DNA WG
Internet-Draft
Expires: November 28, 2005

JinHyeock Choi
Samsung AIT
May 27, 2005

DNA Solution: Link Identifier based approach
draft-jinchoi-dna-protocol2-00.txt

Status of this Memo

Copyright Notice

Abstract

DNA solution consists of 1)link identity detection with a single RA and 2)quick RA acquisition. This draft presents the first component based on Link Identifier. It describes a way for link identity detection with prefix based Link Identifier, 'linkid prefix'. While this document doesn't include the second component, any quick RA acquisition scheme will work with the proposal. The draft is the result of DNA DT discussion.

Table of Contents

1. Introduction 3
1.1 Checking for link change with Link Identifier 3
1.2 Link Identifier based on Prefix 3
1.3 Protocol overview 4
2. New Terms 5
2.1 Linkid prefix 5
2.2 LEAST_VALID_LIFETIME 5
2.3 Linkid lifetime 5
2.4 Linkid Prefix list (LinkidPrefixList) 6
2.5 LPIO (Learned Prefix Information Option) 7
2.6 Identifier bit (I-bit) 7
3. Router Operations 8
3.1 Collecting the prefixes 8
3.2 Selecting the linkid prefix 8
3.3 Advertising the linkid prefix 9
3.4 Graceful linkid prefix change 9
3.4.1 When a new linkid prefix is added. 9
3.4.2 When the linkid prefix lifetime decreases to LEAST_VALID_LIFETIME. 9
3.4.3 When the linkid prefix is removed while its lifetime is greater than LEAST_VALID_LIFETIME. 10
4. Host Operations 12
4.1 Managing the LinkidPrefixList 12
4.2 Checking for link change 13
5. IANA Considerations 14
6. Security Considerations 15
7. Open Issues 16
7.1 Issue 001: LPIO format 16
7.2 Issue 002: Advertising old linkid prefix 16
7.3 Issue 003: Flash renumbering and early reassignment . . . 17
7.4 Issue 004: DAD Interaction 17
7.5 Issue 005: MLD Interaction 17
7.6 Issue 006: Sending RA before completing prefix collection 17
8. Comparison with landmark based approach 18
9. Acknowledgments 19
10. References 20
10.1 Normative References 20
10.2 Informative References 20
Author's Address 21
Intellectual Property and Copyright Statements 22

1. Introduction

Upon establishing a new link-layer connection, a host should detect the identity of the currently attached link to ascertain the validity of the existing IP configuration. If the host is attached to a different link, it also needs to acquire the IP configuration for the new link [4].

DNA solution should be able to 1) check for link change with a single RA (Router Advertisement) and 2) get the RA with minimum latency [8]. This draft presents only the first component, link identity detection. But the proposed Link Identifier based scheme can work with any existing quick RA acquisition method, such as FRD in [16], FastRA in [17] or Hash based scheme in [13].

1.1 Checking for link change with Link Identifier

Usually a host receives the link information with RA (Router Advertisement) messages. But it's difficult for a host to correctly check for link change with a single RA message.

It may be better to design a new way to represent a link, 'Link Identifier'. Each link has its unique Link Identifier and all routers in the link advertise the same Link Identifier in RAs. With a Link Identifier, an RA can represent link identity properly and hosts can check for link change immediately without resorting to approximate knowledge.

When a host receives an RA with the same Link Identifier, it still remains at the same link. If it receives an RA with a different Link Identifier, a link change has occurred and the host is attached to a different link.

1.2 Link Identifier based on Prefix

We may define a new option for linkid. In [14] Erik Nordmark proposed a new option, Location Indication Option and Brett Pentland submitted a draft on linkid [15].

Global prefix is another possible candidate for use as linkid (Link Identifier). If all routers on the link advertise the same prefix as the linkid, the prefix can properly represent the link identity.

Prefixes would have the advantage of being globally unique and requires no additional RA bytes if a router is already advertising the prefix. Only an added flag is needed to indicate that it is also a linkid (Link Identifier). Linkid may need to change as the prefixes used on a link change.

1.3 Protocol overview

The routers on the link collects the prefixes on the link and, as the linkid, pick the smallest prefix among the prefixes with lifetime greater than 3 hours. We call such a prefix 'linkid prefix'.

The routers advertise the linkid prefix in every RA.

If the linkid prefix belongs to the advertising router, it includes the linkid prefix in PIO (Prefix Information Option) with identifier bit (I-bit) set, which indicates that the prefix is the linkid.

If not, the router includes the linkid prefix in LPIO (Learned Prefix Information Option) with identifier bit (I-bit) set, which also indicates that the prefix is the linkid.

The routers use a single linkid prefix at any given time, but due to the possibility that the linkid prefix changing (e.g., when a link is renumbered), the hosts track the list of linkid prefixes they've received in the last 1.5 hours to generate "the linkid prefix list (LinkidPrefixList)".

Take notice that the above doesn't mean that multiple linkid prefixes are assigned on a link at the same time. Not like prefix, the linkid is supposed to be unique at a given moment and the LinkidPrefixList would have only one element for the most of time.

Whenever the host receives an RA with a linkid prefix, a host extracts the prefix to store it in the LinkidPrefixList.

If the host receives an RA with a linkid prefix and the RA also includes a linkid prefix the host knows of, it assumes that it still remains at the same link.

If the host receives an RA with a linkid prefix and the RA doesn't include a linkid prefix the host knows of, it assumes that it has moved to a new link. The host also discards the existing LinkidPrefixList and starts a new one.

This is rather intuitive description and Sec 4 gives more rigorous and detailed one.

2. New Terms

2.1 Linkid prefix

A prefix which plays the role of linkid (Link Identifier).

2.2 LEAST_VALID_LIFETIME

Only a prefix with valid lifetime greater than LEAST_VALID_LIFETIME can be a linkid prefix. When the lifetime of a linkid prefix becomes less than LEAST_VALID_LIFETIME, it can no longer be a Link Identifier.

For the time being, we set the LEAST_VALID_LIFETIME as 3 hours.

LEAST_VALID_LIFETIME is introduced to gracefully change the linkid prefix as described in Sec 3.4.

2.3 Linkid lifetime

RFC 2461 [1] defines default valid lifetime as 30 days and default preferred lifetime as 7 days, which may be too long for a linkid prefix.

Hence, for a linkid prefix, a host keeps a separate lifetime, 'linkid lifetime'.

For each linkid prefix, whenever a host receives an RA with the prefix, the host sets the linkid lifetime as 1.5 hour and starts timer.

When linkid lifetime expires, the host no longer uses the prefix as a linkid. But the host may keep the prefix as an ordinary prefix until its valid lifetime expires.

According to RFC 2461 [1], the maximum of MaxRtrAdvInterval is 1800 secs. Hence, because all RAs are supposed to carry the linkid prefix, linkid lifetime expiration means that the host has lost at least 3 RAs.

Linkid lifetime is introduced to gracefully change the linkid prefix as described in Sec 3.4. It is also intended to deal with flash renumbering and early reassignment as below.

When a linkid prefix is removed from a link, it is recommended that the linkid prefix should not be re-assigned to other link in 3 hours.

The above means that if a host sees the valid prefix which was a

linkid and its linkid lifetime is still left, the host still remains at the same link.

Sub-sec 7.3 gives the detailed analysis.

2.4 Linkid Prefix list (LinkidPrefixList)

A host keeps the linkid prefix list (LinkidPrefixList) per a link. The LinkidPrefixList is the set of linkid prefixes the host has received in the last 1.5 hours.

If the host has declared the movement to a different link, it needs to discard the LinkidPrefixList from the old link.

When the host receives a prefix with identifier bit (I-bit) set, it keeps the prefix in the LinkidPrefixList for the next 1.5 hours.

Take notice that the prefix in the LinkidPrefixList may not be the linkid anymore. Also it's possible for the LinkidPrefixList has more than one prefix.

This is for graceful linkid prefix change. When a linkid prefix is changed in a link, there may be some flapping for a while. A router may advertise the new linkid prefix, whereas the other one the old one. Hence hosts can confuse this as frequent link changes.

To prevent this, hosts keep the LinkidPrefixList and assume they are still at the same link, if they see the prefix in the LinkidPrefixList.

We illuminate the role of the LinkidPrefixList with the following example. Assume a link with two routers R1 and R2. R1 advertise the linkid prefix P1 and R2 advertises the prefix P2. At this stage, hosts would have the LinkidPrefixList {P1}.

R2 starts advertising a new prefix P3, which happens to be the smallest one, hence a new linkid prefix. R2 sends an RA with P3 (with I-bit set) and P1 (without I-bit set). A host sees the RA with a new linkid prefix but will not assume a link change because of P1. The host simply adds P3 to its LinkidPrefixList and the LinkidPrefixList becomes {P1, P3}.

Then because of packet loss R1 doesn't receive the RA and sends an RA with P1 (with I-bit set) but without P3. The host sees an another RA with a different linkid prefix but will not assume a link change because of P1. The LinkidPrefixList is still {P1, P3}

Afterwards R1 belatedly notices a linkid change and starts

advertising P3 as the linkid prefix. Again the host doesn't assume a link change because of P1 and the LinkidPrefixList is {P1, P3}.

During the whole transition period, the prefix P1 in the LinkidPrefixList plays the role of assuring hosts of no link change.

2.5 LPIO (Learned Prefix Information Option)

When routers advertise learned prefixes, they use LPIO instead of ordinary PIO.

LPIO format is TBD but it at least needs to include

- prefix length
- prefix
- Identifier bit (I-bit) (indicating the prefix in the LPIO is the linkid)

Sub-sec 7.1 gives the detailed discussion about LPIO format.

2.6 Identifier bit (I-bit)

A bit in PIO or LPIO which indicates that the prefix in this option is the linkid prefix.

Identifer bit (I-bit) format is TBD.

3. Router Operations

3.1 Collecting the prefixes

To select the linkid prefix, routers should collect all the prefixes on the link.

A router, when it boots or is configured to act as a router (Sec 6.2.2 in RFC 2461 [1]), first sends an RS and waits for RAs to gather prefixes.

Sending one RS and waiting for 4 seconds might suffice; optionally retransmitting the RS once or twice and waiting a total of 8 or 12 seconds gives higher confidence.

From the received RAs, the router extracts only the valid prefixes in PIO and generate the "learned prefix list (LearnedPrefixList)". It is noted that the prefixes the router advertises itself (the AdvPrefixList in RFC 2461 [1]) are not included in the LearnedPrefixList.

The router takes the union of the learned prefix list and the prefixes the router itself advertises to generate the complete prefix list as defined in [9]

Once that process is complete, the router will send RAs and respond to RSs, but not before that.

After initial RS/ RA exchange, the router can send a few closely spaced RAs as specified in RFC 2461 [1] to quickly spread its own information on the link.

Whenever a router receives a new prefix in PIO, it will update its learned prefix list.

Take notice that, for prefix list generation, a router only takes the prefixes in PIO into consideratoin.

3.2 Selecting the linkid prefix

Among the prefixes whose valid lifetime is greater than LEAST_VALID_LIFETIME, i.e. 3 hours, the router picks the smallest prefix as the linkid prefix.

* There are other ways to determine the linkid prefix. We may choose any subset of the complete prefix list and pick the smallest or greatest one of it.

When a router receives a new prefix in PIO, (with or without identifier bit (I-bit) set), if the prefix is smaller than the current linkid prefix and has valid lifetime greather than 3 hours, the router selects the new prefix as a new linkid prefix.

In case the new prefix smaller than the current linkid prefix is advertised in LPIO, the router doesn't change the linkid prefix.

3.3 Advertising the linkid prefix

Whenever a router sends an RA, whehter solicited or unsolicited, it includes the linkid prefix in it. Hence, all RAs carry the linkid prefix.

When a router advertises the linkid prefix, if the linkid prefix is explicitly configured on the router, it sends it in PIO with I-bit set.

If not, the router sends the linkid prefix in LPIO with I-bit set.

3.4 Graceful linkid prefix change

Basic idea is, when a router changes a linkid prefix, for the time being, it sends both 1) new linkid prefix with I-bit set and 2) old linkid prefix without I-bit set to notify hosts that only linkid has been changed without a link change.

3.4.1 When a new linkid prefix is added.

When a router starts advertising a new prefix, if the prefix happens to be the smallest one in the link, a linkid prefix needs to be changed.

First the router sends to all-node multicast address an RA with 1) new linkid prefix with I-bit set and 2) old linkid prefix without I-bit set several times.

Upon receiving this RA, because new prefix is smaller than the current linkid prefix, routers would change the linkid prefix to the new one.

Old linkid prefix (whether its I-bit set or not) assures hosts that they still remain at the same link. So hosts would simply update its LinkidPrefixList without any outward activity, such as sending an RS.

3.4.2 When the linkid prefix lifetime decreases to LEAST_VALID_LIFETIME.

When the valid lifetime of the linkid prefix becomes LEAST_VALID_LIFETIME, i.e. 3 hours, it is no longer a linkid.

A router decreases the prefix lifetime in real-time and at the moment the prefix lifetime becomes 3 hours, the router selects a new linkid prefix and stops advertising the I-bit on the (old) linkid prefix.

For the time being, when the router sends RA, it includes 1) new linkid prefix with I-bit set and 2) old linkid prefix without I-bit set to assure hosts that they still remain at the same link.

3.4.3 When the linkid prefix is removed while its lifetime is greater than LEAST_VALID_LIFETIME.

A router may want to remove the linkid prefix before the valid lifetime decreases to 3 hours.

When a router stops advertising a linkid prefix, it removes the prefix in two steps as below.

First the router picks the second smallest prefix as the new linkid prefix.

It sends to all node multcast address an RA with 1) new linkid prefix with I-bit set and 2) old linkid prefix with 2 hours as valid lifetime & without I-bit set.

Upon receiving this RA, routers would see the (old) linkid prefix in PIO with the valid lifetime less than LEAST_VALID_LIFETIME. Hence they also would select the second smallest one as the new linkid and starts advertising it with I-bit set.

When hosts receives this RA, they would have non-zero linkid lifetime for the (old) linkid prefix, becuase, at least, two RAs with (old) linkid prefix were sent within 1 hour according to MaxRtrAdvInterval value defined in [1]. Therefore, hosts would update its linkid prefix but would not assume a link change because of (old) linkid prefix.

Upon sending such an RA several times, once the router is sure that all routers have changed their linkid, it can remove the (old) prefix linkid entirely by advertising the prefix with zero lifetime value.

For example, assume a router R advertise the linkid prefix P1 and the second smallest prefix is P2.

When the valid lifetime of P1 is 7 days, R wants to remove P1 from the link.

If R immediately advertises an RA with 1) P1 with zero lifetime and 2) P2 with I-bit set, hosts may treat this as a link change, because they would discard P1 entirely when seeing it with zero lifetime.

So instead, R removes P1 in two steps. First, R advertises an RA with 1) P1 with lifetime 2 hours and 2) P2 with I-bit set.

Then hosts would know this is just a linkid change without a link change from (old) linkid prefix P1. Also other routers would know P1 is no longer linkid prefix, because P1 has lifetime less than LEAST_VALID_LIFETIME.

After a while, when R is sure that all nodes on the link have perceived linkid change, it can safely remove P1 by advertising it with zero lifetime.

Also it's recommended that once a prefix has been advertised with a lifetime that results in a prefix invalidation at time Ti, then the router should advertise the prefix with a valid lifetime=0 until time Ti has passed.

4. Host Operations

4.1 Managing the LinkidPrefixList

While routers manage a single linkid prefix, hosts track all the linkid prefixes it has seen in the last 1.5 hours to keep the linkid prefix list (LinkidPrefixList) per a link.

When the host has decides it has moved to a different link, it discards the LinkidPrefixList from the old link.

If a host has no linkid prefix, i.e. its LinkidPrefixList is empty, it sends an RS according to the procedure defined in RFC 2461 [1] to acquire one.

After one RS/ RA exchange, i.e. 4 secs after RS transmission, if no RA with a linkid prefix arrives, the host assumes it is at a non-supporting link and falls back on CPL [9].

Whenever a host receives an RA, it checks whether the RA includes a prefix with identifier bit (I-bit) set.

If the RA contains such a prefix, the host extracts the prefix and sets its linkid lifetime as 1.5 hour and starts timer.

The linkid lifetime for the prefix should decrement in real time that is, at any point in time they will be the remaining lifetime. An implementation could handle that by recording the 'expire' time for the information, or by periodically decrementing the remaining lifetime.

Then the host check for link change as described in Sub-sec 4.2. If the host assumes a link change, it discards the current LinkidPrefixList and makes a new LinkidPrefixList with the new prefix with I-bit set. If not, the host adds the new linkid prefix to the current LinkidPrefixList (unless the prefix already belongs to the LinkidPrefixList).

The host keeps a linkid prefix in the LinkidPrefixList as long as it has non-zero linkid lifetime and valid (prefix) lifetime.

When the linkid lifetime for the prefix expires, the prefix becomes an ordinary prefix. Hosts remove it from the LinkidPrefixList but keep it until its valid lifetime expires.

Take notice that, when the LinkidPrefixList becomes empty, i.e. the linkid lifetime of all prefixes in the LinkidPrefixList expires, the host doesn't assume a link change.

A prefix in the LinkidPrefixList may become invalid (as a prefix). The prefix may be advertised with zero lifetime or the host moves to an another link. In those cases, the host discards the prefix just like an ordinary prefix according to RFC 2461 [1].

4.2 Checking for link change

When a host doesn't have a linkid prefix, i.e. its LinkidPrefixList is empty, it relies on CPL [9] to check for link change.

When a host with a linkid prefix receives a hint of a possible link change, such as Link Up event notification [10], it may send an RS to all-router multicast address to get an RA with a linkid prefix.

After one RS/ RA exchange, i.e. 4 secs after RS transmission, if no RA with a linkid prefix arrives, the host assumes it is at a non DNA link and falls back on CPL [9].

When a host with a linkid prefix receives an RA, whether solicited or unsolicited, that includes a prefix with I-bit set, the host compares its LinkidPrefixList with the set of prefixes in the RA.

If there is a common prefix, i.e. the RA also includes the prefix in the LinkidPrefixList, the host assumes that it still remains at the same link.

Note that the common prefix need not be the linkid prefix in the RA.

If not, the host assumes that it has moved to a new link, discards its LinkidPrefixList and starts a new one.

5. IANA Considerations

This draft will define one new Neighbor Discovery [1] option and a new flag in PIO (Prefix Information Option), which must be assigned Option Type values within the option numbering space for Neighbor Discovery messages:

1. LPIO (Learned Prefix Information Option), described in Sec 2.

2. Identifier bit (I-bit) in PIO (Prefix Information Option), described in Sec 2.

6. Security Considerations

Because this DNA proposal is based on Neighbor Discovery, its trust models and threats are similar to the ones presented in [7]. Nodes connected over wireless interfaces may be particularly susceptible to jamming, monitoring and packet insertion attacks. Use of SEND [6] to secure Neighbor Discovery are important in achieving reliable detection of network attachment. This DNA proposal SHOULD incorporate the solutions developed in IETF SEND WG if available, where assessment indicates such procedures are required.

7. Open Issues

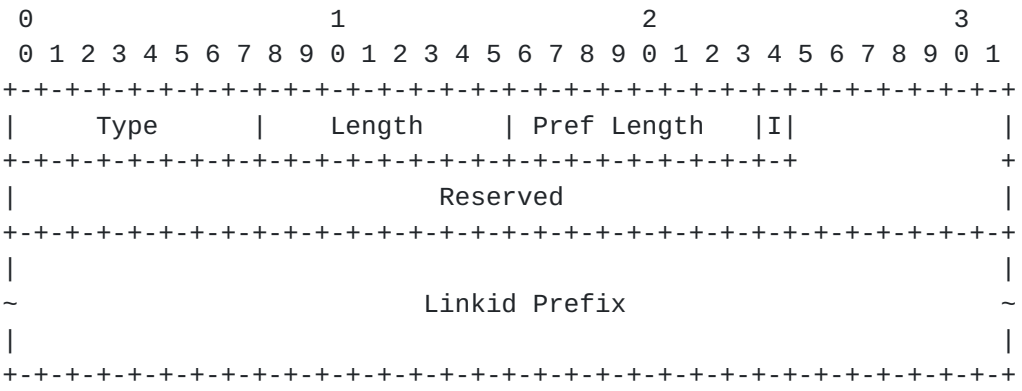
7.1 Issue 001: LPIO format

For LPIO format, there are two approaches under consideration:

1. A minimalist DNA approach - just carry what DNA needs

2. A complete copy of the PIO (Prefix Information Option)

For the first case, we can use the following option similar to landmark option in [13] that includes only prefix length, prefix and identifier bit (I-bit).



For the second case, we can reuse PIO format with a different code. The learning router would not assume it knows anything about the PIO it received but blindly send it (after only changing the option code to "LPIO").

The choice has to do with guessing how the PIO might evolve over time (new flags etc and their semantics). Perhaps the second approach might help as the PIO content evolves but, at this point, it is hard to tell.

We don't see much utility for a middle ground where LPIO includes some things which DNA doesn't need, but doesn't include everything that was in the PIO.

7.2 Issue 002: Advertising old linkid prefix

When a router changes a linkid prefix, to notify hosts that only linkid has been changed without a link change, the router sends both 1) new linkid prefix with I-bit set and 2) old linkid prefix without I-bit set for a while. It is needed to set a normative value about how long the router keeps advertising old linkid prefix after linkid

change. The value under consideration is 1.5 hour.

7.3 Issue 003: Flash renumbering and early reassignment

A useful definition of flash renumbering is that the prefix is removed from a link earlier than its latest advertised expiry time. We define "flash renumbering and early reassignment", as the above plus the prefix being assign to another link before the latest advertised expiry time on the old link.

Flash renumbering and early reassignment does have potential impact on DNA, (when using prefixes to identify links), because the host might move "in parallel" with the reassigned prefix, and therefor fails to detect movement when it in fact moved. Note that the prefix might have been advertised with a zero valid lifetime on the link, but the host might not notice this since it might already have disconnected from the old link when these RAs were sent.

To resolve this issue, this draft proposes that, when a linkid prefix is removed from a link, the linkid prefix should not be re-assigned to other link in 3 hours. With the above, a host would not see the same linkid prefix in two different links because linkid lifetime is 1.5 hour.

7.4 Issue 004: DAD Interaction

The draft needs to describe the DNA interaction with DAD such as what steps a host should go through with respect to DAD. The procedure under consideration is described in Sec 3 in [8]

7.5 Issue 005: MLD Interaction

The draft needs to describe the DNA interaction with MLD such as what steps a host should go through with respect to MLD. The procedure under consideration is described in Sec 3 in [8]

7.6 Issue 006: Sending RA before completing prefix collection

According to Sec 6.1, routers can't respond at all to RSs before completion of the soliciting phase. Brett Pentland proposed to relax this restriction.

8. Comparison with landmark based approach

We present a bit of comparison between linkid based approach in this draft and landmark based approach defined in [13].

C1 Linkid scheme does not add any extra options in an RS, whereas landmark scheme adds a new landmark option in an RS.

C2 Linkid scheme works well irrespective of whether the RA solicited or un-solicited. Routers can announce the linkid prefix in a solicited or unsolicited RA. Whereas in landmark scheme, a router first needs to be solicited to sends a landmark option with yes/no bit.

C3 In linkid scheme, when solicited, routers can send either unicast RA or multicast RA. Whereas, in landmark scheme, the usual response to an RS should be an unicast RA. Hence a soliciting RS should carry TSLLAO [19] and a mechanism is needed to limit the rate at which unicast RAs are sent.

C4 Linkid scheme works well when a link has lots of prefixes; in particular when there are so many prefixes that all the PIOs & LPIOs can not fit in one RA to form a Complete RA.

C5 When a prefix is added or removed in a link, routers may give conflicting information about link identity. Linkid scheme can correctly check for link change even under such an inconsistency.

9. Acknowledgments

The design presented in this document was generated by discussions among the members of the DNA Design Team (JinHyeock Choi, Tero Kauppinen, James Kempf, Sathya Narayanan, Erik Nordmark and Brett Pentland). Design Team members privide enlightening feedback and sometimes even more. Parts of this draft are direct cut and paste from the Design Team mailing list.

10. References

10.1 Normative References

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

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

[3] Hinden, R. and S. Deering, "Internet Protocol Version 6 (IPv6) Addressing Architecture", RFC 3513, April 2003.

[4] Choi, J., "Goals of Detecting Network Attachment in IPv6", draft-ietf-dna-goals-04 (work in progress), December 2004.

10.2 Informative References

[5] Johnson, D., Perkins, C., and J. Arkko, "Mobility Support in IPv6", RFC 3775, June 2004.

[6] Arkko, J., Kempf, J., Sommerfeld, B., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)", draft-ietf-send-ndopt-06 (work in progress), July 2004.

[7] Nikander, P., Kempf, J., and E. Nordmark, "IPv6 Neighbor Discovery (ND) Trust Models and Threats", RFC 3756, May 2004.

[8] Choi, J. and E. Nordmark, "DNA solution framework", draft-ietf-dna-soln-frame-00 (work in progress), April 2005.

[9] Nordmark, E. and J. Choi, "DNA with unmodified routers: Prefix list based approach", draft-ietf-dna-cpl-00 (work in progress), April 2005.

[10] Yegin, A., "Link-layer Event Notifications for Detecting Network Attachments", draft-ietf-dna-link-information-01 (work in progress), February 2005.

[11] Aboba, B., "Detection of Network Attachment (DNA) in IPv4", draft-ietf-dhc-dna-ipv4-11 (work in progress), April 2005.

[12] Pentland, B., "An Overview of Approaches to Detecting Network Attachment in IPv6", draft-dnadt-dna-discussion-00 (work in progress), February 2005.

[13] Narayanan, S., "Detecting Network Attachment in IPv6 Networks (DNAv6)", draft-pentland-dna-protocol-00 (work in progress),

May 2005.

[14] Nordmark, E., "MIPv6: from hindsight to foresight?", draft-nordmark-mobileip-mipv6-hindsight-00 (work in progress), November 2001.

[15] Pentland, B., "Router Advertisement Link Identification for Mobile IPv6 Movement Detection", draft-pentland-mobileip-linkid-03 (work in progress), October 2004.

[16] Choi, J., "Fast Router Discovery with RA Caching", draft-jinchoi-dna-frd-00 (work in progress), July 2004.

[17] Kempf, J., Khalil, M., and B. Pentland, "IPv6 Fast Router Advertisement", draft-mkhalil-ipv6-fastra-05 (work in progress), July 2004.

[18] Daley, G., "Deterministic Fast Router Advertisement Configuration", draft-daley-dna-det-fastra-01 (work in progress), October 2004.

[19] Daley, G., "Tentative Source Link-Layer Address Options for IPv6 Neighbour Discovery", draft-daley-ipv6-tsllao-01 (work in progress), February 2005.

Author's Address

JinHyeock Choi
Samsung AIT
Communication & N/W Lab
P.O.Box 111 Suwon 440-600
KOREA

Phone: +82 31 280 9233
Email: jinchoe@samsung.com

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.