DNA WG JinHyeock Choi
Internet-Draft Samsung AIT
Expires: April 16, 2005 Erik Nordmark
SUN Microsystems
October 16, 2004

DNA with unmodified routers: Prefix list based approach
draft-jinchoi-dna-cpl-01.txt

Status of this Memo

Copyright Notice

Abstract

Upon establishing a new link-layer connection, a host determines whether a link change has occurred to examine the validity of its IP configuration. This draft presents a way to robustly check for link change without assuming changes to the routers. Each link can be uniquely identified by the set of prefixes assigned to it. We propose that, at each attached link, the host generates the complete prefix list, that is, the prefix list contains all the prefixes on the link, and when it receives a hint that indicates a possible link change, it detects the identity of the currently attached link by consulting the existing prefix list. This memo describes how to generate the complete prefix list and to robustly detect the link identity even in the presence of packet losses.

Table of Contents

1. Introduction

When a host establishes a new link-layer connection, it may or may not have a valid IP configuration for the link, such as the subnet prefixes on the attached link or the default router address. Though the host has changed its network attachment point, it may still be at the same link. The term 'link' used in this document is as defined in RFC 2461 [1].

To examine its IP configuration, a host may check for link change, i.e. it verifies whether it is attached to the same or a different link as before [4]. The host can keep current IP configuration if and only if it remains at the same link.

Usually a host receives the link information from RA (Router Advertisement) messages. However, as described in 2.2. [4], it's difficult for a host to correctly detect the identity of a link with a single RA. None of the information in an RA can indicate a link change properly. Neither router address nor prefixes will do.

It may be better to design a new way to represent the identity of a link, Link Identifier as proposed in [8]. Each link has its unique Link Identifier and all routers in the link advertise the same Link Identifier. In [9], Erik Nordmark proposed a new option, 'Location Indication Option', which can serve as Link Identifier. Also Brett Pentland and all submitted a draft about Link Identifier [10].

However, even if some such new scheme is standardized and implemented, hosts would still need to cope with routers which do not (yet) implement such a scheme. Thus it makes sense to write down the rules for how to robustly detect the link identity without assuming changes to the routers.

2. Prefix list based approach

2.1 Approach

Currently there is one thing which can represent the identify of a link,

'The set of all the global prefixes assigned to a link.'

If a host has the complete list of all the assigned prefixes, it can properly determine whether a link change has occurred. If the host receives an RA containing one or more prefixes and none of the prefixes in it matches the previously known prefixes for the link, then it is assumed to be a new link.

This works because each and every global prefix on a link must not be used on any other link thus the sets of global prefixes on different links must be disjoint.

For the purposes of determining the prefixes, this specification uses both 'on-link' and 'addrconf' prefixes [1], that is, prefixes that have either the 'on-link' flag set, the 'autonomous address-autoconfiguration' flag set, or both flags set. This is a safe approach since both the set of on-link and the set of addrconf prefixes must be uniquely assigned to a link.

The difficulty lies both in ensuring that the complete prefix list for a single link is known and preventing prefixes from possibly different links to be viewed as the prefixes for a single link. This is challenging given that a single router advertisements is not required to include all prefixes for the link, router advertisements might be subject to packet loss, new routers and new prefixes (due to renumbering) might appear at any time on a link, and the host might attach to a different link at any time.

If the prefix list determination is incorrect, there can be two different types of failures. One is detecting a new link when in fact the host remains attached to the same link. The other is failing to detect when the host attaches to a different link. The former failure is undesirable because it might trigger other protocols, such as Mobile IPv6 [5], to do unneeded signaling, thus it is important to minimize this type of failure. The latter type of failure can lead to long outages when the host is not able to communicate at all, thus these failures must be prevented.

2.2 Assumptions

In this approach, we assume that an interface of a host can not be attached to multiple links at the same time. Though this kind of multiple attachments is allowed in neither Ethernet nor 802.11b, it may be possible in some Cellular System, especially CDMA.

2.3 Overview

Hints are used to tell a host that a link change might have happened. This hint itself doesn't confirm a link change, but can be used to initiate the appropriate procedures [4].

In order to never view two different links as one it is critical that when the host might have attached to a link, there has to be some form of hint. This hint doesn't imply that a movement to a different link has occurred, but instead, in the absence of such a hint there could not have been an attachment to a different link.

If the IP stack is notified by the link layer when a new attachment is established (e.g., when associating to a different access point in 802.11), this will serve as the hint. It helps to reduce the risk that the assignment of an additional prefix to a link will be misinterpreted as being attached to a different link. Note that this hint is merely a local indication and does not require any protocol changes. For instance, in many implementations this would be an indication passed from a link-layer device driver to the IP layer.

For implementations which do not provide a "link up" indication, the solution is to instead rely on either the receipt of an RA that contains disjoint prefixes from the prefixes that have been collected on the previous link or the lack of scheduled RAs as described in [5], as a hint of an attachment to a possibly different link.

Independent of the type of hint used, once a hint is received the host will start to collect a new set of prefixes for the possibly different link, and compare them with the prefixes known from before the hint. If there is one or more common prefixes it is safe to assume that the host is attached to the same link, in which case the prefixes learned after the hint can be merged with the prefixes learned before the hint. But if the sets of prefixes are disjoint, then at some point in time the host will determine that it is attached to a different link. This process starts when the host is powered on and first attaches to a link.

Since each router advertisement message isn't guaranteed to contain all prefixes it is a challenge for a host to attain and retain the complete prefix list, especially when packets can be lost on the link.

The host has to rely on approximate knowledge of the prefix list using RS/ RA exchanges. Just as specified in [1] the host sends an RS (Router Solicitation) message to All-Router multicast address, then waits for the solicited RAs. If there was no packet loss, the host would receive the RAs from all the routers on the link in a few seconds thereby knowing all the prefixes on the link. Taking into account packet loss, the host may perform RS/ RA exchanges multiple times to corroborate the result.

When a hint indicating a possible link change happens, if the host is reasonably sure that its prefix list is complete, it can determine whether it is attached to the same link on the reception of just one RA containing one or more prefixes.

Otherwise, to make matters certain, the host may need at least to wait for more RAs than a single one, or additionally perform multiple RS/ RA exchanges after the hint. After each RS transmission, the host waits for all RAs that would have been triggered by the RS as one aspect of trying harder, and then sending multiple RSs (and waiting for the resulting RAs) is a way to try even harder.

When hints are received frequently, this means that the host might need to track more than two prefix lists at a time. Essentially, each reception of a hint indicates the start of a new prefix list to track, which may or may not turn out to be disjoint from a previously known prefix list.

All tracking of the prefix lists must take the valid lifetime of the prefixes into account, but apart from this limitation there isn't any harm in the host remembering prefix lists from links it has attached to earlier. The prefix list is maintained separately per network interface.

3. DNA based on the complete prefix list

To each link, the set of prefixes is uniquely assigned. Two separate links have two disjoint sets of prefixes. The two prefix lists of two separate links have no element in common.

The identify of a link can be represented by the prefix list if it correctly includes all the assigned prefixes. We denote the complete prefix list as the list of all the prefixes assigned to the link. Each link has its unique complete prefix list. We also say that the prefix list is complete if all the prefixes on the link belong to it.

In case that a host has the complete prefix list, it can properly determine whether it is attached to the same link or not, when it receives a hint that a link change might have occurred.

This section presents a procedure to generate the complete prefix list and a way to detect the link identity change based on the existing prefix list even in the presence of packet losses.

3.1 Complete prefix list generation

To efficiently check for link change, a host always maintains the list of all known prefixes on the link. This procedure of attaining and retaining the complete prefix list is initialized when the host is powered on.

Usually hosts execute and terminate the process of generating the complete prefix list (of a link) at an old attachment point, before handoff process begins. Though the procedure may take some time, that doesn't matter unless the host moves very fast. A host can generate the complete prefix list with reasonable certainty if it remains attached to a link sufficiently long, approximately 10 seconds.

To generate the complete prefix list, presently the host has to rely on approximate knowledge based on RS/ RA exchanges as follows.

First the host sends an RS to All-Router multicast address. Assuming there is no packet loss, every router on the link would receive the

RS and usually reply with an RA containing all the prefixes that the router advertises.

After an RS transmission, the host waits for all RAs that would have been triggered by the RS. There is an upper limit on the delay of the RAs. MIN_DELAY_BETWEEN_RAS (3 Sec) + MAX_RA_DELAY_TIME (0.5 Sec) + network propagation delay is the maximum delay between an RS and the resulting RAs [1]. 4 seconds would be a safe number for the host to wait for the resulting RAs. Assuming no packet loss, within 4 seconds, the host would receive all the RAs and know all the prefixes.

In case of packet loss, things get more complicated. In the above process, there may be a packet loss that results in the generation of the incomplete prefix list, i.e. the prefix list that misses some prefix on the link. To remedy this deficiency, the host may perform multiple RS/ RA exchanges to collect all the assigned prefixes.

After one RS/ RA exchange, to corroborate the completeness of the prefix list, the host may send one or more RSs additionally and wait for the resulting RAs. Each RS/ RA exchange produces the set of the prefixes on the link. These sets may or may not be identical depending on whether there happened a packet loss or not. Assuming no packet loss, these sets would be the same.

The host takes the union of all these sets to generate the prefix list. The more RS/ RA exchange the host performs, the more probable that the resulting prefix list is complete. Section 4 gives the detailed analysis.

By performing multiple (multicast) RS/ RA exchanges and combining the results, the host can reasonably sure that the existing prefix list is complete.

To ascertain whether its existing prefix list is complete or not, the host can set its own policy. The host may take into consideration the packet loss rate of the link and the number of RS/ RA exchanges it performed.

For example, the host may keep track of how may RS/ RA exchanges it performed on this attachment point, and if this is 3 or more it assumes that the resulting prefix list is complete. But if this is only 1 or 2, the host doesn't assume the completeness of the prefix list.

In general, the higher the error rate, the more RS/ RA exchanges are needed to assure the completeness of the prefix list.

It may happen that the host fails to generate the complete prefix list.

The host may not be sure that the prefix list is complete. Or worse, the host may falsely assume the completeness of the prefix list while it is not.

The host may generate either 1) the incomplete prefix list, i.e. the prefix list misses some prefix on the link or 2) the superfluous prefix list, i.e. the prefix list that contains some prefix that is not assigned to the link.

It is noted that 1) and 2) is not exclusive. The host may generate the prefix list that misses some prefix on the link but includes the prefix not assigned to the link.

Severe packet losses during prefix list generation may cause the incomplete prefix list. Or the host may have undergone a link change before finishing the procedure of the complete prefix list generation. Later we will deal with the case that the host can't be sure of the completeness of the prefix list.

Even if the host falsely assumes that the incomplete prefix list is complete, the pain of that assumption is that the host might later think it has moved to a different link when in fact it has not.

In case that a link change happens, even if the host has the incomplete prefix list, it will detect a link change. Hence the incomplete prefix list doesn't cause a connection disruption. But it may cause excessive signaling messages, for example Binding Update messages in [5].

The superfluous prefix list presents a more serious problem.

Without the notification from the link-layer, the host can't perceive that it has changed its attachment point, i.e. it has torn down an old link-layer connection and established a new one.

So while generating the prefix list, without knowing it, the host may be attached to multiple links in turn and accidentally end up with prefixes from multiple links in the prefix list.

Here is an example. The host begins to collect the prefixes on a link. But before the prefix list generation is completed, without its knowledge, the host moves to a new link. Unaware that now it is at the different link, the host keeps collecting prefixes with RS/ RA exchanges to generate the prefix list. This results in the prefix list containing prefixes from two different links. If the host uses this prefix list, it fails to detect a link change.

Since, in the absence of link-up indications, a RA with a disjoint set of prefixes is treated as a hint, the risk for confusion occurs because the host can not tell from the RAs whether they were solicited by the host. (RFC 2461 recommends that solicited RAs be multicast.)

The simplest approach is for the host to assume that any RA received within 4 seconds after sending a RS, assuming no intervening link-up indication, where solicited, that is, the host has not moved to a different link in that time.

In case the host moves fast, this assumption may lead to the superfluous prefix list generation. The protocol is robust against such confusion when a link-up indication is delivered to the IP layer any time the host might have changed it's attachment.

If the host doesn't have any link-layer event notifications [14], for example link up/down indications, it can't decide whether to use the prefixes it receives for collecting information about the "old" link or about the "new" link.

Maybe this is just something that should be stated as an assumption; We may assume that when a host changes its attachment point, it will be notified of the event and can distinguish the RAs arriving from the old attachment point and the RAs from the new attachment point.

Thus we think the prefix-based approach has a stronger assumption here than the Link Identifier-based approach, because the latter can operate reliably without any link-layer event notifications [14].

3.2 Link identity detection

When a host receives a hint that indicates a possible link change, it initiates DNA procedure to determine whether it still remains at the same link or not. At this time, the complete prefix generation may or may not be finished.

First, if the host has finished the complete prefix list generation and can be reasonably sure of its completeness, the receipt of a single RA (with at least one prefix) is enough to detect the identify of the currently attached link.

Assume that, after the hint, the host receives an RA that contains at least one prefix. Either by an RS transmission or by an arrival of an unsolicited RA, the host can get an RA. The host compares the prefixes in the RA with those in the existing prefix list. If the RA contains a prefix that is also a member of the existing prefix list, the host is still at the same link. Otherwise, if none of the prefixes in that RA matches the previously known prefixes, it is at a different link.

It may happen that the host can not be sure that the prefix list is complete. In this case, the host may use another scheme that makes a decision based on multiple solicited RAs, instead of one RA.

Suppose that before finishing the prefix list generation, the host receives the hint that indicates a possible link change. Then the host can't assume the completeness of the prefix list.

The host can generate another (complete) prefix list to compensate the uncertainty of the existing prefix list. After the hint, it performs one or more RS/ RA exchanges additionally to collect all the prefixes on the currently attached link. With the resulting prefixes, the host generates the second prefix list.

Then the host compares two prefix lists and if the lists are disjoint, i.e. have no prefix in common, it assumes that a link change has occurred. Usually this is done at a new attachment point.

For example, assume that the host keeps track of how many RS/ RA exchanges it performed at a link. If this is 3 or more, the host assumes that it have seen all the prefixes. Suppose that the host has done a single RS/RA exchange, then it receives a link up notification that causes it to initiate the DNA procedure. For a better judgment, the host performs new RS/RA exchanges. If the host tracks the list of prefixes received (from all received RAs) before the link up notification and after the one, it can compare the lists to check for link change. In case that the lists are disjoint, the host can assume it has moved.

The difference with the previous case is that the host doesn't use the first RA received after the hint to make its decision. Instead it waits for the timeout and then use the received set of prefixes to determine whether a link change has occurred. Though slower, this is more robust.

In summary, first a host makes the complete prefix list. When a hint occurs, if the host decides that the prefix list is complete, it will check for link change with just one RA (with a prefix). Otherwise, in case that the host can't be so sure, it will perform additional RS/ RA exchanges to corroborate the decision.

When the host is sure that the prefix list is complete, a false movement assumption may happen due to renumbering when a new prefix is introduced in RAs. We may solve the renumbering problem with minor modification like below.

1) When a router starts advertising a new prefix, for the time being, every time the router advertises a new prefix in an RA, it includes at least one old prefix in the same RA. The old prefix assures that the host doesn't falsely assume a link change because of a new prefix. After a while, hosts will recognize the new prefix as the one assigned to the current link and update its prefix list.

2) To make matters more certain, we may mandate hosts to assume a link change only after a new link-layer connection. It's reasonable to assume that a new link-layer connection precedes a link change.

In this way, we may provide a fast and robust solution. If a host can make the complete prefix list with certainty, it can check for link change fast. Otherwise, it can fall back on a slow but robust scheme. It is up to the host to decide which scheme to use.

4. Protocol Specification

This section provides the actual specification for a host implementing this draft. For generality the specification assumes that the host retains multiple (an unbounded set) of prefix lists until the information times out, while an actual implementation would limit the number of sets maintained.

This description assumes that the link layer driver provides a 'link up' indication when the host might have moved to a different link.

4.1 Conceptual data structures

This section describes a conceptual model of one possible data structure organization that hosts will maintain for the purposes of DNA. The described organization is provided to facilitate the explanation of how the CPL protocol should behave. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

The basic conceptual data type for the protocol is the Candidate Link object. This is an object which contains all the information learned from router advertisement messages that are known to belong to a single link. In particular, this includes

- The prefixes learned from the prefix information options, the A/L bits and their valid and preferred lifetimes.
- The default routers and their lifetimes.
- Any other option content such as the MTU etc.

The lifetimes for the prefixes and default routers in the Candidate Link objects 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.

For each interface, the host maintains a notion of its Current Candidate Link (CCL) object. As we will see below, this might actually be different than the prefix list and default router lists maintained by Neighbor Discovery when the host is in the process of determining whether it has attached to a different link or not.

In addition, the host maintains previous Candidate Link objects. It is TBD whether these should be per interface, shared across interfaces of the same L2 type (e.g., all Ethernet interfaces), or be shared across all interfaces of the host. It isn't currently clear if there are any security issues associated with sharing this information across different interfaces. The previous Candidate Link objects can be found by knowing at least one prefix that is part of the object.

The operations on Candidate Link objects is to create a new one, discard one, and merge two of them together. The issues with merging are discussed in the next section.

For each interface, the host maintains the last time a valid router advertisement was received (called time_last_RA_received in this document), which actually ignores RAs without prefix options, and the last time a link UP indication was received from the link layer on the host (called time_last_linkUP_received in this document). Together these two conceptual variables serve to identify when a RA containing disjoint prefixes can't be due to being attached to a new link, because there was no linkUP indication.

4.2 Merging Candidate Link objects

When a host has been collecting information about a potentially different link in its Current Candidate Link object, and it discovers that it is in fact the same link as another Candidate Link object, then it needs to merge the information between the two objects. Since the CCL contains the most recent information, any information contained in it will override the information in the old Candidate Link, for example the remaining lifetimes for the prefixes. When the two objects contain different pieces of information, for instance different prefixes or default routers, the union of these are used in the resulting merged object.

4.3 Timer handling and Garbage Collection

As stated about the lifetimes for the prefixes and default routers in each Candidate Link object must be decremented in real time. When a prefix' valid lifetime has expired, the prefix should be removed from its object. Likewise, when a default router lifetime has expired, it should be removed from its object. When a Candidate Link object contains neither any prefixes nor any default routers, the object, including additional information such as MTU, should be discarded.

There is nothing to prevent a host from garbage collecting Candidate Link objects before their expire. However, for performance reason a host must be able to retain at least two of them at any given time.

4.4 Receiving link UP indications

When the host receives a link UP indication from its link layer, the only action is to set time_last_linkUP_received to the current time.

4.5 Receiving valid router advertisements

When a host receives a valid router advertisement message (with the validity checks specified in [1]) it performs the following processing in addition to the processing specified in [1] and [2]

If the valid router advertisement does not contain any prefix information options, then not further processing is performed. Note that not even the time_last_RA_received is updated.

If time_last_RA_received is more recent than time_last_linkUP_received, then the host could not possibly have moved to a different link. Hence the only action needed for DNA is to update the current Candidate Link object with the information in the RA, and set time_last_RA_received to the current time.

The remaining processing occurs when a linkUP was received and no RA containing prefix options has yet been received. This is the case when the host needs to perform comparisons of the prefix sets in its Candidate Link objects and the prefix set in the router advertisement. In all these cases time_last_RA_received is set to the current time.

Should the received RA contain at least one prefix which is in the prefix list in the CCL, then the host is still attached to the same link, and just needs to update the CCL with any new information in the RA.

Otherwise, if received RA contains one or more prefixes which is part of the prefix list in some retained Candidate Link object, then the host has most likely moved to that link. As a result the host will retain the content of the CCL for future matching, but switch the CCL to be that matching object. The now new CCL should be updated based on the information in the RA. Then the DNA module informs the Neighbor Discovery module to replace the old information with the information in the new CCL.

It is possible that the above comparison will result in matching multiple Candidate Link objects. For example, if the RA contains the prefixes P1 and P2, and there is one Candidate Link object with P1 and P3 and other Candidate Link object with P2 and P4. This should not happen during normal operation, but if links have been renumbered or physical merged into one before the lifetimes in the Candidate Link objects expired, then the host could observe this. The most sensible action would be for the host to merge all such matching Candidate Link objects together with the information in the receive RA and make this the new CCL. Doing this merging correctly probably requires that each Candidate Link object contains the time it was last updated by a RA, so that more recent information can override older information. Note that this case is one reason one needs to be concerned about security issues when Candidate Link objects are shared across multiple interfaces.

Above the easy case of determining being on the same link has been handled. The remaining cases show up when the first RA after a link UP indication contains prefixes that are new to the host. In this case it isn't obvious whether the host has moved or not. Thus the

host create a new Candidate Link object which it initializes with the information in the received RA, and makes this object the CCL. However, it does not yet treat this as a new link; it is merely a candidate.

Then the host waits for more router advertisement messages. At a minimum it waits for 4 seconds, that is, it starts a timer and router advertisements that are received during that time interval are processed as specified above. This processing might result in finding a prefix in common with a Candidate Link object in which case the host knows whether and to which link it has moved. But should the 4 seconds expire without any common prefix, then it will conclude that it has moved to a new link and inform the rest of the host of the movement. Note that the arrival of a new link UP indication during the 4 second timer must prevent the timer from firing. In this case the host might yet again have moved so it is necessary to restart the process of inspecting the router advertisements.

Subject to local policy and perhaps the hosts knowledge of the packet loss characteristics of the interface or type of L2 technology, the host can try harder than just waiting for 4 seconds. It is allowed to multicast up to MAX_RTR_SOLICITATIONS [1] router solicitation messages spaced RTR_SOLICITATION_INTERVAL apart. In the most conservative approach this means a 12 second delay until the host will declare that is has moved to a new link. Just as above, this process should be terminated should a new link UP indication arrive during the 12 seconds.

4.6 Changing the link in Neighbor Discovery

When DNA detects that it has moved to a different link this needs to cause Neighbor Discovery, Address autoconfiguration, and DHCPv6 to take some action. While the full implications are outside of the scope of this document, here is what we know about the impact on Neighbor Discovery.

Everything learned from the router advertisements on the interface should be discarded, such as the default router list and the on-link prefix list. Furthermore, all neighbor cache entries, in particular redirects, need to be discarded. Finally the information in the Current Candidate Link object is used to create a new default router list and on-link prefix list.

5. Performance Analysis

In this section, we compute the probability that a host fails to generate the complete prefix list and consequently assumes a link change erroneously.

Suppose, in a link, there are N routers, R[1], R[2],...., R[N].

Each R[i] advertises the Router Advertisement RA[i] with the prefix P[i].

It is the worst case that each router advertises the different prefix. It is necessary to receive all the RA[i] to generate the complete prefix list.

We assume there is a host, H, and when the host sends a Router Solicitation, let P be the probability that it fails to receive a RA[i] because of a RA loss. For the simplicity, we disregard RS losses.

So when the sends a Router Solicitation, the probability that it will receive all RA[i] is (1-P)^N.

Let's assume the host performs RS/ RA exchange T times, 1,2,..,T.

Let S[k] be the set of all RAs which the host H successfully receives at k-th RS/RA exchange. The probability that R[i] belongs to S[k] is (1-P).

Let PL[k] be the set of prefixes which are made from S[k], i.e. the set of P[j] such that RA[j] belongs to S[k]. Obviously, the probability that P[i] belongs to PL[k] is also (1-P).

Let PL be the union of all PL[k], from k=1 to k=T. PL is the prefix list made from performing RS/ RA exchange T times.

1) The probability of the complete prefix list generation

First the probability that P[i] belongs to PL is 1-P^T. The probability that the prefix list PL is complete is (1-P^T)^N.

For example, assume the error rate is 1 % and there are 3 routers in a link, then, with 2 RS/ RA exchanges, the probability of generating an accurate Complete Prefix List is roughly 99.97 %.

At this point, assume that the host H receives a hint that a link change might have happened and consequently initiates the procedure of checking a link change.

2) The false DNA probability if the host checks for link change with one RA.

Assume one RA, whether solicited or unsolicited arrives. If the host H makes a decision based solely on the RA and the prefix list, the probability that it falsely assume a link change is P^T.

For example, given the error rate is 1%, with 2 RS/ RA exchanges, the probability of false movement detection is 1/ 10000.

3) The false DNA probability if the host checks for link change with additional RS/ RA exchanges.

Instead of depending on the single RA, the host H performs additional RS/ RA exchange U times, 1,2...U. Then the probability that H falsely assumes a link change is

[P^T + P^U - P^(T+U)]^N.

For example, given the error rate is 1 % and there are 3 routers in a link, if the host H performs 2 RS/ RA exchanges before the hint and 1 RS/ RA exchange after one, the probability of false movement detection is roughly 1/1000000.

In the above formula, the result goes to P^(U*N) as T goes infinity. The term P^(U*N) results from the probability that the host receives no RA during U RS/ RA exchange after the hint. To see that it still remains at the same link, a host needs to receive at least one RA.

We think it is reasonable to assume that the RS will be retransmitted until at least one RA arrives. If we take a one more assumption that the host receives at least one RA, the probability will be

[[P^T + P^U - P(T+U)]^N - P^(U*N)]/ [1- P^(U*N)]

The above converges to zero as T approaches infinity.

6. IANA Considerations

No new message formats or services are defined in this document.

7. Security Considerations

Because DNA schemes are 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 [6] to secure Neighbor Discovery are important in achieving reliable detection of network attachment. DNA schemes SHOULD incorporate the solutions developed in IETF SEND WG if available, where assessment indicates such procedures are required.

The threats specific to DNA are that an attacker might fool a node to detect attachment to a different link when it is in fact still attached to the same link, and conversely, the attacker might fool a node to not detect attachment to a new link.

The first form of attack is not very serious, since at worst it would imply some additional higher-level signaling to register a new (care-of) address. The second form of attack can be more serious, especially if the attacker can prevent a host from detecting a new link. The protocol as specified would require an attacker to be on-link and be authenticated and authorized to send router advertisements when Secure Neighbor Discovery [6] is in use. However, even without SEND, an attacker would need to send router advertisements containing the prefixes to which it wants the host to be unable to detect movement. This can be done for a small number of prefixes, but it isn't possible for the attacker to completely disable DNA for all possible prefixes on other links.

8. Examples

This section contains some example packet flows showing the operation of prefix based DNA.

8.1 Example with link-up indication

Assume the host has seen no link-up indication for a long time and that it has the prefixes P1, P2, and P3 in its prefix list for the interface.

The IP layer receives a link-up indication. This hint makes it multicast a Router Solicitation and start collecting the received prefixes in a new list of prefixes.

The host receives a Router Advertisement containing no prefixes. This has no effect on the algorithm contained in this specification.

The host receives a Router Advertisement containing only the prefix P4. This could be due to being attached to a different link or that there is a new prefix on the existing link which is not announced in RAs together with other prefixes, and a spurious hint. In this example the host decides to wait for another RA before deciding.

One second later a router advertisement arrives which contains P1 and P2. As a result the "new" prefix list has P1, P2, and P4 hence is not disjoint from the "old" prefix list with P1, P2, and P3. Thus the host concludes it has not moved to a different link and its prefix list is now P1, P2, P3, and P4.

Some time later a new link-up indication is received by the IP layer. Triggers sending a RS.

A Router Advertisement containing P5 and P6 is received by the host. Based on some heuristic (the assumed frequency of prefixes being added to an existing link) this time the host decides that it is on a new link.

One second later a Router advertisement with prefix P7 is received.

Thus the prefix list now contains P5, P6, and P7.

8.2 Example without link-up indication

Assume the host has collected the prefixes P1, P2, and P3 in its prefix list for the interface.

The host receives a Router Advertisement containing only prefix P4. The fact that P4 is disjoint from the prefix list makes this be treated as a hint. This hint makes the host multicast a Router Solicitation and start collecting the received prefixes in a new list of prefixes, which is initially set to contain P4.

The host receives a Router Advertisement containing no prefixes. This has no effect on the algorithm contained in this specification.

The host receives a Router Advertisement containing only the prefix P4. This could be due to being attached to a different link or that there is a new prefix on the existing link which is not announced in RAs together with other prefixes. In this example the host decides to wait for another RA before deciding.

One second later a router advertisement arrives which contains P1 and P2. As a result the "new" prefix list has P1, P2, and P4 hence is not disjoint from the "old" prefix list with P1, P2, and P3. Thus the host concludes it has not moved to a different link and its prefix list is now P1, P2, P3, and P4.

Some time later the host receives a Router Advertisement containing prefix P7. This is treated as a hint since it is not part of the current set of prefixes. Triggers sending a RS and initializing the new prefix list to P7.

A Router Advertisement containing P5 and P6 is received by the host. This is disjoint with both of the previous prefix lists, thus the host might be attached to a 3rd link after very briefly being attached to the link with prefix P7.The host decides to wait for more RAs.

One second later a Router advertisement with prefix P7 is received. It still isn't certain whether P5, P6, and P7 are assigned to the same link (and without a link-up indication such uncertainties do exist).

A millisecond later a Router Advertisement with prefixes P6 and P7 is received. Now the host knows that P5,P6, and P7 are assigned to the same link.

Four seconds after the RS was sent and no RA containing P1, P2, P3, or P4 has been received the host can conclude with high probability that it is no longer attached to the link which had those prefixes.

9. References

9.1 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., "Detecting Network Attachment in IPv6 Goals", draft-ietf-dna-goals-03 (work in progress), October 2004.

9.2 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-jinchoi-dna-soln-frame-00 (work in progress), July 2004.

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

[10] Pentland, B., "Router Advertisement Link Identification for Mobile IPv6 Movement Detection",

draft-pentland-mobileip-linkid-02 (work in progress), July 2004.

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

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

[13] Daley, G., "Deterministic Fast Router Advertisement

Configuration", draft-daley-dna-det-fastra-00 (work in progress), July 2004.

[14] Yegin, A., "Link-layer Event Notifications for Detecting Network Attachments", draft-ietf-dna-link-information-00 (work in progress), September 2004.

Authors' Addresses

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

Erik Nordmark
Sun Microsystems
17 Network Circle
Menlo Park, CA 94043
USA

Phone: +1 650 786 2921
EMail: erik.nordmark@sun.com

Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Disclaimer of Validity

Copyright Statement

Acknowledgment

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