Internet Draft
<draft-kitamura-ipv6-neighbor-cache-update-00.txt>

H. Kitamura
NEC Corporation
S. Ata
Osaka City University
M. Murata
Osaka University

Expires March 2010
October 19, 2009

IPv6 Neighbor Cache Update
<draft-kitamura-ipv6-neighbor-cache-update-00.txt>

Status of this Memo

This Internet-Draft will expire on January 2010.

Copyright Notice

Abstract

This document describes IPv6 neighbor cache update methods from external nodes. In current communication environments, using status of IP addresses is frequently changed by several operations, such as disconnecting / connecting nodes to networks at mobile environments, suspending / hibernating / resuming nodes, and consuming large number of IP addresses dynamically when ephemeral address [Ephemeral] function is enabled.

When using status of an IP address is changed from "Used" to "Not Used", its related used resources such as neighbor cache entry should be deleted cooperatively from security enhancement viewpoint and efficient resource management viewpoint.

[RFC4861] defines neighbor discovery methods and describes how to manage neighbor cache entries by neighbor discovery messages. However, [RFC4861] does not define quick and clear neighbor cache update (delete) functions.

In order to meet above request, this document proposes two types of IPv6 neighbor cache update (delete) methods from external nodes. One is a heuristic type method that does NOT require neighbor discovery message extensions. The other is an explicit type method that requires small neighbor discovery message extensions.

1. Introduction

In current communication environments, using status of IP addresses is frequently changed (from "Used" to "Not Used") by several operations, such as disconnecting / connecting nodes to networks at mobile environments, suspending / hibernating / resuming nodes, and consuming large number of IP addresses dynamically when ephemeral address [Ephemeral] function is enabled.

When using status of an IP address is changed from "Used" to "Not Used", its related used resources such as neighbor cache entry should be deleted cooperatively from security enhancement viewpoint and efficient resource management viewpoint. (This also follows the manner "Leave everything neat and tidy when you go behind you".)

[RFC4861] defines neighbor discovery methods and describes how to manage neighbor cache entries by neighbor discovery messages. However, [RFC4861] does not define quick and clear neighbor cache update (delete) functions.

In order to meet above request, this document proposes two types of IPv6 neighbor cache update (delete) methods from external nodes.

2. Problems on IPv6 Neighbor Cache Entry State Aging

By following the [RFC4861] definition, typical neighbor cache entry state aging procedures can be shown in Fig. 1.

Fig.1 shows a case of an Edge Router's neighbor cache entry state aging.

```
               Edge Router          Client Node
                    |                    |
        NC State    |                    |
        ~~~~~~~~    |<-------Main--------|
                    |<---Communication---|
     ---------------|=NS================>|
       INCOMPLETE   |                    |
     ---------------|<================NA=|
                    |--------Main------->|
        REACHABLE   |----Communication-->|
                    |                    |
                    |                    |
                    |                    |==================
     ---------------|                    | Session finished
                    |                    |
            STALE   |                    |
              .     |                    |
              .     |                    |
              .     |                    |
              .     |                    |
                    |                    |
```

Fig. 1 Overview of Neighbor Cache Entry State Aging

When the Client Node finishes a communication session that uses an IP address and releases the IP address, status of the IP address is changed from "Used" to "Not Used" and state of the corresponding neighbor cache entry on the Edge Router reaches STALE state.

The STALE state is also aged by the timer. However, compared to the other states, the STALE state continues for a long time (typically one day). When the Edge Router issues a packet which looks up this neighbor cache entry, its state is proceeded to next state (DELAY).

If the IP address is released and the status of it is "Not Used", this action is never taken, and the neighbor cache entry on the Edge Router stays at the STALE state for a long time and is not deleted.

In current communication environments, using status of IP addresses is changed (from "Used" to "Not Used") frequently. It becomes a problem that the corresponding neighbor cache entry is not deleted cooperatively.

3. Neighbor Cache Update (Delete) Methods

In order to meet above request, this document proposes two types of IPv6 neighbor cache update (delete) methods from external nodes. One is a heuristic type method that does NOT require neighbor discovery message extensions. The other is an explicit type method that requires small neighbor discovery message extensions.

3.1 Heuristic Type Neighbor Cache Update (Delete) Method

Fig. 2 shows abstract of a heuristic type neighbor cache update method. A simple method to delete a neighbor cache entry is to proceed the entry's state from the STALE to the DELAY. After the DELAY state is finished, it enters the PROBE state. After the PROBE state is finished, the neighbor cache entry is deleted.

In order to proceed the entry's state from the STALE to the DELAY from a external node (Client Node), Client Node issues a special packet that requires to issue a packet which looks up this neighbor cache entry.

We can use any types of packet for this special packet if it satisfies the above conditions. In this case, an NS (Neighbor Solicitation) message is chosen. In order to look up the target neighbor cache entry at the Edge Router, source (target) address of the NS is set to the released (Not Used) IP address.

This method does not require neighbor discovery message extensions. Only by issuing a special NS message from Client Node when an address is released, the corresponding target neighbor cache entry is deleted.

```
               Edge Router          Client Node
                    |                    |
        NC State    |                    |
        ~~~~~~~~    |<-------Main--------|
                    |<---Communication---|
     ==============-|=NS================>|
       INCOMPLETE   |                    |
     ---------------|<================NA=|
                    |--------Main------->|
        REACHABLE   |----Communication-->|
                    |                    |==================
                    |                    | Session is finished:
                    |                    | IP Address is Released
                    |                    |  [Used -> Not Used]
                    |                    |
     ---------------|<..............u-NA-|  NC Update is Started!
                    |                    |
            STALE   |<-**************-NS-| (Src= Not Used IP Addr)
                    |                    |
     ---------------|-NA-**************->| (Reached but Ignored)
            DELAY   |                    |
     ---------------|=NS================>|
                    |                    |
            PROBE   |=NS================>|
                    |                    |
                    |=NS================>|
     ==============-|                    |
         (deleted)
```

Fig. 2 Heuristic Type Neighbor Cache Update (Delete) Method

3.2 Explicit Type Neighbor Cache Update (Delete) Method

Fig. 3 shows abstract of a explicit type neighbor cache update method. In order to delete the target neighbor cache entry, NA (Neighbor Advertisement) message format is extended.

Fig. 4 shows an extended NA message format. Two flags are newly introduced. One is D flag to order to delete a target neighbor cache entry. If the state of the target entry stays at the REACHABLE state, the delete operation ordered by the D flag is not executed. The other is F flag to force to execute delete operation ordered by the D flag. The target entry is deleted at wherever the state of it stays. Only when D flag is set, F flag is considered.

```
               Edge Router          Client Node
                    |                    |
        NC State    |                    |
        ~~~~~~~~    |<-------Main--------|
                    |<---Communication---|
     ==============-|=NS================>|
       INCOMPLETE   |                    |
     ---------------|<================NA=|
                    |--------Main------->|
        REACHABLE   |----Communication-->|
                    |                    |==================
                    |                    | Session is finished:
                    |                    | IP Address is Released
                    |                    |  [Used -> Not Used]
                    |                    |
     ---------------|<..............u-NA-|  NC Update is Started!
                    |                    |
            STALE   |<-##############-NA-|  Extended NA
                    |                    |
     ==============-|                    |
         (deleted)
```

Fig. 3 Explicit Type Neighbor Cache Update (Delete) Method

```
     0                   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      |     Code      |          Checksum             |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |R|S|O|D|F|                 Reserved                            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    +                                                               +
    |                                                               |
    +                       Target Address                          +
    |                                                               |
    +                                                               +
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |   Options ...
    +-+-+-+-+-+-+-+-+-+-+-+-+-
```

D: Delete flag.
F: Force Delete flag.

Fig. 4 Extended NA (Neighbor Advertisement) Message Format

3.3 Combined Type Neighbor Cache Update (Delete) Method

It is possible to design (Heuristic and Explicit) Combined type neighbor cache update method. Fig. 5 shows it.

It is designed that unknown extended messages is ignored in [RFC4861]. When a node who does not understand NA extension receives the extended NA message for neighbor cache update, the message is ignored and the target neighbor cache entry is not deleted.

Even if the entry is not deleted by the explicit method, the entry is deleted by the following heuristic method.

```
               Edge Router          Client Node
                    |                    |
        NC State    |                    |
        ~~~~~~~~    |<-------Main--------|
                    |<---Communication---|
     ==============-|=NS================>|
       INCOMPLETE   |                    |
     ---------------|<================NA=|
                    |--------Main------->|
        REACHABLE   |----Communication-->|
                    |                    |==================
                    |                    | Session is finished:
                    |                    | IP Address is Released
                    |                    |  [Used -> Not Used]
                    |                    |
     ---------------|<..............u-NA-|  NC Update is Started!
                    |                    |
            STALE   |<-##############-NA-|  Extended NA
         (deleted?) |                    |
                    |<-**************-NS-| (Src= Not Used IP Addr)
                    |                    |
     ---------------|-NA-**************->| (Reached but Ignored)
            DELAY   |                    |
     ---------------|=NS================>|
                    |                    |
            PROBE   |=NS================>|
                    |                    |
                    |=NS================>|
     ==============-|                    |
         (deleted)
```

Fig. 5 Combined Type Neighbor Cache Update (Delete) Method

4. Security Considerations

Security Considerations of Neighbor Discovery [RFC4861] can also be applied to Neighbor Cache Update. Additional security enhancement features are provides in the Neighbor Cache Update methods, because they can provide unnecessary neighbor cache entry deleting functions.

5. IANA Considerations

This document has no actions for IANA.

Appendix A. Implementations

The neighbor cache updated specification has been implemented under the following environments, and its basic functionalities have been verified

```
  OS:   FreeBSD6.2R (32bit / 64bit)
  CPU:  i386 / amd64
```

Acknowledgment

A part of this work is supported by the program: SCOPE (Strategic Information and Communications R&D Promotion Programme) operated by Ministry of Internal Affairs and Communications of JAPAN.

References

Normative References

[RFC4861] T. Narten, E. Nordmark, W. Simpson, and H. Soliman, "Neighbor Discovery for IP Version 6 (IPv6)", RFC 4861, September 2007

[RFC4862] S. Thomson, T. Narten, and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, September 2007

Informative References

[Ephemeral] H. Kitamura, S. Ata, and M. Murata, "IPv6 Ephemeral Addresses" <draft-kitamura-ipv6-ephemeral-address-01.txt> work in progress, July 2009

[Uncertain] H. Kitamura, S. Ata, and M. Murata, "Harmless IPv6 Address State Extension (Uncertain State)" <draft-kitamura-ipv6-uncertain-address-state-01.txt> work in progress, July 2009

Authors' Addresses

Hiroshi Kitamura
Service Platform Research Laboratories, NEC Corporation
(Igarashi Building 4F) 11-5, Shibaura 2-Chome,
Minato-Ku, Tokyo 108-8557, JAPAN
Graduate School of Information Systems,
University of Electro-Communications
5-1 Chofugaoka 1-Chome, Chofu-shi, Tokyo 182-8585, JAPAN
Phone: +81 3 5476 9795
Fax: +81 3 5476 1005
Email: kitamura@da.jp.nec.com

Shingo Ata
Graduate School of Engineering, Osaka City University
3-3-138, Sugimoto, Sumiyoshi-Ku, Osaka 558-8585, JAPAN
Phone: +81 6 6605 2191
Fax: +81 6 6605 2191
Email: ata@info.eng.osaka-cu.ac.jp

Masayuki Murata
Graduate School of Information Science and Technology, Osaka Univ.
1-5 Yamadaoka, Suita, Osaka 565-0871, JAPAN
Phone: +81 6 6879 4542
Fax: +81 6 6879 4544
Email: murata@ist.osaka-u.ac.jp