Proactive DAD: A Fast Address-Acquisition Strategy for Mobile IPv6 Networks

The increasing number of Wi-Fi-compatible mobile devices highlights various wireless access challenges, including the need for smooth hand offs between Internet attachment points in mobile IPv6 networks. To confirm address uniqueness in a new domain, mobile nodes must run duplicate address detection (DAD), which is a time-consuming process. The Proactive DAD approach uses topology information and layer-2 signals to predict the new network domains prior to or in parallel with layer-3 hand offs. Experimental results show that P-DAD can significantly reduce both hand-off latency and packet loss.

Chien-Chao Tseng
National Chiao-Tung University

Yung-Chang Wong
Providence University

Li-Hsing Yen
Chung Hua University

Kai-Cheng Hsu
National Chiao-Tung University

As Wi-Fi compatibility becomes standard in mobile devices, demand for wireless Internet access services is increasing dramatically. Developers designed Mobile IPv6 (MIPv6)[1] to let mobile nodes remain reachable and maintain ongoing connections while changing their points of Internet attachment. This hand-off process can involve activities at various layers. In MIPv6, hand off consists of three phases:

- the *link-change detection* phase,[2] in which the mobile node realizes the need to acquire a new IP configuration;
- the *address acquisition* phase, in which the node configures a valid IP address to use on the new network domain; and
- the *binding update* phase, in which the node informs the network of its new IP address.

Of these three phases, address acquisition is the most time-consuming because it requires duplicate address detection (DAD). A mobile node can acquire an IPv6 address through stateless[3] or stateful[4] configuration; either way, a DAD procedure must confirm address uniqueness in the new domain. DAD's default execution time is at least one second. During that interval, all active connections are suspended, which is unacceptable for most real-time applications.

Although techniques such as Optimistic DAD (O-DAD)[5] and Advanced DAD (A-DAD)[6] have improved this latency, both approaches have limitations. With our Proactive DAD approach, our goal is

 Published by the IEEE Computer Society 1089-7801/06/$20.00 © 2006 IEEE

to address those limitations and eliminate hand-off latency by using topology information and layer-2 signals to predict the new network domains prior to or in parallel with MIPv6 hand offs. Our experimental testing shows that P-DAD can significantly reduce DAD hand-off latency. We describe our study results here, following a discussion of existing approaches and an overview of how P-DAD works.

Fast Address-Acquisition Methods

Mobile nodes initiate DAD procedures by sending neighbor solicitation (`NeighborSol`) messages to the address being checked. If the mobile node receives a defending neighbor advertisement (`NeighborAdv`) corresponding to the `NeighborSol` within `RetransTimer` milliseconds (ms), the mobile node deconfigures the target address. Otherwise, it issues another `NeighborSol`. This solicit-and-wait process can repeat at most `DupAddrDetectTransmits` times. Given `RetransTimer` and `DupAddrDetectTransmits` default values — which are 1,000[7] and 1,[3] respectively — the DAD execution time is at least one second. Although we could set `RetransTimer` to a smaller value to speed up DAD, doing so increases the probability of missing defending `NeighborAdvs`.

There are currently two approaches to improving DAD: O-DAD, which is an Internet Engineering Task Force standard, and A-DAD.

Optimistic DAD

O-DAD[5] lets nodes use addresses before DAD has checked their uniqueness. If the DAD procedure later reports that an address is already in use, the mobile node using it must immediately deconfigure it. This can penalize both the mobile node (by breaking ongoing connections) and the node that rightfully owns the address (because it will receive misdirected packets). O-DAD is beneficial, however, if address collision probability is low.

Advanced DAD

In A-DAD,[6] an access router uses standard DAD to verify IP addresses in a network domain. It then maintains all unique IP addresses in an address pool for allocation to arriving mobile nodes. These addresses are considered reserved. Some devices, however, can attempt to configure reserved addresses by means other than A-DAD. When this happens, the access router will receive a `NeighborSol` destined for a pooled address; to avoid address collision, it must silently delete the address from its pool.

With A-DAD, mobile nodes obtain duplication-free addresses as a part of the standard router discovery process when they enter new subnets. Entering mobile nodes multicast router solicitations (`RouterSol`) to all access routers to obtain essential router information. A-DAD extends `RouterSol` to include an option that notifies access routers of a duplication-free care-of-address (CoA) request. This option includes the mobile node's previous CoA and link-layer address as an attachment. When it receives such a `RouterSol`, the access router

- selects and removes an address from its address pool;
- creates a neighbor cache entry that associates the selected address with the mobile node's link-layer address;

A-DAD performance depends on the address pool's size: it must be large enough to accommodate potential mobile nodes.

- creates a host route entry using the mobile node's previous CoA and link-layer address;
- creates a `RouterAdv` with an enabled new CoA reply option (`NCoA Reply`) that includes the address selected from the pool;
- sends the `RouterAdv` directly to the mobile node's previous CoA using the host route entry; and
- deletes the host route entry.

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

A-DAD performance depends on the address pool's size: it must be large enough to accommodate potential mobile nodes, but without wasting too much address space. Also, an A-DAD access router must maintain hard states. That is, A-DAD no longer works if the temporarily stored address pool is missing, which can occur during access router reboot.

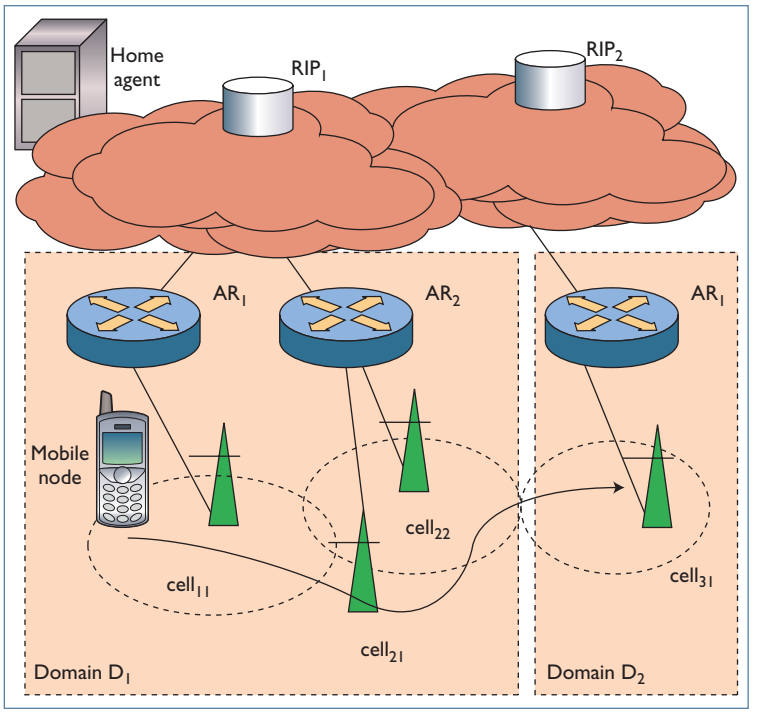

Figure 1. An IPv6 network with regional information point (RIP) servers. RIPs maintain tables with access routers (AR_1, AR_2, and AR_3).

Predicting Mobile Node Movement

A layer-2 hand off causes a layer-3 hand off when a node changes network domains. There are no standard ways to detect the need for a layer-3 hand off. Although it's possible to infer domain change by the last-received router advertisements' expiration, advertisement lifetimes are typically in the order of minutes, so the approach isn't timely.

Some researchers[8] have proposed detecting domain change by exploiting topology information — that is, the association between access points and access routers. Given this topology information, mobile nodes can determine whether the new and old access points are in the same network domain. Consequently, the mobile node can detect the need for a layer-3 hand off as soon as it knows the new access point.

Our work also relies on detecting movement using topology information, but we focus on stateless, rather than stateful, address configuration. In stateless address configuration, mobile nodes form valid IP addresses without the aid of a Dynamic Host Configuration Protocol (DHCP) server.

P-DAD

Our basic idea is to run DAD before or during layer-3 hand off by using topology information and layer-2 signals to predict the new network domain. Unlike O-DAD, our approach guarantees IP address uniqueness prior to use. Unlike A-DAD, P-DAD doesn't require a reserve of IP addresses and thus better utilizes address space. Also, P-DAD access routers need only maintain soft state.

Anticipated Network Architecture

Figure 1 shows an example of an IPv6 network. Our approach assumes that each domain uses a regional information point (RIP) server, which maintains mobile-node attachment point (MAP) tables that store connected-to relationships between access routers and access points in the serving domain. Each MAP table entry is a tuple $\langle p, q, f \rangle$, where p is an access point's basic service set ID (BSSID), q is the access router IP address that p connects to, and f is the prefix advertised by q (used for stateless address autoconfiguration).

Table 1 shows example MAPs. Developers can implement a RIP as either a stand-alone server or an add-on software module in access routers. They can manually configure the MAP information, which rarely changes.

Each RIP periodically exchanges its MAP table with neighbor RIPs. Consider Figure 1 as an example. RIP_1 learns its neighbor domain's topology by exchanging its MAP table with that on RIP_2. Table 2 shows the resulting MAP table on RIP_1 (RIP_2). Upon entering a new domain, a mobile node requests a copy of the MAP from the new RIP server. The mobile node thus knows all the access-point and access-router associations and network prefixes in the current and all surrounding network domains.

Each access router must maintain a registration cache. Each mobile node that has its IP address verified via P-DAD maintains the following data in the registration cache:

- the home address,
- the pre-allocated new CoA, and
- the CoA's lifetime value.

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

The Protocol

A mobile node obtains the serving RIP's IP address as part of the standard router discovery process. We assume that all access routers know their RIP server's IP address. When a mobile node sends a

Table 1. Example mobile-node attachment point (MAP) tables: before exchange (a) on RIP_1 and (b) on RIP_2.

AP's Basic service set ID	AR's IP address	Advertised prefix
(AP_{11}) 00-01-4A-C3-07-01	(AR_1) 2001:0E10:6440:0001::1	Prefix1
(AP_{21}) 00-01-4A-C3-07-02	(AR_2) 2001:0E10:6440:0002::1	Prefix2
(AP_{22}) 00-01-4A-C3-07-03	(AR_2) 2001:0E10:6440:0002::1	Prefix2

(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	Prefix3

(b)

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

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	$Prefix_3$

`RouterSol`, the access router returns a `Router-Adv` with an optional field notifying the mobile node of the serving RIP's IP address. After locating the serving RIP, the mobile node can then request its MAP table.

Given the MAP information, a mobile node can determine whether to prepare a layer-3 hand off when a layer-2 hand off is about to occur. First, the mobile node uses some mechanism[9] to discover the next access point for a layer-2 hand off. It can then use the MAP's connect-to information to determine which access routers correspond to the candidate access points. If the access routers differ from the current serving access router, the mobile node realizes a layer-3 hand off is needed before it actually conducts a layer-2 hand off. Therefore, the MAP's layer-2 information and the layer-2 hand off's next candidate access points can jointly assist the mobile node in determining whether a layer-3 hand off is imminent and (if so) start DAD proactively.

The following CoA pre-allocation algorithm (Algorithm 1) shows how a mobile node conducts a layer-3 hand off:

1. Before switching to the next access point, the mobile node extracts the new access router's associated prefix from the MAP table and uses the prefix to generate a tentative CoA `addr`. The mobile node then sends the new access router a `CoA_preAllocate Request` message with parameter `addr` to test for uniqueness.
2. Upon receiving the `CoA_preAllocate Request (addr)`, the new access router checks its registration cache and, if necessary, performs a standard DAD to see if `addr` is unique. It reports the result back to the mobile node via a `CoA_preAllocate Reply (U)` message: the `U`-bit is set only if the uniqueness check is passed.
3. If the `U`-bit is set, the mobile node stores the pre-allocated CoA in its registration cache and sends the new access router a `CoA_activation Request` message to activate the pre-allocated CoA after the node associates with the new access point. If the `U`-bit isn't set or the mobile node receives no reply, the mobile node forms a CoA using the standard stateless address auto-configuration procedure, and proceeds to Step 5.
4. Upon receiving the `CoA_activation Request` message, the new access router removes `addr` from the registration cache and acknowledges the mobile node through a `CoA_activation Reply` message. If the mobile node doesn't receive a `CoA_activation Reply` in time, it performs a standard stateless address configuration procedure.
5. The mobile node informs the home agent (HA) of its current location through a `Binding Update` message.

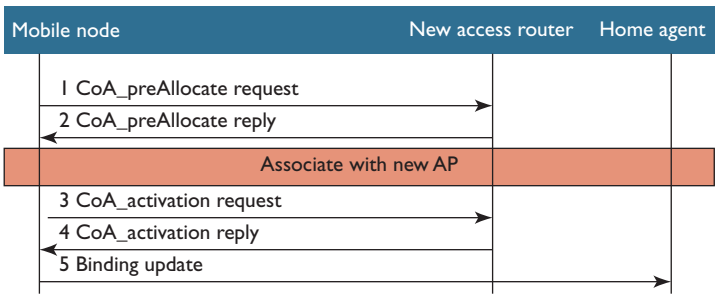

Figure 2. Care-of-address pre-allocation. The mobile node forms a new CoA before switching to the next access point (AP).

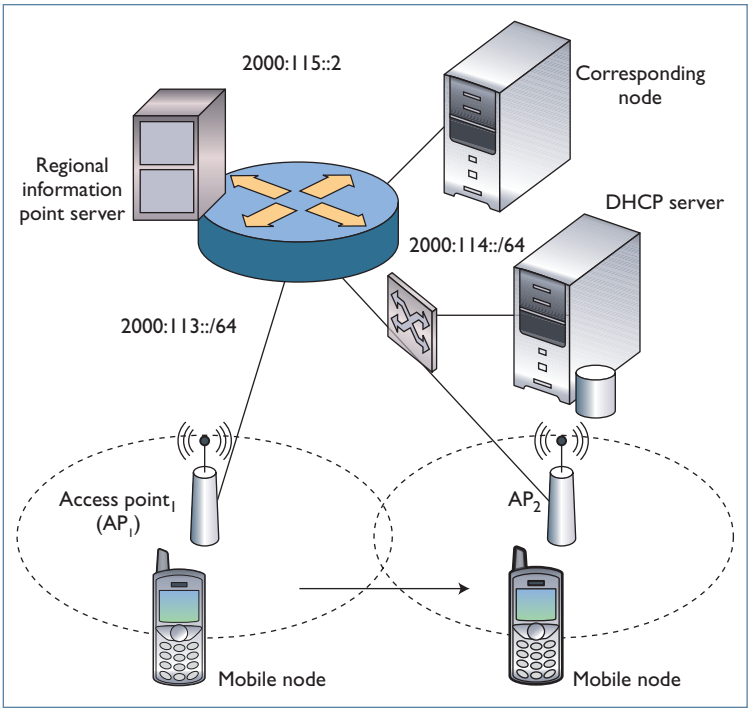

Figure 3. Experimental setup. We installed an access router, a Dynamic Host Configuration Protocol (DHCP) server, and mobile nodes to measure layer-3 hand-off delay and any lost packets caused by hand off.

Figure 2 shows the message flow of Algorithm 1.

In contrast to A-DAD, an access router in our protocol need only maintain soft state. That is, a crashed or malfunctioning access router causes mobile nodes to perform only standard stateless address configurations.

Performance Evaluation

We conducted experiments to measure layer-3 hand-off delay.

Figure 3 shows the experimental setup. We used a PC server with Linux Kernel 2.6 and IPv6 router advertisement daemon `radvd` version 0.8 (http://v6web.litech.org/radvd) as an IPv6 router. The `radvd` sends `RouterAdv` messages to local LANs periodically and to nodes that request them through `RouterSol` messages. These messages are required for IPv6 stateless address autoconfiguration. We used DHCPv6 (version 0.85), which was originally developed by a SourceForge project (http://dhcpv6.sourceforge.net). Finally, we equipped the mobile node with two identical Intersil prism2-based IEEE 802.11b wireless interfaces and placed it where it could associate with either AP_1 or AP_2.

In each experiment, a corresponding node generated packets at a constant rate (one per 20 ms) and lost a sequence of packets during the hand-off period. The mobile node also lost a sequence of packets during the hand-off period. We recorded both the time ($t1$) when the node received the last packet before a hand off and the time ($t2$) when it received the first packet after the hand off. The hand-off delay was measured as ($t2 - t1$). To measure hand-off delay, we followed five steps:

1. Before hand off, associate the mobile node's interface 1 with AP_1. Configure interface 1 through standard stateless address autoconfiguration.
2. Start generating and transmitting packets.
3. Detach the CoA of interface 1. Directly associate the mobile node's interface 2 with AP_2.
4. Configure a new CoA for interface 2 through standard stateless address autoconfiguration.
5. Perform Mobile IPv6 binding update.

This procedure doesn't consider the erratic layer-2 hand-off delay. Step 3 emulated breaking the link to AP_1; because we performed Step 4 immediately after Step 3, no move-detection delay occurred.

Also, we slightly modified Step 4 to measure the layer-3 hand-off delay for stateless address autoconfiguration with A-DAD and P-DAD. For A-DAD, we configured a new CoA for Step 4's interface through a DHCP server. For P-DAD, we configured a new CoA right after Step 4's execution. Table 3 summarizes the empirical results from 10 experimental runs.

As the table shows, the original layer-3 hand-off delay was more than 1,400 ms, which is unacceptable for voice over IP applications. In contrast, A-DAD's hand-off delay was approximately 83 ms, whereas P-DAD's was around 48 ms with low variation, which should meet time-

Table 3. Layer-3 hand-off delay and lost packets for 10 experimental runs.

Metrics	Standard stateless address configuration	Stateless address configuration with A-DAD	Stateless address configuration with P-DAD
Hand-off delay	Avg. 1,419.2 ms	Avg. 83.6 ms	Avg. 48.4 ms
	Std. 906.9	Std. 16.07	Std. 11.6
Number of lost packets	Avg. 70	Avg. 2.5	Avg. 1.4
	Std. 45.2	Std. 0.7	Std. 0.5

critical applications' delay requirements. Furthermore, the original layer-3 hand off lost 70 packets; this number fell to 2.5 and 1.4 packets with A-DAD and P-DAD, respectively.

In the near future, multiple wireless network interfaces — including Universal Mobile Telecommunications System (UMTS), Wi-Fi, and Wi-MAX — will be equipped with mobile nodes. This will make seamless hand off between heterogeneous networks a critical challenge. We're now working to adapt P-DAD to this kind of environment.

Acknowledgments

National Science Council grants NSC 94-2219-E-009-006, NSC 94-2745-E-126-003-URD, and NSC 94-2213-E-216-001 supported this work.

References

1. D. Johnson, C. Perkins, and J. Arkko, *Mobility Support in IPv6*, IETF RFC 3775, June 2004; www.rfc-editor.org/rfc/rfc3775.txt.
2. J.H. Cho and G. Daley, *Goals of Detecting Network Attachment in IPv6*, IETF RFC 4135, Aug. 2005; www.rfc-editor.org/rfc/rfc4135.txt.
3. S. Thomson and T. Narten, *IPv6 Stateless Address Autoconfiguration*, IETF RFC 2462, Dec. 1998; www.rfc-editor.org/rfc/rfc2462.txt.
4. R. Droms, ed., *Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*, IETF RFC 3315, July 2003; www.rfc-editor.
org/rfc/rfc3315.txt.
5. N. Moore, *Optimistic Duplicate Address Detection for IPv6*, IETF RFC 4429, April 2006; www.rfc-editor.org/rfc/rfc4429.txt.
6. Y.H. Han, S.H. Hwang, and H. Jang, "Design and Evaluation of an Address Configuration and Confirmation Scheme for IPv6 Mobility Support," *Proc. IEEE Wireless Comm. and Networking Conf.*, IEEE Press, 2004, pp. 1270–1275.
7. T. Narten, E. Nordmark, and W. Simpson, *Neighbor Discovery for IP version 6 (IPv6)*, IETF RFC 2461, Dec. 1998; www.rfc-editor.org/rfc/rfc2461.txt.
8. C.C. Tseng et al., "Topology-Aided Cross-Layer Fast Handoff Designs for IEEE 802.11/Mobile IP Environments," *IEEE Comm.*, vol. 43, no. 12, Dec. 2005, pp. 156–163.
9. C.C. Tseng et al., "Location-Based Fast Handoff for 802.11 Networks," *IEEE Comm. Letters*, vol. 9, no. 4, 2005, pp. 304–306.

Chien-Chao Tseng is a professor in the Department of Computer Science at National Chiao-Tung University, Hsin-Chu, Taiwan. His research interests include wireless Internet, handover techniques for heterogeneous networks, and mobile computing. Tseng has a PhD in computer science from Southern Methodist University. Contact him at cc tseng@csie.nctu.edu.tw.

Yung-Chang Wong is an associate professor in the Department of Computer Science and Information Engineering at Providence University, Taiwan. His research interests include personal communications, wireless Internet, and wavelength-division multiplexing optical networks. Wong has a PhD in computer science and information engineering from National Chiao-Tung University, Taiwan. He is a member of the IEEE and the Institute of Electronics, Information, and Communications Engineers (IEICE). Contact him at ycwong@pu.edu.tw.

Li-Hsing Yen is an associate processor in the Department of Computer Science and Information Engineering at Chung Hua University, Taiwan. His current research interests include mobile computing, wireless networking, and distributed algorithms. Yen has a PhD in computer science from National Chiao-Tung University, Taiwan, and is a member of the IEEE. Contact him at lhyen@chu.edu.tw.

Kai-Cheng Hsu is a graduate student in the Department of Computer Science and Information Engineering at National Chiao-Tung University, Taiwan. His research interests include design and analysis of wireless networks, Internet communications, and mobile computing systems. Hsu has a BS in computer science and information engineering from National Chiao-Tung University. Contact him at kcshu@csie.nctu.edu.tw.