Ad Hoc Networking with Mobile IP

Hui Lei
Computer Science Department
Columbia University
New York, NY 10027
lei@cs.columbia.edu

Charles E. Perkins
Sun Microsystems Inc.
901 San Antonio Rd.
Palo Alto 94303
charles.perkins@sun.com

Abstract —

An *ad hoc* network is the cooperative engagement of a collection of wireless mobile nodes. In this paper we describe a mechanism that allows the construction of such an ad hoc network and provides continuous Internet access to the entire ad hoc network. Connectivity within our ad hoc environment is provided by operating each mobile node as a specialized router. Internetworking the whole ad hoc network in the Internet is provided by adaptation of the Mobile IP protocol so that a foreign agent can service a mobile node even if the latter is not within the communication range of the former. Further, we make use of a route manager to coordinate the operations by the routing and mobile IP daemons on kernel's routing table. We also discuss some practical issues encountered in the course of a real implementation.

Keywords: Ad hoc networks, Mobile IP, Wireless access to the Internet.

I. Introduction

Battery powered, untethered computers are likely to become a pervasive part of our computing infrastructure. An *ad hoc network* is the cooperative engagement of a collection of these mobile computers. For instance, any number of people who carry a mobile computer could conceivably enter a conference room and agree to form a temporary wireless network among themselves. Information often needs to be shared between the mobile nodes and between a mobile node on the ad hoc network and a stationary node on the Internet. In order to support these two kinds of information sharing, one has to be able to provide routing within an ad hoc network and to interconnect the ad hoc network with the Internet.

In [7], Perkins and Bhagwat proposed an approach to routing in an ad hoc network. The basic idea is to treat each mobile node as a router and to run a modified Routing Information Protocol [3] among them. The modified RIP is designed to accommodate the highly dynamic, self-starting behavior of mobile nodes and to account for the properties of the wireless broadcast medium. The following modifications to RIP are proposed:

- Each route table entry is augmented with a sequence number, which is originated by the destination node. The use of sequence numbers guarantees loop-free paths at all instants.
- An average settling time is maintained for each destination. The advertisement of routes which may not have settled is delayed in order to damp the fluctuations in route table entries.
- The existing triggered updates mechanism is deployed to propagate route table changes throughout the ad hoc network as soon as any topological changes are noticed.

Our work is a continuation of the above work. Our goal is to provide continuous Internet access to the *entire* ad hoc network. The IETF Mobile IP protocol [9] addresses the problem of mobile node internetworking in the Internet, assuming that a care-of address is available within the communication range of the mobile node. However, such an assumption may not always be true in an ad hoc network. Although it can be assumed that at least one foreign agent exists that bridges the ad hoc network and the

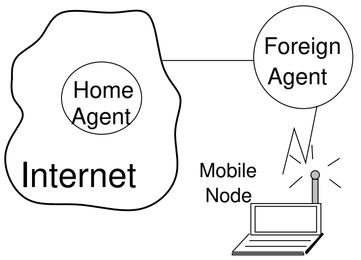

Fig. 1: Basic Operation of Mobile IP

Internet, it may not be the case that every mobile node is within the communication range of a foreign agent. We present a mechanism that effectively extends foreign agent coverage to the whole ad hoc network. A further issue is that both the routing daemon (`routed`) and the mobile IP daemon (`mipd`) need to manipulate the kernel's routing table. Hence we have to make sure that the routing table is always maintained in a valid state.

The rest of the paper is organized as follows. Section II covers the basic features of Mobile IP that are relevant to our study. Section III describes some small changes made to the Mobile IP protocol in order to interconnect an ad hoc network with the Internet. Section IV details our effort to coordinate the route manipulations by `routed` and `mipd` daemons. Section V discusses some practical issues encountered during our implementation.

II. Mobile IP Basics

In this section we shall discuss aspects of the Mobile IP protocol that are relevant to our ad hoc networking study. We emphasize the specific features of the Mobile IP implementation we have been working with.

The basic operation of Mobile IP is illustrated in Figure 1. A mobile node has a home agent. Each home agent maintains a *home list*, which identifies those mobile nodes that it is configured to serve, along with the current location of each of these mobile nodes, if known. When a mobile node connects to a network, it must perform a registration process with some foreign agent attached to the network before datagrams will be delivered to it. The address of the foreign agent, supplied as the mobile node's care-of address, defines the mobile node's current location. The combination of a home address and a care-of address is known as a *binding*. Each foreign agent maintains a *visitor list*, which identifies those mobile nodes that are currently registered with it, and delivers arriving datagrams addressed to one of the visiting mobile nodes locally to the mobile node.

A cache agent is an agent that caches the current binding of a mobile node and forwards datagrams directly to the mobile node. A mobile node's home agent always serves as its cache agent, but using the techniques of route optimization [5], any other computer may also store location information for mobile nodes and thus be a cache agent. In particular, and in our implementation, a mobile node's previous foreign agents may also function as a cache agent and bypass the default reliance on routing through the home agent, thus offering smoother handoffs [8]. It has even been proposed that routers be able to snoop route optimization messages so that they can become intermediate cache agents, but such techniques are fatally vulnerable to unauthenticated maliciously transmitted cache updates.

In our environment, the Mobile IP protocol is executed by a `mipd` daemon which also implements the abovementioned route optimization protocol. When a mobile node is connected to a new network, it discovers new foreign agents by executing a modified form of ICMP Router Discovery [2]. The router advertisements carry a mobility extension and are thus called agent advertisements. The mobile node performs a registration to create a mobility binding. Specifically, the following things will happen after a successful registration:

1. The home agent, and possibly the previous foreign agent, will be notified of the new binding. The kernel's routing table on these agents will be updated with a route to the mobile node via its current care-of address. The new routing table entry is called a *location entry*, and it is tagged with a `RTF_LOCATION` flag. These agents later can direct datagrams to the mobile node's current location using tunneling.

2. The new foreign agent adds the mobile node to its visitor list. It also adds a host route to the mobile node in its routing table in the kernel. The routing table entry, called a *visitor entry*, carries a `RTF_VISITOR` flag. When the foreign agent receives a (possibly tunneled [6]) datagram destined to the mobile node, it will decapsulate the datagram, if necessary, and deliver the datagram locally to the wireless interface.

3. The mobile node adds a host route to the foreign agent. The mobile node also establishes the new foreign agent as its default router, through which the mobile node can access other networks across the Internet.

Each location entry or visitor entry has a timeout associated with it, ensuring that a stale entry does not persist forever.

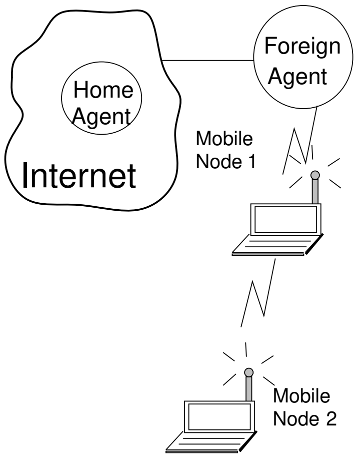

Fig. 2: Ad Hoc Networking with Mobile IP

III. Internetworking an Ad Hoc Network

The Mobile IP protocol allows transparent routing of IP datagrams to mobile nodes. It implicitly assumes that a foreign agent is always available on the network to which the mobile node is attached and that the foreign agent is within the transmission range of the mobile node. The foreign agent essentially acts as a relay between the mobile node and the rest of the Internet. We decided to employ Mobile IP to interconnect an ad hoc network with the Internet. Although it is reasonable to require that some foreign agent exists that bridges between the Internet and the ad hoc network, the foreign agent may not be able to communicate directly with every mobile node in the ad hoc network. This situation is illustrated in Figure 2, where the foreign agent and mobile node 2 are not within each other's wireless transmitter range, but mobile node 1 is within the overlap between the range of the foreign agent and that of mobile node 2. Our goal was to make the internetworking possible with minimal changes to the Mobile IP protocol.

We need to enable unicasting between the foreign agent and every mobile node on the ad hoc network. This can be achieved by making the foreign agent run our improved routing daemon `routed`. In other words, the foreign agent is made to participate in the routing protocol of the ad hoc network. In this way, mobile nodes within range of the foreign agent can cooperate to forward unicast messages between the foreign agent and other mobile nodes outside the range of the foreign agent, as long as those other mobile nodes are close to some mobile node that is within range.

Mobile nodes effect agent discovery by broadcasting agent solicitations and/or listening for agent advertisements broadcast by the foreign agent. So, we need a mechanism to effectively propagate these broadcast messages. We made some further modifications to the `routed` program. When run on a foreign agent, `routed` will pick up the agent advertisement from the loopback interface and include it as an extension to the route table entry for itself (in our improved version of `routed`, an entry needs to be made in the route table for the local node, because sequence numbers are generated by the destination computer in each route table entry).

Piggybacked with the route table entry for the foreign agent, the agent advertisement can reach the `routed` process on each mobile node throughout the ad hoc network. When run on a mobile node that is not within the range of the foreign agent, `routed` will relay the agent advertisement message to mobile IP on the local node, both periodically and upon receiving an agent solicitation from the loopback interface, as follows. `Routed` on a mobile node needs to relay agent advertisements periodically to the local node, since agent advertisements from the foreign agent do not reach the mobile node. According to mobile IP protocol, `mipd` on a mobile node may send out agent solicitations. When this happens, some entity has to send the answering advertisement via a unicast. Since the foreign agent is out of reach, we require `routed` to reply to agent solicitations issued from `mipd` on the local node. The solicitations are known to be sent by the local instance of `mipd`, when they are received from the loopback interface. We expect that if `routed` relays advertisements periodically, there will be no need for `mipd` to ever send agent solicitations. However, in our implementation the foregoing situation is properly handled if it arises.

Following a successful registration, a mobile node will create a default route via the foreign agent in the kernel's routing table. This may be a problem in an ad hoc environment. When the foreign agent is not within the range of the mobile node's wireless transmitter, the default route won't go through, because ARP [10] isn't able to perform address resolution for a foreign agent that is several hops away. The correct router in this case is the next hop on the shortest path to the foreign agent. In other words, the foreign agent is only an "indirect" gateway on the default route; the direct gateway can be determined by looking up the routing table for the foreign agent. Hence we tag the default route established by `mipd` in an ad hoc environment with a new flag `RTF_INDIRECT`.[1] When sending out an IP datagram,

[1] By convention, a default route is also tagged with a `RTF_GATEWAY` flag. `RTF_GATEWAY` indicates that the route is an indirect route since the destination is not directly connected and has to be reached via a gateway. In contrast,

the kernel IP code needs to search its routing table to choose a matching route entry. If the selected route entry carries an `RTF_INDIRECT` flag, IP is made to perform another look-up, only this time for the indirect gateway returned from the first look-up. It is the link-layer address determined in the second look-up process that shall be used to output the IP datagram. We note that when we apply the `RTF_INDIRECT` flag to the mobile IP default route, we don't have to be concerned with whether the foreign agent is actually multiple hops away. If the foreign agent is only one hop away, by the nature of the `routed` operation, a host route (one with the `RTF_HOST` flag set) to the foreign agent will have been added in the routing table. This is the entry that will be located by the second look-up. IP will then decide that the link-layer address of the foreign agent will be used to send out the datagram, which is exactly what we desire.

IV. Coordinating Route Manipulations

Two separate daemon processes, `mipd` and `routed`, can and typically do run concurrently on a computer in our ad hoc environment. Since they both can issue requests to modify the kernel's routing table, it is critical to ensure that the routing table is always kept in a valid state.

In general, routes discovered by `routed` correctly reflect the present topology of the interconnection. Routes created by `mipd`, on the other hand, only serve to indicate the logical associations between a foreign agent and a mobile node. Therefore, when both daemons request that a route be established to the same destination, but via different gateways, the gateway specified by `routed` should be picked. However, the `RTF_VISITOR` flag , which `mipd` running on a foreign agent uses to indicate a visitor route entry, needs to be preserved when a `mipd` route entry is superseded by a `routed` entry. The kernel needs this information to process datagrams destined to its visiting mobile nodes.

We have to accommodate the relatively independent behaviors of the two daemons. It is possible that, on some particular node, one of the two daemons is not used. For example, one mobile node may decide that it will exchange datagrams with peer nodes in the ad hoc network, but not with anyone on the Internet. In such a case, this mobile node doesn't need to invoke `mipd` at all. More often, one daemon may be started and terminated without knowledge of whether the other daemon is up running. Upon exit, each daemon will enter a request that all the routes they have created be cleaned up. If one daemon exits while the other one is still running, the routes created by the latter should be installed in the kernel's routing table exactly as they were specified, even though those routes may have been superseded by the routes created by the exiting daemon. Similarly, when both daemons are running, one of them can add or delete a route to some destination regardless whether the other one maintains a route to the same destination. However, routing information to some destination should be removed from the routing table only if both daemons have deleted their routes to the destination.

`Routed` and `mipd` are cooperative processes, but from a software engineering point of view, the coupling of these two processes should be minimized. We have developed a route manager program, `rtmgrd`, to coordinate the route manipulation operations requested by `routed` and `mipd`. `Rtmgrd` runs as a separate daemon and should be started before `routed` and `mipd` and terminated after them. Some simple changes have been made in `routed` and `mipd` so that they always send their route manipulation requests to `rtmgrd`. `Rtmgrd`, based on the source of the request and the history of route operations, decides an appropriate action to be taken in the kernel's routing table. The following rules are implemented in `rtmgrd`:

- A `routed`-supplied gateway always supersedes a `mipd`-supplied gateway to the same destination.
- The `RTF_VISITOR` flag associated with a route entry, if any, should be preserved.
- When only one of `routed` and `mipd` has requested that a route be created to some destination, the routes should be installed in the kernel's routing table exactly as specified.

We illustrate the operations of `rtmgrd` with an example.

Consider the scenario in Figure 2, where the foreign agent (FA) and mobile node 2 (MN_2) need to enlist the services of mobile node 1 (MN_1) to forward datagrams for them. Suppose `routed` and `mipd` are running on all three computers of interest. On MN_2, `routed` requests the addition of a route to FA via gateway MN_1. After a successful registration with FA, `mipd` on MN_2 will request the addition of a host route to FA. `Rtmgrd` correctly decides to install the route requested by `routed` in the routing table. All the way over on FA, `mipd` will request the addition of a host route to MN_2 with `RTF_VISITOR` flag, while `routed` on the same machine has requested the addition of a route to MN_2 via MN_1. Although it eventually installs an indirect route to MN_2 with MN_1 as the gateway, `rtmgrd` will also tag this route entry with `RTF_VISITOR` flag, as specified by `mipd`.

the `RTF_INDIRECT` flag introduced here indicates that even the gateway specified is not directly connected, but several hops away.

Now suppose for some reason, MN_2 deregisters from FA. Mipd on FA will request the deletion of the visitor route entry it has created. All rtmgrd needs to do in this case is to reset the RTF_VISITOR flag in the corresponding route entry. Apart from that, the route entry remains the same on FA, as long as routed wants it to. For example, mobile node MN_2 could move to a different FA2 while still remaining in range of MN_1, and even within range of FA, for the purposes of ad hoc networking.

V. Implementation Considerations

We have implemented our mechanism in both OS/2 and AIX for use with the mobile computers in our lab. Our implementation includes modifications to Berkeley's routed, minor changes to the Mobile IP program mipd developed at IBM Watson, and coding of the route manager rtmgrd. In this section, we detail some important aspects of the implementation.

Traditionally, a routing protocol involves routers that connect different networks. Each router transmits router advertisements on all its interfaces. In our environment, all the routers participating in the routing protocol may be considered to be part of a single logical non-broadcast network, i.e., the ad hoc network. Each router only needs to send out the advertisements through its wireless interface. So a foreign agent doesn't need to broadcast routing information from its wired interfaces even if it is perfectly capable of doing so.

RIP employs an update technique of *split horizon* to counteract (but not completely solve) the slow convergence problem [1], [4]. When using split horizons, a router records the interface over which it received a particular route and does not propagate its information about that route back over the same interface. Such a technique is not applicable in a wireless ad hoc network, due to the inherent nonuniform propagation characteristics of wireless transmissions, and the fact that most nodes have only a single network interface.

In an ad hoc network, it is common for one mobile node to forward IP datagrams for two other mobile nodes which cannot themselves directly exchange datagrams. Since the forwarding node is sending the datagram out the same wireless interface on which the datagram arrived, the ICMP *redirect* mechanism [11] would be triggered, if care were not exercised. If the forwarding node sends the original sender an ICMP redirect message, the original sender will incorrectly modify its kernel's routing table, treating the destination node as within its transmission range. In order to avoid this, we have modified ICMP so that it doesn't send redirect messages when forwarding along an ad hoc route, and it doesn't update an ad hoc route upon receiving a redirect. We recognize ad hoc routes by associating a flag RTF_ADHOC with each routing table entry created by our routed.

Mobile IP specifies that advertisements for foreign agents contain a sequence number in the range from 0 to 0xffff. A foreign agent, after booting, uses the number 0 for its first advertisement. Each subsequent advertisement will use the sequence number one greater, with the exception that the sequence number 0xffff is followed by sequence number 256. Accordingly, when a mobile node receives an agent advertisement with a sequence number no greater than that in the previous advertisement from the same agent, it assumes a rollover of the sequence number if the new sequence number is equal to or greater than 256. Otherwise it assumes that the foreign agent has rebooted, and it will perform another registration with the same foreign agent. The implication of this convention is that we have to prevent a mobile node from receiving duplicate advertisements. Duplicate advertisements are possible because routed engages in relaying of advertisements. If a mobile node always receives duplicate copies of advertisements that have a sequence number less than 256, it will reregister constantly even if the foreign agent has not gone down, resulting in what we call *thrashing*. In order to prevent thrashing, we require that:

- When routed on a mobile node can receive the agent advertisements directly, i.e., when the mobile node is within the range of the foreign agent, routed will not relay the advertisements to the local node.
- Routed on a mobile node should refrain from relaying the same agent advertisement more than once. Otherwise, since routed and mipd operate in a relatively asynchronous fashion, it is possible that two consecutive advertisement relays by routed to the local node originate from the same agent advertisement.

In cases when more than one foreign agent are available, routed can make an intelligent relay decision. As advertisements are propagated to routed along with the distance information of the foreign agent, routed is able to single out the agent that is closest, and to relay the advertisements from that agent only. Considering the highly dynamic topology of interconnections in an ad hoc network, routed shall switch to advertising a new foreign agent to its local mipd only when the distance to the new agent is significantly shorter than that to the current agent.

Finally, we note that nodes in an ad hoc network will typically appear to reside on different networks, as indicated by the network number defined by their

IP address. Therefore, limited broadcast rather than directed broadcast should be used to advertise route and agent information. Furthermore, when adding an ad hoc route, say, for destination `dst` and gateway `gw`, the determination of the proper local interface can no longer be based on matching addresses. Instead, the routing table has to be consulted to locate the corresponding interface for the specified gateway `gw`. This is feasible because a route to `gw` must have been added in the routing table.

VI. Conclusion

We have designed and implemented an ad hoc networking mechanism that provides convenient connectivity for mobile computers and continuous interconnectivity with the Internet. Routing within the ad hoc network is provided by modeling the mobile nodes as routers and running a modified RIP protocol among them. We have augmented Mobile IP so that it can internetwork an entire ad hoc network in the Internet. With the coordination of a route manager, the routing and mobile IP daemons cooperate with each other while maintaining high autonomy.

We are in the process of tuning and evaluating our mechanism. We also hope to explore further the application-level support needed to automatically enable use of the networking layer routing capabilities to provide simple access to conferencing and workplace tools for collaboration and information sharing.

VII. References

[1] Douglas E. Comer. *Principles, Protocols, and Architecture*, volume 1 of *Internetworking with TCP/IP*. Prentice Hall, Englewood Cliffs, N.J., second edition, 1991.

[2] Stephen E. Deering, Editor. ICMP Router Discovery Messages. RFC 1256, September 1991.

[3] C. Hedrick. Routing Information Protocol. RFC 1058, June 1988.

[4] J. Jaffe, A. Baratz, and A. Segall. Subtle design issues in the implementation of distributed routing algorithms. *Computer Networks and ISDN Systems*, (12):147–158, 1987.

[5] David B. Johnson and Charles E. Perkins. Route Optimization in Mobile-IP. draft-ietf-mobileip-optim-05.txt, November 1996. (work in progress).

[6] Charles Perkins. IP Encapsulation within IP. RFC 2003, May 1996.

[7] Charles Perkins and Pravin Bhagwat. Routing over Multi-hop Wireless Network of Mobile Computers. *SIGCOMM '94 : Computer Communications Review*, 24(4):234–244, October 1994.

[8] Charles Perkins, Andrew Myles, and David Johnson. IMHP: A Mobile Host Protocol for the Internet. *Computer Networks and ISDN Systems*, 27(3):479–491, December 1994.

[9] C. Perkins, Editor. IPv4 Mobility Support. RFC 2002, October 1996.

[10] David C. Plummer. An Ethernet Address Resolution Protocol: Or Converting Network Protocol Addresses to 48.bit Ethernet Addresses for Transmission on Ethernet Hardware. RFC 826, November 1982.

[11] W. Richard Stevens. *TCP/IP Illustrated, Volume 1: The Protocols*. Addison-Wesley, Reading, Massachusetts, 1994.