets of an IPv6 header)

SRC is an IPv6 source address

DEST is an IPv6 destination address

EXT is zero or more IPv6 extension headers

the prefix "o" means "outer"

the prefix "i" means "inner"

The presence of the inner IPv6 header is indicated by the IANA-assigned value 41 (decimal) in the Next Header field of the last outer header. If there are no outer extension headers (oEXT) present, this is the Next Header field in the oNAF part of the outer IPv6 header; otherwise, it is the Next Header field of the last ("rightmost") outer extension header.

To enable deletion of redundant IPv6 addresses, three new "Next Header" values are introduced:

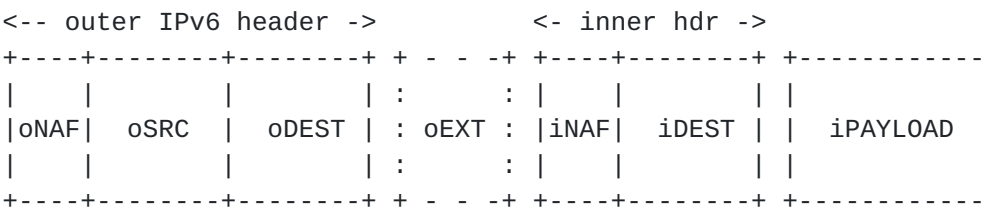
IPv6_NO_SRC (value TBD) - indicates an IPv6 header with its Source Address field removed

IPv6_NO_DEST (value TBD) - indicates an IPv6 header with it Destination Address field removed

IPv6_NO_ADDRS (value TBD) - indicates an IPv6 header with both of its address fields removed

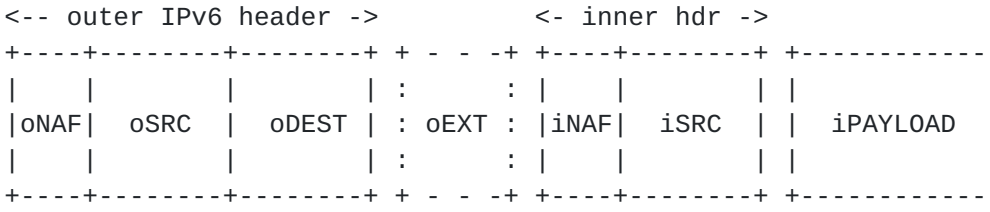
When performing the encapsulation, the encapsulating node compares the addresses in the outer and inner IPv6 headers and produces packets as follows:

If oSRC == iSRC & oDEST != iDEST, produce a packet of the following format:



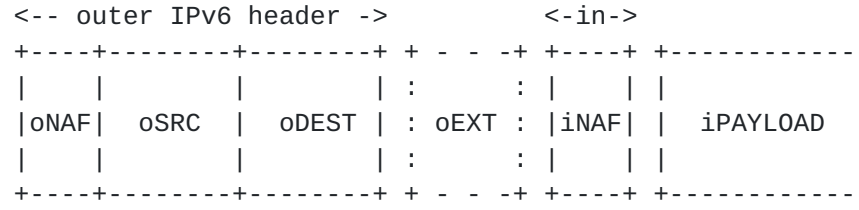
and set the Next Header field of the last outer header to IPv6_NO_SRC.

If oSRC != iSRC & oDEST == iDEST, produce a packet of the following format:



and set the Next Header field of the last outer header to IPv6_NO_DEST.

If oSRC == iSRC & oDEST == iDEST, produce a packet of the following format:



and set the Next Header field of the last outer header to IPv6_NO_ADDRS.

Otherwise, produce the normal encapsulated format with a full inner IPv6 header, identified by a Next Header value of 41.

When performing a decapsulation, the decapsulating node uses the Next Header value of the last outer header to determine which, if any, of the addresses were deleted from the inner IPv6 header, and restores them from the coresponding addresses received in the outer IPv6 header, to produce the original encapsulated packet:

3. Issues

[This part isn't done yet. The following are the authors' notes to themselves, identifying issues to be discussed in the next version

of this draft]

- Discuss MTU and fragmentation considerations when using this technique. No particular problems, because the transformations all increase the available MTU, rather than reduce it, compared to the normal encapsulation case.

- Note that the technique described herein can and should be applied recursively, when a node is the entry/exit point of a tunnel within a tunnel (within a tunnel....).

- Observe that the same technique can be used when the last outer header is not a standard IPv6 extension header with a Next Header field, e.g., when doing UDP tunneling or GRE tunneling. In those cases, three new code points will have to be assigned in whatever "next header" code space is used by those particular headers (e.g., well-known port numbers for UDP, or ethertypes for GRE), to identify

the three forms of IPv6 headers with deleted addresses.

- Explain why we don't also propose deleting other possiby redundant fields in the iNAF part of the inner header. (The reason has to do with maintaining 64-bit alignment of all headers, for efficient memory access. In cases where saving every byte or bit matters, there already exist IPv6 header compression standards that work across multiple headers, including encapsulations. However, if this spec is adopted, those other standards should be updated to take into account the three new variants of the IPv6 header defined here.)

- Discuss backwards-compatibility issues, i.e., ensuring that these forms of encapsulation are not used by a tunnel entry-point without assurance that the tunnel exit-point understands and implements them.

- Perhaps add some words suggesting that, when there is a choice of addresses for the outer header, an effort be made to pick ones that are the same as ones present in the inner header, whenever possible.

n. Security Considerations

[haven't thought about this yet]

m. IANA Considerations

This specification requires the assignment of three new 8-bit Protocol Type values to be used in IPv6 Next Header fields. It is suggested that those new Protocol Types be named as follows:

```
    IPv6_NO_SRC
    IPv6_NO_DEST
    IPv6_NO_ADDRS
```

References

[TBD]

Change History

None.

Acknowledgements

[TBD: acknowledge previous examples of this general idea, e.g., RFC 2004]

Authors' Addresses

Steve Deering
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
Phone: +1 408 527 8213
Wmail: deering@cisco.com

Brian Zill
Microsoft Research
One Microsoft Way
Redmond, WA 98052
USA
Phone: +1 425 703 3568
Email: bzill@microsoft.com