Mobile Networks and Applications 11, 215–227, 2006
© 2006 Springer Science + Business Media, LLC. Manufactured in The Netherlands.
DOI: 10.1007/s11036-006-4474-8

Enabling Intelligent Handovers in Heterogeneous Wireless Networks

ROBERT C. CHALMERS
Department of Computer Science, University of California, Santa Barbara, CA 93106, USA

GOVIND KRISHNAMURTHI
Communications Systems Lab, Nokia Research Center, Burlington, MA 01803, USA

KEVIN C. ALMEROTH
Department of Computer Science, University of California, Santa Barbara, CA 93106, USA

Published online: 31 March 2006

Abstract. In the future Wireless Internet, mobile nodes will be able to choose between providers offering competing services at a much finer granularity than we find today. Rather than months, service contracts may span hours or minutes. Connectivity, however, is just one of many possible services. Providers will begin to offer network and application-level services targeted at improving the overall wireless experience of the user. Determining the best path through the various networks will require accurate information describing which services are being offered by each provider. In this paper, we model the process of propagating this information as an instance of a distributed, hierarchical cache. Access routers actively discover and collect information about the immediate network neighborhood on behalf of mobile nodes. Mobiles fill their own caches through queries to their local access routers, and then employ the cached information to make informed, intelligent handover decisions. Through simulation, we show that high cache hit rates at the mobile node can be achieved even when the discovery process at the access router is incomplete. In comparison to static and centralized approaches, our dynamic approach requires less configuration and maintenance, avoids single points of failure, and provides a scalable solution that spans administrative domains.

Keywords: mobility, handover, candidate access router discovery

1. Introduction

The development and proliferation of wireless, mobile technologies has revolutionized communications. Ubiquitous connectivity, however, has yet to be achieved, especially for data services. The problem is one of scale. Cellular systems provide large coverage areas, but traditionally offer low bandwidth connectivity and limited support for data traffic. On the other hand, wireless data networks such as 802.11 (WLAN or WiFi) offer broadband access, but only within a limited physical range. Recent interest has focused on leveraging the best of both technologies in the form of hot-spot deployments [4, 5, 11]. Using hot-spots, providers can offer subscribers not only wide-area connectivity through the cellular infrastructure, but also increased bandwidth via WiFi access points deployed in high concentration areas such as malls or airports.

Hot-spots, and overlay networks in general [16], complicate handover[1] decisions for mobile nodes. For instance, as a mobile node (MN) transitions into a hot-spot, it must initiate a vertical handover [14,21] between two distinct systems (cellular and WiFi). Moreover, once within the hot-spot, a number of competing providers may offer different services, such as Quality of Service (QoS) for voice or smooth handover protocols [6,7] to improve latency. As the Wireless Internet matures, mobile nodes will be faced with an increasing array of choices and will be required to make more informed and intelligent decisions concerning handover.

Current mobile systems primarily base handovers on physical-layer properties such as received signal strength (RSS), bit-error rate (BER), or signal-to-noise ratio (SNR) [11]. In future systems, policy-based handover algorithms [20] will supplement these physical-layer properties with higher-level information describing the **capabilities** of each target access point (AP) and corresponding access router (AR).[2] These capabilities describe which services are supported by a particular access router, as well as properties of the connection, such as available bandwidth or cost.

The solution that we propose, dyCARD—dynamic Candidate Access Router Discovery [18], takes a distributed, learning-based approach to the problem of collecting and

[1] A handover is the transition a mobile node makes between two distinct networks or between wireless access points within a single network.

[2] An access router is the first-hop IP router for a mobile node. An access point is simply a link-layer bridge connecting the AR to the wireless network. For the purposes of this paper, we will consider an AP and a base-station to be synonymous.

distributing these high-level capabilities. In dyCARD, the mobile node's current access router assists the MN by collecting, in advance, the capabilities of its immediate neighbors. Initially, neighbors discover one another through the handover patterns of the mobile nodes [10]. Once discovered, two neighboring routers exchange capabilities directly. As a mobile node prepares for handover, it queries its local access router for information regarding reachable access points. The returned capabilities can then be passed as parameters to an appropriate policy-based handover algorithm.

In addition to implementing a prototype in Linux, we simulate dyCARD in ns-2 to evaluate the effectiveness of the discovery process. To this end, we model the system as a distributed cache. We measure the fill ratio in each AR's cache of neighboring access points, as well as the cache hit rate for individual mobile nodes resolving reachable APs. We compare dyCARD to static, pre-configured caches, as well as two server-based, centralized schemes. In short, we find that dyCARD compares favorably to non-dynamic approaches, especially for moderate and high mobility. dyCARD is less sensitive than either server-based scheme to delay between access routers or the back-end server. Moreover, dyCARD provides support for capabilities that change over time (e.g., prices that change with respect to the available bandwidth) since routers exchange information directly, allowing them to refresh dynamic capabilities periodically.

The remainder of the paper is organized as follows. We begin in Section 2 by further motivating the topic with an example. We review related work in Section 3. Section 4 provides an overview of the dyCARD protocol. In Section 5, we present evaluation results from our simulations. We discuss future directions in Section 6, and conclude the paper in Section 7.

2. Motivation

Consider a shopping mall of the future where a number of providers offer competing network services (see figure 1). While roaming through the mall, a user selects an appropriate network based on her current context, as well as which services are being offered at agreeable prices. For instance, a user may enter a store that offers its customers free connectivity with any purchase, but advanced services such as application-level proxies are not available. The decision to handover from a more extensive mall-wide provider to the store's local network depends on a number of factors including the user's intention to purchase something.

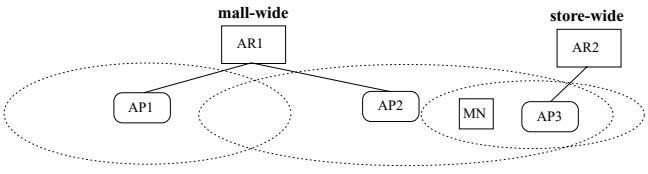

Figure 1. An example scenario where a MN has the choice of handing-over from a mall-wide provider to a store-wide service. The regions demarcated by dotted lines represent the coverage areas of the respective APs.

Although this example may seem slightly exaggerated, the underlying concepts are not. As the Wireless Internet grows, mobile users will encounter an expanding array of options when deciding to handover. Current strategies, based on simple physical-layer information, are inadequate when faced with the challenges of navigating a complex wireless topology where connectivity is both hierarchical and heterogeneous [5]. For instance, a mobile hosting a voice call should prefer an access router offering QoS even though the signal strength is slightly lower than another AR's that is offering only best-effort service. In the future, it will become necessary for a mobile node to consider the capabilities of each candidate access router in relation to the user's requirements as an integral part of any handover decision.

When discussing heterogeneity, it is important to note that heterogeneity does not imply simply integrating different access technologies such as macro-cellular, micro-cellular, BlueTooth and WiFi [19]. Certain vendors may target key application areas, such as multimedia, providing proxies at the edge of the network to automatically transform multimedia streams [2,11], thus improving performance for small wireless devices. Other vendors may offer IP-level services such as Fast Mobile IP [7] or Context Transfers [6] to improve handover latencies. Finally, competing vendors may offer short-term special rates for raw connectivity in reaction to reduced system utilization. For users, the most important criteria are application-level performance and cost. The best target for handover is the one that maximizes the quality of the connection as it is perceived by the end-user [16].

The key to enabling this level of heterogeneity and choice is to provide the mobile node with enough information concerning competing IP and application-level services so that it can make intelligent handover decisions. To achieve this goal, there are two primary alternatives:

1) force the mobile node to query each AR individually, or
2) provide the necessary information through the MN's current access router.

The former option is problematic, especially for mobile's with a single interface since the collection process would require the mobile to disconnect from its current access point, thus disrupting any on-going communication.

With respect to the latter AR-assisted approach, there still remain a number of possibilities:

1) fully provision all routers with the capabilities of their neighbors,
2) maintain a centralized map in a back-end server [3], or
3) allow each router to dynamically discover its neighbors and exchange capabilities directly.

The first two options are adequate for small networks operated by a single provider, but more complex topologies require a more dynamic approach. dyCARD offers a flexible approach

that does not require cooperating providers to exchange and maintain extensive maps of neighboring access networks. Server-based schemes suffer from well-known scalability issues inherent to centralized architectures, and they present a single point of failure for the protocol. Finally, only a dynamic approach can support truly dynamic capabilities. For instance, prices for different services may change frequently based on the current demand for each service. A server-based approach would require that all changes be posted to the central server and then broadcast out to all ARs, introducing serious scalability concerns.

So, why would two competing providers choose to cooperate at all? In fact, cooperation is already quite prevalent within the current cellular phone system in the form of roaming agreements. Cellular phone providers allow subscribers to roam in competing networks in order to maintain high customer satisfaction. Today, customers expect ubiquitous connectivity, and a failure to provide connectivity results in lost customers. It is to the advantage of the cellular provider to out-source connectivity in order to retain their current customer base, even at a loss of immediate profits. Moreover, roaming offers a means for providers to court new customers with additional services and exceptional quality.

We expect that future wireless operators will continue to form roaming agreements with their competitors. In the future, however, more than simple connectivity may be at stake. Customers will request advanced services and improved bandwidth when they become available, even if it is through a competing provider.

3. Related work

The issue of Candidate Access Router Discovery (CARD) has been addressed, in part, by the Seamless Mobility (Seamoby) working group of the IETF. Seamoby has produced a problem statement for CARD [19], as well as a detailed set of requirements [8]. Two competing protocols were discussed as candidates for standardization: a server-based approach [3], and the dynamic approach we present herein [18]. The working group was unable to reach a consensus on which approach was most appropriate.

Thus, the resulting working group document [9] fails to define a means for the initial discovery of neighboring routers. In other words, the current CARD proposal assumes that the access routers have *a priori* knowledge of their physical neighborhood. The document simply defines the message structures used for communication between the protocol participants (MNs and ARs), similar to those described in Section 4. In this work, we describe a complete solution to CARD that takes a dynamic approach to the issue of neighbor discovery. We compare our approach to both static and server-based schemes in Section 4.5.

CARD is just one part of a family of smooth handover protocols whose purpose is to minimize handover time and the disruption experienced while moving between access points. Pagtzis and Kirstein propose an architecture for Proactive Mobility in which access routers pre-allocate care-of addresses for mobile nodes and multicast packets to neighboring routers while the mobile is in transit [10]. Although the authors mention AR capabilities and introduce the notion of using the mobility of MNs to aid access routers in identifying neighbors, they do not pursue the idea of using that information to improve MN handover decisions in heterogeneous environments.

Similar to Proactive Mobility, Fast Handover protocols [7] attempt to smooth out handovers by forwarding packets from the previous access router to the new AR while the mobile node is in transit. Context Transfers [6] allow a mobile node to forward state maintained at the previous AR, such as QoS parameters, to the new access router during handover, forgoing otherwise necessary renegotiation after handover. These techniques, however, make an implicit assumption that the current access router has some *a priori* knowledge concerning the next AR. For example, in Fast MIPv6, the current access router issues proxy router advertisements for a neighboring AR [7]. dyCARD provides the means to gather this information dynamically. In this way, dyCARD provides a service, not only to mobile nodes, but to other protocols at the access router.

In 1994, Katz realized that the future of wireless systems is moving toward hybrid networks, a combination of pico-, micro- and macro-cellular systems that takes advantage of the strengths of each architecture [5]. This resulted in a number of techniques for performing vertical handovers between hybrid networks [11,16,21]. Traditionally, however, these techniques force a very strict ordering on the layering of networks. For the most part, mobile nodes know *a priori* which systems are available, and have a preexisting precedence for each network based on the maximum bit-rate. Although dyCARD itself is not a handover algorithm, the protocol provides the means for mobile nodes to collect adequate information in order to make more advanced handover decisions. This opens a new frontier for policy-enabled handover algorithms [20].

4. dyCARD

In this section, we describe the operation of dyCARD. The concept is simple. Access routers collect information describing what is available in the immediate neighborhood in order to help the mobile node select the best path through the network. The protocol takes a learning-based approach where neighboring access routers discover one another through the handover patterns of mobile nodes [10].

dyCARD consists of two distinct, though related, components: 1) Physical Neighbor Discovery (PND), and 2) Target AR (TAR) Resolution. Physical Neighbor Discovery is the process by which two access routers with overlapping coverage areas discover one another and share information about the services each supports. TAR Resolution is initiated by

the mobile node in order to resolve the capabilities of each candidate target for future handovers.

In the following sub-sections, we discuss the general environment in which dyCARD exists, and then describe each protocol component in more detail. We conclude the section with a discussion of security, focusing specifically on techniques to mitigate the effect of malicious mobile nodes.

4.1. Environment

We make the underlying assumption that future wireless networks will be all-IP networks. This is already true for WiFi networks, and the next generation of cellular phone systems are expected to employ native IPv6. By abstracting ourselves away from the link-layer and physical characteristics of individual technologies, we can describe the protocol in more general terms, using consistent terminology. In this section, we define this terminology as it pertains to the physical environment of a wireless network.

At the network layer, a mobile node's point of attachment in a visited network is an access router (AR). The access router may support multiple access points (AP) which provide wireless link-layer connectivity. Each access point can be identified by a unique value, or tuple, referred to herein as the AP's Basic Service Set Identifier (BSSID). This identifier is broadcast periodically which allows a mobile node to compile a set of reachable access points.[3] Typically, any information regarding the access router associated with a particular access point is not available unless the mobile node actively associates with that access point.

To clarify the description of the protocol and the associated examples, we will assume throughout the following discussion that each access point represents a distinct IPv6 subnet, and is thus associated with a unique subnet prefix. Accordingly, upon handing-over to a new access point, the MN must configure a new unique **care-of** address (e.g., IPv6 stateless address configuration) [12,17] which is local to that subnet.

4.2. Physical neighbor discovery

The coverage area of a given access router is simply the total area that falls within transmission range of any of the AR's wireless access points. In dyCARD, an access router leverages the mobility pattern of the mobile nodes to discover the local topology of overlapping access points in its own coverage area. In short, *if a mobile node can handover between two access points, then the associated access routers are considered to be physical neighbors*.

The Physical Neighbor Discovery process is outlined in Figure 2. In this example, the mobile node is handing-over between two access points (step a), from the previous AP (pAP) to the new AP (nAP). Each access point is associated

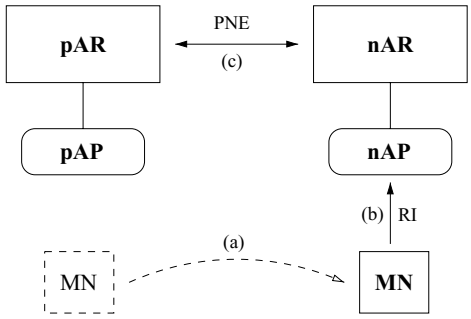

Figure 2. A sample message exchange during Physical Neighbor Discovery.

with an access router, pAR and nAR respectively.[4] Upon handover to the new access point, the mobile node first configures a care-of address and performs any necessary authentication and authorization with nAR, as dictated by the access network. Then, the mobile node sends a Router Identity (RI) message to the new access router (step b), describing both the source and destination of the handover. Specifically, the message carries:

- the BSSID of the new access point, nAP;
- the BSSID of the previous access point, pAP; and
- the IP address of the previous access router, pAR

In cases where there is no previous point of attachment (i.e., *disjoint* handovers[5]), the information pertaining to the previous AP and AR is omitted from the message.

Upon receiving a Router Identity message, the access router updates its view of its own locally connected access points using the BSSID of nAP from the message. The access router maintains a list of local APs as soft-state that is refreshed with each newly received RI message. In this way, we leverage the inherent learning capabilities of the dyCARD protocol to allow an access router to discover local access points. This is important since there currently exists no standardized method for an access router to detect attached APs. Of course, this information can be statically configured by an administrator, but this does not account for possible failures and limits reconfiguration.

If the received RI message contains the address of the previous access router (i.e., the handover is not disjoint), the new AR sends a Physical Neighbor Exchange (PNE) message to pAR containing the BSSIDs of the two APs and the identity of the mobile node (step c in figure 2). Upon receipt of this message, pAR performs a number of checks to ensure the validity of the information provided by the MN (see Section 4.4). If the report is considered valid, pAR creates or updates an entry for the tuple, <nAR,nAP>, in its Physical Neighbor Cache (PNC). Then, pAR replies to nAR with the result of the validation. If pAR accepted the report (i.e., pAR

[3]The BSSID is specific to WiFi, but the concept can easily be applied to other wireless technologies.

[4]nAR and pAR could also be a single router with two access points.

[5]Disjoint handovers indicate a disruption in connectivity where a MN is not connected to any AP for some period of time.

considers nAR to be physical neighbor), nAR updates its own cache with the tuple <pAR,pAP>.

As a result of the Physical Neighbor Exchange, each MN handover creates or refreshes entries in the PNCs of both neighboring routers. After a time, if no handover occurs between the two routers, the PNC entries will timeout and be removed. By employing soft-state, the protocol gracefully handles failures in neighboring access routers or their APs. Moreover, changes in the topology, such as a new or relocated access point, will be discovered dynamically as soon as a mobile node transitions to or from the affected AP.

In addition to discovering the local topology of neighboring access routers, dyCARD provides a mechanism for neighbors to exchange **capabilities**. Capabilities are a description of the services that a given access router can offer a mobile node, such as available bandwidth, cost, or the presence of other protocols and services.

4.3. Target AR resolution

TAR resolution is the process by which a mobile node associates capabilities with each candidate access point prior to making a handover decision. To initiate TAR resolution, the mobile node issues a Candidate List Request (CLR) message to its current access router. The CLR message contains a list of the AP BSSIDs that are currently reachable by the mobile node. Reachability pertains to the set of APs from which a mobile node can currently receive beacons. In order for a neighboring router to be considered a candidate for handover, one of its access points must be within range of the mobile node.

Upon receiving a Candidate List Request, the current access router translates the list of access points included in the message to a set of candidate access routers (CARs) using the mappings present in its Physical Neighbor Cache. If the mobile node includes a **profile** describing its preferences as part of the CLR message,[6] the access router can filter the CAR list based on the contents of the profile and the capabilities of each candidate AR, possibly choosing a subset of the most promising candidates. The access router then returns the CAR list to the mobile node in a Candidate List (CL) message with partial capabilities accompanying each CAR entry.

Once TAR Resolution is complete, the mobile node can employ an appropriate handover algorithm, passing in the resolved capabilities, its own profile, and the physical-layer properties of each reachable access point in order to select a target for handover. In cases where the current access router is unable to resolve a given AP (a cache miss), the mobile node has only the physical-layer properties of the access point to consider during TAR resolution. In the case of an empty CL message, the handover decision is based solely on the physical properties of each reachable AP, identical to a traditional handover algorithm. Thus, the effectiveness of an informed handover depends directly on the completeness of Physical Neighbor Discovery. We explore this dependency in more detail in Section 5.4.

[6] A mobile node's profile could already be available at the current AR due to a context transfer from the previous access router, or as part of the MN's Authentication, Authorization and Accounting (AAA) context retrieved from the MN's home network.

4.4. Security

Next, we consider the security aspects of dyCARD. In a recent work, Shim et al. present a general analysis of the security issues related to Candidate Access Router Discovery [15]. Here, we focus specifically on mechanisms in dyCARD designed to limit the impact of malicious mobile nodes. Much of the detail presented in this section derives from lessons learned while actually implementing a prototype of the protocol.

The key problem is that an access router creates state in its Physical Neighbor Cache in response to the information provided by mobile nodes in the form of Router Identity messages. With a sufficient number of bogus entries, a mobile node could overrun the router's memory unless the size of the PNC is strictly limited. Once we restrict the total size of the cache, however, we run the risk of replacing valid entries with erroneous ones, thus directly affecting support for non-malicious mobile nodes.

In reality, it is impossible to eliminate all attacks on the protocol since access routers depend on reports from mobile nodes in order to learn about their physical neighborhood. However, attacks are made extremely difficult with the introduction of three validation checks (discussed below) which access routers perform in order to verify the reports made by MNs. As a result, any concerted attack would require a very large number of local mobiles impersonating an equally large number of nodes spread across the network. Moreover, due to the nature of soft-state, the effort would need to be sustained in order to deny service to valid mobile nodes.

4.4.1 Authentication and authorization

In order to secure the protocol, all participants must be able to mutually authenticate one another with explicit authorization to exchange dyCARD messages. Inter-AR authorization could be performed through a AAA architecture, leveraging the service-level agreement between the participating domains to securely prepare keying material for authentication [1]. For the mobile node, authentication and authorization should be part of the process required to initially access the visited network, and thus should occur prior to any dyCARD messages being exchanged between the AR and MN.

Authentication provides a means to uniquely identify the mobile node. In cellular systems, this identification might be the International Mobile Subscriber Identifier (IMSI) from the phone's SIM card [15]. For AAA-based authentication, the user's Network Access Identifier (NAI) would be used. Unauthenticated identifiers, such as the mobile node's care-of address, are inadequate since a node can generate any number of IPv6 addresses, and thus appear to be many mobiles from the point of view of the access router.

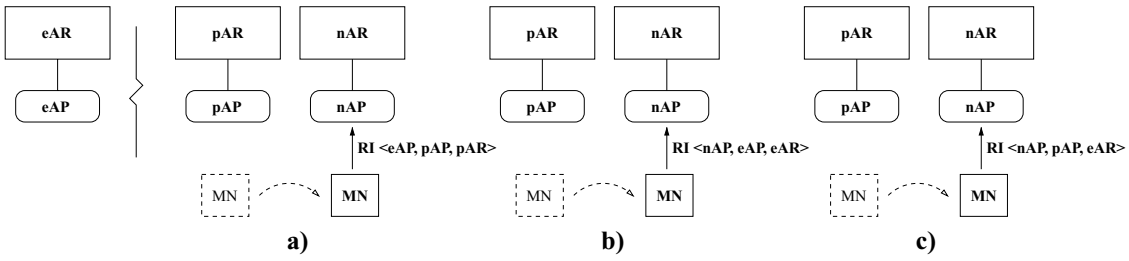

Figure 3. Three basic forms of erroneous RI messages: the MN lies about a) its current AP, b) its previous AP, or c) its previous router. A valid RI message should contain the tuple <nAP,pAP,pAR>. In each case, eAR and eAP are erroneous in that they are not physical neighbors of <nAR,nAP>.

Although mobile nodes are authenticated and authorized to send RI messages, access routers should not blindly trust MN reports. A mobile node could mistakenly report a disjoint handover, or a malicious MN with stolen credentials could impersonate another mobile. Moreover, colluding mobile nodes could share credentials to mount a distributed attack on the protocol. Therefore, authentication and authorization are not enough. Access routers must validate the reports that they receive from mobile nodes.

4.4.2. Report validation

For a single malicious node acting alone, an erroneous RI can take on a combination of three basic forms (see figure 3): the MN lies about a) its current AP, b) its previous AP, or c) its previous router. In the first case, the mobile node provides an invalid BSSID for its current access point. In the second case, the previous AR and AP are associated, but are not neighbors of the current access router. In the last case, the previous access point is not associated with the reported previous AR, but is actually a neighboring AP. To catch these three errors, the access routers perform the following three validity checks for each RI message:

1. the current AR checks the current AP against a list of authorized local access points;
2. the previous AR checks that the previous AP exists as a local entry in its PNC; and
3. the previous AR verifies that the mobile node was recently present.

In short, the checks ensure that the two access points do indeed belong to the two access routers, and that the previous AR is indeed physically adjacent.[7]

To combat distributed attacks by colluding mobile nodes, we tag each cache entry with the identity of the reporting mobile. We can then bound the total number of cache entries attributed to any one mobile node. This technique limits the effectiveness of a valid, although malicious, mobile node sharing its credentials with others. Moreover, a smart cache replacement policy should be employed to ensure that valid entries are given highest priority. We propose two simple rules that reflect this goal:

1. favor entries that have been recently referenced in Candidate List messages; and
2. favor entries created from local Router Identity messages over those created in response to remote PNE messages.

Both of these rules favor information gathered from locally connected mobile nodes, thus diminishing the effect of a distributed attack.

[7]There is also the possibility that a MN reports an alternate valid AP for the previous AR. This case is not problematic, however, since the two ARs are actually neighbors and the total number of these entries is necessarily limited by the number of valid APs.

5. Evaluation

In this section, we present the methodology and results of our analysis of the dyCARD protocol. First, we describe our implementation experience. Then, we develop a comprehensive simulation environment in which we study the factors contributing to the performance of the protocol.

5.1. Prototype implementation

The process of implementing the protocol brought forth a number of interesting issues that were not initially considered during protocol design. Our exploration of these issues is reflected in the level of detail afforded the protocol description in the preceding section. Moreover, implementation provided us with a means to validate the concepts on real systems.

The complete protocol is composed of a set of kernel modules developed for Linux 2.4.19. We also implemented the AR functionality for FreeBSD 3.2. In Linux, a core module provides common utility routines and exposes two distinct interfaces to the rest of the kernel. One interface exposes hooks into the MN and AR functionality of the protocol. The second interface provides a registration service for external modules to register specialized support routines that are used by dyCARD. This allows separate modules to be developed independently and loaded dynamically to perform such tasks as managing capabilities or performing customized target selection.

In order to validate the effectiveness of the protocol, we constructed a small 802.11b test-bed consisting of one mobile node and three access routers, each with a single access point. With the testbed, we confirmed the correct operation of the

protocol, as well as demonstrated its effectiveness in allowing the mobile node to select between multiple AP's based on high-level properties, such as the advertised bandwidth and cost of each link. Working with real nodes, however, limits the number of scenarios that one can effectively explore with repeatable experiments. Moreover, it provides limited insight into the dynamics of the protocol. For example, how does the protocol behave as the number of mobile nodes increases, or the topology of ARs vary? How does the protocol compare to a static approach or server-based schemes? In order to evaluate more extensive scenarios, simulations are a necessary next step.

5.2. Simulation

To simulate dyCARD, we modeled the protocol in ns-2. In addition to implementing the base protocol, it was necessary to extend ns-2 to properly represent access routers supporting multiple access points, as well as to provide limited IPv6 functionality. We created a simple node to act as an access point, forwarding traffic between the wired and wireless networks. Each AP transmits data on a unique channel, analogous to using a distinct frequency. All APs however, broadcast periodic beacons[8] over a common channel so that all mobiles within range can receive them.

Although not part of the dyCARD protocol, it was necessary to implement a rudimentary algorithm to initiate handovers. Our algorithm is based on the signal strength of each beacon received. We define three thresholds for the signal strength: **thresh_in**, **thresh_out** and **thresh_crit**. **Thresh_in** is the minimum signal strength necessary to consider a neighboring access point as a candidate for handover. **Thresh_out** determines the transition region [21] where a mobile node begins the process of TAR resolution and eventually target selection. Once the current signal strength reaches **thresh_crit**, a mobile node will hand-over immediately whether or not the dyCARD discovery process has completed. For these simulations we are not particularly concerned with limiting ping-pong effects. So, hysteresis and dwell timers [21] are not employed in an attempt to keep the handover algorithm uncomplicated.

To compare dyCARD with possible server-based approaches, we modeled a single back-end server that maintains a complete mapping for all access routers and their associated APs. Since no definitive description of a server-based protocol exists [3], we experimented with two alternative schemes: **parallel** queries and **serial** queries. The parallel scheme performs server queries in the background, responding immediately to CLR messages with any cached mappings. The serial scheme queries the back-end server to resolve any uncached mappings before replying with a CL message.

[8]Access points generate beacons at 100 ms intervals.

Table 1
Average handover rates for differing parameters of a sample simulation. Also shown are the total number of handovers throughout the simulation, as well as the period between handovers (90th percentile).

Rate	MNs	Speed	Pause	Handovers	Period
0.05	10	0.5 m/s	10 s	30	194.63 s
0.21	10	3 m/s	10 s	125	90.04 s
0.25	50	0.5 m/s	10 s	149	224.86 s
0.43	10	10 m/s	10 s	256	51.05 s
0.50	100	0.5 m/s	10 s	300	254.20 s
0.70	150	0.5 m/s	10 s	419	240.04 s
0.91	50	3 m/s	10 s	546	90.57 s
1.69	100	3 m/s	10 s	1016	96.53 s
2.05	50	10 m/s	10 s	1227	42.55 s
2.76	150	3 m/s	10 s	1657	87.86 s
4.34	100	10 m/s	10 s	2605	43.02 s
6.30	150	10 m/s	10 s	3777	41.55 s
7.61	150	10 m/s	0.1 s	4563	36.77 s
13.78	150	20 m/s	0.1 s	8270	19.68 s
17.96	150	30 m/s	0.1 s	10774	14.17 s
24.09	150	40 m/s	0.1 s	14456	10.33 s
27.72	150	50 m/s	0.1 s	16630	9.35 s
33.95	150	60 m/s	0.1 s	20368	7.17 s
41.92	150	80 m/s	0.1 s	25149	5.73 s
47.66	150	100 m/s	0.1 s	28596	4.96 s

5.3. Simulation parameters

For our simulations, we employ 802.11 at the physical/MAC layers; however, we are not currently focused on the effect of any particular link technology. We chose a transmission range of 40 meters. At this range, the signal strength thresholds, **thresh_in**, **thresh_out** and **thresh_crit**, translate to distances of 34.5, 36 and 39.5 meters respectively. These particular values were chosen empirically to limit the likelihood of disconnection while still providing a reasonable amount of time to complete the discovery process, prior to reaching the critical threshold.

Five sets of AP topologies were generated. Each AP topology covers a 200 × 200 meter grid. Access points were randomly chosen and placed in the grid so as to provide complete wireless coverage with no intervening gaps. Once the grid was covered, any remaining APs were randomly placed within the grid. Simulations were run with topologies consisting of 20 access routers, each having between 1 and 3 associated access points. For each handover, a mobile node considers 4 to 6 access points on average.

Five distinct sets of mobility patterns were generated with a way-point model for each combination of 10, 50, 100 and 150 MNs with a pause time of 10 seconds and maximum speeds of 0.5, 3 and 10 m/s. Further mobility patterns were generated for 150 MNs with a shorter pause time and higher speeds in order to simulate scenarios with higher handover rates (see Table 1). All simulations ran for a total of 600 seconds.

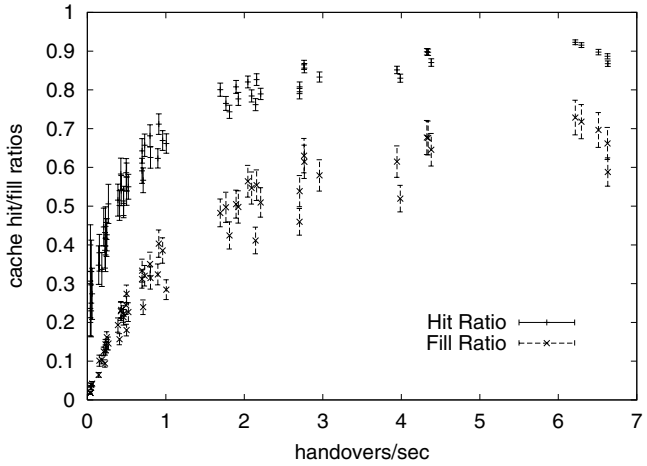

Figure 4. Average cache hit and fill ratios (99% confidence intervals).

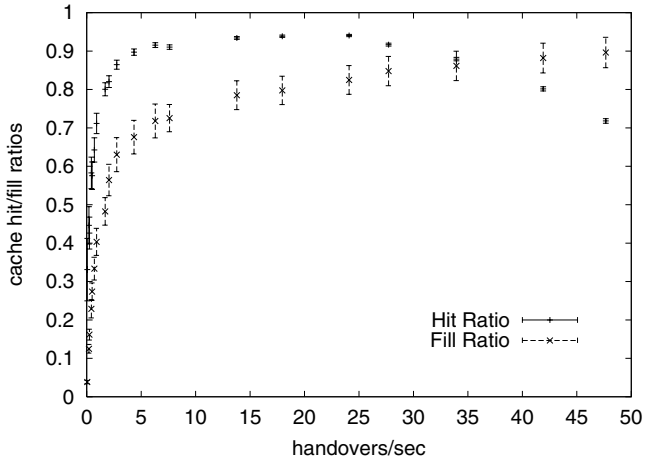

Figure 5. An extended view of average cache hit and fill ratios.

5.4. Simulation results

The goal of this section is to evaluate whether dyCARD effectively maintains neighboring relationships between access routers, and distributes capabilities to mobile nodes. To this end, we can view dyCARD as a distributed cache management protocol. Each access router maintains a Physical Neighbor Cache (PNC) with information regarding its local neighborhood. Mobile nodes maintain their own caches of reachable access points which they annotate with capabilities through requests to their local ARs. In order for intelligent handovers to be effective, a MN's cache should be relatively complete.

To analyze the performance of dyCARD as a distributed cache, we employ two primary metrics: the **cache fill ratio** and the **cache hit ratio**. The cache fill ratio measures the percentage of neighboring cache entries present in each access router's PNC. In other words, the cache fill ratio reflects how full the PNC is for each access router. The cache hit ratio, on the other hand, reflects how often a mobile node can resolve capabilities for targets being considered for handover.

Figure 4 presents both metrics for each simulation instance (error bars in all graphs indicate 99% confidence intervals). The two metrics are graphed with respect to the average handover rate for the entire network. The handover rate provides a good measure of the activity level of the network particularly since handovers initiate the population of the AR caches. A handover rate of 2.0, for instance, indicates that 1200 handovers occurred during the 600 seconds of simulation. A handover rate of 4.0 reflects twice as much activity. Table 1 relates handover rate to the underlying properties of the simulation that affect handover, namely, the total number of mobile nodes, as well as their maximum speed and pause time.

It should be noted that the absolute value of the handover rate is not itself a viable indicator of the activity at any particular access router, especially over short periods. To clarify this point, consider a single simulation instance with an average handover rate of 27.72 (150 MNs at 50 m/s in Table 1). Looking at the activity at individual routers, we find that the busiest AR handled 436 handovers over a one second period. Ninety percent of all routers had bursts of more than 188 handovers per second.

Returning to figure 4, it is interesting to note the presence of small clusters of five points visible throughout the data. Each cluster represents simulation runs using one of the five matched sets: topology and mobility pattern. It is clear from this graph that, although some variability exists, neither the AP topology nor the MN mobility pattern plays a critical role in the overall behavior of the protocol. Throughout the remainder of the analysis, we present results from a single topology with the understanding that all topologies produce similar results.

Our expectation is that as the handover rate increases, AR caches will maintain higher fill ratios since more RI messages are received from mobiles with details of the physical neighborhood. Figure 4 confirms this hypothesis with cache fill ratios reaching around 60–70% for moderate handover rates. What is more interesting, however, is that the cache hit ratio exceeds the cache fill ratio, reaching 90%. One would expect that the cache hit rate would be somehow constrained by the fill ratio since the former depends directly upon the latter.

The reason behind this discrepancy can be explained in terms of cache locality. Consider the relationship between the two caches, an access router's PNC and a MN's cache of resolved APs, as two levels of a hierarchical cache. The MN's cache is a proper subset of the AR's PNC since all reachable access points are by definition within the neighborhood of the current AR. The access router's PNC, the higher level cache, has the potential for far more entries than are necessary to achieve a 100% cache hit rate in a single mobile's cache. In other words, the mobile node depends on a subset of the complete neighborhood map, and this subset changes slowly with time as the mobile moves through the topology. So, high cache hit rates at the mobile node can be achieved with lower fill ratios at each access router.

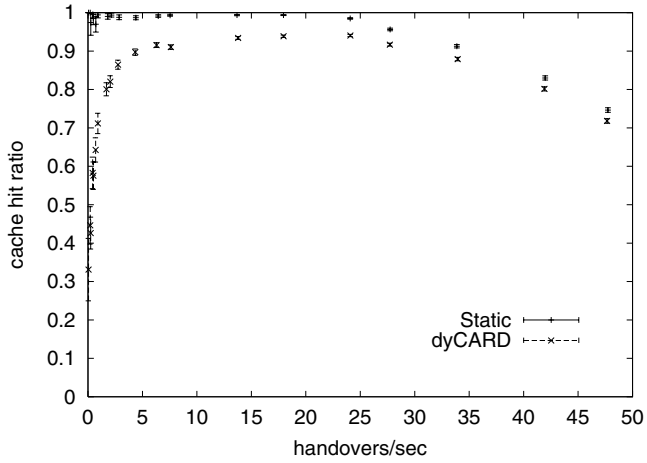

Figure 6. A comparison of cache hit ratio for dyCARD and a static approach.

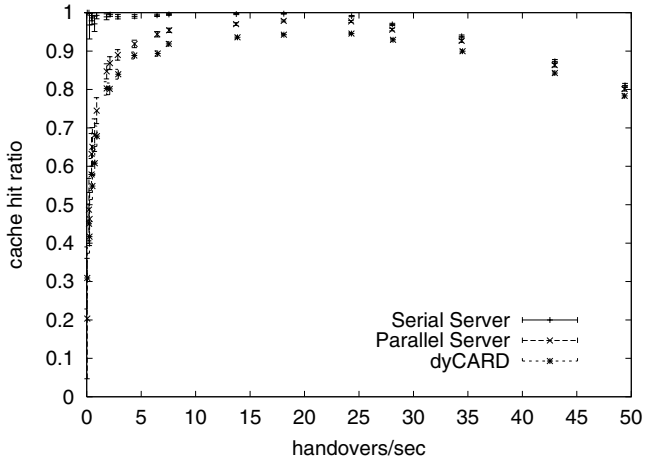

Figure 7. A comparison of cache hit ratio for dyCARD and server-based schemes.

One issue that is not apparent from figure 4 is how extreme speed effects the population of the mobile's cache. In figure 5, we extend the simulation set to include much higher handover rates resulting from higher MN speeds and shorter pause times (see Table 1). As the handover rate increases, the cache hit ratio continues to increase until the mobile's speed no longer allows it to complete the resolution process effectively. As speeds increase, the time available to query the local AR is reduced. The result is a considerable drop in the measured cache hit rate although the cache fill ratio continues to climb.

To determine whether this eventual degradation in the hit rate is specific to the dynamic nature of dyCARD, we compare dyCARD with a static approach in figure 6. In the static approach, we pre-load all AR PNCs with complete neighborhood information, resulting in 100% fill ratios throughout the simulation. As can be seen, the static approach suffers the same fate as mobile speeds increase. Thus, this degradation seems inherent to any discovery-based method. We will return to this issue later in the analysis.

Our expectation is that a static approach should perform better than a dynamic approach, such as dyCARD, since each access router has complete knowledge of its local neighborhood. This expectation is confirmed by figure 6, but the difference between the two approaches diminishes significantly as the rate of handovers increases. To compare the two graphs empirically, we employ the **root mean square difference (rmsd)** which measures the average point-wise distance between the two graphs. In figure 6, a static approach offers a **rmsd** of 0.29 over dyCARD. This result is misleading, however, since the majority of that difference occurs at extremely low handover rates (see Table 2). If we focus on scenarios with handover rates greater than 3.0, the **rmsd** is only 0.06. In other words, for simulations over three handovers/sec, dyCARD achieves average cache hit rates within 6% of a static approach.

Next, we turn our attention to how dyCARD compares to server-based approaches. As discussed in Section 5.2, we simulated two server schemes: **parallel** and **serial**. Again, we would expect a server-based approach to perform better than a dynamic approach since ARs can query the back-end server which has *perfect* knowledge. In figure 7, the **parallel** scheme performs marginally better than dyCARD with a measured **rmsd** of 0.05. The **serial** scheme performs very well at low handover rates with a **rmsd** of 0.42 under 3.0 handovers/sec. When mobiles are moving slowly, there is ample time to query the server directly. As the rate of handovers increases, however, improvement over dyCARD diminishes to 0.06, similar to the static approach (see Table 2).

Our base topology in figure 5 employs link delays of 10 ms and 20 ms, respectively, for the links between APs and ARs (*AP Delay*) and those between ARs and the core router that interconnects all access routers (*AR Delay*). To fairly compare dyCARD with each server scheme, we reduced all link delays in figure 7 to 0.1 ms, including the delay between the core router and the back-end server (*Server Delay*). Due to the lower delays, the cache hit ratio for dyCARD improves between figures 5 and 7.

Thus, figure 7, provides a rather optimistic view. To better understand how delay impacts each server scheme, as well as dyCARD, we increased the *AR Delay* and *Server Delay* by factors of 10, from 0.1 ms up to 10 seconds. Figure 8

Table 2
The **rmsd** of alternative schemes with respect to the base dyCARD protocol for different handover rates.

	Handover rate			
Scheme	>0.0	<3.0	>3.0	>20.0
Static	0.291	0.408	0.058	0.035
Serial server	0.301	0.421	0.063	0.037
Parallel server	0.053	0.068	0.032	0.025
AP capabilities	0.087	0.120	0.029	0.020
Retain	0.087	0.063	0.106	0.146

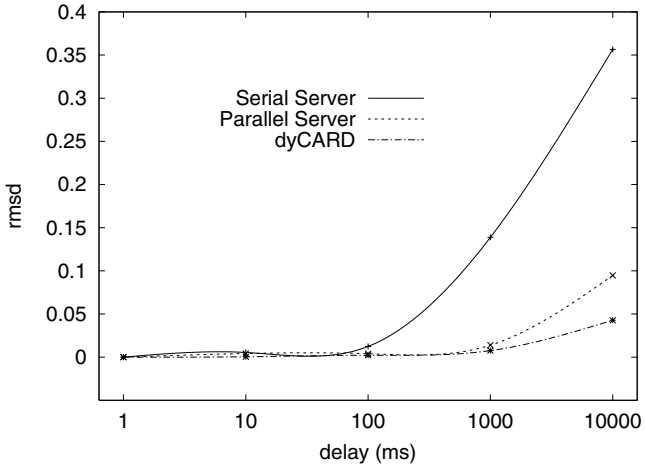

Figure 8. The effect of delay expressed in terms of the **rmsd** from the minimum delay of 0.1 ms.

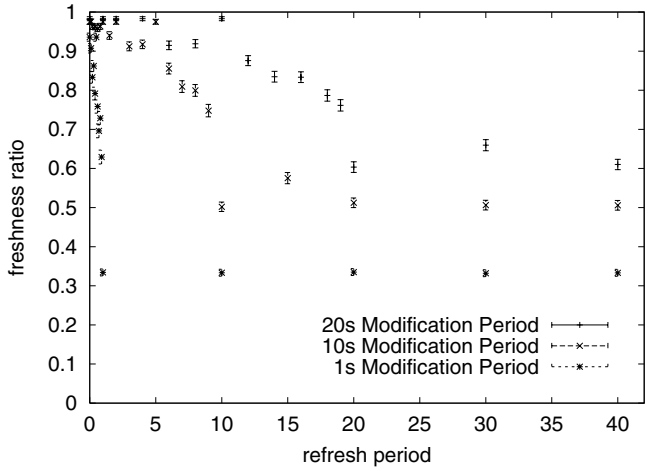

Figure 9. The effect of dynamic capabilities on freshness.

shows that dyCARD and both server-based approaches are indeed affected by the delay between network components. We express the magnitude of the change in terms of the **rmsd** measured from the minimum delay of 0.1 ms for each scheme. For dyCARD, increased delay results is a slight reduction in the cache hit ratio, 0.04 for the longest *AR Delay* of 10 seconds.[9] The **parallel** scheme also performs reasonably well with a worst-case decrease of 0.10. The **serial** scheme, however, begins to degrade rapidly as the delay increases. With a 1 second *Server Delay*, the performance of the **serial** scheme is reduced by 0.14, and reaches more than 0.35 for a 10 second delay.

The **serial** scheme suffers due to the fact that every mobile request incurs a high *Server Delay* unless all requested BSSIDs are already cached at the AR. For both dyCARD and the **parallel** scheme, the mobile receives an immediate, although possibly incomplete, reply. The delay for these two schemes is masked since the exchange between network elements is performed outside the MN's critical resolution path. The result may be slightly slower population of the AR caches, but as we have shown, a low fill-ratio is less detrimental to the mobile node than is a slow resolution process.

One problem with a server-based approach is that it becomes difficult to manage capabilities that change frequently since access routers do not communicate directly. In dyCARD, access routers can periodically push updates for dynamic capabilities to neighboring routers. With frequently modified capabilities, however, we face the possibility that mobiles might base handover decisions on stale information, especially if the *refresh period* is greater than the *modification period*. In other words, mobiles may receive stale capabilities from their local AR if the capabilities are changing faster than the two neighboring ARs are refreshing them.

[9]The delay experienced by dyCARD is actually twice the *AR Delay* since each message traverses two links.

We explore the issue of freshness in figure 9. Our metric, the **freshness ratio**, measures the ratio of cache entries at the mobile node that have up-to-date capabilities at the time that handover decisions are made. In each graph, the freshness ratio drops at a linear rate for *refresh periods* less than the *modification period*. Once the *refresh period* reaches the *modification period*, however, the freshness ratio plateaus. Although the plateau provides a worst-case measure for each respective *modification period*, we believe the associated freshness ratio is too low for most handover scenarios. Unless the handover algorithm is resistant to stale information or the particular capability does not require accurate representation, deployments should ensure that the configured *refresh period* between two neighbors is significantly less than the shortest *modification period* in use. Although it is difficult to discern from Figure 9, a *refresh period* of exactly half the *modification period* results in an optimal freshness ratio (a discontinuity in the linear decrease) of more than 90% in all three cases.

One interesting aspect of each graph is that we can approximate the slope (m_p) of the linear portion as we increase the *modification period* (p):

$$m_{p_2} \approx \frac{m_{p_1}}{\frac{p_2}{p_1}} \tag{1}$$

By increasing the *modification period* by a factor of 10, we decrease the rate of decay in the freshness ratio by the same factor of 10. This estimate provides a means for operators to fine-tune the *refresh period*, finding an appropriate tradeoff between freshness and communication overhead. Reducing the *refresh period* necessarily increases the volume of traffic exchanged between neighboring ARs.

So far, the analysis has concentrated primarily on understanding how dyCARD performs with respect to other approaches. Now, we explore the question of whether there exists room for improvement in the base protocol itself. We present two optimizations which work to improve the measured cache hit ratio. We find that significant improvement

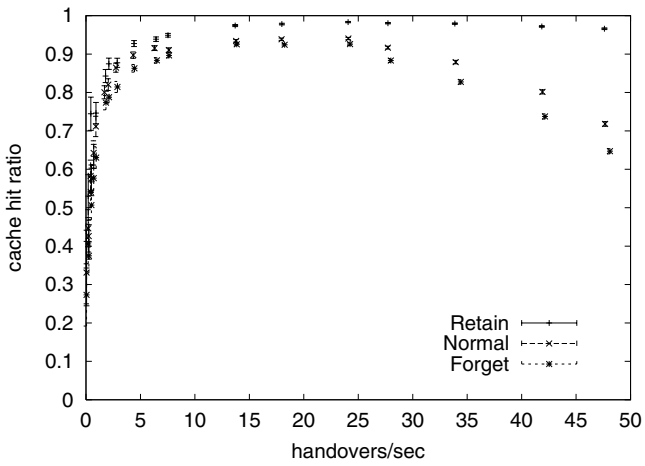

Figure 10. The effect of memory at the MN on the cache hit ratio.

is possible, especially at high handover rates where all approaches begin to exhibit degraded performance.

First, we attempt to implicitly increase the cache hit rate by improving the cache fill ratio at each access router. If we return to the model of dyCARD as a distributed cache then one way in which to improve the fill ratio of the PNC is to increase the size of the *fill unit*. Currently, entries in the PNC are created at the granularity of individual access points. Each handover creates or refreshes a single AR to AP mapping. If we expand this *fill unit* from a single access point to all APs associated with the same access router, we increase the measured cache fill ratio by 0.15. With this optimization, neighboring routers exchange lists of their local access points as capabilities. In this way, each handover results in a complete exchange of local topology information which reduces the number of handovers necessary to achieve full PNCs. The increased fill ratios have a significant impact on the cache hit rate when mobility is low (see **AP Capabilities** in Table 2), but the benefit is reduced as the rate of handovers increases.

Our second optimization addresses the specific problem of high mobility, and the resultant degradation of the cache hit ratio. For this experiment, we never expire cache entries at the mobile node. Thus, the MN **retains** all discovered capabilities for the extent of the simulation. In the base dyCARD protocol, cache entries are flushed when the associated access point is no longer reachable. An AP is considered unreachable when the MN misses three consecutive beacons. As a mobile moves through the topology, it normally forgets information that might be useful again after a rather short period, especially when mobility is high.

Figure 10 demonstrates the effect that memory has on the MN's cache hit rate with three protocol instances: **Normal**, **Forget** and **Retain**. **Normal** represents the base protocol and is identical to Figure 5. **Forget** represents one extreme in which the mobile node flushes its cache after every handover. Flushing results in only a slight drop in the hit ratio which leads us to conclude that **Normal** actually retains very little information between handovers. With **Retain**, the cache hit rate rises considerably, reaching more than 98% with measured improvement of 0.15 for handover rates greater than 20.0. By extending the lifetime of cache entries at the MN, we can overcome the previously observed degradation at high handover rates. Longer lifetimes, however, come with the price of reduced cache freshness. A more general solution may be to reduce the frequency of handovers by relegating high-speed mobiles to umbrella cells with larger coverage areas [13].

So, what conclusions can we draw from the analysis? First, the level of cache retention at the mobile node has the most significant impact on the resulting cache hit rate (see Table 2). In other words, what a mobile remembers is more important than what a mobile discovers from its local AR. A mobile node can compile a fairly complete view of its own neighborhood by combining the partial views it receives from each access router. As a mobile moves through the network, each access router contributes a piece to the MN's view of the network.

This observation leads us to conclude that the most significant aspect of any CARD protocol is whether the local AR can resolve a mobile's request for information before the MN is forced to handover. The completeness of that information is less critical than the timeliness of the reply. In this regard, dyCARD surpasses either server-based approach. Because the two phases of *discovery* and *resolution* are decoupled, dyCARD can provide an immediate response to mobile requests. Moreover, dyCARD is more resilient to delay within the wired network. Thus, dyCARD is applicable to a wider range of access networks with varying topologies and delay characteristics.

A dynamic approach also has a number of less empirical benefits. In particular, dyCARD was specifically designed to operate between domains. Static and server-based schemes are typically considered as intra-domain solutions. Unlike a server-based approach, dyCARD has no single point of failure since it is a fully distributed protocol. Finally, due to its dynamic nature, dyCARD requires significantly less configuration and maintenance than any of the alternatives solutions.

6. Future work

So far, we have focused on evaluating dyCARD from the perspective of its caching behavior. To claim, however, that dyCARD actually improves handover performance requires the integration of dyCARD with a particular handover algorithm that can take advantage of the high-level information provided by cached capabilities. Wang et al. propose a policy-based handover algorithm which employs a utility function to select the best handover target based on a number of criteria, such as available bandwidth and cost [20]. The range of capabilities they consider, however, is limited due to the fact that they have no means to collect general information about each access point and router. dyCARD is the first step

towards making policy-based handovers possible in heterogeneous wireless networks. As future work, we will further pursue the issue of handover performance by developing a number of intelligent handover algorithms which work in concert with dyCARD.

7. Conclusion

In this paper, we present dyCARD, a dynamic protocol for Candidate Access Router Discovery. In dyCARD, access routers leverage MN handovers to automatically discover neighboring ARs [10]. Once discovered, neighboring routers exchange capabilities describing the services being offered to mobile nodes. In turn, mobile nodes can leverage the information collected by their current access router to make informed, intelligent decisions for future handovers.

We show that a dynamic approach to CARD is indeed achievable through a prototype implementation, as well as a detailed simulation environment. From the simulations, we conclude that a static or server-based approach is most appropriate for small networks with extremely low mobility since a dynamic approach relies on mobility within the network in order to populate AR caches. For more active networks, however, dyCARD is the superior solution since it scales to the inter-domain, introduces no single point of failure, and requires minimal configuration or maintenance.

A dynamic approach is also more resistant to fluctuations in network delay than either of the server-based schemes that we investigated. The **serial** server scheme, in particular, is sensitive to increased delay since every MN request is penalized by the round trip between the access router and back-end server. This delay can become especially problematic in scenarios with high mobility since the time available to MNs for resolution is further reduced. dyCARD masks inter-AR delay by communicating outside the critical period, replying immediately to MN requests. In dyCARD, discovery and resolution are separated, leading to a more robust and responsive system.

Another major benefit of a dynamic approach is that it directly supports capabilities that change over time. We find that to maintain optimal freshness, the *refresh period* should be exactly half the *modification period*. For routers with many dynamic capabilities, however, a single optimal *refresh period* may not exist. Thus, operators will be forced to tradeoff between achieving optimal freshness and limiting the communication between access routers. To this end, we provide an estimate for the decay rate of the freshness ratio as the *modification period* is adjusted.

In the end, we believe that the future of wireless networking, whether we call it the Wireless Internet or not, will ultimately depend on how successfully mobile nodes are able to manage heterogeneity. The first step in that evolution has already begun in the deployment of hot-spots where mobiles must navigate between heterogeneous technologies. In the future, service-level heterogeneity will further complicate the handover decisions of mobiles, but with complexity comes choice.

A mobile node of the future will be able to choose between providers at a much finer granularity than we find today. Rather than months, service contracts may span just hours or minutes. Current strategies based on simple physical-layer information will not be adequate when faced with the challenges of navigating a complex wireless topology in which connectivity is both hierarchical and heterogeneous. dyCARD empowers mobile nodes with the information necessary to navigate this complex scape of the future Wireless Internet.

Acknowledgment

The authors wish to acknowledge Dirk Trossen, Hemant Chaskar, and Eunsoo Shim for their integral contributions to the concept and design of dyCARD. We would also like to thank Josep M. Blanquer and Toni Batchelli for their thorough review and welcome comments.

References

[1] R. Chalmers and K. Almeroth, A security architecture for mobility-related services, Journal of Wireless Personal Communications. (to appear) (2004).

[2] A. Fox, S.D. Gribble, Y. Chawathe, E.A. Brewer and P. Gauthier, Cluster-based scalable network services, Operating Systems Review 31(5) (1997) 78–91.

[3] D. Funato, X. He, C. Williams, A. Takeshita, M.D. Ali and J. Nakfour, Geographically adjacent access router discovery protocol, Technical Report draft-funato-seamoby-gaard-*.txt, Internet Engineering Task Force (2002).

[4] P.S. Henry and H. Luo, WiFi: What's next. IEEE Communications Magazine 40(12) (2002) 66–72.

[5] R.H. Katz, Adaptation and mobility in wireless information systems, IEEE Personal Communications 1(1) (1994) 6–17.

[6] J. Kempf (Editor), Problem description: Reasons for performing context transfers between nodes in an IP access network. Request for Comments (Informational) 3374, Internet Engineering Task Force (2002).

[7] R. Koodli (Editor), Fast handovers for mobile IPv6. Technical Report draft-ietf-mobileip-fast-mipv6-*.txt, Internet Engineering Task Force (2002).

[8] G. Krishnamurthi (Editor), Requirements for CAR discovery protocols. Technical Report draft-ietf-seamoby-card-requirements-*.txt, Internet Engineering Task Force (2002).

[9] M. Liebsch and A. Singh (Editors), Candidate access Router discovery. Technical Report draft-ietf-seamoby-card-protocol-*.txt, Internet Engineering Task Force (2003).

[10] T. Pagtzis and P. Kirstein, A model for proactive seamless IP mobility and mobility-hop routing, in: *Proceedings of 10th IEEE International Conference on Networks (ICON'02)* (Singapore, 2002).

[11] K. Pahlavan, P. Krishnamurthy, A. Hatami, M. Ylianttila, J. Mäkelä, R. Pichna and J. Vallström, Handoff in hybrid mobile data networks, IEEE Personal Communications 7(2) (2000) 34–47.

[12] C. Perkins and D. Johnson, Mobility support in IPv6, in: *Proceedings of Mobicom'96* (Rye, NY, USA, 1996).

[13] T. Rappaport, *Wireless Communications: Principles and Practice*, Prentice Hall (Upper Saddle River, NJ, 1996).

[14] S. Saunders, K. Kelly, S. Jones, M. Dell'Anna and T. Harrold, The indoor-outdoor radio environment, Electronics and Communication Engineering Journal 12(6) (2000) 249–261.

[15] E. Shim, J.P. Redlich and R. Gitlin, Secure candidate access router discovery, in: *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC'03)*. New Orleans, LA, USA (2003).

[16] M. Stemm and R.H. Katz, Vertical Handoffs in wireless overlay networks, ACM Mobile Newtorks and Applications 3(4) (1998) 335–350.

[17] S. Thomson and T. Narten, IPv6 stateless address autoconfiguration. RFC 2462, Internet Engineering Task Force (1998).

[18] D. Trossen, G. Krishnamurthi, H. Chaskar, R. Chalmers and E. Shim, A Dynamic protocol for candidate access-router discovery. Technical Report draft-trossen-seamoby-dycard-*.txt, Internet Engineering Task Force (2002a).

[19] D. Trossen, G. Krishnamurthi, H. Chaskar and J. Kempf, Issues in candidate access router discovery for seamless IP-level handoffs. Technical Report draft-ietf-seamoby-cardiscovery-*.txt, Internet Engineering Task Force (2002b).

[20] H. Wang, R. Katz and J. Giese, Policy-enabled handoffs across heterogeneous wireless networks, in: *Proceedings of IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'99)* (New Orleans, LA, USA, 1999).

[21] M. Ylianttila, M. Pande, J. Mäkelä and P. Mähönen, Optimization scheme for mobile users performing vertical handoffs between IEEE 802.11 and GPRS/EDGE networks, in: *Proceedings of IEEE Global Telecommunications Conference (GLOBECOM'01)* (San Antonio, TX, USA (2001).

Robert C. Chalmers received his B.S. in Computer Science from the California State Polytechnic University, Pomona in 1997, and his M.S. in 2003 from the University of California in Santa Barbara. He is currently a Ph.D. candidate at the University of California in Santa Barbara where his main research interests focus around leveraging intelligence within the network. Particularly, he has studied multicast and its effect on resource utilization, as well as how to provide services for small, mobile devices in edge networks. He was awarded the Ericsson Fellowship in 2001 and is currently a Eugene Cota-Robles Fellow.
E-mail: robertc@cs.ucsb.edu

Govind Krishnamurthi received an M.S. (Electrical Engineering) and Ph.D. degree (Computer Engineering) from the University of Washington and the Iowa State University, in 1997 and 1999 respectively. He spent the summer of 1995 as an intern at Bellcore, Morristown, NJ. He is a recipient of the Research Excellence Award from the Iowa State University for his Ph.D. thesis. Since the summer of 1999 he has been a Senior Research Engineer at the Nokia Research Center, Boston, MA. He has authored several publications and holds 3 patents. His current interests deal with QoS, location based services and security issues in IP based wireless networks.
E-mail: govind.krishnamurthi@nokia.com

Kevin C. Almeroth earned his Ph.D. in Computer Science from the Georgia Institute of Technology in 1997. He is currently an associate professor at the University of California in Santa Barbara where his main research interests include computer networks and protocols, multicast communication, large-scale multimedia systems, and performance evaluation. At UCSB, Dr. Almeroth is a founding member of the Media Arts and Technology Program (MATP), Associate Director of the Center for Information Technology and Society (CITS), and on the Executive Committee for the University of California Digital Media Innovation (DiMI) program. In the research community, Dr. Almeroth is on the Editorial Board of IEEE Network, has co-chaired NGC 2000, Global Internet 2001, NOSSDAV 2002, and MMNS 2002; has served as tutorial chair for several conferences, and has been on the program committee of numerous conferences. Dr. Almeroth is serving as the chair of the Internet2 Working Group on Multicast, and is a member of the IETF Multicast Directorate (MADDOGS). He is also serving on the advisory boards of several startups including Occam Networks, NCast, Hidden Footprint, and the Santa Barbara Technology Incubator.
E-mail: almeroth.cs.ucsb.edu