8-2001

Autoconfiguration, Registration and Mobility Management for Pervasive Computing

Archan MISRA
Singapore Management University, archanm@smu.edu.sg

Subir DAS
Telcordia Technologies

Anthony MCAULEY
Telcordia Technologies

Sajal K. DAS
University of Texas at Arlington

DOI: https://doi.org/10.1109/98.944000

Published in IEEE Personal Communications, Volume: 8, Issue: 4, Aug 2001, Pages 24-31.
http://doi.org/ 10.1109/98.944000

Abstract

In the vision of pervasive computing, users will exchange information and control their environments from anywhere using various wireline/wireless networks and computing devices. We believe that current protocols, such as DHCP, PPP, and Mobile IP, must be enhanced to support pervasive network access. In particular, this article identifies three fundamental functions: autoconfiguration, registration, and mobility management, that need such enhancements. Realizing that the IP autoconfiguration capabilities must be extended to configure routers and large dynamic networks, we first describe our autoconfiguration solution based on the Dynamic Configuration and Distribution Protocol (DCDP). Second, we discuss why providing user-specific services over a common infrastructure needs a uniform registration protocol, independent of the mobility and configuration mechanisms. We present an initial version of the Basic User Registration Protocol (BURP), which provides secure client-network registration and interfaces to AAA protocols such as Diameter. Finally, we discuss the Dynamic Mobility Agent (DMA) architecture, which provides a hierarchical and scalable mobility management framework. The DMA approach allows individual users to customize their own mobility-related features, such as paging, fast handoffs, and QoS support, over a common access infrastructure and to select multiple global binding protocols as appropriate.

Autoconfiguration, Registration, and Mobility Management for Pervasive Computing

Archan Misra, Subir Das, and Anthony McAuley, Telcordia Technologies
Sajal K. Das, The University of Texas at Arlington

Ensuring seamless, technology-independent connectivity is an important first step to realizing pervasive access to data and voice services. This requires integration of two wireless access paradigms:

- Packet-based wide-area cellular networks, based on standards such as General Packet Radio Service (GPRS) [1] or Universal Mobile Telecommunications Services (UMTS) 2000 [2]
- Packet-based local area networks, such as IEEE 802.11 [3] LANs or low-cost short-range Bluetooth [4] links

These technologies promise to usher in the first wave of ubiquitous, device-independent backbone network access. Over a slightly longer timeline, the emergence of low-cost, low-power localized radio technologies is expected to lead to a new generation of networked-enabled devices, enabling the creation of more advanced, localized, and context-aware services, especially in traditionally non-networked environments, such as homes and shopping malls.

Although such a definition of pervasive computing is clear from a user perspective, the technological path for building such an *anytime, anywhere* networking environment is less clear. A useful way forward is to contrast the characteristics of such pervasive networks with traditional networks. In traditional computing, users work through *powerful hosts* attached to well-managed networks. The network topology is *manually crafted and configured* and *mobility is confined to hosts*. In pervasive computing, users use a wide variety of devices, many of which are only temporarily associated with an individual user. A quantum increase in the number of network-enabled nodes, as well as the need to establish dynamic connections between such nodes, makes the manual configuration of individual nodes (and even individual networks) impractical. Furthermore, the pervasive network is characterized by much stronger *application heterogeneity*; accordingly, the pervasive access infrastructure must provide an individual user the means to tailor a common access infrastructure to his/her service and mobility-related needs in a device and link-layer independent fashion.

Obviously, realizing this goal of pervasive access to network resources and applications will require enhancements at several layers of the conventional protocol stack. The bulk of pervasive computing research focuses on the service or middleware layer and is typically concerned with how nodes, *that already have basic network connectivity*, cooperate to provide users with intelligent, context-aware services in a secure and authenticated manner. In this article, however, we concentrate on providing the network access. We believe that the potential for an extremely large number of network-enabled devices and the need to provide uniform features over widely varying link technologies lead to formidable challenges. *Link-layer independence can only be achieved by defining our solutions at or above the network (IP) layer*. As part of our research in the network-layer aspects of a pervasive computing environment, we have so far focused on three specific functions where the state of the art needs to be augmented (Table 1).

Dynamic IP Configuration: To establish basic IP-level connectivity, a node must be configured with certain information, such as IP addresses and addresses of key servers (e.g., Domain Name Service, DNS). Existing configuration proto-

This work was performed while Archan Misra, currently at IBM Research, was with Telcordia Technologies.

cols such as Point-to-Point Protocol (PPP) [5], Dynamic Host Configuration Protocol (DHCP) [6], and Mobile IP [7] (with foreign agents) can configure individual hosts. However, the pervasive environment will require such autoconfiguration to be performed over entire networks of nodes, often connected in dynamic topologies. Later we introduce our *Dynamic Configuration Distribution Protocol* (DCDP)-based approach.

User Registration: Service providers must be able to authenticate, authorize, and account each user. This is specially important in pervasive networks, where the user will not only be mobile but also associated with a static set of devices. Current registration solutions, such as PPP with its well defined authentication, authorization, and accounting (AAA) interface and Mobile IP with its newly defined AAA interface [8] are inapplicable or inappropriate for certain future access scenarios, which require extensible client-to-network registration. We will explain why such registration should be separate from the choice of configuration and binding protocols, and provide the initial specification of our *Basic User Registration Protocol* (*BURP*).

Mobility Management: "Mobility support" will mean very different things in different contexts. Thus, a mobile worker needing simple Web access needs only basic connectivity; there is no need for the user to be locatable or to ensure seamless redirection of ongoing connections (at the IP layer) during a change in the point of attachment. In contrast, a user of voice over IP (VoIP) services will need paging support to receive incoming calls, and is likely to expect fast and lossless handoffs of ongoing conversations during a move. Current IP-layer mobility solutions, such as Mobile IP (MIP), lack flexible support for several such features. Moreover, they do not support the ability to simply localize certain local mobility features independent of the global mobility management scheme. We will show how our *dynamic mobility agent* (*DMA*)-based mobility solution allows a hierarchy in the mobility management architecture and enables the user to customize his or her mobility feature set *while using a common access infrastructure.*

Functions	Our solutions	Complementary protocols
Configuration	DCDP	DRCP, IPv6 autoconfiguration
Registration	BURP	Diameter
Mobility management	DMA	Mobile IP, SIP

Table 1. *Key functions and protocols for pervasive access.*

Dynamic IP Parameter Autoconfiguration

This section describes our approach to IP autoconfiguration of large dynamic networks. We first motivate the need for a mechanism to automate the distribution of IP configuration information (e.g., IP addresses) in pervasive networking environments and then present an overview of our new protocol, DCDP.

Why Autoconfiguration?

If successful, pervasive computing will lead to the proliferation of networked devices on a scale never experienced before. Even if the nodes were static, manually configuring potentially billions of devices would be too time-consuming and error-prone. Consider, for example, the *office environment*, where IP-networked nodes could include copiers, printers, projectors, phones, cameras, and vending machines. The need for such autoconfiguration capabilities becomes even more acute when one considers the *networked home* of the future, with IP-enabled appliances, such as microwave ovens, thermostats, alarm clocks, speakers, and various kinds of sensors. Clearly, we cannot expect ordinary individuals to tinker around with netmasks, default gateways, and MTU sizes. A robust and fast plug-and-play solution is needed which provides reconfiguration when nodes exhibit individual or collective mobility (e.g., when moving nodes to new rooms).

Current Solutions

Current solutions focus on autoconfiguring individual nodes on a single link (IP subnet). Popular link configuration protocols (LCPs), such as PPP [5] for serial links and DHCP [6] for broadcast LANs, have clients (hosts) dynamically requesting configuration parameters from a preconfigured server on the link. Newer LCPs, such as IPv6 stateless autoconfiguration [9] and the Dynamic and Rapid Configuration Protocol (DRCP) [10], provide more flexibility but continue to focus on the configuration of single links and assumes the server information can be manually preconfigured.

New "zeroconf" LCPs allow configuration of a subnet with no user configuration, but do not allow automatic configuration of an arbitrary topology of routers and links. Mobile ad hoc networking (MANET) scenarios go beyond an individual link; but MANET solutions have largely focused on the routing problem in isolated islands (which allow nodes to use arbitrary addresses). Providing globally routable addresses and autoconfiguring services such as DNS have not been addressed.

DCDP Overview

DCDP evolved from the Dynamic Address Allocation Protocol (DAAP) [11], which was conceived as a mechanism to automate the distribution of IP address pools to a hierarchy of DHCP servers. Besides improving on DAAP's top-down address distribution mechanism, DCDP provides autoconfiguration of additional IP-related parameters and capabilities, such as the location of DNS or SIP servers.

Our autoconfiguration approach is *modular* in that we retain the use of conventional LCPs, such as DHCP or DRCP. DCDP merely serves as the *macro autoconfiguration solution*, in that it acts as a recursive mechanism for distributing valid and unique address pools and other configuration information to dynamically assigned LCP servers. DCDP is built around a temporary, bidirectional (*logical*) distribution tree that spans all subnets. DCDP, moreover, maintains no state beyond its own configuration information and does not use any periodic messages.

DCDP uses a transactional model whereby nodes are either requestors of or responders to individual configuration requests. A requester asks for configuration information from a DCDP entity. The DCDP responder subleases part of the available address pool and gives other configuration information to the requesting node. By recursively splitting the address pool down the distribution hierarchy, DCDP can automatically distribute address pools to each link.

DCDP Characteristics —

Scalable: DCDP is a top-down modular protocol whereby configuration information is distributed without central control or global knowledge. DCDP also leaves the formation of individual IP links to a separate LCP. Alternative bottom-up (e.g., clustering-based solutions) or centralized approaches are not suitable for use in large dynamic topologies.

Aggregatable Addresses: To distribute the available pool to another DCDP requestor, DCDP uses a very simple binary splitting approach: it splits the currently available pool into two equal halves. However, when responding to an LCP request DCDP responds by simply providing a (configurable) chunk of addresses (say 256). This simple partitioning rule

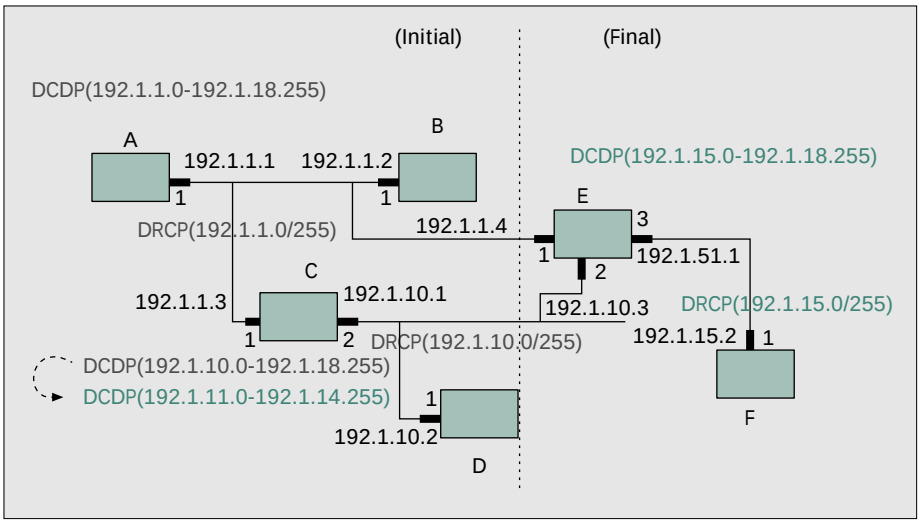

Figure 1. *Network autoconfiguration via DCDP + DRCP.*

allows the use of compact CIDR-like notation for the address pools or chunks, simplifying routing and significantly reducing the length of DCDP packets. It also proves very robust in dynamic topologies; when combined with address reclamation, unused addresses can always be used elsewhere (even if not providing the optimal hierarchy).

Robust: DCDP provides several unique features for efficient and rapid network reconfiguration. DCDP address pools have a priority associated with them: positive priorities identify pools with global validity, and negative priorities imply locally generated pools. Such priority allows DCDP to efficiently reconfigure addresses when networks merge or split. Since the merger of private pools can give rise to possible addressing conflicts, DCDP allows one pool to *poison* the other pool, thus automatically reconfiguring one of the merged subnetworks. DCDP automatically overwrites lower-priority configuration parameters, allowing networkwide renumbering simply by manually configuring a higher-priority (fresher) pool at a single node in the network.

An Illustrative Example — To explain DCDP operation, we consider its use, in conjunction with the DRCP subnet configuration protocol, in configuring an entire network. We have implemented and verified this operation of DCDP and DRCP (in Linux) on our prototype testbed with up to 15 laptops in seven networks [11, 12]. Here we give an illustrative example in a smaller network.

Consider the topology in Fig. 1 and assume that only nodes A, B, C, and D (to the left of the vertical dotted line) are present initially. The DRCP process on all nodes initially tries to configure their interfaces, but fails to do so in the absence of response from any DRCP server node. The DRCP process on each node then asks their DCDP process (if one exists) for configuration information. If there is no pool, this also fails.

In this example let us now assume that A's DCDP is given an address pool, say 192.1.1.1.0–192.1.18.255, from our GUI (which allows a user to set this pool). A's DCDP can now give a chunk (192.1.1.0/255) to its DRCP process, which will configure its only interface with the address 192.1.1.1. After node A configures its interface, nodes B and C configure their interfaces marked 1 using DRCP, getting 192.1.1.2 and 192.1.1.3 as their respective addresses.

Interface 2 on C remains unconfigured, however, because no DRCP master process exists for the subnet associated with interface 2. DRCP has advertised the fact that node A is the DCDP server. Node C can then request node A, which uses binary splitting and leases the pool 192.1.10.0–192.1.18.255 to node C and keeps addresses 192.1.2.0–192.1.9.255 . Finally, the DRCP process in node C associated with interface 2 obtains a chunk (192.1.10.1/255) from this DCDP pool and configures the interface with the address 192.1.10.1. DRCP also subsequently configures interface 1 of node D with the address 192.1.10.2.

To carry the example a step forward, consider what happens when nodes E and F show up. While DRCP is adequate to configure interfaces 1 and 2 on E with addresses 192.1.1.4 and 192.1.10.3, respectively, interface 3 on E, as well as node F, cannot be configured by DRCP alone. Accordingly, node E issues a request for DCDP pools to the candidate DCDP nodes, A and C. Both then split up their available DCDP address pools using binary splitting and offer address pools 192.1.6.0–192.1.9.255 and 192.1.15.0–192.1.18.255, respectively, to node E. Since both offered pools are of the same size, node E accepts any one offer (say 192.1.15.0–192.1.18.255) and confirms the lease to node C; it subsequently allocates the chunk (192.1.15.0/255) to the DRCP process associated with interface 3. DRCP then configures interface 3 on node E with the address 192.1.15.1 as well as interface 1 on node F with the address 192.1.15.2.

While we have discussed the use of DCDP for network address configuration, it should be clear that DCDP can distribute additional configuration parameters, such as DNS server location. In one experiment, we had DCDP and DRCP not only distribute the DNS location, but also put the dynamically allocated addresses in the DNS database.

DCDP Performance

DCDP provides rapid autoconfiguration. The total configuration latency of the network is essentially proportional to the

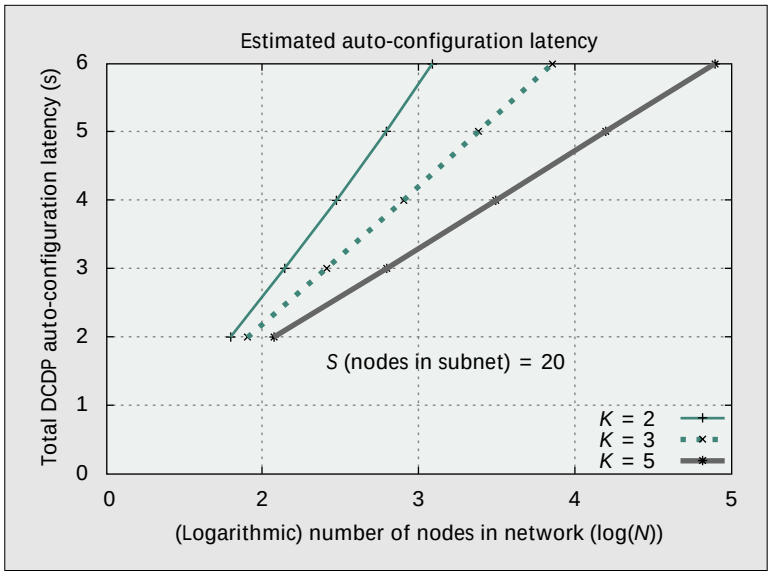

Figure 2. *DCDP auto-configuration latency for varying network sizes.*

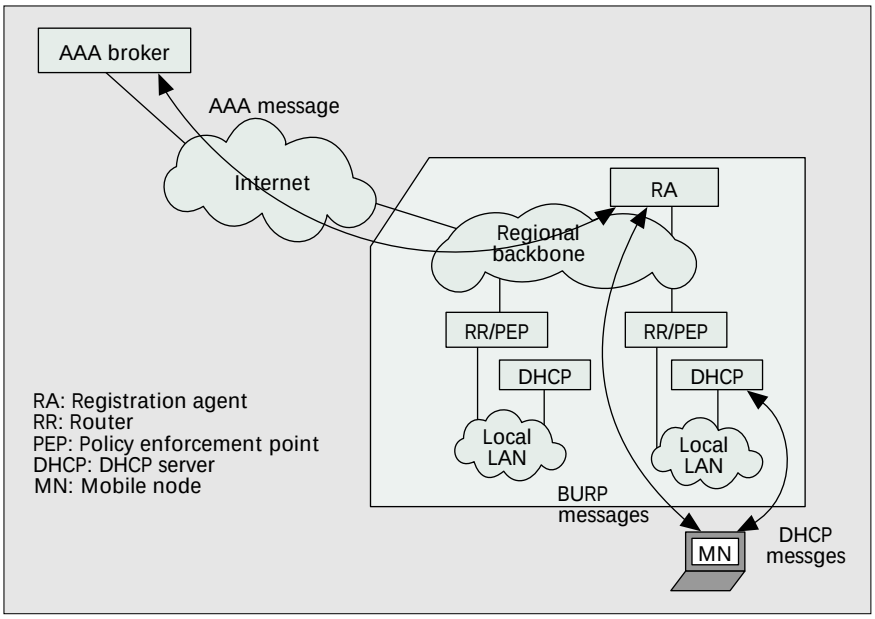

Figure 3. *A network access using BURP: an example scenario.*

height of the virtual spanning tree. This configuration time is proportional to the log of the number of nodes. More exactly, consider a full K-ary DCDP tree, where each node represents the DCDP server for a subnet. Assume also that each subnet has S nodes. A DCDP tree with a depth of d thus has $K^d - 1/K - 1$ distinct DCDP nodes, and the corresponding network comprises a total of $S * K^d - 1/K - 1$ nodes. Figure 2 shows the estimated autoconfiguration latency for $S = 20$ and $K = 2, 3$, and 5 , as the number of total nodes in the network increases. The initial offset is due to the LCP (DRCP) requesting DCDP configuration on an interface only after ~ 1 s latency. The graph is extremely encouraging. For example, if $S = 20$ and $K = 3$, DCDP can configure 7280 nodes ($d = 6$) in only ~ 6 s!

Seamless User Regristration

In this section, we first discuss why a distinct link-layer independent registration protocol is necessary in future pervasive access scenarios. We then explain why current approaches to user registration fall short of the requirement for a customizable access to network resources and present our initial thoughts and naive design of the Basic User Registration Protocol (BURP).

Why User Registration?

To enable a user to access their *individualized* network services over a publicly shared access infrastructure, it is not enough to simply **configure** the user's node with the appropriate IP configuration parameters. To provide intelligent services that extend beyond basic *packet-level connectivity*, the network must associate the identity of the user with the specific configured device. It is only by providing a secure **registration** mechanism that the network can identify the identity of a user, and consequently the access and service privileges associated with that node. Such determination is extremely important in ensuring the commercial viability of this pervasive access paradigm. As an example, consider an airport terminal offering public 802.11 LAN-based wireless access to the Internet. In this scenario DHCP is preferred to PPP, since it can provide configuration parameters (e.g., a valid IP address) without any unnecessary framing overhead. While basic connectivity may be a user-agnostic service, enhanced services will be available only to appropriate user subsets. Thus, common users may only obtain the complimentary basic Web access; however, higher-priority users, such as airport employees or travelers with pre-existing agreements, may be able to obtain additional location-based services, such as access to the current terminal layout information or the nearest printer. Other users may need premium QoS support for real-time applications, such as VoIP. Clearly, a generic flexible registration scheme is needed to establish the context for authenticated and accountable access to higher service abstractions.

Current Solutions

In today's world, registration and configuration is usually a part of the configuration protocol used. For example, Internet service providers (ISPs) currently use RADIUS [14] over PPP [5] for authentication and authorization of their dialup users. The network provides a dumb pipe and all-or-nothing access. At present, LAN-oriented configuration protocols, such as DHCP or DRCP, have no support for user registration; they only provide a valid address to a node in the network. The recently proposed IEEE 802.1X [13] mechanism does provide a port-based authentication scheme for wireless LAN users; however, this is again 802.1X-specific and also provides all-or-nothing connectivity.

Current IP mobility solutions, on the other hand, integrate registration and configuration support with a specific binding mechanism. Binding is the mechanism by which the mobile user informs a centralized registry of its current location, thus allowing oneself to be locatable by others. Most approaches typically combine the registration phase implicitly with either the binding or autoconfiguration functions. For example, mobility management solutions, such as MIP or Session Initiation Protocol (SIP), integrate registration with the binding function. Registration in MIP consists of negotiating lifetimes and authentication information with the foreign agent (FA), as well as the home agent (HA), *as part of the binding update process.* Additionally, MIP has recently joined forces [8] with AAA protocols (e.g., Diameter [15] or RADIUS) to provide a mechanism by which the HA interacts with AAA servers to verify the identity and rights of a specific user. A SIP-based mobility solution, while using different protocols and message formats, also follows a similar approach, where the user is authenticated at the SIP server during the processing of a SIP REGISTER message. These solutions are effective only when the node uses either MIP or SIP to support mobility management. The airport access scenario, however, provides an easy example of cases where such MIP or SIP-based binding may not be required. A mobile worker simply accessing the Web (*pull* model) does not need any binding functionality since there is no need for continuous locatability or in-session packet redirection. A flexible registration protocol, independent of any specific configuration or binding mechanism, is thus clearly needed.

Basic User Registration Protocol

BURP is our attempt to develop a common access-technology-independent higher-layer protocol that allows a user to register in the local network by providing identity and authentication information to the local network. The network can then use the AAA infrastructure (Fig. 3) to validate the user for authorization and accounting purposes. BURP provides a mechanism to achieve seamless registration and access control in environments where user/node configuration is performed via proto-

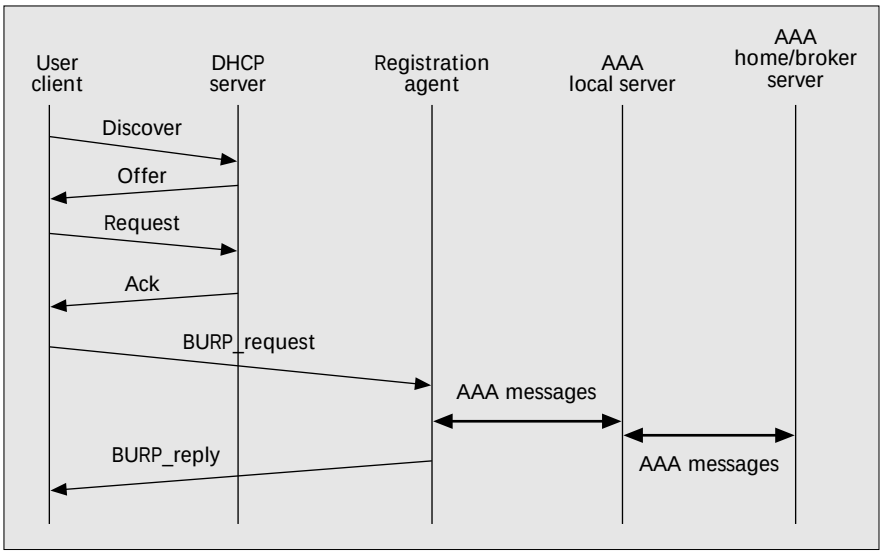

Figure 4. *BURP message flow.*

cols such as DHCP, DRCP, or IPv6 stateless autoconfiguration [9]. BURP is a higher-layer protocol and interacts with a registration agent (RA) in the local network. In this particular example scenario (Fig. 3), the RA resides at the first hop router. For flexible access control and authentication, it may be necessary to place the RA in a separate server in the local network beyond the edge router; in such cases, it is important to devise a mechanism that informs the user of the location or address of the RA (server). We believe this information can be carried as extensions or options to traditional protocols, such as ICMP router advertisement or LCPs (e.g., DHCP or DRCP). If such extensions are not available, the user must have a fallback mechanism to discover the server location.

Unlike autoconfiguration and mobility management protocols, the registration protocol design and specifications are still rather immature; we therefore focus more on the generic requirements/features of BURP and illustrate the protocol message flow in the following:

- BURP is a simple user-network protocol and works for both IPv4 and IPv6.
- BURP is independent from node configuration protocols. It does not provide mobility support, but works with any mobility protocol, such as Mobile IP.
- The BURP client interacts only with a local RA, which may reside on any node in the local domain.
- BURP does not control any firewall/policer directly (to control the packet forwarding), but can work with any policing protocol, such as COPS.
- The RA does not exchange any new interdomain AAA message, but works with any AAA protocol (e.g., Diameter or RADIUS).
- BURP allows various ways of identifying a user, such as NAI [17] and FQDN. However, one default globally unique identifier specific to this protocol will be supported.
- BURP creates a local security association (LSA) between a visiting client (user) and access router (server) in the visited network. However, it does not assume that the client and server will share preestablished LSA or public key certificates.
- BURP has a flexible mechanism for specifying extensible support for various authentication schemes.
- BURP offers protection against replay and man-in-the-middle attacks.
- BURP supports challenge/response authentication whenever necessary.
- The BURP client delivers all the user parameters required by an AAA protocol.

By using BURP, network providers in future pervasive computing environments will have better information and control of network usage. Being a higher-layer protocol, BURP requires no change to the TCP/IP stack and can easily be implemented on a variety of devices with varying operating systems. Figure 4 presents an example BURP message flow in an environment where DHCP is used as configuration protocol. Once the interface configuration phase is over, the BURP client sends a registration request (BURP_REQUEST) to the RA, which in turn replies (BURP_REPLY) to the client after proper authentication. We assume that the RA acts as an adapter, with one interface logically understanding BURP messages and the other communicating with an AAA protocol.

The BURP_REQUEST may include an authentication token using a preestablished security association with its AAA home (AAAH) or AAA broker (AAAB). The RA will then contact the AAA local (AAAL) for authentication. Receiving an AAA request from the RA, AAAL will do the network-to-network AAA using Diameter messages and obtain the keys to establish an LSA with the client (from the AAAH or AAAB). It is possible for the AAAH to send challenges or other requests which may trigger a BURP_AAA_CHALLENGE message from the RA to the user. Once the RA receives a response from AAAL it sends a BURP_REPLY to the client. There are two types of BURP reply: BURP_ACK and BURP_NACK, which allow or deny access to the network, respectively.

Scalable Hierarchical Mobility Management

Given the wide variety in device capabilities, access technologies and user profiles, a mobility solution for pervasive computing must offer customizable 'link-layer independent' mobility support. Such support can range from ensuring simple intermittent backbone connectivity to seamless redirection of ongoing connections during node movement. In this section, we consider the shortcomings of current IP-based mobility solutions and then provide an overview of our hierarchical Dynamic Mobility Agent (DMA) architecture.

Why Mobility Management?

While managing user and node mobility is important even in current networking environments, the pervasive arena introduces additional constraints and challenges which must be addressed. Current IP mobility solutions have very limited deployment; moreover, different device sets (e.g., pagers, cellular phones, and PDAs) are managed by logically (often non-IP) separate networks, each customized to a specific service profile. In contrast, the pervasive vision assumes that a single management infrastructure will manage the mobility of potentially billions of such heterogeneous devices. Application and service profiles will exhibit large variations in their mobility needs. To ensure that a common infrastructure can support such device and application heterogeneity, we need to make the mobility solutions extremely *customizable*. In particular, the access infrastructure should allow the use of one or more global binding protocols.

Current Solutions

Mobile IP [7], the standard solution for IP mobility management, attempts to maintain any existing network layer connections by redirecting packets addressed to the mobile node's (MN's) *permanent home address* to its temporarily assigned and topologically correct *care-of address* (CoA). A specific node, the HA, located in the MN's home network is responsible for acting as the MN's global point of contact and performs this redirection by intercepting and tunneling packets to the CoA. However, as documented in [18], MIP and enhancements thereof (e.g., Mobile IPv6 [19]) all lack a hierarchical framework and suffer from drawbacks such as high update latency, high global signaling overhead, and lack of support for paging and fast handoffs. The Internet Engineering Task Force (IETF) is currently investigating techniques to improve the handoff latency [20] in MIP. Extensions to SIP have also been proposed [21] to provide application-layer mobility support. By using an application-layer mobility management approach, SIP-based mobility makes individual applications aware of and responsible for managing host mobility. However, such a solution still suffers from the absence of a hierarchy, and ties the user to a single binding protocol.

To address the problems of latency and global signaling, several hierarchical IP management techniques have been proposed recently. By localizing most of the mobility-related updates to the current *domain*, all these protocols alleviate the scaling and latency concerns to a significant degree. Among the alternative proposals, Cellular IP [22] and HAWAII [23] define host-based routing approaches, whereby the MN maintains a single domain-wide CoA and routing tables are appropriately modified to reflect the MN's current point of attachment. In contrast, mechanisms such as Regional Tunnel Management [24] and Hierarchical MIP [25] associate an MN with multiple CoAs, each resolving the MN's location at a particular depth in the management hierarchy. All these proposals, however, implicitly assume the use of MIP as a global binding technique.

The DMA Architecture

The Dynamic Mobility Agent (DMA) architecture is a two-level, hierarchical mobility management technique that separates intradomain from global (interdomain) mobility management. The DMA architecture uses the Intra-Domain Mobility Management Protocol (IDMP) [26] to manage intradomain mobility, as well as to support optional features such as fast handoff and paging. The architecture is based on having a mobility agent (MA) manage all the local (intradomain) mobility support desired by a specific MN; by using two separate CoAs, intradomain mobility becomes completely transparent to the global Internet.

From the pervasive networking viewpoint, key elements of the DMA design include:

- **Independence from a global binding protocol**: Not only can IDMP be combined with multiple global binding protocols such as MIP or SIP, such a global binding mechanism can be completely absent.
- **Customizable intradomain mobility support**: IDMP allows MNs and users to request layer-independent support for features such as fast handoffs and paging. Moreover, users are free to request such optional support features only if required by their specific application suite.
- **Customizable QoS Support**: The DMA architecture uses a hierarchical Differentiated Services (Diffserv) framework to integrate QoS assurances with mobility management. Network nodes are responsible for ensuring not just basic connectivity, but also the necessary level of QoS guarantees, as an MN roams within the domain.

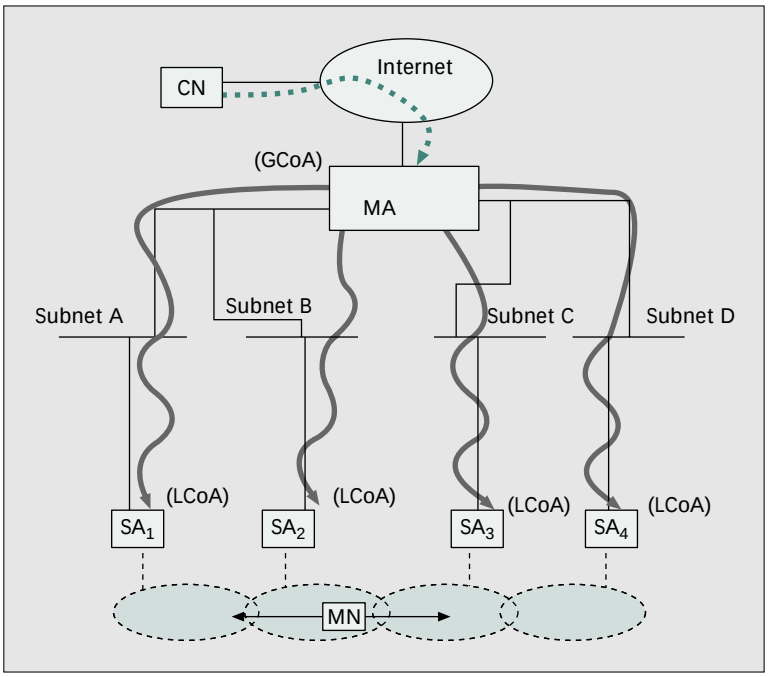

Figure 5. *IDMP logical elements and architecture.*

Figure 5 depicts the functional layout of IDMP. The MA is similar to an FA of MIP, except that it resides higher in the network hierarchy (than individual subnets) and provides the MN a stable point of attachment throughout the domain. Subnet agents (SAs) are functionally very similar to Mobile IP FAs and manage the configuration of MNs at each individual subnet. An MN has two separate CoAs:

- *Global CoA* (GCoA): This identifies the MN's current location only up to a domain-level granularity and remains unchanged as long as the MN stays in the current domain. This is the address used by global binding protocols such as MIP or SIP.
- *Local CoA* (LCoA): This has only local (domain-wide) validity and identifies the MN's present subnet of attachment. On every change in subnet, the MN obtains a new LCoA and informs its MA of this new *local* binding.

Figure 6 shows the potential IDMP messaging flow (including the QoS-related signaling, which will be explained later) for initial movement into the domain. In addition to the LCoA (which changes with every change in subnet), IDMP's configuration phase provides a newly arrived MN with a designated MA and a GCoA. Packets from a remote CN, tunneled or directly transmitted to the GCoA, are intercepted by the MA and then forwarded (by reencapsulation) to the MN's LCoA.

Optional Features: Fast Handoff, Paging, and QoS Support

IDMP provides customizable, link-layer-independent support for certain features, such as fast handoffs, paging, and QoS assurances, that are logically independent of the global binding protocol used. Both fast handoff and paging support in the DMA architecture use some form of multicasting and are logically represented in Fig. 7. An MN desiring fast handoff support during an impending change in the point of attachment

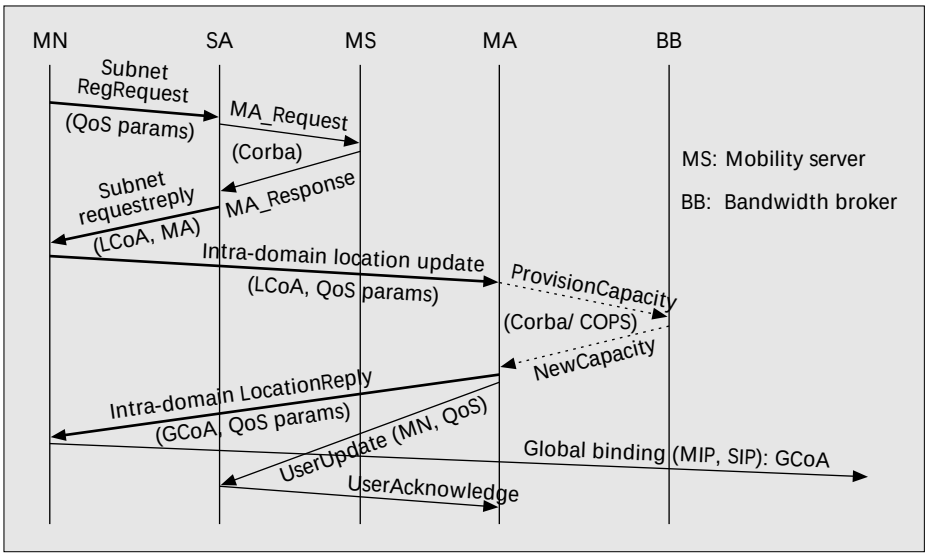

Figure 6. *IDMP message flow (with QoS extensions using BB).*

sends a *MovementImminent* message to the MA whenever it senses (via layer 2 triggers) the possibility of movement. The MA then proactively multicasts inbound packets (solid lines in Fig. 7), for a limited duration, to the SAs that are neighbors of the MN's current point of attachment (to SA_1 and SA_3 in Fig. 7), where such packets are temporarily buffered. Such proactive buffering not only reduces or eliminates the handoff packet loss, it also ensures that the MN's packets are available immediately after it attaches at the new SA. IDMP's paging operation is functionally very similar to the fast handoff mechanism. In the paging mode, an *idle* MN does not perform any location update or registration as long as it stays within a paging area (PA) comprising multiple subnets. On receipt of an incoming packet for an idle MN, the MA buffers it and multicasts a *PageSolicitation* (dashed lines in Fig. 7) to the MN's current PA (PA_2 in Fig. 7), requesting the MN to reregister at the MA with a new and currently valid LCoA. Further details on paging and fast handoffs in DMA are available in [27].

To provide redundancy and scalability, the DMA solution uses load balancing algorithms to dynamically distribute incoming MNs among the candidate MAs. To provide integrated QoS support, DMA uses a centralized bandwidth broker (BB)-based approach, which leverages the Diffserv framework, to dynamically provision resources for MNs as they move within the domain. When a user first registers in a new DMA domain, it can signal its QoS requirements. These requirements are relayed to the mobility server (MS) (MA_Request message in Fig. 6), which uses this information to assign the MN an appropriate MA. On subsequent movement within the domain, the MA is responsible for transferring (*UserUpdate* message in Fig. 6) the MN's QoS profile to the new SA, eliminating the need for QoS re-negotiation by the MN. Any resource provisioning within the domain is handled by the MA through appropriate requests (*ProvisionCapacity* message in Fig. 6) to the BB. Full details of the mechanism and architecture for optional QoS support are provided in [28].

We have implemented the IDMP functional specifications by modifying the Stanford University Mobile IP Linux code and have tested its operation on our testbed [29]. We are currently working to demonstrate the utility of different load-balancing algorithms and the BB-based QoS provisioning architecture.

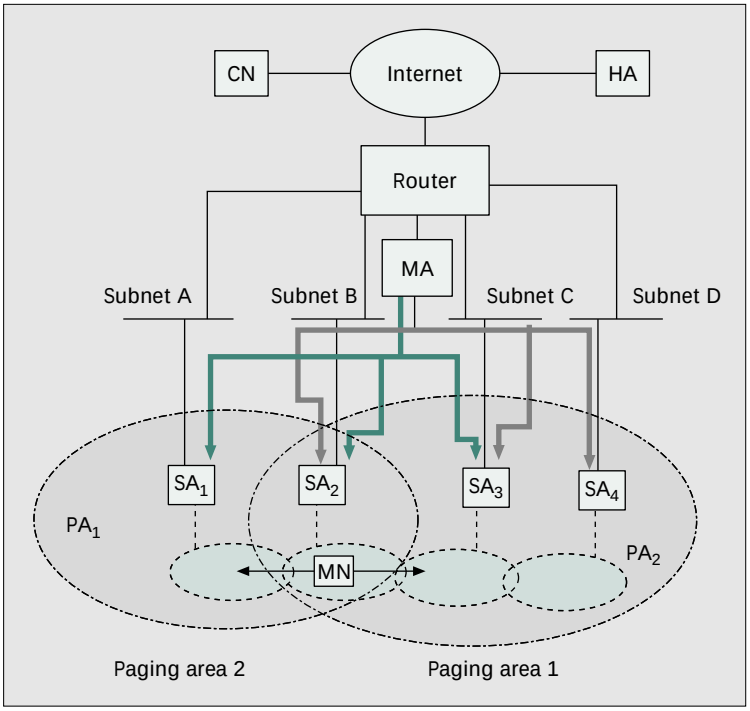

Figure 7. *IDMP fast handoff/paging.*

Conclusion

Pervasive computing will usher in a quantum increase in the number of networked nodes and also lead to application heterogeneity, increased dynamicity of the network topology, and the use of diverse link layers. To face these future challenges, we must enhance many existing network protocols. Here, we have argued about the types of enhancements needed to IP-layer autoconfiguration, user-to-network registration, and mobility management solutions.

We first show why manual configuration of individual hosts and nodes is impractical in future networks, characterized by a significantly larger number of networked nodes and considerably more dynamic topologies. We then describe our Dynamic Configuration Distribution Protocol (DCDP) for autoconfiguring large networks with IP addresses and other information. DCDP provides a technology-independent bidirectional spanning-tree-based approach for robust and rapid network autoconfiguration. Our current prototype is based on IPv4; we believe that additional research is necessary to enhance the protocol for IPv6 by leveraging the richer semantics of IPv6 addressing. To use DCDP as a tool for generic information dissemination, it needs to be interfaced to other service discovery mechanisms, such as Bluetooth's Service Discovery Protocol specification or the IETF's Service Location Protocol [30].

We then describe why service providers require the development of a standardized mechanism to authenticate a user to the network. Such authentication should be independent of the configuration and binding protocols (unlike current solutions

such as PPP or MIP) and will be an important ingredient in developing the pervasive vision of user-specific context-sensitive services. Our Basic User Registration Protocol provides such a simple UDP-based mechanism; however, we need to perform additional research to understand the impact of QoS and policy negotiations on the BURP authentication process.

We finally present the flexible DMA approach for mobility management. DMA uses IDMP, a hierarchical protocol that allows an MN the flexibility of specifying localized mobility features of interest. The DMA approach also permits different users to use one or more global binding solutions, as appropriate, to provide any needed global reachability. The DMA approach combines bandwidth-broker-based dynamic provisioning with appropriate mobility agent assignment algorithms to provide an integrated framework for QoS support.

While the protocols presented here significantly enhance the capabilities of future pervasive networks, several additional problems, such as security, still need to be completely worked out. The memory and processing requirements of our solutions are other important issues needing further investigation. For successful application in the pervasive environment, the protocols must be lightweight enough to be deployed in handheld and other capacity-constrained devices.

References

[1] P. Rysavy, "General Packet Radio Service (GPRS)," *GSM Data Today J.* (online), Sept. 1998.

[2] E. Lycksell *et al.*, "IMT-2000 Standards: Network Aspects," *IEEE Pers. Commun.*, vol. 4, Aug. 1997, pp. 20–29.

[3] Wireless LAN Alliance, "The IEEE 802.11 Wireless LAN Standard," http://www.lana.com/intro/standard/intro.html

[4] R. Mettala, Ed., "Bluetooth Protocol Architecture, v. 1.0," Aug. 1999, http://www.bluetooth.com/developer/download/download.asp?doc=175

[5] W. Simpson, "The Point to Point Protocol (PPP)," Internet STD 51, July 1994.

[6] R. Droms, "Dynamic Host Configuration Protocol," IETF RFC 2131, Mar. 1997.

[7] C. Perkins, "IP Mobility Support for IPv4, revised," draft-ietf-mobileip-rfc2002-bis-02.txt, IETF, July 2000, work in progress.

[8] C. Perkins, "Mobile IP Joins Forces with AAA," *IEEE Pers. Commun.*, Aug. 2000, pp. 59–61.

[9] S. Thomson and T. Narten, "IPv6 Stateless Address Autoconfiguration," IETF RFC 2462, Dec. 1998.

[10] A McAuley *et al.*, "Dynamic Registration and Configuration Protocol," draft-itsumo-drcp-00.txt, IETF, July 2000, work in progress.

[11] A. McAuley and K. Manousakis, "Self Configuring Networks," *Proc. 4th Adv. Telecommun. and Info. Dist. Res. Conf.*, College Park, MD, Mar. 2000.

[12] K. Manousakis *et al.*, "Configuring an Entire Network with DCDP/DRCP," *Proc. 5th Adv. Telecommun. and Info. Dist. Res. Conf.*, College Park, MD, Mar. 2001.

[13] Tony Jeffree, Ed., "IEEE Draft P802.1X/D10: Standards for Local and Metropolitan Area Network: Standard for Port Based Network Access Control," Jan. 16, 2001.

[14] C. Rigney *et al.*, "Remote Authentication Dial In User Service (RADIUS)," IETF RFC 2865, June 2000.

[15] P. Calhoun *et al.*, "Diameter Base Protocol," draft-calhoun-diameter-17.txt, Sept. 2000, work in progress.

[16] S. Das, A. Mcauley and B. Patil, "Basic User Registration Protocol (BURP)," Minutes, AAA-WG, IETF-48, Pittsburgh, PA, July 2000, work in progress.

[17] B. Aboba and M. A. Beadles, "The Network Access Identifier," RFC 2486, Jan. 1999.

[18] S. Das *et al.*, "TeleMIP: Telecommunication Enhanced Mobile IP Architecture for Fast Intradomain Mobility," *IEEE Pers. Commun.*, Aug. 2000, pp. 50–58.

[19] D. Johnson and C. Perkins, "Mobility Support in IPv6," draft-ietf-mobileip-ipv6-13.txt, IETF, Nov. 2000, work in progress.

[20] K. El Malki, Ed., *et al.*, "Low Latency Handoffs in Mobile IPv4," draft-ietf-lowlatency-handoffs-v4-00.txt, IETF, Febr. 2001. work in progress.

[21] E. Wedlund and H. Schulzrinne, "Mobility Support Using SIP," *Proc. 2nd ACM Int'l. Wksp. Wireless Mobile Multimedia*, Aug. 1999, pp. 76–82.

[22] A. Campbell *et al.*, "Cellular IP," draft-ietf-mobileip-cellularip-00.txt, IETF Jan. 2000, work in progress.

[23] R. Ramjee *et al.*, "IP Micro-Mobility Support Using HAWAII," draft-ietf-mobileip-hawaii-01.txt, July 2000, work in progress.

[24] E. Gustafsson, A. Jonsson and C. Perkins, "Mobile IP Regional Tunnel Management," draft-ietf-mobileip-reg-tunnel-04.txt, Mar. 2001, work in progress.

[25] H. Soliman *et al.*, "Hierarchical MIPv6 Mobility Management," draft-soliman-mobileip-hmipv6-02.txt, IETF, Feb. 2001, work in progress.

[26] A. Misra *et al.*, "IDMP: An Intra-Domain Mobility Management Protocol Using Mobility Agents," draft-mobileip-misra-idmp-00.txt, IETF, July 2000, work in progress.

[27] A. Misra *et al.*, "IDMP-Based Fast Handoffs and Paging in IP-Based Cellular Networks," *Proc. IEEE 3GWireless Conf.*, 2001, San Francisco, pp. 427–32.

[28] A. Misra *et al.*, "Integrating QoS Support in TeleMIP's Mobility Architecture," *Proc. IEEE Int'l. Conf. Pers. Wireless Commun.*, Hyderabad, Dec. 2000, pp. 57–64.

[29] K. Chakraborty *et al.*, "Implementation and Performance Evaluation of TeleMIP," *Proc. ICC 2001*, Helsinki, Finland, June 2001.

[30] E. Guttman, C. Perkins, J. Veizades and M. Day, "Service Location Protocol, Version 2," IETF RFC 2608, July 1999.

Biographies

ARCHAN MISRA (archan@us.ibm.com) is currently a research staff member at IBM T J Watson Research Center, Hawthorne, New York, where he is investigating issues related to providing a customizable architecture for seamless, secure, and mobile access to heterogeneous applications over multiple wide-area and local access technologies. Prior to joining IBM, he spent over three years as a research scientist at Telcordia Technologies, where he was involved in the design of network-layer mobility and auto-configuration schemes and in analyzing scalable architectures for QoS provisioning for VoIP and data in IP networks. He is co-developer of the IDMP and DCDP protocols and has published over 30 research papers in the areas of mobility management, network auto-configuration, Internet congestion control, and buffer management. He has developed several prototypes and enhancements for Mobile-IP-based seamless connectivity over various wireless networks and is currently very interested in the security and authentication aspects of mobile computing. He received his M.S. and Ph.D. degrees in electrical and computer engineering from the University of Maryland at College Park in 1996 and 2000, respectively, and his B.Tech in electronics and communication engineering from the Indian Institute of Technology, Kharagpur, in 1993.

SUBIR DAS [M] (subir@research.telcordia.com) received an M.Tech. degree from the Institute of Radio Physics and Electronics, Calcutta University, in 1990, and a Ph.D. degree in 1996 from the Indian Institute of Technology, Kharagpur, India. Since 1999 he has been at Telcordia Technologies and is currently a research scientist in the Wireless IP Research Group. During 1997–1999, he was a faculty member in the Electronics and Electrical Engineering Department, Indian Institute of Technology, Kharagpur. He has designed several protocols and architecture for next-generation wireless networks, particularly in the areas of auto-configuration, registration, and mobility management. He has also developed several prototypes for Mobile IP and SIP-based seamless connectivity in wireless LAN-based networks. His current research interests include mobility management in next-generation wireless access systems, wireless multimedia, user registration, and auto-configuration of ad hoc mobile networks. He is very active in IETF.

ANTHONY MCAULEY (mcauley@research.telcordia.com) received his Ph.D. from Hull University, England, in 1985. He was a research fellow at Caltech from 1985 to 1987. Since 1987 he has been at Telcordia and is currently a director in the Wireless IP Research group. He created and works on projects including protocols for complete network auto-configuration (including DRCP and DCDP), self-managed virtual networks and architectures, and protocols for future IPv4 and IPv6 wireless access. He has built several mobile internetworking systems on Linux that included software he wrote for auto-configuration, protocol boosters, multicast proxies, and a transport protocol. In the past, he worked on IP multicasting in ad hoc and large-scale networks (including the Comprehensive Test Ban Treaty network), efficient error correction and detection codes, and VLSI chip design for everything from microprocessors to asynchronous packet switches.

SAJAL K. DAS [M] received a B.Tech. degree in 1983 from Calcutta University, an M.S. degree in 1984 from the Indian Institute of Science at Bangalore, and a Ph.D. degree in 1988 from the University of Central Florida at Orlando, all in computer science. Currently he is a full professor of computer science and engineering and alsofounding director of the Center for Research in Wireless Mobility and Networking (CReWMaN) at the University of Texas at Arlington. During 1988–1999, he was on the faculty of computer science at the University of North Texas, Denton. He is a recipient of the Honor Professor Award from UNT in 1991 and 1997 for best teaching and scholarly research, and UNT's Developing Scholars Award in 1996 for outstanding research. His current research interests include resource management in wireless networks, mobile computing, QoS provisioning, wireless Internet, mobile multimedia, distributed/parallel computing, and network performance modeling and simulation. He has published over 170 research papers in these areas and directed several funded projects. He is a member of the ACM.