Proactive DAD: An L2-assisted Fast Address Acquisition Strategy for Mobile IPv6 Networks

Chien-Chao Tseng
Department of Computer Science and Information Engineering
National Chiao-Tung University
Hsinchu, Taiwan 300

Yung-Chang Wong[1]
Department of Computer Science and Information Engineering
Providence University
Taichung, Taiwan 433

Li-Hsing Yen
Department of Computer Science and Information Engineering
Chung Hua University
Hsinchu, Taiwan 300

Kai-Cheng Hsu
Department of Computer Science and Information Engineering
National Chiao-Tung University
Hsinchu, Taiwan 300

[1] Contact author: Yung-Chang Wong, E-mail: ycwong@pu.edu.tw

Abstract

In Mobile IPv6 (MIPv6) networks, a mobile node (MN) can acquire an IPv6 address through stateless or stateful configuration at new points of network attachment. In either way, duplicate address detection (DAD) procedure is needed to confirm the uniqueness of the address. We propose a proactive design to speedup the DAD procedure. Experimental results show that the proposed method reduces the DAD delay significantly.

Key words: Mobile IPv6, fast handoff, address acquisition, DAD.

1. Introduction

Recently, demand for wireless Internet access services is preeminent since Wi-Fi compatible products have become a standard component in mobile devices. Mobile IPv6 (MIPv6) [1] was designed to allow mobile nodes (MNs) to be reachable and maintain ongoing connections while changing their points of attachment to the Internet.

The process of changing points of attachment to the Internet is referred to as *handoff*, which may involve activities at various layers. In MIPv6, a handoff consists of three phases: link-change detection, address acquisition, and binding update. Link-change detection concerns how an MN realizes the need to initiate the process of acquiring a new IP configuration. The IETF Detecting Network Attachment (DNA) working group has been chartered to define an improved scheme for determining whether a link change has occurred [2]. After the link change, address acquisition configures a valid IP address to be used on the new network domain. Binding update is to inform the network of the MN's new IP address. Among these three phases, address configuration is most time-consuming. This is due to the lengthy execution of duplicate address detection (DAD). This article investigates the acceleration of DAD procedure.

An MN can acquire an IPv6 address through stateless [3] or stateful [4] configuration. In either way, DAD procedure is needed to confirm the uniqueness of the address in the new network domain. The default execution time for DAD is at least one second. During that interval, active connections, if any, are all suspended. This is unacceptable for most real-time applications.

Some work has been done on improving the DAD latency. *Optimistic DAD* (O-DAD) [5] allows the use of an address before DAD has checked its uniqueness, which is beneficial if the probability of address collision is low. If the DAD procedure reports later that this address is already in use, the MN must immediately stop using the address and deconfigure it. This may incur some penalty to the MN, in the form of breaking ongoing connections, and some penalty to the rightful owner of the address, since it will receive misdirected packets. In *Advance DAD* (A-DAD) [6], a router maintains a pool of IP addresses that have been confirmed unique on a network domain. These addresses can therefore be safely allocated to arriving MNs without being checked again by DAD. The performance of A-DAD depends on the size of the address pool: it must be sufficiently large to accommodate potential MNs while not wasting too much address space. Besides, the router must maintain hard states.

This article aims to eliminate the handoff latency caused by DAD. The basic idea is to conduct DAD procedure prior to or in parallel with the associated Layer 3 (L3) handoff. This is possible because we can predict the new network domain the MN is entering with the help of topology information and layer 2 signals. Unlike O-DAD, this approach guarantees the uniqueness of IP addresses before their usages. Compared with A-DAD, this approach does not reserve IP addresses and thus has a better utilization of address space. Furthermore, a router in our approach needs only maintain soft state.

The rest of this article is organized as follows. Section 2 introduces related work on fast address acquisition. Then, in Section 3, we present the proactive DAD procedure. Experimental results are presented in Section 4 and Section 5 concludes this article.

2. Related work

A DAD procedure verifies the uniqueness of an address before its usage. An MN initiates a DAD procedure by sending a NEIGHBOR SOLICITATION (*NeighborSol*) message destined for the address being checked. If the MN receives a defending NEIGHBOR ADVERTISEMENT (*NeighborAdv*) corresponding to the issued *NeighborSol* within *RetransTimer* ms, the MN deconfigures the address in question. Otherwise, the MN issues another *NeighborSol*. This solicit-and-wait process may repeat at most *DupAddrDetectTransmits* times. With the default values of *RetransTimer* and *DupAddrDetectTransmits*, which are 1000 [7] and 1 [3], respectively, the DAD execution time is at least one second. We could set *RetransTimer* to a smaller value to speedup the DAD procedure. However, doing so also increases the probability of missing defending *NeighborAdv*'s.

In A-DAD [6], an access router (AR) uses standard DAD to verify the uniqueness of IP addresses before these addresses are allocated to MNs. All IP addresses that have been proven unique are maintained in an address pool. These addresses are considered reserved. Some device, however, may still acquire an already-reserved address by means other than A-DAD. When this happens, the AR will receive a *NeighborSol* that is destined for some address in the pool. The AR must silently delete the address from its address pool to avoid address collision.

With A-DAD, an MN obtains a duplication-free address as a part of the standard router discovery process conducted when the MN enters a new subnet. Not knowing the existence of any router, the MN multicasts ROUTER SOLICITATION (*RouterSol*) to all ARs for essential router information. A-DAD extends *RouterSol* to include an option that notifies AR of the request of a duplication-free CoA. Attached with this option are MN's previous CoA and link-layer address. When such a *RouterSol* is received, an AR performs the following procedure:

Step 1. Select and remove an address from its address pool.
Step 2. Create a neighbor cache entry that associates the selected address with the MN's link-layer address.
Step 3. Create a host route entry using the MN's previous CoA and link-layer address.
Step 4. Create a ROUTER ADVERTISEMENT (*RouterAdv*) with NCoA Reply option enabled. The option includes the address selected from the pool.
Step 5. Send the *RouterAdv* directly to the MN's previous CoA using the host route entry.
Step 6. Delete the host route entry.

When the MN receives a *RouterAdv* with NCoA Reply option set, it takes the address specified in the option field as its new CoA. The MN thereby acquires a duplication-free address without performing DAD during the handoff.

An AR in A-DAD needs to maintain hard states. That is, A-DAD no longer works if the address pool kept in a temporary storage is missing, which may occur when the AR reboots.

As mentioned, an L2 handoff may or may not cause an L3 handoff. It depends on whether the handoff involves changing network domain. There are no standard ways to detect the need for an L3 handoff. A possible indication of domain change is the expiration of the last received router advertisements. However, the lifetime of such advertisements is typically in the order of minutes, which makes this method not a timely approach. Some researchers [8] have proposed to exploit topology information, the association between access points (APs) and ARs. With topology information, an MN can determine if the new and old APs are in the same network domain. Consequently, the need for an L3 handoff can be detected as soon as the new AP is known. In this work, we take the same idea of detecting movement based on topology information. However, in [8] a stateful address configuration is assumed, while we focus on stateless address configuration.

3. Proactive DAD (P-DAD)

In this section, we first present the network architecture with assumptions made by the proposed scheme. The protocol is then detailed.

3.1 Anticipated Network Architecture

Figure 1 shows an example of an IPv6 network. Our approach assumes the deployment of a server called Regional Information Point (RIP) in each domain. An RIP maintains a Mobile-node Attachment Point (MAP) table which stores connected-to relationship between ARs and APs in its serving domain. Each entry of the MAP table is a tuple $\langle p,q,f \rangle$, where p is the BSSID of an AP, q is the IP address of the AR to which p connects, and f is the prefix advertised by q (used for stateless address auto-configuration). Example MAPs are shown in Table 1. RIP can be implemented either as a standalone server or as an add-on software module in ARs. The information contained in MAP could be manually configured, since such information rarely changed in most cases.

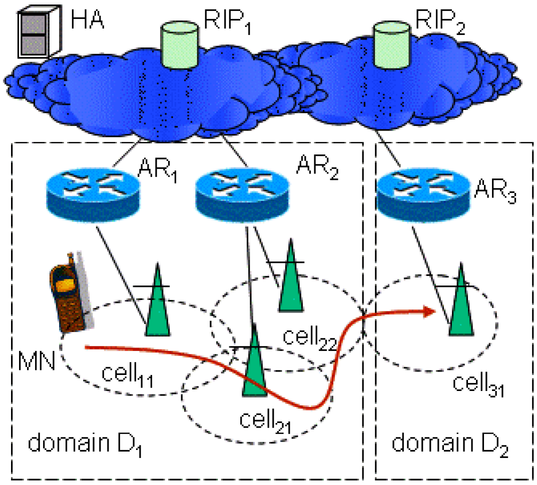

Figure 1. An IPv6 network with Regional Information Points.

AP's BSSID	AR's IP address	Advertised prefix
(AP$_{11}$) 00-01-4A-C3-07-01	(AR$_1$) 2001:0E10:6440:0001::1	$prefix_1$
(AP$_{21}$) 00-01-4A-C3-07-02	(AR$_2$) 2001:0E10:6440:0002::1	$Prefix_2$
(AP$_{22}$) 00-01-4A-C3-07-03	(AR$_2$) 2001:0E10:6440:0002::1	$Prefix_2$

(a)

AP's BSSID	AR's IP address	Advertised prefix
(AP$_{31}$) 00-01-4A-C3-07-04	(AR$_3$) 2001:0E10:6440:0003::1	$Prefix_3$

(b)

Table 1. Example MAP tables: before exchange (a) on RIP_1 (b) on RIP_2.

Every RIP periodically exchanges its MAP table with neighbor RIPs. Consider Figure 1 as an example. RIP_1 learns the topology of its neighbor domain by exchanging its MAP table with that on RIP_2. The resulting MAP table on RIP_1 (RIP_2) is illustrated in Table 2. An MN, on entering a new network domain, requests a copy of the MAP from the new serving RIP. The MN thus knows all the AP-AR associations and network prefixes in the current and all surrounding network domains.

AP's BSSID	AR's IP address	Advertised prefix
(AP_{11}) 00-01-4A-C3-07-01	(AR_1) 2001:0E10:6440:0001::1	$prefix_1$
(AP_{21}) 00-01-4A-C3-07-02	(AR_2) 2001:0E10:6440:0002::1	$Prefix_2$
(AP_{22}) 00-01-4A-C3-07-03	(AR_2) 2001:0E10:6440:0002::1	$Prefix_2$
(AP_{31}) 00-01-4A-C3-07-04	(AR_3) 2001:0E10:6440:0003::1	$Prefix_3$

Table 2. Example MAP table: after exchange (both on RIP_1 and on RIP_2).

Each AR is required to maintain a *registration cache*. For each MN that has its IP address verified via P-DAD, an entry in the registration cache maintains the following data:

- The home address.
- The pre-allocated new care-of-address (CoA).
- The lifetime value of the CoA.

If a node tentatively configures one of these pre-allocated CoAs and attempts to test it using DAD, the AR responds to the DAD message, indicating that the address is already assigned.

3.2 The Protocol

An MN obtains the serving RIP's IP address as a part of the standard router discovery process. We assumes that all ARs know the IP address of the RIP that serve them. When receiving a *RouterSol* from an MN, the AR returns a *RouterAdv* with an optional field that notifies the MN of the serving RIP's IP address. After locating the serving RIP, the MN can then request an MAP table from the RIP.

With the information contained in MAP, an MN can determine whether it needs to prepare an L3 handoff when an L2 handoff is about to occur. After the MN has discovered the next candidate APs for an L2 handoff by some next AP discovery mechanism [9], the MN can use the connect-to information in MAP to determine the ARs corresponding to the candidate APs. If the ARs are different from the current serving AR, the MN realizes the need of an L3 handoff before it actually conducts an L2 handoff. Therefore the L2 information contained in MAP and the next candidate APs in L2 handoff together can assist the MN to determine whether an L3 handoff is imminent and starts DAD proactively.

Algorithm I shows how an MN conducts an L3 handoff.

Algorithm I (CoA pre-allocation)

Step 1. Before switching to the next AP, the MN extracts the prefix associated with the new AR from MAP table, and generates a tentative CoA *addr* based on the prefix. Then the MN communicates with the new AR for uniqueness test through an *CoA_preAllocate Request* message with parameter *addr*.

Step 2. Upon reception of *CoA_preAllocate Request* (*addr*), the new AR checks its registration cache and, if necessary, performs a standard DAD to check if *addr* is unique. The result is reported back to the MN via message *CoA_preAllocate Reply* (*U*), where the *U*-bit is set if the uniqueness check is passed and unset otherwise. In the former case, the AR stores the pre-allocated CoA in its registration cache.

Step 3. If the MN receives *CoA_preAllocate Reply* with the *U*-bit set, the MN sends an *CoA_activation Request* message to the new AR to activate the pre-allocated CoA after associating with the new AP. If the *U*-bit is not set or the MN receives no reply, the MN forms a CoA by means of the standard stateless address auto-configuration procedure, and goes to Step 5.

Step 4. Upon receipt of the *CoA_activation Request* message, the new AR removes *addr* from the registration cache, and acknowledges the MN through an *CoA_activation Reply* message. If the MN does not receive *CoA_activation Reply* in time, it performs a standard stateless address configuration procedure.

Step 5. MN informs HA of its current location through a *Binding Update* message.

Figure 2 illustrates the message flow of Algorithm I.

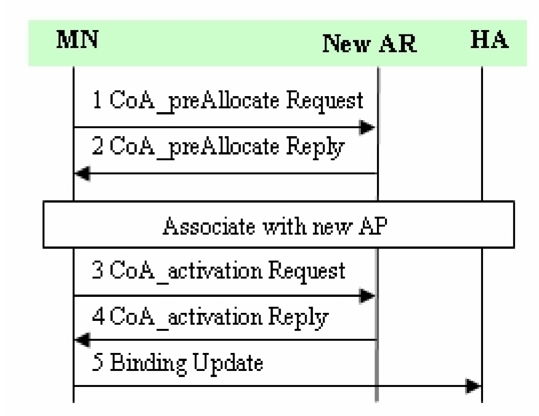

Figure 2. CoA pre-allocation.

In contrast to A-DAD, an AR in the proposed protocol need only maintain soft state. That is, a crashed or male-functional AR only causes MNs to perform standard stateless address configurations.

4. Performance Evaluation

We conducted experiments to measure L3 handoff delay and lost packets caused by handoff for standard DAD, A-DAD, and P-DAD.

Figure 3 shows the experimental setup. A PC server with Linux Kernel 2.6 and IPv6 Router Advertisement Daemon[2] (*radvd*) was used as an IPv6 router. The *radvd* sends *RouterAdv* messages to local LANs periodically and when requested by a node sending a *RouterSol* message. These messages are required for IPv6 stateless address auto-configuration (*SAA*). The *dhcpv6*[3], which was originally developed by a project at Sourceforge, was used as a DHCP implementation. The MN was equipped with two identical Intersil prism2-based IEEE 802.11b wireless interfaces, and was located in a place where it could associate with either AP1 or AP2. In each experiment, a corresponding node (CN) generates packets at a constant rate (one per 20 ms). A sequence of packets was lost during the handoff period. The time t_1 when the last packet was received before a handoff, and the time t_2 when the first packet was received after the handoff, were both recorded. The handoff delay was measured as t_2-t_1. The following procedure measures L3 handoff delay.

1. Before handoff, associate the MN's interface 1 with AP1. Configure interface 1 through standard SAA.

[2] *radvd* ver. 0.8, http://v6web.litech.org/radvd
[3] *dhcpv6* ver. 0.85, http://dhcpv6.sourceforge.net

2. Start generating and transmitting packets.
3. Detach the CoA of interface 1. Directly associate the MN's interface 2 with AP2.
4. Configure a new CoA for interface 2 through standard SAA.
5. Perform Mobile IPv6 binding update.

This procedure did not consider the erratic layer-2 handoff delay. Step 3 emulated breaking the link to AP1. Because Step 4 was carried out immediately after Step 3, no move detection delay occurred.

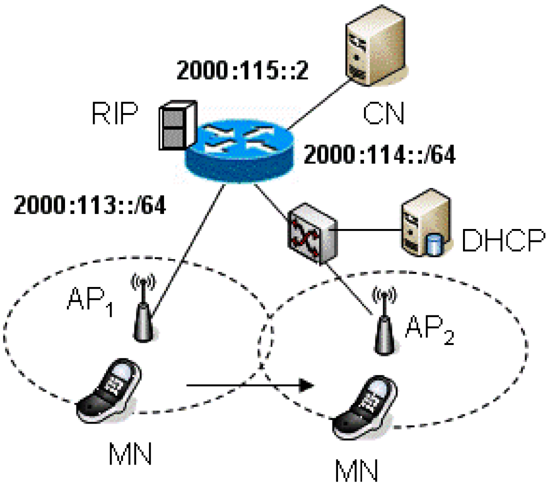

Figure 3. Experimental Setup.

Step 4 of the above procedure was modified slightly to measure the L3 handoff delay for SAA with A-DAD and SAA with P-DAD, respectively. For SAA with A-DAD, a new CoA was configured for interface 2 in Step 4 through a DHCP server; while for SAA with P-DAD, a new CoA was configured right after the execution of Step 4. Table 3 summarizes the empirical results, where each value was measured based on ten experimental runs.

Metrics	Standard stateless address configuration	Stateless address Configuration with A-DAD	Stateless address Configuration with P-DAD
Handoff delay	Avg. 1419.2 ms Std. 906.9	Avg. 83.6 ms Std. 16.07	Avg. 48.4 ms Std. 11.6
Number of	Avg. 70	Avg. 2.5	Avg. 1.4

lost packets	Std. 45.2	Std. 0.7	Std. 0.5

Table 3. Means and standard deviations of L3 handoff delay and the number of lost packets.

According to Table 3, the original L3 handoff delay was larger than 1400 ms. Such delay is unacceptable for VoIP applications. The handoff delay for A-DAD was around 83 ms. Using P-DAD, the handoff delay dropped to around 48 ms with low variation, which should meet the delay requirement of time-critical applications. Furthermore, the number of lost packets in original L3 handoff was 70 packets. This number fell to 2.5 and 1.4 packets when A-DAD and P-DAD was used, respectively.

5. Conclusion

We have reviewed several techniques that aim to reduce the overhead introduced by duplicate address detection during Mobile IPv6 handoff. This article proposes a proactive design, *named P-DAD*, to speedup the detection. The experimental results show that the resulting handoff delay meets the delay requirement of VoIP applications if the MN can perform address pre-configuration.

Acknowledgement

This work was supported by the National Science Council under Grands NSC 94-2219-E-009-006, NSC 94-2745-E-126-003-URD, and NSC 94-2213-E-216-001.

References

1. D. Johnson, C. Perkins, and J. Arkko, ``Mobility Support in IPv6," RFC 3775, June 2004.
2. J.H. Cho and G. Daley, ``Goals of Detecting Network Attachment in IPv6," RFC 4135, Aug. 2005.
3. S. Thomson and T. Narten, ``IPv6 Stateless Address Autoconfiguration," RFC 2462, Dec. 1998.
4. R. Droms, Ed., ``Dynamic Host Configuraion Protocol for IPv6 (DHCPv6)," RFC 3315, July 2003.
5. N. Moore, ``Optimistic Duplicate Address Detection for IPv6," Internet-Draft, draft-ietf-ipv6-optimistic-dad-06.txt, Sep. 2005.
6. Y.H. Han, S.H. Hwang, and H. Jang, ``Design and Evaluation of an Address Configuration and Confirmation Scheme for IPv6 Mobility Support," in

Proceedings of 2004 IEEE Wireless Communications and Networking Conference, p. 1270-1275, March 2004.

7. T. Narten, E. Nordmark, and W. Simpson, ``Neighbor Discovery for IP version 6 (IPv6)," RFC 2461, Dec. 1998.
8. C.C. Tseng, L.H. Yen, H.H. Chang, and K.C. Hsu, ``Topology-aided Cross-layer Fast Handoff Designs for IEEE 802.11/Mobile IP Environments," *IEEE Communications*, vol. 43, no. 12, pp. 156-163, Dec. 2005.
9. C.C. Tseng, K.H. Chi, M.D. Hsieh, and H.H. Chang, ``Location-based Fast Handoff for 802.11 Networks," *IEEE Communications Letters*, vol. 9, no. 4, pp. 304-306, April 2005.