End-system based Mobility Support in IPv6[1]

Chuanxiong Guo, Haitao Wu, Kun Tan, Qian Zhang, Jingmin Song, Junfeng Zhou, Christian Huitema, and Wenwu Zhu

ABSTRACT

Numerous mobility solutions have been proposed in the past, but none of them have been widely deployed today. To address the deployment difficulty in previous work, we propose an *end-system based* mobility solution for IPv6 (EMIPv6). In our design, we adhere to the end-to-end principle [1] by *directly* performing connection maintenance and data packet delivery between the two communicating hosts. And we leverage distributed hash table-based peer-to-peer (P2P) systems to carry out self-organized, scalable and robust name lookup for mobile hosts. When the mobility messages such as address updates cannot be delivered directly between the end hosts (e.g., due to firewalls, NATs, or simultaneous movement), we propose a distributed subscription/notification (S/N) service on top of the previously introduced P2P overlay to deliver them. This leads to a complete end-system solution with small handoff latency and efficient packet delivery. Our simulation results showed that our scheme achieves small name resolution latency by considering host heterogeneity into the design. We have implemented EMIPv6-based end-systems. The experiments in our testbed demonstrated that a complete end-system based mobility solution is technically feasible and is easy to deploy in the real world without the need of introducing new network components.

Keywords: Mobility management, name resolution, connection continuity, distributed subscription/notification service, peer-to-peer overlay.

1. INTRODUCTION

With rapid progresses in wireless, Internet, and embedded system technologies, more and more Internet applications such as Web browsing, e-mail, Instance Messaging, and VoIP (voice over IP) are available on mobile devices such as laptops, PDAs, and cellular phones. However, due to the mobility nature of mobile devices, Internet-based applications face challenges in wireless networks. When a mobile device moves across wireless networks, the location of the device may change frequently, and as a result, its IP address may change accordingly. The current Internet uses IP address for both host identification and packet routing. Due to IP address change (e.g., when a mobile host roams between an 802.11 WLAN and a 3G network), on-going connections of a mobile

[1] Manuscript received Nov. 30, 2004; revised May 29, 2005. Chuanxiong Guo is with the Institute of Communications Engineering, Nanjing 210007, China (xguo@ieee.org). Haitao Wu, Kun Tan, Qian Zhang, Jingmin Song, Junfeng Zhou are with Microsoft Research Asia, Beijing 100080, China (t-hwu, kuntan, qianz, t-jmsong, t-jfzhou@microsoft.com. Christine Huitema is with Microsoft Corporation (huitema@microsoft.com). Wenwu Zhu is with Intel Corporation (wenwu.zhu@intel.com). This work was performed when Chuanxiong Guo and Wenwu Zhu were associated with Microsoft Research Asia.

host will break and peer nodes may have difficulty to locate a mobile host (that is, identify its IP address. Location management and name to IP address resolution are used identically in this paper). Two key challenges for mobility support in wireless IP networks are therefore how to maintain continuity of the on-going connections and how to provide location management.

To date, many mobility management schemes have been proposed to address these two issues (e.g., [2-15]). For connection maintenance, solutions such as Mobile IP (MIPv4 and MIPv6) [2-3], ROAM [13], and Mobile Tapestry [14] introduce yet another level of indirection in the network using rendezvous points, from which a mobile node can always send and receive data packets. We note that rendezvous points, however, cause packet delivery inefficiency and are not necessarily needed.

For location management, many schemes [5-7] rely on dynamic DNS (DDNS) [16] or similar methods [8-11] to track the location of a mobile device. Though there are still no address update traffic measurements for mobile IP networks, the measurements at cellular networks showed that the location updating rate of cellular phones can be 10 times that of the call rate [17]. The current WLAN hot-spots have much smaller coverage than that of cellular systems. Hence it is reasonable to predict that location updates in mobile IP networks will be more frequent than that in cellular networks. These solutions therefore may face single point of failure and scalability problems due to tremendous location updates. And dynamic updates may also impact the performance of existing DNS system, which relies heavily on caching to perform well. Moreover, solutions such as DDNS also need considerable administrative effort. Recent research revealed that the DNS system may suffer from operational errors such as mis-configurations [18].

In our previous work [15], we proposed a seamless roaming solution for vertical handoffs between WLAN such as 802.11b and WWAN such as GPRS and CDMA1X. We designed a connection manager (CM) to actively sense network conditions and a virtual connectivity (VC) to maintain on-going connections. VC and this work share the same design philosophy in that they all adhere to the end-to-end (E2E) principle for connection maintenance. Nonetheless, VC is specifically designed for seamless vertical handover. And since VC is designed to work for both IPv4 and IPv6, it needs transport layer information (i.e., port number) for connection maintenance. Furthermore, VC depends on dynamic DNS for location management and relies on a centralized subscription/notification (S/N) service to handle connection maintenance under NAT (network address translator) [19] and simultaneous movement.

More detailed review of the current mobility management schemes is given in the following section.

In this work, in order to overcome shortcomings of the previous schemes and provide an easy way for deployment, we further develop and implement a complete *end-system* based mobility solution for IPv6 (EMIPv6) by

leveraging P2P technology. For connection maintenance, we introduce an E2E connection maintenance procedure to directly exchange mobility messages between two communication peers. And in the case that mobility messages cannot be directly delivered (e.g., due to firewall/NAT separation and simultaneous movement), we propose a distributed S/N service which is built on top of an already widely deployed P2P overlay, named PNRP (Peer Name Resolution Protocol) [20] to help the E2E procedure to deliver them. For location management, we extend the PNRP overlay, by considering host heterogeneity in overlay construction and routing, to efficiently locate a mobile node no matter where it moves. EMIPv6 has the following attractive properties:

? Ease of deployment and administration

EMIPv6 is a fully end-system based solution. It enhances both mobile and correspondent hosts while keeping the network core untouched. Here we argue that it is easier to upgrade the end hosts than the network core. This is based on the observation that the function provided by the network core is almost the same since the invention of the Internet: forwarding packet as fast as possible. Meanwhile, mobile devices such as cellular phones are upgraded much more frequently.

? Small handoff latency and efficient data packet delivery

Small handoff latency is a critical feature for interactive, real-time applications such as VoIP. EMIPv6 achieves small handoff latency and efficient data packet delivery by performing connection maintenance and data packet delivery between communication peers from end to end.

? Self-organizing, scalability and robustness

EMIPv6 achieves self-organizing, scalability and robustness by leveraging P2P technologies into mobility management. The PNRP [20] overlay is used to carry out self-organized location management for mobile devices. Furthermore, a distributed S/N service is built on top of PNRP for connection maintenance under firewall/NAT or simultaneous movement. In this way, EMIPv6 is able to support a (virtually unlimited) large number of mobile devices, avoid single point of failure, and moreover, it is self-organized so that administrative cost can be minimized.

? Application transparency

Currently, most (legacy) Internet-based applications are developed without considering mobility. It is desirable that these applications can also have mobility support. EMIPv6 achieves application transparency by using a node-pair binding cache (NPBC), which keeps the relationship between the original and current addresses. With NPBC, the application only sees the original addresses no matter how the current addresses change.

? Security

An insecure mobility management scheme may cause more problems than the benefits it brings. EMIPv6 achieves secured mobility management by performing return routability testing and using IPSEC or cryptographically generated addresses (CGA) [21-22] to protect mobility messages.

Though previous schemes may have some of the above properties, to the best of our knowledge, EMIPv6 is the first scheme that possesses all of them. We have implemented EMIPv6 for PCs, PocketPCs, and Smart-phones and performed extensive experiments to verify its properties. Our experiences have convinced us that EMIPv6 is easy to deploy due to the fact that only software upgrade for end hosts is needed and that the PNRP overlay has been already widely deployed.

The rest of this paper is organized as follows. In Section 2, we discuss the related work. We present the detailed design of EMIPv6 in Section 3. We first present an E2E mobility module for connection maintenance; we then introduce a distributed subscription/notification (S/N) service to handle mobility under firewall/NAT and simultaneous movement which cannot be successfully handled only by the two communication peers; after that, we describe a PNRP overly for location management. Implementation details and experiments are given in Section 4. Section 5 concludes the paper.

2. RELATED WORK

Mobility management is one of the key issues to enable a ubiquitous all-IP wireless Internet. In literature, there has been many related works for device location and connection maintenance.

Mobile IP (MIPv4 and MIPv6) [2-3] perhaps is the most influential mobility management scheme. MIP provides mobility support at IP layer. It introduces a home agent in the network. Each mobile node is assigned a global unique home address (HoA). A correspondent node is expected to get the HoA of a mobile node via the existing DNS system. When a mobile node is away from its "home network", it always registers its current care-of address (CoA) at its home agent in its 'home network' (the home agent may be dynamically discovered). The home agent therefore maintains a mapping relationship between the home address and the care-of address of a mobile node. A correspondent node can always send packets to a mobile node via its home agent. In order to improve route efficiency, route optimization is introduced to enable data packet exchange between the communication peers directly. Our design differs from MIP significantly in that we do not need a home agent for either location management or data forwarding.

Cellular IP [23], HAWAII [24], and HMIPv6 [25] are three schemes that provide micro mobility support. They utilize the hierarchical structure of the network to localize address registration and packet routing when mobile hosts are roaming within a homogeneous local network. We note that micro-mobility is an ability that cannot be provided by pure end-to-end mobility schemes such as EMIPv6. This disadvantage, however, can be alleviated

when combining EMIPv6 for macro-mobility management and some link layer mobility solutions (e.g., [26] [27]) for micro-mobility management.

Targeting at overcoming some of the drawbacks (such as performance and survivability) of MIP, in [10], the authors proposed a MIP-LR scheme. The idea of MIP-LR is to introduce a service node named HLR (home location register). Before launching a connection to a mobile host, the peer first queries HLR to get the peer's current IP address. The HLR is not necessarily located at the home network of a mobile node. The authors further introduced translation servers (TS) or quorum consensus (QC) algorithms to reliably locate a HLR [11]. After getting the current address of a mobile node from HLR, the data packet is delivered directly between communication peers without the involvement of HLR. We note that EMIPv6 differs from MIP-LR in that EMIPv6 a self-organizing system whereas MIP-LR needs to deploy the HLR servers. Furthermore, MIP-LR is targeted for enterprise environment whereas EMIPv6 is for Internet-like environment.

HIP (Host Identity Payload) [8, 9] can be considered as a layer 3.5 solution for mobility. It decouples network and transport layers by introducing a statistically global unique Host Identity. In this way the transport connections are bound to Host Identity, not IP address. The mobility issue can therefore be solved by mapping different IP addresses to the unchanged Host Identities. Different from HIP, our observation in EMIPv6 is that, we can resolve the mobility problem by utilizing the addressing architecture of IPv6 without introducing yet another global ID space at transport layer.

TCP-R [5] and Migrate [6] are transport layer approaches to solve mobility for TCP. Both of them extend TCP states and introduce new procedures to maintain connection continuation for TCP during handover. A mobile host always tries to update its current IP address to its peers by using the newly introduced mobility procedure when it moves to a new wireless network. In [28], an extension is proposed for Migrate to support simultaneous movement based on DNS querying. Both TCP-R and Migrate rely on dynamic DNS update to locate a mobile node. We note that relying on dynamic updates in DNS may cause performance issues to the existing DNS system, which relies on extensive caching to perform well. Furthermore, dynamic updates may cause scalability problem if the number of mobile nodes is large and a large number of location update traffic is generated.

At application layer, schemes have been proposed for different applications case by case. For example, SIP [29] can be extended to support mobility by re-sending the INVITE message to the peer to re-establish a session when IP address of a mobile host changes. The application level approaches [29-30] do not need to revise the TCP/IP stack (which usually resides in OS kernel), however, they need to re-establish the connections after address change; hence the handoff latency may be large.

Recently, several P2P based mobility support schemes have been proposed. ROAM [13] is a scheme based on I3 [31], an overlay Internet infrastructure. Since ROAM is an indirection based approach and I3 is built on P2P, ROAM is flexible and scalable in providing mobility support. However, the end-to-end semantic of Internet communication does not hold anymore (this, however, should be considered as a character instead of a drawback, since one of the design goals of ROAM is to achieve location privacy). In Mobile Tapestry [14], the authors use the publishing and routing mechanism of Tapestry, to provide rapid mobility support, and hierarchical mobility can be provided to group mobile devices by further introducing yet another level of indirection. Both ROAM and Mobile Tapestry use rendezvous points in P2P overlay for data packet delivery. Our scheme differs from both ROAM and mobile tapestry in that we only use P2P overlay for name resolution and certain mobility message delivery, whereas data packet forwarding is still performed at IP layer.

We also note that there are several related works on name resolution which are not directly related to mobility management [32-34]. The intentional naming system (INS) [32] uses a descriptive language to describe name (or resource) and a late binding to integrate name resolution and message routing. The purposes of PNRP and INS are different, with INS is intended to resource discovery under small scale dynamic networks (e.g., several hundred to a few thousand nodes), whereas PNRP is to provide DNS-alike service in an Internet alike environment. It is a nature step to use the lookup service provided by the current P2P overlays for name resolution, e.g., [33-34]. In this sense, our PNRP approach is similar with these schemes in that our scheme also achieves $log(n)$ hop routing and needs to maintain an application level routing cache. Besides the detailed technical differences with the existing P2P schemes, a very important characteristic of EMIPv6 is that PNRP is not only for name resolution but also an indispensably part for connection maintenance.

3. END-SYSTEM BASED MOBILITY SUPPORT IN IPV6

There are three major components in EMIPv6: an E2E mobility module, a distributed S/N service, and a PNRP overlay. The E2E mobility module and distributed S/N service are to maintain connection continuity of on-going connections; and the PNRP overlay is to provide location management for mobile devices.

Traditionally, connection maintenance and location management were studied separately. In EMIPv6, these two parts are integrated seamlessly by using end-system based P2P technology: The DHT-based PNRP overlay is used not only for location management, but also to build a distributed S/N service to help the E2E mobility module to perform connection maintenance. This distributed S/N service helps the E2E mobility module to deliver mobility messages in the case when those messages cannot be directly delivered via IP layer. In the following three sub-sections, we present these three parts in detail.

3.1 End-to-End Connection Maintenance

In this sub-section, we focus on the E2E mobility module for connection maintenance. The E2E mobility module is located at IP layer. We first introduce the addressing model and the core data structure, and then introduce the connection maintenance procedure which is used to maintain the consistence of the data structures between communication peers. After that, the IPv6 header extensions which are introduced to carry the newly introduced mobility messages and data packet processing procedures are illustrated in detail. Lastly we address the security issues in EMIPv6.

3.1.1 Addressing Model and Node-pair Binding Cache

One key property of EMIPv6 is that we do not need a unique "home address"[2] for mobile devices to maintain connection continuity. In this design, we use original address (*orig_addr*), current address (*curr_addr*), and previous address (*prev_addr*) concepts. When two peers first begin to communicate, the addresses they use are called original addresses. When a mobile node moves from one network to another, the newly attained address is the current address. The addresses used by a node between original address and current address are called previous addresses. In our addressing model, we assume that the address used by one mobile node will not be reused by another host after the mobile node moves away and releases that address. See Section 3.1.1.1 on how to achieve this. In what follows, the following terminologies are used:

local_orig/curr/prev_addr, represent the local original/current/previous addresses of a mobile node.

remote_orig/curr/prev_addr, represent the original/current/previous addresses of a peer node.

Two peers in communication maintain a same data structure, which we call node-pair binding cache (NPBC) entry. The NPBC data structure is as follows.

```
struct NPBC_Entry {
    local_orig_addr; remote_orig_addr;
    local_curr_addr; remote_curr_addr;
    local_prev_addr_list;
    remote_prev_addr_list;
};
```

where *local_orig_addr* and *remote_orig_addr* are addresses used by a node and its peer at the beginning of communication, respectively. *local_curr_addr* is the current address used by the node, *local_prev_addr_list* is

[2] Keeping the " home address" concept can simplify the design (e.g., no address ambiguity problem). However, we need to introduce additional devices/services to manage the " home addresses". We therefore introduce the " original address" concept.

the list that contains the previous addresses of the node, *remote_curr_addr* is the current address used by its peer, and *remote_prev_addr_list* is the list that contains the previous addresses of the peer.

In EMIPv6, applications only see the original addresses during the communication life time no matter how the current addresses change. The {*local_orig_addr, remote_orig_addr*} pair does not change during its life time after it is created, thereby making mobility transparent to higher-layer protocols and applications. Current addresses are used by the IP layer for routing purpose. *local_curr_addr* is updated once the address of the node is changed, and *remote_curr_addr* is updated once address update from the peer is received.

Note that original address in EMIPv6 is conceptually different from home address in that the original address of a mobile node seen by different peers may be different. For example, suppose node *A* changes its address from IP_{A1} to IP_{A2}, then to IP_{A3}. It starts a connection with node *B* at IP_{A1} and starts a connection with node *C* at IP_{A2}. Under this condition, the original addresses of *A* seen at node *B* and *C* are IP_{A1} and IP_{A2}, respectively.

A node-pair entry can be created when there are packet exchanges between the two communication peers. When to create a node-pair is decided by certain pre-defined rules. The following are some examples:

? Create NPBC entry if there exists VoIP traffic between two communication peers;

? Create NPBC entry if there exist FTP connections between two peers, and the traffic volume exceeds certain threshold;

? Do not create NPBC entry if only HTTP traffic exists between two peers (since HTTP sessions generally are short-lived).

Detailed description of rules is out the scope of this paper.

When there is no packet exchange between a node-pair for a threshold of *T* seconds (default to 3 minutes), the node-pair will be deleted. The expiration timer is renewed for each packet exchanged between the peers.

In NPBC, *local_prev_addr_list* is used to track the address change of the node itself and *remote_prev_addr_list* is used to track the address change of the peer. With the help of the previous address list, a node may be able to simultaneously receive packets from both previous and current addresses of its peer, as will be demonstrated in the first experiment of Section 4. Hence more seamless handoff can be achieved.

3.1.1.1 Address Ambiguity Avoidance

Though not introducing yet another network ID space such as home address [2-3] or Host Identity [8] is a very attractive feature of EMIPv6, lacking of an unchanged unique ID may cause the following address ambiguity problems which must be addressed.

Case 1: different nodes are assigned the same IP address

Suppose node *A* with address IP_A is communicating with mobile node *B* with address IP_B. The node-pair is $<IP_A, IP_B>$. Then *B* moves away and gets a new address IP_B'. Transport layers at *A* will still see IP_B due to the mapping provided by the NPBC entry. Suppose then a node *C* gets address IP_B at *B*'s previous location and uses IP_B to communicate with *A*. This will cause: 1) When the transport layer at *A* receives a packet with IP_B, it does not know whether it is from *B* or *C*; 2) When the transport layer at *A* transmits a packet to IP_B, the IP layer does not know whether this packet is for *B* or *C*.

We notice that this issue can be solved within the IPv6 addressing architecture [35-36]. Due to the unique address architecture of IPv6, it is possible to always assign unique IPv6 addresses to different nodes. When stateless address configuration scheme is used, the interface id of an address can be constructed from the lower layer MAC address, or by hashing one's public key [22]. In the case that stateful address configuration scheme (e.g., DHCPv6) is used, since IPv6 has very large address space, assigning address by simply increasing the interface id is good enough to overcome address ambiguity. Suppose a DHCP server assigns 2^{20} addresses per second, it will take about 557844 years to assign overlapped addresses when the interface id is of 64 bit length.

Case 2: different applications in a same node-pair may use different original addresses

Suppose there are hosts *A* and *B* with original addresses IP_{A1} and IP_{B1}, respectively. App_1 at *B* setups up a connection between nodes *A* and *B*. A node-pair is created, using IP_{A1} and IP_{B1} as the original addresses. Then *A* changes its address from IP_{A1} to IP_{A2} after sometime. Due to the connection maintenance procedure, the current address of *A* is updated in both NPBC entries at *A* and *B*. Then App_2 at *B* starts and would like to setup a new connection with *A*. It revolves the address of *A* via PNRP, and gets *A*'s current address IP_{A2}. If *B* uses IP_{A2} to setup connection with *A*, the mobility module at *B* will be bypassed, and IP_{A1} (the original address of *A*) is not used in communication. This may cause problem if it is required that these two applications should use the same address for node *A*; for example, there exist one IPSEC SA to protect all the traffic between *A* and *B*.

In order to enforce the applications to use the same original address, the following procedure is proposed: after the resolution module returns IP_{A2}, the NPBC cache is looked up using IP_{A2} as the current destination address. If a NPBC entry is found, we substitute IP_{A2} with the original destination address (in this case, IP_{A1}) before deliver the address to the caller application. In this way, a unique original address of the peer is provided to higher-layer applications.

3.1.2 Connection Maintenance Procedure

The connection maintenance procedure in EMIPv6 is depicted in Figure 1. When a mobile node *A* changes its address, it first sends a CoTI (Care-of Test Init) message to *B*. This message normally is sent to *B* via IP layer. However, when the IP layer cannot deliver CoTI directly (due to firewall/NAT or simultaneous movement), the

E2E mobility module will ask the distribute S/N service to deliver it. CoTI is to inform *B* that *A* has changed address. After *B* receives CoTI, it will send back a CoT (Care-of Test) message to *A*'s claimed current address. If *A* can receive this CoT, it then sends a BU (Binding Update) message to *B* to update its current address. After receiving BU, *B* will update its NPBC entry and send back a BA (Binding ACK). The connection maintenance procedure is finished after *A* receives BA.

There are following cases that the connection maintenance procedure invokes the S/N service to deliver a CoTI message: 1) the mobile node knows (via peer probing and negotiation) that the corresponding node is behind a NAT/firewall box; 2) the mobile node sends CoTI via both IP layer and the S/N service directly (to maximize receiving probability and reduce handoff delay); 3) or after one or more timeouts (due to simultaneous movement) of sending CoTI via IP layer. In case 3), it is possible that the S/N service is (mistakenly) triggered due to packet loss caused by congestions or wireless interference. Nonetheless, CoTI message can still be delivered and the connection maintenance procedure is unaffected.

The CoTI/CoT exchange is a kind of return routability test, to make sure that node *A* is really at its claimed current address (note that even if *A* and *B* have shared secret, *B* can not trust that *A* is really at its claimed current address without CoTI/CoT exchange). After that, *A* can send BU to *B*. Note that though BU and BA are signed by the generated shared key (which is generated from the token carried in CoT), the security is not enough. This is because a malicious node can act as node *A*, and after CoTI/CoT exchange, can easily hi-jack the traffic sent from *B* to *A*. For this reason, we further introduce IPSEC or CGA [22] to protect the BU/BA messages. The detailed security issues are further discussed in Section 3.1.5.

From the procedure we can calculate that the handoff latencies in EMIPv6 (note that the handoff latency does not include delays such as network detection time and DAD (Duplicate Address Detection) time). Node *A* should be able to send packet using its new address after sending out BU. Hence the handoff latency perceived by node *A* is 1 RTT (round trip time) in this case. It is possible that BU is dropped by the network but the following data packets with *A*'s new address correctly received by *B*. If this is the case, *B* should drop or buffer the new packets until *A* re-transmits BU to correctly establish *B*'s binding cache (In our implementation, we drop the packets when the binding update procedure is not complete. This makes the implementation simple and resilient to Denial-of-Service attack). Node *B* should be able to send packet to *A*'s new address once it sends out BA. Hence the handoff latency perceived by *B* is 1.5RTT. It is interesting to observe that the mobile node always has smaller handoff latency than the correspondent node. As we have indicated previously that there are cases (simultaneous movement or NAT/firewall) that the CoTI message may not be able to deliver to the peer node directly. In the case that a mobile node need to use the distributed S/N service to deliver CoTI, the handoff latencies are

approximately the same as before if we treat the distributed S/N as a "virtual interface" (see Figure 6) under the network layer. A more concrete expression of the handoff latencies in this case are $D_{timout} + D_{A->B}^{PNRP-S/N} + D_{B->A}^{IP}$ at node *A* and $D_{timout} + D_{A->B}^{PNRP-S/N} + RTT_{A-B}^{IP}$ at node *B*, where $D_{timeout}$ is the timeout value caused by previous IP layer CoTI message, $D_{timeout}$ may be zero if *A* sends CoTI using both PNRP and IP layer simultaneously, $D_{A->B}^{PNRP-S/N}$ is the delay from *A* to *B* via the distributed S/N and $D_{B->A}^{IP}$ is the delay from *B* to *A* via IP layer, respectively. In the case of simultaneous movement, the handoff latency is the minimum value of the two simultaneous handoff procedures.

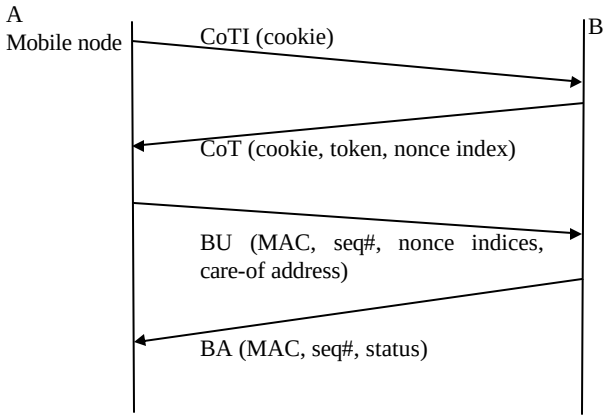

Figure 1. The connection maintenance procedure for EMIPv6. CoTI/CoT for return routability testing and BU/BA exchange for address updating.

3.1.3 IPv6 Extension Headers for Mobility Management

In this design, several 'new' IPv6 extension headers are introduced to carry the mobility messages. These extension headers and mobility options are borrowed heavily from MIPv6. This is intentional, since though the design philosophies of the two approaches are very different, the format of the mobility signaling should be approximately the same. Moreover, by doing so, it should be easier to make EMIPv6 compatible with MIPv6 to some extend (in fact, our EMIPv6 implementation can act as correspondent node for MIPv6).

Mobility Header: Mobility header is used by a mobile node to exchange mobility management messages with its peers. The following messages are defined for mobility header: Peer Probing and Negotiation (PPN), Care-of Test Init (CoTI), Care-of Test (CoT), Binding Update (BU), Binding ACK (BA), Binding Refresh Request (BRR), and Binding Error (BE). The formats of the messages except PPN are almost the same as that of MIPv6. PPN message is used for the peers to exchange information at the beginning of communication. Using PPN, a node can learn whether its peer supports EMIPv6, whether the peer is mobile or under NAT/firewall, and which security

method to use, when to create node-pair, etc. And the nodes also use PPN to exchange PNRP IDs and certified peer addresses (CPAs).

Original Address Destination Option Header: This header is the same as the Home Address Destination Option Header of MIPv6. We re-interpret the home address option header in our EMIPv6 context. The original source address is carried by the Destination Option extension header. It is used in packets sent by a mobile node when the local original and current addresses are not the same.

Routing Header (type 2): It is used to carry the original destination address from a peer to a mobile node, when the remote original and current addresses are not the same.

3.1.4 Data Packet Processing

When a node sends a packet with original addresses {*local_orig_addr, remote_orig_addr*}, it will use the original addresses to look up the NPBC cache. If no entry can be found, the packet is delivered directly; otherwise, the E2E mobility module substitutes the source address with *local_curr_addr*, the destination address with *remote_curr_addr* in the IP header, and carries *local_orig_addr* with an original address destination option header, and *remote_orig_addr* with a routing header (type 2). Then the packet is sent out directly to the receiver's current address.

The procedure on the receiving side is simpler as compared with the sending procedure, since the original addresses are carried in the extension headers. If a packet is received without an original address destination option header or a routing header (type 2), the packet is processed without the mobility module involved. Otherwise the packet is checked to verify if there is a NPBC entry for the packet. If the NPBC entry does exist, the source and destination addresses of the packet are substituted with the values carried in the mobility headers before the packet is delivered to the transport layer, otherwise, the packet should be dropped. The interaction between the mobility module and IPSEC will be discussed in the next sub-section.

3.1.5 Security

In this design, CoTI/CoT exchange can make sure that the sender of BU is really at its claimed current address. However, it cannot prevent a malicious user from acting as a mobile node to hi-jack the traffic. Here we further introduce IPSEC or CGA, to secure the BU/BA exchange.

For IPSEC, the credentials used to establish the IKE SA may be obtained from PNRP or the pre-installed public/private key pairs. Then the IPSEC SA which is needed to protect BU/BA can be established using the IKE SA. Using BU as an example, the message format with transport ESP is as follows.

IPv6 header (source = current address,
destination = peer address)
Destination Options header

Original Address option (original address)
ESP header in transport mode
Mobility header
Binding Update
Alternate current address option (current address)
ESP trailer

We note that IPSEC and EMIPv6 can help each other in that mobility messages can be protected by IPSEC, and IPSEC SAs can survive address change events with the help from EMIPv6. Based on IPv6 protocol, the IPSEC headers are behind of destination option and routing headers. Therefore, The IPSEC implementation needs to be aware of the original address destination option and the routing header (type 2). When sending data packets, IPSEC needs to use the original addresses carried in the extension headers to lookup the SPD (security policy database) and SAD (security association database) databases; when receiving, IPSEC can be unaware of mobility since the original addresses have already been placed in the right place.

For CGA, by generating one's IP address from its public key (see [22] for CGA details), the mobile node can attach its public key together with the BU message, and then sign the whole message with its private key. The receiver therefore can check the relationship between the IP address and the public key and further verify whether the message has been tramped by re-computing the signature.

To use IPSEC and CGA in EMIPv6 have both advantages and disadvantages. Using IPSEC to protect mobility messages has the advantage that EMIPv6 can help IPSEC SAs to survive address change. However, IPSEC SAs needs several messages to setup. CGA is light-weighted and does not need pre-existing shared secret. But it requires the IP address generated from the public key. The hosts can negotiate which method to use based on their preference during the peer probing and negotiation phase. The time cost for both IPSEC and CGA for address authentication are light-loaded, hence their contribution to handoff delay is neglectable.

3.2 Distributed S/N Service for Mobility Message Delivery

There are mobility cases that cannot be handled only by the two communication participants: 1) unidirectional connection setup caused by NAT or firewall, and 2) simultaneous movement. Using mobility under firewall/NAT as an example, suppose mobile node *A* is communicating with a node *B*, which is behind a firewall/NAT box. After *A* moves to a new network attachment and changes its IP address, it will not be able to directly notify *B* of its new address due to the separation of firewall/NAT. Similar difficulty exists for simultaneous movement, where both nodes will send mobility messages to their peers' old IP addresses. We note that since NAT is very popular in IPv4, the unidirectional connection setup caused by NAT will exist at least in the IPv4 to IPv6 transition period. And since more and more devices go mobile and connect to the Internet, simultaneous movement will happen occasionally if not often. These two cases therefore need to be addressed in any mobility schemes.

In this paper, we introduce a distributed subscription/notification (S/N) service, which is built on top of a PNRP overlay, to address these two mobility cases. The idea is that both communication peers subscribe the IP address change events of their peers via the distributed S/N service. In the case that a mobile node cannot deliver a mobility message via IP layer, it delivers this message as a notification to its peer via the distributed S/N service.

Since both participants in communication have interest to the address change events of their peers, the 'subscription' operation is therefore implicit and no explicit subscription message exchange is needed. It is nature to build a 'notification' semantic on top of the 'lookup' service provided by P2P overlay. Many P2P networks can be used for distributed S/N. But since we use PNRP for location management, PNRP is thus the best choice for our S/N service. In this way, connection maintenance and location management are integrated together.

We describe how the distributed S/N service can solve the simultaneous movement issue (similar procedure can be applied to the firewall/NAT case). Suppose mobile nodes *A* and *B* move simultaneously. *A* and *B* will try to send CoTI messages directly to the old locations of *B* and *A*, respectively. These messages, however, will be dropped due to simultaneous movement. Using node *A* as example, after timeout (another choice is that *A* sends out CoTI via S/N and IP layer at the same time to reduce handoff latency in the case the application is delay sensitive), it will try to deliver the CoTI packet via the distributed S/N service. The underlying PNRP overlay will then route this message. This message will reach one of *B*'s neighbors, and will finally reach *B* from that neighbor (see Figure 2). In PNRP, a node actively maintains the relationship with its neighbors. Hence, even under firewall/NAT or simultaneous movement, if a message reaches one of its PNRP neighbors, the message can then reach the destination from its neighbors. We note that the distributed S/N together with NAT traversal proposals such as Teredo [37] can maintain connections for mobile nodes even when both of them are behind NAT. The details are omitted here due to space limitation.

Though PNRP overlay can be directly used for mobility message delivery, the handoff latency in this case may be large. The message in average needs to traverse $O(\log N)$ logical hops to reach the destination. In order to reduce the handoff latency, we introduce a NHT (neighbor hint table) to build a distributed S/N on top of PNRP.

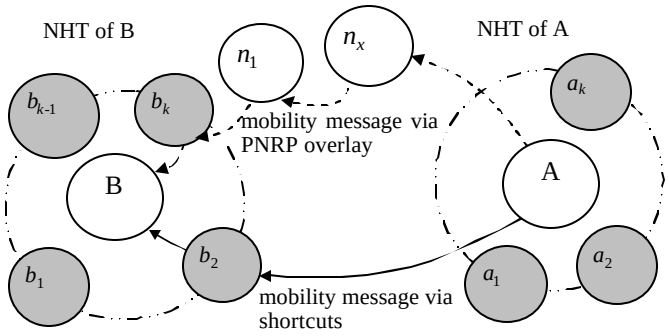

Figure 2. Using distributed S/N service to deliver mobility massages. The short-cut created by the S/N service improves the performance from $O(\log N)$ to 2 hops.

3.2.1 Neighbor Hint Table

Due to the specific design of PNRP, as indicated in Figure 2, if a message reaches one of the neighbors of a destination, then the neighbor can successfully send the messages to the destination. Based on this observation, a node can form a neighbor hint table (NHT) from its selected neighbors. Two communication peers then exchange their NHT to create short-cut for mobility message delivery. When a node would like to send a CoTI message to its peer using PNRP overlay, instead of delivering the message using PNRP routing, it will send the message to the nodes in the NHT table of the peer as illustrated in Figure 2. Then those nodes in NHT will forward the message to the destination. We note that due to the dedicated design of PNRP ID, most of the neighbors of a mobile node are expected to belong to a same authority. The neighbors of a mobile node thus are expected to have incentive to forward packets for it.

The entries of a NHT table are sorted based on the following criteria:

? Trustiness: a neighbor with a higher level of trust is preferred against a neighbor with lower trust;

? Fixed/mobile attribute: fixed nodes should be preferred since the address of mobile nodes may change;

? Proximity: a neighbor that is physically closer to the node is preferred over other neighbors.

The NHT addresses are then arranged into a matrix with m rows and n columns. The entries of this matrix are filled from the first row one by one with the sorted addresses. For example, when m=2 and n=3, the matrix is filled according to sequence (1,1), (1,2), (1,3), (2,1), (2,2), (2,3). When sending a CoTI message, the entries in the first row will be tried simultaneously, if all the entries in the first row fail, the second row will be tried.

It is obvious the value setting of *m* and *n* is a tradeoff between reachability and traffic overhead. Based on our reachability study on various life time distribution (exponential, uniform, and pareto), we use *m*=2 and *n*=3 in this design, which can provide good reachability and small traffic overhead. (See [38] for more detail.)

3.2.1.1 NHT Maintenance

In order to keep NHT up-to-date, an efficient maintenance scheme is important. We propose a maintenance scheme combining periodic and trigger-based maintenances.

In normal case, a node periodically updates its NHT table by sending maintenance messages to its peers. Based on our analysis and simulation [38], if the mean lifetime of each host is on the order of hours, 10 minutes maintain period and 6 NHT neighbors can provide >99.99% reachability probability.

However, there still exists the most extreme case that most of the entries in NHT become invalid before the next round of periodic maintenance. In this case, trigger-based maintenance is performed. The distributed S/N actively monitors the availability of its neighbors. When it detects a considerable modification of its NHT table, it will send a maintenance message to its peers to update the NHT.

3.2.2 Virtual S/N Interface and Circular Dependency Avoidance

From the functionality provided by the distributed S/N service (i.e., mobility message notification), we can treat it as a special "network interface" to the E2E mobility module, as illustrated in Figure 6. When the virtual interface receives a mobility message from the E2E mobility module, it will up-call the distributed S/N module to deliver the message. By introducing the virtual interface concept, the connection maintenance procedure is the same no matter the message is delivered via IP layer or PNRP overlay.

However, we have a circular dependency problem between the distributed S/N and the E2E mobility module. From layering point of view, the E2E mobility module locates at IP layer and the distributed S/N as a service locates at application layer. This means that the distributed S/N may trigger IP level connection maintenance procedure, which unfortunately depends on the distributed S/N itself to perform well in some mobility scenarios. Therefore, a circular dependency occurs. This may cause unwanted NPBC node pairs to be created, and in the worst case, the number of unwanted NPBC node pairs may increase exponentially [38].

The solution to this problem, as we observe from the phenomenon, is to cut the inter-dependency between the distributed S/N and the IP layer mobility management. In this work, we do not create node-pairs for PNRP packets. When the E2E mobility module receives or sends a packet, it checks if the message is a PNRP message; if it is, no node-pair will be created, thereby breaking the inter-dependency condition.

We note that the E2E mobility module just treats the distributed S/N as a transmission channel. The authentication and encryption of mobility messages are performed by the E2E mobility module at IP layer.

3.3 PNRP Overlay for Name Resolution

The 'lookup' service of P2P overlay [e.g., 39-40] can be used for self-organized and scalable location management in EMIPv6. In this work, we use a DHT-based PNRP (Peer Name Resolution Protocol) overlay which is originally proposed in [20] for distributed name resolution in the Internet. As part of the Windows P2P software development kit (SDK), PNRP has been widely distributed and is readily available on computers that run Windows XP (and will be available in the next version of Windows CE). In the rest of this sub-section, after briefly introducing the PNRP overlay, we present enhancements for PNRP to better support wireless mobile networks by considering the characteristics of mobile hosts such as host heterogeneity and trustiness relationship.

3.3.1 PNRP Overlay

In PNRP overlay, each node is assigned a PNRP ID. A PNRP ID is a 256 bit number that is the concatenation of a 128 bit *P2P ID* and a 128 bit *Service Location ID*. P2P ID is hashed from a node's name and its public key. Service Location is hashed from application specified values such as port number. PNRP IDs in the PNRP overlay are arranged in a circular number space. Each node participates in the overlay by responding or forwarding name resolution request. Only the owner of a PNRP ID can respond to a request. This makes sure that the returned IP address of the queried node is always up-to-date. Note that this feature is critical to mobility management, since mobile nodes may frequently change their network locations. The response to a resolution request contains a certified peer address (CPA), which includes: the PNRP ID of the node, the current IP address, the validity interval of the CPA, the public key for the PNRP ID, and the signature of the CPA based on public/private key pair. If a node receives a resolution request for other nodes, it tries to find a next node which has a PNRP ID closer to the destination and forwards the message to it.

In order to achieve efficient overlay routing, each active node in PNRP overlay maintains a multiple-level routing cache as illustrated in Figure 3. A routing cache is a collection of CPAs representing knowledge about selected participants in the PNRP overlay. Each level represents a segment of the total PNRP ID number space. The top level of the cache spans the entire number space. The next level down spans a smaller segment of the number space, clustered around the PNRP ID of the node. Each subsequent level spans a progressively smaller part of the number space, always around the same PNRP ID. There is a maximum number of entries (k) allowed at each cache level. We call nodes in the lowest level of the cache the neighbors of the node. When a node changes its IP address, it will immediately notify its neighbors. When a node adds a new CPA into its cache, it will flood the new entry to all its neighbors.

The routing algorithm of PNRP is simple: when a node receives a resolution request, if it is the destination of the request, it responses to the request with its CPA; otherwise, it forwards the request to the next node in its routing

cache whose PNRP ID is 'closest' to the destination. Due to the structured multiple-level routing cache, like most of the current DHT-based P2P overlays [39-40], PNRP achieves O($log_k N$) hops in message delivery, where k is the number of entries in each cache level, and N is the number of active nodes in the overlay. For description simplicity, we omit the technical details such as PNRP overlay discovery, joining/leaving a PNRP overlay, cache maintenance, split detection and repairing. As compared with centralized location management schemes, PNRP needs additional traffic to maintain its structure. However, this additional traffic is affordable. For example, for a 10 million nodes PNRP overlay, the per-node background traffic is only about 500 bytes per second. Readers are referred to [20] for more information.

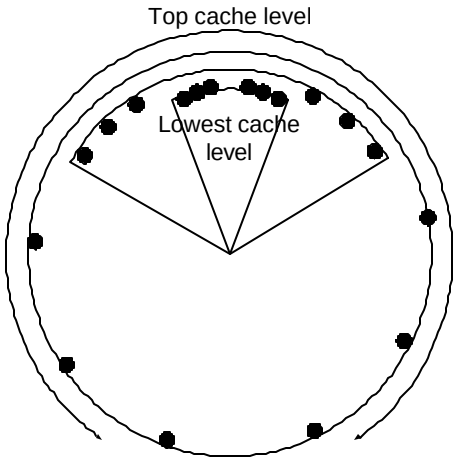

Figure 3. Multi-level routing cache of PNRP.

3.3.2 Enhanced PNRP for Mobility

The original PNRP protocol is designed for Internet case without considering mobility. In this work, we introduce several enhancements for PNRP to better support mobile devices.

First, we differentiate mobile nodes from fixed nodes. Mobile nodes may frequently change their addresses due to their mobile nature, and in general have limited bandwidth, memory space, and computational power. PNRP nodes thus may not want to route messages over mobile nodes. To achieve so, a mobile node announces that it is mobile by setting a "NO_CACHE" flag in its CPA. Other nodes will not add a mobile node into their routing caches unless the mobile node is their neighbor (i.e., locating at their lowest level of cache).

Second, nodes owned by one authority (i.e. one user/family/organization) are assigned with PNRP IDs that are clustered together in PNRP numeric space. For example, an end user may own several mobile and fixed nodes. It may be desirable for him to organize these nodes to be neighbors mutually. In this way, one's fixed nodes are motivated to carry traffic for his mobile nodes. We note that this feature can be added without any revision to the PNRP design. As mentioned before, PNRP ID is composed of a P2P ID and a Service Location part. Therefore,

nodes owned by the same authority may share the same P2P ID, but with different Service Locations. Note that this extension can provide incentive for fixed nodes to carry traffic for mobile nodes.

Third, host heterogeneity is exploited to further enhance PNRP performance. The Internet is intrinsically heterogeneous. Different hosts have different network bandwidth, memory size, and computational power. It is therefore desirable that nodes with different capacities take different responsibilities in the PNRP overlay. In this extension, nodes with higher capacity can choose larger k (number of entries in each row of the routing cache) for their routing cache, and become power nodes (PN) in the PNRP overlay. A node advertises its k in its CPA (an interesting problem is how to infer k without explicit advertisement, we leave this for further study). When cache replacement algorithm is executed to select new CPAs, nodes with larger k are preferred.

The routing algorithm is also revised based on the value of k. Instead of choosing the node which is numerically closest to the destination, a node N chooses the next hop N_i that satisfies: $N_i = \arg\min_{N_x}(|N_x - N_{dest}|/k_x)$ for $N_x \in Cache(N)$ and $|N_x\text{-}N_{dest}| < |N\text{-}N_{dest}|$, where N_{dest} is the PNRP ID of the destination, and $Cache(N)$ is the set of routing cache of node N. $|N_x\text{-}N_{dest}|$ is the numeric distance between N_x and the destination N_{dest}, and $|N_x\text{-}N_{dest}|/k_x$ is the expected distance between the next node (the node which N_x chooses to forward the message) and the destination. $|N_x\text{-}N_{dest}|/k_x$ is derived from PNRP's property that the distance between N_x and N_{dest} is expected to reduce k_x times when N_x routes the message to the next node. The intuition behind this strategy is that we look two steps ahead. Instead of finding the nearest cached node to the destination, we find the (power) node which is expected to be able to forward the message to a node which is nearest to the destination. The introducing of power node improves PNRP in several aspects: 1) different nodes in PNRP overlay can contribute resources based on their capacities, and the average number of hops for a message is significantly reduced; 2) as we will observe via the following simulation, a few percentage of power nodes is able to significantly improve the performance.

We evaluate PNRP and the enhancements via simulation. In the following simulations, we consider two classes of nodes: normal nodes with k set to 4 and power nodes with k set to 64. All PNRP nodes are uniformly distributed in PNRP space. The overlay has run enough time to let the active nodes tune their routing caches. For each run of simulation, 10,000 resolve requests are issued between randomly selected sources and destinations. We use the latency distribution measured from [41] to simulate one-hop lookup delay.

Figure 4 depicts the mean number of logical hops a resolution traverses versus the number of active nodes in the PNRP overlay. As indicated in Figure 4, the mean number of hops increases logarithmically as the number of active nodes increases. We also have evaluated the performance of PNRP overlay under different fraction of power nodes. A very interesting phenomenon is that only a small fraction of power node is able to reduce the

average lookup hops significantly. This phenomenon can be explained as follows. Suppose the k values for power nodes and normal nodes are k_{PN}, and k_{NN}, respectively. From PNRP's routing strategy, we know that the distance reduced at each hop by power nodes is k_{PN}/k_{NN} times than that of normal nodes. Hence if we have 1 power node among k_{PN}/k_{NN} normal nodes, almost all traffic will be routed merely with the power nodes. As illustrated in Figure 4, 6.25% (4/64=6.25%) fraction of power nodes in PNRP is able to reduce about 2 resolution hops.

Figure 5 depicts the cumulative distribution function (CDF) of the resolution latency in PNRP. In this simulation, we also consider node failure rate. We start a PNRP overlay with 80,000 nodes and after some time bring down nodes with a certain failure rate. We observe that the latency distribution is significantly affected by the node failure rate. This is because the failed nodes may still be cached in the routing caches of some active nodes. Again we see that a small fraction of power nodes is able to significantly reduce the resolution latency even with large node failure rate.

In summary, we consider there are several unique characteristics in PNRP and its extension, 1) resolution response is given by the node who owns the PNRP ID, thus there is no cache invalidity issue; 2) devices with good trustiness relationship are clustered in PNRP ID space; 3) host heterogeneity is considered in our design to improve performance; and 4) PNRP is designed not only for name resolution, but also an indispensable part for connection maintenance as we have shown in Section 3.2.

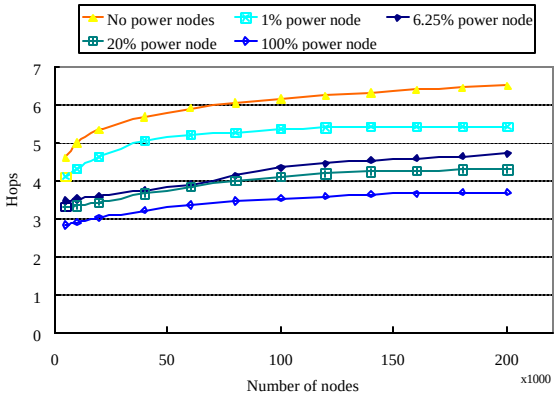

Figure 4. Mean resolution hops vs. number of active nodes.

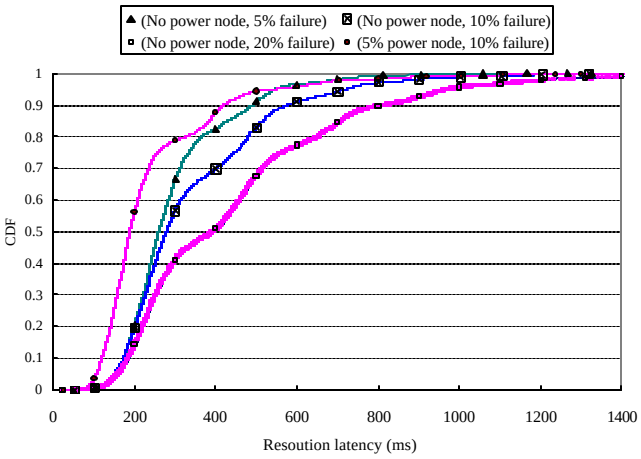

Figure 5. CDF of the resolution latency.

4. IMPLEMENTATION AND EXPERIMENTS

4.1 Implementation Description

We have implemented EMIPv6 in both Windows XP and Windows CE operating systems. The implementation architecture is illustrated in Figure 6. The three major components of EMIPv6 are emphasized with gray color. Besides the three major components, we introduce a handoff decision maker (HDM) to decide to which interface (and address) the host should switch to when mobility events happen. The implementations for Windows XP and CE versions share the same source tree and use conditional definitions to configure platform related setups. The implementation consists of more than 8500 lines of C code (the kernel part) and 13500 lines of C++ code (the distributed S/N, PNRP extension, and HDM).

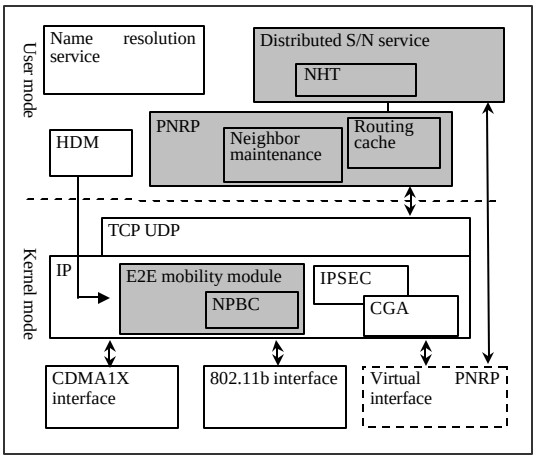

Figure 6. The implementation architecture of EMIPv6. There are three major components: an E2E mobility module, a distributed S/N service, and a PNRP overlay.

The E2E mobility module performs the connection maintenance procedure and maintains the NPBC cache. It is built together with the Windows IPv6 stack, which is located at the OS kernel. Readers are referred to [42] for the major data structures of the IPv6 implementation. The establishment of the NPBC entries is triggered by data packets sent/received. The connection maintenance procedure is triggered by commands from HDM. Note that as to the E2E kernel, the distributed S/N is abstracted to a "virtual PNRP interface" under IP layer (see Figure 6). The E2E mobility management kernel therefore has a uniform implementation in that it does not need to distinguish whether it is using a real network interfaces or a P2P overlay for mobility message delivery.

The PNRP module is already in Windows XP/SP2. It provides APIs for distributed name to IP address resolution. We made several extensions to PNRP so that it exports several APIs, such as to get the entries in PNRP cache and send mobility messages, to the distributed S/N service.

The distributed S/N service is built on top of PNRP, and it helps the E2E kernel to deliver mobility messages. It forms NHT using nodes gotten from the lowest level of cache (i.e., the neighbors of the node) or dedicated nodes and performs NHT maintenance operations. The distributed S/N also registers a callback function in the PNRP module which will be invoked by PNRP when CoTI or NHT maintenance messages are received by PNRP.

The role of HDM is to monitor the status of the network interfaces and to make handoff decision based on user-defined rules once the status of the network interfaces change. Here the rules are used to define user preferences. For example, "use Ethernet if it available, else use WLAN, else use CDMA 1X". Detailed description of user preference is out the scope of this paper.

Since MIPv6 is also available in Windows XP and Windows CE as Technology Preview [43], our implementation provides method to manually configure which mobility stack to use. And a node with our EMIPv6 implementation can also act as a CN node if the corresponding mobile node only supports MIPv6. Our EMIPv6 implementation therefore provides potential simultaneous usage together with MIPv6.

4.2 Experiments

In this sub-section, we use two experiments to demonstrate how EMIPv6 keeps connection continuity. In the first experiment, vertical handoff between CDMA1X and WLAN is presented; in the second experiment, handoff with the help from distributed S/N service is studied. In both experiments, the mobile node has two network interfaces and at least one interface is always on. Note that when a mobile node has only one single interface, the communication interrupt time may be larger due to the time costs of network detection and IP address verification. Since we focus on handoff latency caused by EMIPv6, we do not use the single interface scenario.

In the first experiment, there are two nodes in communication. Node *A* is in dual mode with a CDMA1X card and an 802.11b WLAN card. Node *B* is a desktop PC connected to the network via Ethernet. There is a TCP connection between node *A* and *B*. Data packets are transferred from *B* to *A*. The bandwidths from *B* to *A* under

CDMA1X and WLAN are about 90 Kb/s and 2 Mb/s, the RTTs are about 600ms and 20ms, respectively. At first, the 802.11b card of *A* is out of signal range, and *A* uses the CDMA1X card for communication. Then node *A* moves into the coverage of WLAN, the 802.11b card becomes active. Since WLAN has much higher bandwidth, the connection is switched from CDMA1X to WLAN. We captured the packets sent and received at node *A*. Figure 7 depicts the time-sequence plot of this TCP connection. From the figure, we observe that node *A* sends out CoTI at 34.526s, receives CoT at 34.547s. It then sends out BU immediately and the connection maintenance procedure is finished after BA is received at 34.568s. The whole handoff procedure ended in 42ms.

In this experiment, since CDMA1X is always on and much slower than 802.11b, we observed a very interesting phenomenon: after handoff from CDMA1X to 802.11b, node *A* can still receive the data packets from the previous CDMA1X pipe. The triggered ACKs will be transmitted via 802.11b and will trigger more data packets to be transmitted to *A*. Due to the 'hole' caused by the data in the CDMA1X pipe, fast retransmit will be triggered. In this experiment, we observe two fast retransmit before the data in the CDMA1X pipe is drained. Due to the fact that EMIPv6 can receive packets from both pipes, there is no packet drop. Based on this experiment, we note that if TCP is handoff aware, it may adapt to the network condition of 802.11b more quickly and even better performance can be obtained.

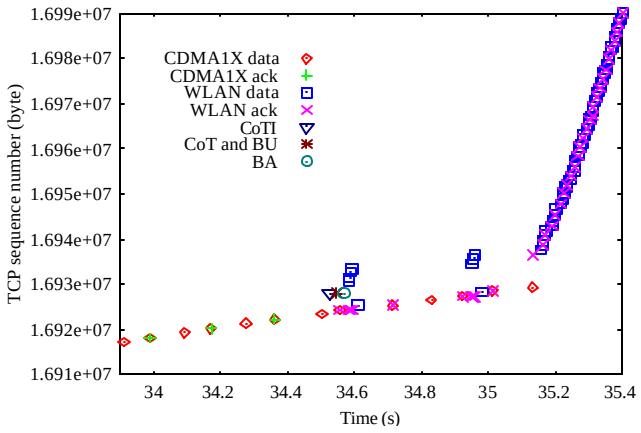

Figure 7. Handoff from CDMA1X to WLAN.

The network topology of the second experiment is depicted in Figure 8. In the second experiment, there are also two nodes in communication. Mobile node *A* is communicating with a fixed node *B*, which is behind firewall/NAT. Node *A* is equipped with an 802.11b WLAN (11Mb/s) and an Ethernet interface (100Mb/s), the WLAN interface is always on in this experiment. Nodes *A* and *B* are communicating using real-time VoIP. The bi-directional VoIP stream uses GSM 06.10 codec and the frame interval is 20 ms/frame. The VoIP packet size from the codec is 33

bytes, so the codec rate is 13 Kb/s. With 12 bytes application header and 8 bytes UDP header, the payload to IP is 53 bytes. The round trip time between *A* and *B* is 60ms. Node *A* at first uses its Ethernet interface, switches to WLAN at some time (by unplugging the Ethernet wireline), and then switches back to Ethernet (by re-plugging the Ethernet wireline).

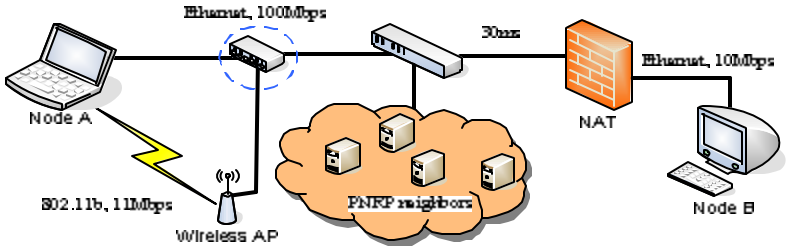

Figure 8. Network topology for mobility experiment under NAT environment.

Due to the fact that *B* is behind firewall, *A* will not be able to send CoTI to *B* directly after the Ethernet interface of *A* has been unplugged. Hence *A* will use the distributed S/N to deliver the CoTI message. The time-sequence plots of the two handovers are shown in Figure 9. Note D(*x*) represents packets from *A* to *B* with sequence number *x*, and Note D'(*y*) represents packets from *B* to *A* with sequence number *y*, respectively.

We observe that, with the help from the distributed S/N service, the connection is maintained successfully for both handoffs and the voice interruption during the handoff periods is almost unnoticeable. The handoff latencies introduced by our protocol is 70 ms (about 1RTT) for node *A* and 103 ms (about 1.5RTT) for node *B*, which is verified by our experiment results. The detailed procedures for handoff are as following:

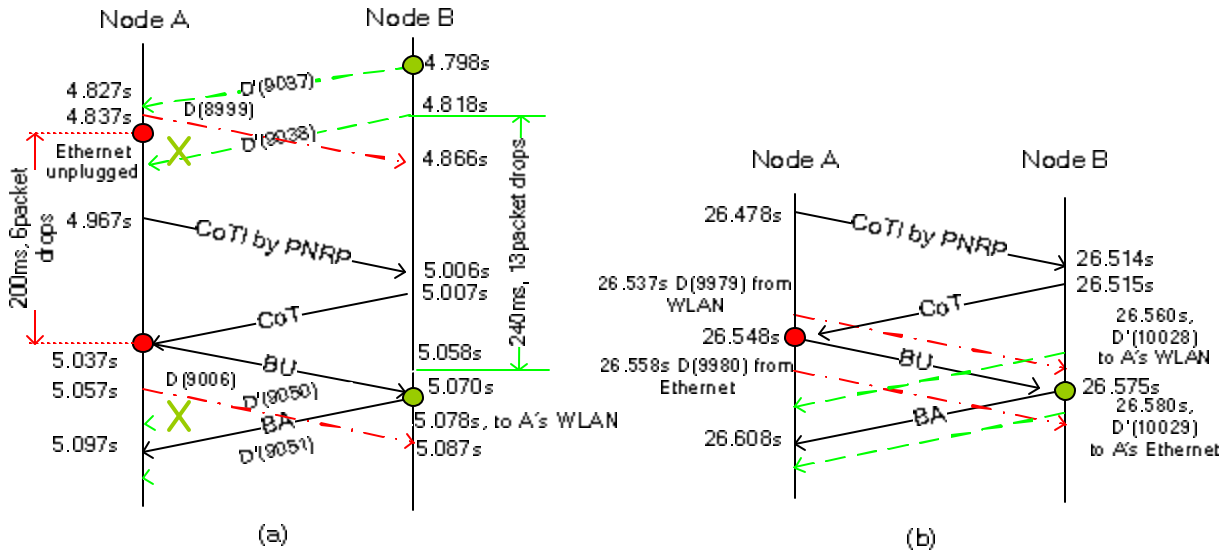

Figure 9. Time-sequence plot for handoffs via PNRP. (a) From Ethernet to WLAN. (b) From WLAN to Ethernet.

Ethernet to WLAN handover: The Ethernet interface is unplugged at time around 4.837s, and it takes the system about 120ms to detect the unplug event and *A* sends out COTI via PNRP at 4.967s. *B* receives CoTI at 5.006s and sends back CoT, then A sends BU at 5.037s and receives BA at time 5.097s. The handoff delays at node *A* and *B* are 70 ms and 103 ms, respectively. Only after node *B* receive the BU at time 5.070s, it will send VoIP packets to *A*'s WLAN interface, which means the packet sent to the *A*'s Ethernet interface between 4.818s and 5.058s, around 240 ms interval, will be dropped. Since the VoIP application uses 20 ms frame size, totally we observed 13 packet drops from node *B* to *A*. From *A* to *B*, node *A* will use WLAN to send packets to *B* after handoff, and we observe 6 packet drops from *A* to *B*[3]. We notice that the time for the system to detect the unplugged event (about 120 ms) is larger than the handoff latency caused by EMIPv6 in this experiment; hence a very important topic for further research is how to reduce the network detection time.

WLAN to Ethernet handover: The procedure for the handover from WLAN to Ethernet is similar, which is shown in Figure 9(b). Since the WLAN interface is still available after the Ethernet wire-line is plugged in, much smoother handoff is achieved. In experiment, we observed zero packet loss in both directions since the WLAN interface of *A* is always available during handoff. The handoff delays are 70 ms and 97 ms at node *A* and *B*, respectively. The handoff delays are approximately the same for handoffs from Ethernet to WLAN and from WLAN to Ethernet.

[3] Note that the VoIP application should generate about 10 packets in this 200ms time interval in theory. However, we noticed that only 6 packets were generated. The reason for this phenomenon might be that, when the Ethernet is unplugged, the OS is busy in the kernel mode for about 120ms, and the interrupts generated by the sound card might consequently be dropped by the OS.

We also have performed experiments for simultaneous movement with the help of distributed S/N. The result is similar to that of the second experiment, except that both nodes simultaneously initiate the E2E connection maintenance procedure.

5. CONCLUSION

We have presented EMIPv6, an end-system based mobility solution, which solves connection maintenance and location management for mobile hosts in an integrated and self-organizing way. The connection maintenance is performed at IP layer between communication peers from end to end without relying on additional network components. And a DHT-based PNRP overlay is constructed for name to IP address resolution by considering the heterogeneity of end-systems. Moreover, we further construct a distributed S/N service based on the PNRP overlay for mobility handling under firewall/NAT or simultaneous movement.

By performing mobility management from end to end and leveraging P2P technology, EMIPv6 has the following key features: 1) it is a complete end-system based solution and does not introduce any additional network components; 2) it is self-organized, scalable and robust, and provides small handoff latency without network administrative burden; 3) and it handles complicated mobility scenarios such as mobility under firewall/NAT or simultaneous movement.

We have implemented EMIPv6 in the Windows XP/SP2 (desktop and laptop) and Windows CE (Pocket PCs and smart-phones) operating systems and performed extensive experiments in our testbed. We have evaluated the performance of our extended PNRP design via simulation with traces from real world. Our experiments and simulation results have convinced us that our scheme fulfills its design goals (i.e., ease-of-deployment, small handoff latency and efficient data packet delivery, self-organizing and scalable and robust, transparent to application, and secure). Since EMIPv6 does not introduce additional network components, we expect it to be a quick and alternative way to provide IP layer mobility support for mobile wireless networks.

There remain some important research areas along this end-system based approach. First, the benefits of our scheme partly depend on users' participating in the PNRP overlay. Hence some incentives should be introduced to attract end users to forward packets for others. Second, it should be interesting to further optimize the performance of EMIPv6 once more measurement results are available from real deployment.

ACKNOWLEDGEMENT

We thank Jawad Khaki for introducing PNRP to us when we were seeking ways to decentralize the S/N service. We are grateful to Bernard Aboba, Tuomas Aura, Pradeep Bahl, Siamak Poursabahian, Helen J. Wang, and Zhensheng Zhang for their valuable discussions and suggestions. We thank our visiting students Weidong Shang and Tong Yuan for helping the first author to build a concept-proof prototype, and Xianlong Fan for writing the

VoIP application which we used in our experiments for both Windows XP and CE. We also would like to thank Yunxin Liu for helping us solve many network emulator related problems.

REFERENCES

[1] J. H. Saltzer, D. P. Reed, and D. D. Clark, "End-to-end arguments in system design," ACM trans. Computer Systems, vol. 2, no. 4, 1984.

[2] C. Perkins, Editor, " IP mobility support for IPv4," RFC 3344, http://www.ietf.org/rfc/rfc3344.txt.

[3] D. Johnson, C. Perkins, and J. Arkko, "Mobility support in IPv6," RFC 3775, http://www.ietf.org/rfc/rfc3775.txt.

[4] F. Teraoka, K. Uehara, H. Sunahara, and J. Murai, "VIP: A protocol providing host mobility," CACM 38(8): 67-75 (1994).

[5] D. Funato, K. Yasuda, and H. Tokuda "TCP-R: TCP mobility support for continuous operation," In Proc. IEEE ICNP, pages 229–236, Atlanta, Georgia, October 1997.

[6] A. C. Snoeren and H. Balakrishnan, "An end-to-end approach to host mobility," in Proc. Mobicom'00.

[7] P. Nikander, J. Lundberg, C. Candolin, and T. Aura, " Homeless Mobile IPv6" , IETF draft, work in progress, February 2001.

[8] R. Moskowitz, " Host identity payload architecture," IETF Draft, work in progress, February 2001.

[9] Pekka Nikander, Jukka Ylitalo, and Jorma Wall "Integrating security, mobility and multi-homing in a HIP way," In Proc. Network and Distributed Systems Security Symposium (NDSS'03), February 2003.

[10] R. Jain, T. Raleigh, C. Graff, and M. Bereschinsky, "Mobile Internet access and QoS guarantees using Mobile IP and RSVP with location registers," in Proc. ICC'98, June 1998.

[11] R. Jain, T. Raleigh, D. Yang, L. Chang, C. Graff, M. Bereschinsky, and M. Patel, " Enhancing survivability of mobile Internet access using Mobile IP with location registers," in Proc. Infocom 99.

[12] D. A. Maltz and P. Bhagwat, "MSOCKS: An architecture for transport layer mobility," in Proc. infocom 1998.

[13] S. Zhuang, K. Lai, I. Stoica, R. Katz, and S. Shenker, " Host mobility using an Internet indirection infrastructure," in Proc. Mobisys 2003.

[14] B. Y. Zhao, A. D. Joseph, and J. Kubiatowicz, "Supporting rapid mobility via locality in an overlay network," Technical report, Computer Science Division, University of California, Berkeley, 2003.

[15] Chuanxiong Guo, Zihua Guo, Qian Zhang, and Wenwu Zhu, "A Seamless and Proactive End-to-End Mobility Solution for Roaming Across Heterogeneous Wireless Networks," IEEE Journal on Selected Areas in Communications, vol. 22, no. 5, June 2004.

[16] P. Vixie, editor, " Dynamic updates in the domain name system (DNS UPDATE)," RFC2136, April 1997.

[17] Sami Tabbane, " Location management methods for third-generation mobile systems," IEEE communication magazine, August 1997.

[18] Vasileios Pappas, Zhiguo Xu, Songwu Lu, Daniel Massey, Andreas Terzis, and Lixia Zhang, "Impact of configuration errors on DNS robustness," in Proc. SIGCOMM 2004.

[19] K. Egevang and P. Francis, "The IP network address translator (NAT)," RFC1631.

[20] Introduction to Windows Peer-to-Peer Networking, http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/winxppro/deploy/p2pintro.asp.

[21] G. O'Shea and M. Roe, "Child-proof authentication for MIPv6 (CAM)," Computer Communication Review, Vol. 31, No.2, April 1999.

[22] Tuomas Aura, " Cryptographically generated addresses (CGA)." In Proc. 6th Information Security Conference (ISC'03), volume 2851 of LNCS, October 2003.

[23] A. Valko, " Cellular IP: A new approach to Internet host mobility," Computer Communication Review, Vol. 29, No.1, pp. 50–65, January 1999.

[24] R. Ramjee, T. La Porta, S. Thuel, K. Varadhan, and S. Wang, " HAWAII: A domain-based approach for supporting mobility in wide-area wireless networks," in Proc. IEEE ICNP, 1999.

[25] H. Soliman, C. Castelluccia, K. El-Malki, and Ludovic Bellier, "Hierarchical mobile IPv6 mobility management (HMIPv6)" , IETF Draft, work in progress, June, 2003.

[26] H. Yokota, A. Idoue, and T. Hasegawa, " Link layer assisted Mobile IP fast handoff method over wireless LAN

networks," in Proc. Mobicom 2002.

[27] Rajeev Koodli, " Fast Handovers for Mobile IPv6," IETF draft, http://www.ietf.org/internet-drafts/draft-ietf-mipshop-fast-mipv6-03.txt.

[28] S. Tilak and N. B. Abu-Ghazaleh, "A concurrent migration extension to an end-to-end host mobility architecture," Mobile Computing and Communications Review, 5(3):26–31, July 2001.

[29] T. Kwon, M. Gerla, Sajal Das, and Subir Das, " Mobility management for VoIP service: Mobile IP vs. SIP," IEEE Commun. Mag., October 2002.

[30] V. C. Zandy and B. P. Miller, "Reliable network connections," in Proc. Mobicom 2002.

[31] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana, " Internet indirection infrastructure," in Proc. SIGCOMM 2002.

[32] W. Adjie-Winoto, E. Schwartz, H. Balakrishnan, and J. Lilley, "The design and implementation of an intentional naming system," in Proc. SOSP'99.

[33] R. Cox, A. Muthitacharoen, and R. Morris, " Serving DNS using a peer-to-peer lookup service," in IPTPS'02.

[34] S. Ajmani, D. Clarke, C. Moh, and S. Richman, " ConChord: Cooperative SDSI certificate storage and name resolution," in IPTPS'02.

[35] R. Hinden and S. Deering, " Internet Protocol, version 6 (IPv6) specification," RFC 2460.

[36] R. Hinden and S. Deering, "IP version 6 addressing architecture," RFC 2373.

[37] C. Huitema, "Teredo: Tunning Ipv6 over UDP through NATs," IETF Draft, work in progress, June 6, 2003.

[38] Chuanxiong Guo, Haitao Wu, Kun Tan, Qian Zhang, Wenwu Zhu, and Christian Huitema, "End-to-end mobility support in IPv6 using peer-to-peer technologies," MSR Technical Report, MSR-TR-2004-29, March 29, 2004.

[39] I. Stoica, R. Morris, D. Liben-NoWell, D. Karger, M. Kaashoek, F. Dabek, and H. Balakrishnan, " Chord: A scalable peer-to-peer lookup protocol for Internet applications," IEEE/ACM trans. Networking, vol. 11, no. 1, February 2003.

[40] A. Rowstron and P. Druschel, " Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems," in Proc Middleware 2001.

[41] J. Jung, E. Sit, H. Balakrishnan, and R. Morris, " DNS performance and the effectiveness of caching," IEEE/ACM trans. Networking, vol. 10, no. 5, 2002.

[42] R. Draves, A. Mankin, and B. Zill, " Implementing IPv6 for Windows NT," in Proc. 2nd USENIX Windows NT Symposium, August 1998.

[43] Microsoft TechNet, "Mobile IPv6 Support in Microsoft Windows," http://www.microsoft.com/technet/community/columns/cableguy/cg0904.mspx.

LIST OF FIGURE CAPTIONS

Figure 1. The connection maintenance procedure for EMIPv6. CoTI/CoT for return routability testing and BU/BA exchange for address updating.

Figure 2. Using distributed S/N service to deliver mobility massages. The short-cut created by the S/N service improves the performance from O(logN) to 2 hops.

Figure 3. Multi-level routing cache of PNRP.

Figure 4. Mean resolution hops vs. number of active nodes.

Figure 5. CDF of the resolution latency.

Figure 6. The implementation architecture of EMIPv6. There are three major components: an E2E mobility module, a distributed S/N service, and a PNRP overlay.

Figure 7. Handoff from CDMA1X to WLAN.

Figure 8. Network topology for mobility experiment under NAT environment.

Figure 9. Time-sequence plot for handoffs via PNRP. (a) From Ethernet to WLAN. (b) From WLAN to Ethernet.

LIST OF TABLE CAPTIONS

BIOGRAPHIES

Chuanxiong Guo [S'99-M'01] received a Ph.D. degree in communications and information systems in 2000 from the Institute of Communications Engineering, Nanjing 210007, China. In 2001 he spent the year to run a research project sponsored by the National High Technology Research and Development Program of China (the 863 Program). He was with Microsoft Research Asia (MSRA) from 2002 to 2004, first as an associate researcher and postdoc, then as a researcher. He is now with the Institute of Communications Engineering, Nanjing. His research interests lie in the filed of networking, encompassing scalable protocols and algorithms design and analysis for wired and wireless networks, multi-service support, mobility management, networking support in operating systems, Internet worm/virus detections and defenses.

Haitao Wu was born in 1976. He received his Bachelor degree in Telecommunications Engineering and Ph.D degree in Telecommunications and Information System, in 1998 and 2003 respectively, both from Beijing University of Posts and Telecommunications (BUPT). Currently he is an Associate Researcher in wireless networking group of Microsoft Research Asia (MSRA). His research interests are QoS, TCP/IP, P2P, and wireless networks.

Kun Tan (M'03) received the B.E., M.E. and Ph.D. degree in Computer Science and Engineering from Tsinghua University, Beijing, China, in 1997, 1999, and 2002 respectively. He joined Microsoft Research Asia as an

Associate Researcher, in April 2002. His research interests include transport protocols, congestion control, delay-tolerant networking, and wireless networks and systems.

Qian Zhang (M'00-SM'04) received the B.S., M.S., and Ph.D. degrees from Wuhan University, China, in 1994, 1996, and 1999, respectively, all in computer science. She joined Microsoft Research, Asia, Beijing, China, in July 1999. Now, she is the research manager of the Wireless and Networking Group. Dr. Zhang has published about 90 refereed papers in international leading journals and key conferences in the areas of wireless/Internet multimedia networking, wireless communications and networking, and overlay networking. She is the inventor of about 30 pending patents. Her current research interest includes seamless roaming across different wireless networks, multimedia delivery over wireless, Internet, next-generation wireless networks, P2P network/ad hoc network. She also participated many activities in the IETF ROHC (Robust Header Compression) WG group for TCP/IP header compression.

Dr. Zhang is the Associate Editor for IEEE Transactions on Vehicular Technologies and IEEE Transactions on Wireless Communications. She is now also serving as Guest Editor for special issue on wireless video in IEEE wireless Communication Magazine. Dr. Zhang has received TR 100 (MIT Technology Review) world's top young innovator award. She also received the Best Asia Pacific (AP) Young Researcher Award elected by IEEE Communication Society in year 2004. She received the Best Paper Award in Multimedia Technical Committee (MMTC) of IEEE Communication Society. Dr. Zhang is a member of the Visual Signal Processing and Communication Technical Committee and the Multimedia System and Application Technical Committee of the IEEE Circuits and Systems Society. She is also a member and chair of QoSIG of the Multimedia Communication Technical Committee of the IEEE Communications Society.

Jingmin Song received his Ph.D degree from Beijing University of Aeronautics and Astronautics in Computer Science, in 2001. He is currently working in Technology Transferring Group in MSR Asia. His interests include networking, multimedia and embedded systems.

Junfeng Zhou is a Software Design Engineer of the Wireless and Networking Group, Microsoft Research Asia (MSRA). He joined MSRA since April 04 and worked on the End-system based Mobility Project. Before that, he received his Master degree in Computer Science in 2004 from Beijing University of Posts and Telecommunications (BUPT). His research interests are embedded system, TCP/IP and P2P.

Christian Huitema is "Director of Wireless Networking" at Microsoft, in the "Windows Networking & Devices" group. His group is in charge of the wireless developments in Windows, and in particular of developing support for 802.11 (Wi-Fi). Until September 2004, he was working as "architect" for Windows Networking, with a special interest for IPv6, IPSEC, Wireless, Peer-to-Peer and home networking. Until January 2000, he was chief scientist, and Telcordia Fellow, in the Internet Architecture Research laboratory of Telcordia, working on Internet Quality of Service and Internet Telephony. Prior to that, he was a researcher at CNET and then at INRIA in France, where he worked on innovative communication protocols, software and compilers, including an IP based H.261 videoconferencing system, IVS, doing video over the Internet in 1994.

He has written several books and publications. He was a member of the Internet Architecture Board (IAB) from 1991 to 1996, its chair between April 1993 and July 1995. He was a trustee of the Internet Society from 1995 to 2001. He was a member of the board of the SIP Forum from October 2001 to September 2003.

Wenwu Zhu (S'91-M'96-SM'01) is Co-Director of Communication Technology Lab China since September 2004. Prior to his current post, he was with Microsoft Research Asia first as a researcher in Internet Media Group and later as Research Manager of Wireless and Networking Group. From 1996 to 1999, he was with Bell Labs, Lucent Technologies, NJ, as a Member of Technical Staff during 1996-1999. From 1988 to 1990, he was with the Graduate School, University of Science and Technology of China (USTC), and Chinese Academy of Sciences (Institute of Electronics), Beijing, China. He has published over 200 refereed papers in the areas of wireless/Internet multimedia delivery, and wireless communications and networking. He participated activity in the IETF ROHC WG on robust TCP/IP header compression over wireless links. He is co-inventor of over 20 pending patents. His current research interest is in the area of wireless communication and networking, and wireless/Internet multimedia communication and networking.

Dr. Zhu has been on various editorial boards of IEEE journals such as Guest Editors for the Proceedings of the IEEE and IEEE JSAC, AE for IEEE Transactions on Mobile Computing, IEEE Transactions on Multimedia, IEEE Transactions on Circuits and Systems for Video Technology. He received the Best Paper Award in IEEE Transactions on Circuits and Systems for Video Technology in 2001. He also received the Best paper award on Multimedia Communication in 2005. Dr Zhu currently is also the Chairman of IEEE Circuits and System Society Beijing Chapter and the Secretary of Visual Signal Processing and Communication Technical Committee. He is a member of Eta Kappa Nu, Multimedia System and Application Technical Committee and Life Science Committee in IEEE Circuits and Systems Society, and Multimedia Communication Technical Committee in IEEE Communications Society.

Wenwu Zhu received the B.E. and M.E. degrees from National University of Science and Technology, China, in 1985 and 1988, respectively, the M.S. degree from Illinois Institute of Technology, Chicago, and the Ph.D. degree from Polytechnic University, Brooklyn, New York, in 1993 and 1996, respectively, all in Electrical Engineering.