Secure Mobile Networking
Fourth Quarterly Report

James R. Binkley and John McHugh
Portland State University

14 October 1996

Project Status Overview

Summer was very busy and much progress was made. It is reasonable to say that the project is up to speed at this point.

At the beginning of summer we were able to add 1 fulltime person and 3 Research Assistants to the project. In spring, David Reeder graduated from PSU with a B.S. in Computer Science and at the beginning of summer he joined our project as a fulltime employee. The three R.A.s are Xu Hao, Radheka Godse, and Bjorn Chamblis. Radheka and Bjorn are new graduate students as of summer.

Accomplishments for the summer include:

- In early summer we made an alpha release of our Mobile-IP implementation for FreeBSD 2.1 which included routing daemons for mobile nodes and agents, utilities for making status enquires of the routing daemons, and drivers for ISA and pcmcia cards.
- New personnel were trained in our Mobile-IP implementation and brought up to speed in terms of project plans.
- An X-based user interface was created for the Mobile Node daemon. The aim of the tool is to allow the user to control and observe various aspects of mobile status including especially security policy and setup, mobile configuration management, and various runtime statistics including agent signal strength. There is a section below with more details on the tool.
- We have completed a port of the NRL IPSEC stack into our BSD based kernel (FreeBSD) and intend to use this as a foundation for our fall work on integration of IPSEC and Mobile-IP routing. We used the NRL June release.

- We have for the most part finished the coding of our authenticated ad hoc routing protocol (referred to in previous reports as "network authentication"), which is intended as a replacement for ARP. Initial testing has shown proof of concept. We intend to finish testing this quarter, integrate control of this sub-system with the X interface, and deploy it on our Mobile-IP network at the end of the test period. This protocol has both network security and important redundancy aspects. We will provide more details below.

- We enhanced our Mobile-IP implementation to allow quick handoffs between radio-based agents. This work was primarily aimed at providing redundancy along the lines of switching between agents when one agent should fail. It also applies to path testing between MN and HA. If the route via one FA to the HA should fail and another FA is available, we will try it. This work will provide us with a basis for what we call Foreign Agent redundancy later down the road.

Mobile-IP

An alpha release of a Mobile-IP based on draft-15 was made around the end of June. Our implementation at that time had around 12000 lines of C code (counting *.c, and *.h but not kernel mods or the X user interface which was produced after the alpha release). The implementation included our ISA and PCMCIA Wavelan drivers that have been ported from the original MACH versions developed by Anders Klemet. The release included man pages for the routing daemons (MN and agent (HA/FA), man pages for the configuration files, source, makefiles, kernel patches for FreeBSD 2.1 and for a few related binaries (route/netstat), and an implementation guide. To our horror, we discovered a nasty kernel bug in August, and patched the implementation to fix it.

We have also stumbled upon a few Wavelan enhancements, in one case provided by a user, and have kept the drivers available and up to date. Although we have not kept statistics on how many times either the Mobile-IP implementation or the Wavelan drivers have been fetched, they seem to be reasonably popular. For example, we have been contacted by a student of JJ Garcia-Luna-Aceves at UC Santa Cruz doing mobility research who wanted the drivers.

The ftp site is:
`ftp://zymurgy.cs.pdx.edu/pub/mobility`.

The alpha release is also findable from our Secure Mobile Network web page:
`http://www.cs.pdx.edu/research/SMN/index.html`.

We should point out that the effort to complete a Mobile-IP implementation is huge. We are skeptical that there might be a complete Mobile-IP implementation anywhere. For example, our release is aimed only at Foreign Agent support and includes no support for a COA-based Mobile Node. This is because we intend to work on Foreign Agent redundancy issues (and in fact have already done work this summer along those lines). Also the COA-based system is probably impractical (at least for us) in a useful sense, as we lack the needed administration to setup DHCP (or some other mechanism) to dynamically allocate per-subnet IP addresses on real local networks. We have reviewed draft-17 over the summer and made a list of missing features. However we are attempting to minimize any more Mobile-IP work. Our current implementation suffices for our research needs, although we may want to add a IETF required missing feature or two along security lines as we go along. Our goal at this point (and in fact at mid-summer) was to only *enhance* the system along our desired lines of research in security and redundancy. This work is beyond the limits of the IETF Mobile-IP specification.

Considerations Regarding Timestamp Replay In Mobile-IP

We implemented the timestamp replay protection mechanism in Mobile-IP. Timestamp replay protection is done between the Home Agent and Mobile Node and applies to the UDP registration and reply messages. The HA provides 32 bits worth of NTP time (seconds) and the Home Agent may send a NAK and a better time if it believes the MN is out of sync. We did the obvious thing here which was to make the MN implementation willing to accept a time update from the HA (there is also a crude attempt at calculation of round trip latency made in our system). Our implementation allows the MN to choose between ignoring the HA and somehow updating its own time or accepting a HA update. However we currently have no better choice than to accept the HA's time. We have noticed that our PCs will typically get out of synchronization a few times per hour which may be acceptable. However latency due to congestion or number of hops combined with the need for synchronized time is worrisome. We performed an outrageous latency test by taking a MN home, and by running it over a local Foreign Agent connected to the PSU network over a 28.8k SLIP connection. We subjected the slip link to extreme degradation by using the popular UNIX ncftp client to perform an FTP download which used up all bandwidth on the slip connection. There were 100s of queued packets typically (and certainly some packets were dropped) as NCFTP typically does 30K i/o chunks (presumably in an effort to take advantage of ever-increasing TCP windowing sizes). Mobile-IP registration packets had difficulty getting through and we found that we had to tune our retry mechanism to make it slower in order to make it more robust. More surprising, log messages in the Mobile Node's security log showed that it was receiving HA NAKS with the new HA timestamp the same as the current MN's time. The HA logs showed that the MN was truly out of time syncronization with it on the order of 5 seconds or more. The queueing of packets at the slip client/server was real and caused an immense slow down of MN control packets. The bottom line here is that latency is problematic. We currently tolerate "time slop" of about five seconds. That may not be enough. From a security point of view, the time slop needs to be as small as possible. From a network latency point of view, it needs to be as long as possible.

One thought is that Mobile-IP packets should be given higher priority by IP than other packets (certainly bulk FTP packets).

There is interest in Mobile-IP circles in building FA systems where there is a hierarchy of FAs, say an interior "mother FA" that communicates status between MNs and HAs, and leaf FAs that actually deal with handoff between visiting MNs and local cells. The goal of such an architecture is to cut down on MN to HA exchanges over the Internet and thus reduce Internet traffic. The timestamp mechanism where HAs need to sync the MN mitigates against such an architecture.

We submit that Mobile-IP's approximation of NTP time via a few bits in a Mobile-IP registration packet is probably insufficiently complex. However what other possibilities exist for synchronization of time?

1. one might wish for an atomically-powered PCMCIA clock card, but certainly we know of no such card available now. We have been told that such cards might actually be available in the future.

2. we might use GPS, but GPS will not work indoors.

3. we might try and use NTP itself, and once we have IPSEC protection between the HA and the MN, we will try it and see how much bandwidth is actually used.

Lab and Wavelan Considerations

At this point we are only attempting to modestly extend our Mobile Network. We intend to add a few Mobile Routers (agents) for other buildings on campus over the course of the year. We also hope to place one Foreign Agent (FA) at Oregon Graduate Institute, which should be a rather interesting exercise both in network administration (they have a firewall. The FA will be installed as a "bastion host") and may shed light on how timestamp-relay based Mobile IP works across a number of Internet hops (currently 7 and subject to change).

We made a minor modification to our Wavelan drivers over summer that allows us to dynamically change the Wavelan Network ID (NWID). The NWID is used to allow Wavelan networks to logically distinquish between NWID Network I and NWID Network J. I and J will not receive packets from each other (they actually still use the available bandwidth, but the radio modems ignore packets from NWIDs not their own). We can boot laptops or mobile routers and set them to a different NWID. This allows us to setup a logical Wavelan link that occupies the same physical space as another network and is incredibly convenient for "lab testing". As a result, we can setup FAs on another NWID, and in essence have a small "virtual" lab that exists in the same place as a real radio network.

Authenticated Ad Hoc Routing

We are currently primarily engaged in our work to make Mobile-IP more secure and are in the process of finishing one of the two major portions of that work which we have called in the past "Network Authentication" and will refer to in the future as "Authenticated Ad Hoc Routing". (The second portion is a combination of Mobile-IP and IPSEC at the routing layer and will be discussed below). It should be emphasized that this work is an extension of Mobile-IP and is fundamentally a link-layer routing protocol that is analogous to the ISO ES-IS protocol and somewhat similar in function to ARP. The protocol is important from both routing and network security points of view.

Let us briefly outline how the system works. We have experimentally enhanced the ICMP router discovery protocol (RFC 1256) along lines suggested by the Mobile-IP implementation. In the current Mobile-IP system, only agents beacon. In our system, all entities including both agents and Mobile Nodes beacon. Beacons are either broadcast to limited broadcast (which is what we are currently doing) or multicast. We have added three Tag Length Value (TLV) suffixes that imitate and physically follow the Mobile-IP extension that is added to the RFC1256 router discovery packet. (From here on in, we call this a "beacon"). Logically the packet may consist of the following sections:

```
(ip header,
 ICMP router discovery header,
 Mobile-IP extension portion (required by Mobile-IP),
 new MAC address TLV,
 optional id string TLV,
 network authentication extension,
 optional ad hoc authentication extension)
```

The IP address of the sender is found in the ICMP router discovery header. The MAC address portion contains the MAC address of the sender. It is up high for reasons of "convenience". Our daemons are applications after all, and it is not easy in a BSD kernel stack to get MAC info upstairs to the application layer. Convenience here is close to necessity.

An optional information string may be provided. Our agents currently state where they are, along the lines of "Guinness: PCAT room 128D". MNs might state their user or might simply not use this facilty. The location string may optionally be placed in the agent or MN configuration file. If found, it will be placed in the beacon. This information is displayed in our X user interface for the MN daemon (section on this below).

One or two authentication extensions must follow. Our authentication mechanism here is analogous to the Mobile-IP authentication mechanisms; i.e, we authenticate via a shared secret based on a MAC algorithm (currently MD5, but there is a spi in the authentication extension, and the algorithm in use could of course be changed). We will discuss the authentication header more below, but for now there are several important points that need to be made:

1. note that we are essentially associating a secret key with a link. Key distribution (static for us of course) is viewed as belonging to the multicast key distribution problem, as obviousally one system sends out a broadcast/multicast beacon and N systems receive it.

2. The authenticated information logically includes at least the following information: (ip address, MAC address, and some control information about the system (it's a FA, HA, or MN)). The authentication is done over the ICMP header (where the IP address is), and all higher headers up to and including the spi in the authentication extension.

3. there are at least two possible keys that may be concatenated.

Any agent (FA or HA) or MN that belongs to the same logical link define themselves in terms of a fundamental network authentication extension. A second ad hoc key may be generated on the spot and given away for a brief period for ad hoc (face to face) meetings between MNs. For example, the key might be generated via a shared passphrase fed to MD5 when visitors come to your site or when you have a face to face rendezvous with some other Mobile Node. This key exists primarily so that no user ever gives away the site network authentication keys. It should be pointed out that an implementation might choose to have a list of keys on a per agent (HA or FA) basis for the network key and index these via agent IP address. In our implementation, we currently only have the network key (and ad hoc secondary key) supported, but this is an implementation detail and for now we our implementation will suffice for proof of concept. It should be noted that we accept beacons if either key is successfully authenticated. This allows MNs that are at home to speak to the ad hoc systems (the visitors) directly and retain their own current agent connection. It also allows a site manager to teach a FA about the ad hoc key for the duration of a meeting, thus visitors may be allowed controlled access to the Internet. The more important aspect there is that a site manager would NOT have to give a visitor a local network key that was in use by others (especially Mobile Users who are local and can thus not be knocked off a net link due to a key change). We reject beacons that fail authentication or lack it. In summary, beacons provide an authenticated MAC and IP address pairing.

Some basic design choices were made here. For the most part they were made due to reasons of Mobile-IP protocol coherence. One might ask the following pointed questions about our choices:

1. why not enhance ARP? One might argue that ARP was not designed to be enhanced (even though this has actually been done in the past). We felt that

since Mobile-IP had already added extensions to ICMP router discovery and had a reasonable TLV way for doing that, it was natural to enhance the Mobile-IP beacon. The MAC enhancement and location enhancements actually have general utility for Mobile-IP by themselves.

2. why not use IPSEC AH? The strongest reason is that we wanted to have a mechanism where we could have two authentication keys available, even if one of them was rarely used and it is not clear how AH keys might be provided in pairs. There would need to be a definition of AH made that would allow more than one key per packet. It is also much more convenient to imitate Mobile-IP where we can do the work at the application layer. One of the more compelling aspects of our Mobile-IP implementation so far is that we have attempted to minimize kernel modifications as much as possible. (We will outline the kernel modifications needed for this work below). Of course the most honest reason is sheer convenience because we did not until just recently have AH available. There is probably some way AH could be used. It should be pointed out that by putting an IP, MAC pairing above IP (as an ICMP payload), one has made the use of IPSEC possible here and there is no IPSEC definition or applicability for ARP.

Ad Hoc Routing Considerations

From the routing point of view, the major benefits are as follows:

1. link-layer reachability between nodes (agents and MNs or MNs and MNs) is estabilished simply because hearing a beacon defines reachability. A higher-level ad hoc routing protocol along lines discussed elsewhere by Dave Johnson (ad hoc routing driven by a source-routing technique), or Charlie Perkins (ad hoc routing according to a vector distance technique) might use parts of this protocol or be based on top of it. It is not unusual for routing protocols to seek to establish the existance of systems via "echos" or logical "hellos".

2. network redundancy is improved simply because the rather absurd restriction on same IP subnet-only access is removed. Two Mobile Nodes from different subnets can talk to each directly as long as they can hear each other. This is a simple form of ad hoc routing (link-level only) and does not allow cross-hop routing. No intermediate router is needed.

Before we go on, we review a couple of points we have made before in regard to IP subnets and radio-based topologies. We also will explain the routing theory behind our beaconing sub-system.

First of all, no one should ever make the mistake of assuming that radio based links (when they are broadcast based and not point to point) are topologically similar to ethernet or other wire-based technologies. With ethernet, the same subnet prefix implies that system A on a wire CAN actually hear system B (barring too many collisions). The notion of an IP subnet is 1 to 1 with the notion of a wire. Subnetting is not a meaningful concept on a radio link. It is an optimization on the problem of how a sender determines who is local and who is not. After all the classic IP routing algorithm is:

```
if same subnet
        use ARP
else
        send packet to default router
```

With radio the fact that 3 systems share subnet IP addresses even if they are close does not mean that they can talk to each other. Imagine a radio cell with three systems, A, B, C. B is an agent in the center of the cell. A and B are Mobile Nodes and are diametrically opposed at the edge of the agent's cell. Thus A and B might not be able to talk to each other directly. The consequences here are profound. If A tries to talk to B, according to most IP kernel stack implementations, it will attempt to use ARP. This won't work. It must instead send the packets to the agent. On the other hand, the agent B, must not try and use a routing redirect to tell C to talk to some other router D. B has no way of knowing whether or not C and D can talk directly. We simply use beacons here to install link layer routes and our resulting routing algorithm is roughly:

```
if a link-layer route exists (because of beacons)
        use it
else
        use the default
```

Of course one may also make the observation that Mobile-IP renders subnetting absurd too. After all a Mobile Node may wander into a link at an agent that contains other Mobile Nodes or may simply find itself in an ad hoc situation with other MNs, none of which share the same IP subnet. Mobile-IP in the current draft attempts to fixup some of the link-layer toplogical glitches via fiddling with ARP. For example, at a Home Agent, the MN should issue a gratuitious ARP when it returns from aboard. The HA should also proxy ARP for the MN while it is away on business. However these mechanisms are limited and do not address all of the topological problems, some of which may be built into implementations. For example, if you have a BSD kernel and two MNs that share the same subnet are at different Foreign Agents, they will not be able to talk to each other. This is because as long as the assumption that same subnet == direct link is present (due to a routing table entry (BSD) or worse built into the code (brand X)), then one MN will attempt to ARP for its peer as opposed to using Mobile-IP (which would work). In our current routing system, the

MNs (shared subnet, but at different FAs) would simply use the default route (to the agents), and thus can take advantage of Mobile-IP in order to find each other.

In order to implement our current version of link-level only ad hoc routing, we had to change some fundamental kernel assumptions. Kernel modifications themselves were small but profound. First we modified the operating system so that ARP could be turned off on a per interface basis. This is done with an ifconfig -arp flag. Ironically the flag existed in the BSD kernel, but did nothing. We taught the ARP code in the kernel (which is actually fundamental to driver use since the ARP code is always used for mapping IP addresses to MAC addresses) to ignore incoming ARP packets and not use ARP on output. The ARP cache mechanism is still maintained for mapping IP to MAC addresses. However in the case of the ad hoc system, in general, (but see below for an important tie-in between Mobile-IP and this routing system at agents), an authenticated beacon will cause the Mobile-IP routing daemon (MN or agent daemon) to insert a IP/Mac mapping. Unlike with ARP, the daemon tracks the routing entries and times them out. This gives us link-layer routes (IP to MAC as gateway). It also gives us daemon control over link-layer routes as opposed to having them magically appear just because an ARP was done due to a packet sent to a local subnet IP address.

A more subtle and invidious hazard had to be negotiated too in terms of how link-layer routes were inserted. The ARP mechanism in 4.4BSD is tied to a particular per interface route table entry that has the flags UC. UC here means "user" and "clone". We call this the "clone route". The clone route is of type LLINFO; i.e., it is a link layer route and its gateway may be a MAC address. Its destination is the subnet ip address. The clone route is installed when an interface is configured with ifconfig and an IP address and subnet mask is bound to the interface. The clone route itself has an empty gateway, which represents the link layer part. When ip output tries to send a packet to an ip destination that matches the clone route in a routing table lookup, a link-layer route is created (cloned) for that ip destination. ARP will go off and fill the MAC address in. Thus the clone route causes link layer routes to same subnet destinations to magically appear in the routing table. On the input side, packets coming in from a subnet/link peer will cause ARP to allocate via cloning a link-layer route for the peer. Thus a routing table entry is used to implement the subnet/ARP idea.

This semantic is not exactly hard-wired into the BSD implementation. We choose to rework our Mobile-IP routing mechanism for adding default routes and link-layer routes so that they did not depend on having the clone route present. Thus if we choose to use our authentication/ad hoc mechanism, our daemons on a per interface basis remove the clone route (and turn ARP off on the interface). With no clone route present in the radix tree, and with a link-layer route (dst=IP/gateway=MAC) installed due to an authenticated beacon, packets will be forwarded locally when a system is actually local. If it is not, packets will be sent to the default and Mobile-IP will come into play for Mobile Nodes (or not) that are elsewhere.

We expect that agents will be configured at boot with ARP on their ethernet side and with ad hoc on their wireless side. MNs are capable of being configured dynamically via our X user interface and can change between "modes" (routing state must be flushed). Our user interface will show the user whether or not beacons are being received that are authenticated, incorrectly authenticated, or lack authentication (presumably you are in ARP land). The user may then decide to dynamically change the link mechanism. However the system itself does not currently change the mechanism, which is a conservative approach. We simply want to use the user interface to better inform the user about toplogy and security aspects.

The traditional ugly question of scalability should be raised. Some have denigrated ES-IS in the past and promoted ARP, but if you can't make the subnet "net==link" assumption it is hard to see how you can ask on demand as opposed to simply be told somehow. Beacons are one way to be told. We also must keep the lack of transitivity with radio links in mind.

So let us accept for the moment that beaconing is OK, but is less scalable. How can we make it more scalable? We observe that Mobile Nodes do not need to beacon as much as agents. So far we have only done minimal testing (more follows this quarter) but we are considering setting the MN beacon rate to be 5 to 10 times slower than that of agents. We feel that MNs for the most part need to correspond with the agent, and MN to MN interaction may exist but is rare. As a result, our implementation sends a beacon out in front of the MN UDP registration messages (current rate at stable state is about 1/3 of the agent lifetime, which is 1/3 of 2 minutes, hence one reregistration every 40 seconds). A beacon rate of once per 10 seconds is probably fine for MNs with current usage. If a group of us sits down in a conference room, we will have established connectivity by the time we get around to actually doing anything. We suggest that the beaconing rate should be tied to motion and to the rate of MN beacons received from other MNs. If an MN discovers MANY MN beacons, it may slow its own rate down. On the other hand, if it moves it might speed its rate up. Of course, the question to ask here is how does an MN tell its rate of speed? There is room for future research here.

Ad Hoc Routing Security Considerations

There are probably at least four threats that are addressed here:

1. ARP spoofing in the active sense; i.e., a spoofing system steals the IP address of another system and uses it. With traditional ARP spoofing, the spoofer only changes its IP address and retains its MAC address.

2. denial of service due to loss of a link-layer route courtesy of an ARP spoof.

3. denial of service due to unauthenticated ICMP beacons sent by a fake agent. This is more dangerous in a radio topology if handoff mechanisms (like our current handoff mechanism) are based on signal strength.

4. the possibility that a user might due to frustration decide to give away the network link key. This is why we want to emphasize and make available a key that can be easily manufactured, destroyed, and given away in short term face to face ad hoc situations. In such a scenario, MNs from different domains may be present and all may be disconnected from the Internet.

#1 and #4 are probably the most important cases that need to be considered. #2 is an outgrowth of #1. #3 is a very real possibility and should not be considered unimportant. We will not say anything more about it as our beacons are authenticated by definition.

Without explaining exactly how to do this, (although we fear the mechanism is all too obvious), we have "successfully" performed an ARP spoof at a Foreign Agent. One MN (the attacker) overwrote the ARP cache for another MN (the victim) at a Foreign Agent. The result was that the attacking MN (which assumed the IP address of the innocent MN) was able to steal both the link layer routing entry AND the tunnel between the Home Agent and the Foreign Agent. As a result, without running Mobile IP, the attacking system was able to use the spoofee's Mobile IP setup. The only Mobile-IP attribute that the attacking system needed was the ability to bind the default route to a agent not on its subnet. Of course, at a Home Agent the attacker would not need that ability, but at HOME, the scenario is exactly equivalent to traditional ARP spoofing. On a radio link, it is further possible that an ARP override might not be detectable by the victim, due to the non-transitive topology we discussed earlier in this report. Possibly, something even worse might be accomplished, as there might be a way for an attacker to feed the Mobile-IP UDP reply packets returning from the HA back to the victim and thus keep the Mobile-IP connection fresh.

In summary we don't believe that ARP is a very good foundation for Mobile-IP, especially on radio links where a radio cell might easily extend outside buildings, even to passing vehicles. Mobile-IP has decent mechanisms for authentication of its UDP control messages including replay prevention. However we hijacked an authenticated MN/HA UDP Mobile-IP connection. Further if there had been authentication between the Mobile Node and the Foreign Agent, it would not have mattered. Mobile-IP authentication mechanisms only apply currently to its UDP control packets (registration/reply). We might judge the hijacking of such a Mobile-IP connection to be a more serious breach than a traditional ARP spoof, since we highjacked an authenticated tunnel.

One might think that ARP over ethernet is fine since ethernet links are (maybe) more subject to managerial controls than radio links. Whether this is an illusion or not is something we are not going to discuss here. However it may very well be a dangerous illusion and there is nothing to prevent our authenticated beacons from being used on ethernet too. IPSEC can be used to authenticate and encrypt ordinary data traffic, or applications like ssh/slogin might be used for authenticated and encrypted TCP sessions. However neither Mobile-IP or IPSEC address ARP as a protocol. Use of IPSEC might reduce such a highjacking to simple denial of service (we hope). However our mechanism would prevent such a denial of service attack (loss of link-layer routing) or at least make it more difficult.

The use of authenticated MAC and IP addresses changes the playing field for spoofing. An attacking system must now spoof both the MAC address and IP address. Certainly some lan controllers have programmable MAC addresses and it is possible to adopt some other system's MAC address. Note however that since the attacker intends to use the MAC address of the victim, the victim might see returning packets (or all packets) due to the shared unicast MAC address. For example, think again about our radio setup with the three systems, A, B (FA), and C. If A spoofs C, C might not be able to see A's packets. However packets returning from B to A would be seen by C. C therefore has an opportunity that it would not have had with a simple ARP/IP spoof, where the attacker's MAC address is different. If A starts a TCP connection, C would issue a TCP reset against it. Therefore what is needed here is that C should be "alert" and notice anything peculiar going on at the link. It should be able to see spoofing. What is needed is a user interface that informs the user about network stack security events of interest, TCP resets, UDP ICMP no such port errors and the like.

On the other hand, an attacker might simply be passive, but if it is passive it isn't gaining anything that it can't do by analysis of local traffic without going to the trouble of capturing packets and reprogramming its ethernet controller. IPSEC addresses such a threat.

IPSEC provides network and transport authentication, encryption, issues and we feel that this ad hoc mechanism is complimentary with IPSEC. An authenticated ad hoc link-layer routing protocol coupled with IPSEC mechanisms between MN and HA may be said to form the "moat" and "castle wall" of a more secure roaming node.

One interesting question remains and that is what if anything should be done about possible replay attacks? What if an attacker soaks up your link-layer authenticated beacon at one FA and goes to another and plays it back? In this case, the victim would not be present to notice what is going on. Possibly replay attacks are a higher level problem and we should not worry about them here. After all, all you gain is a route in either a Mobile Node or agent. However, we have ruled replay attacks out at agents due to careful implementation. Our agents will not install the link-layer route (at either HA or FA), until the Mobile-IP UDP registration packet returns successfully. This means that an attacker must also minimally successfully breach the MN/HA Mobile-IP security relationship. Hence to some extent, we are taking advantage of Mobile-IP replay protection and authentication. The remaining question may be reduced to what, if anything should be done about MN to MN relations? Of course in that case, spoofing must be done locally. It is possible that we might use a challenge-response mechanism of some sort but we would like to get advice on this problem space from other quarters. Opinions and thoughts of any readers are appreciated and requested. For now, we observe that close attention to what is happening on the link layer (is anybody using my MAC address?) will suffice.

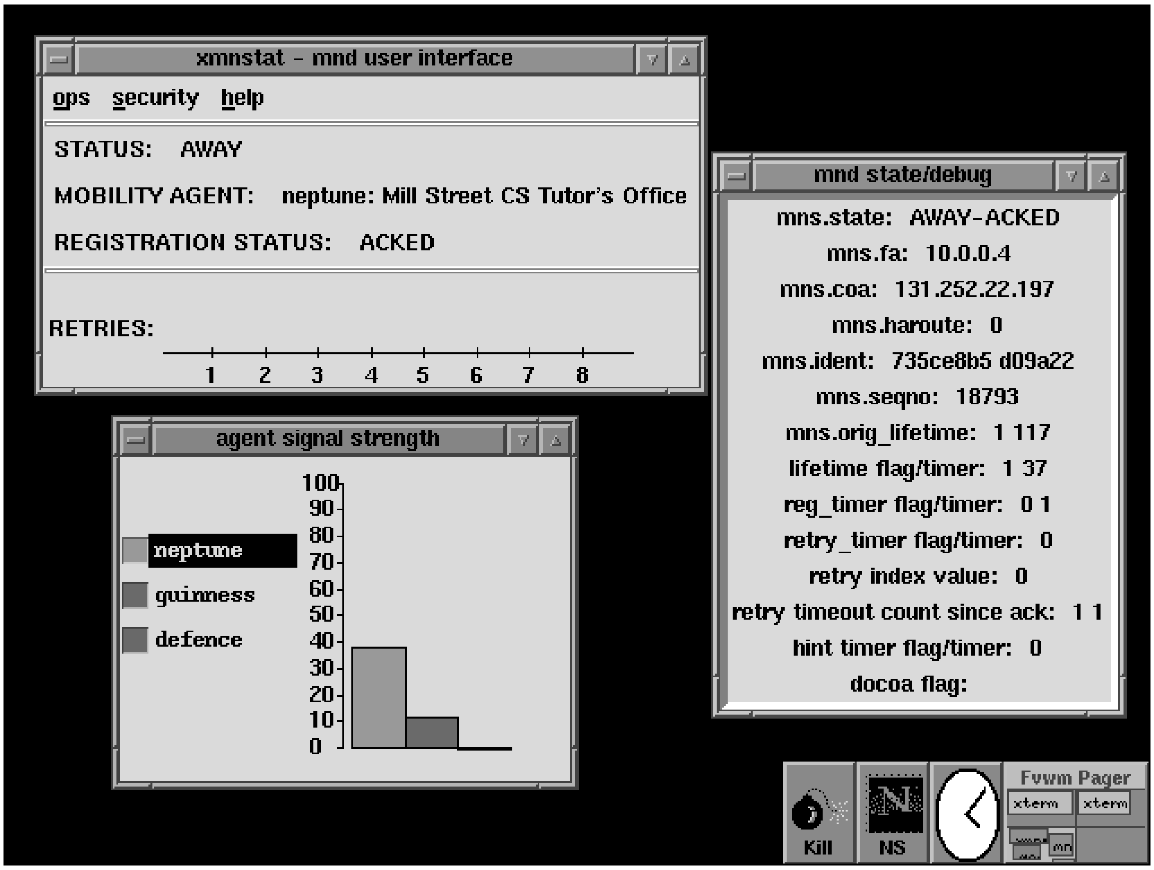

Figure 1: Mobile-IP Status Windows

X-based User Interface for Mobile Routing Daemon

We feel that it is fundamentally necessary to build a GUI as a visible controller for mobile interactions, especially security interactions. Our visual display utility is called “xmnstat”. We will use this tool to control and view various state and security parameters of the Mobile-IP node. For example, there are screens that show current agent status, a running histogram of available agent signal strength, and prototypes of network authentication (link/ad hoc) security and (not yet, but soon) IPSEC policy. We present a couple of the screens here and will briefly discuss them. The implementation is being done in TCL by one of our graduate student RAs. In terms of security mechanisms, the ui will offer certain policy mechanisms to the user, and then if the user changes the policy, the ui uses a UNIX socket (no off machine control - we view our laptop systems as single user systems) to instruct the Mobile Node daemon, which in turn will manipulate the operating system as needed.

(The screens are available in color on our web site as the following urls:

1. http://www.cs.pdx.edu/research/SMN/index.html/summer96/status.gif

2. http://www.cs.pdx.edu/research/SMN/index.html/summer96/linkauth.gif)

Figure 1, the "status" windows show various windows associated with normal Mobile-IP operation and is NOT a prototype. One may see the current state (ATHOME or AWAY) and information about current agent connections in either small or large amounts of detail. The latter of course is primarily useful for debugging. The most "demoable" part here is the running 5 second histogram display of agent strength. We scale Wavelan signal strength on an arbitrary 0..100 percent scale (which is not how the Wavelan driver scales it), and regard signal strength $< 10\%$ as bad. The agent strength sub-window shows that we currently can hear 3 agents, and are hooked up to neptune (FA) as it has the strongest signal strength. Connectivity to guinness and defence are weak, which makes a lot of sense given that they are in another building.

Figure 2, the "link/ad hoc authentication" security screen is currently a rough prototype and will change. We will allow the user to do the following:

1. choose between ARP and ad hoc (authenticated beacon) modes. All state will be flushed in the MN daemon as appropriate when a mode switch is done.

2. try and supply various hints about what kind of link you have wandered into; .e.g, if we are in authenticated link mode, we increment counters (but do not store or setup routing) that will allow the user to see that there are agents out there that are not speaking authenticated beacons. Thus the user may decide what the policy will be for this link. To begin with, information about agents "in the other mode" will take more of a "dumb lights on the dashboard" approach; i.e., we will show the existance of entities in the other mode, but not provide a nice list of them. We may (as time permits) attempt to enhance the user interface in this regard and show a list of agents that we can't talk to.

3. If the user is using authenticated beacons, the user may make certain simple policy choices about those beacons as we have two possible keys in place. The only real functionality here is focused on the secondary optional ADHOC network key. The user may choose to accept/reject beacons with such a key. By default, such beacons are rejected. The user can put a ad hoc key in place, start sending beacons with the ADHOC MAC hash, and at the conclusion of a meeting, disable the ADHOC interface. We can also generate such a key for face to face meetings and distribute it (you can assume the distribution mechanism is out of band BUT in point of fact we have a simple user interface that feeds a passphrase to a MAC algorithm and generates such a key. This

is a mere feature). This mechanism may be of use in disconnected operational situations.

The "ask" buttons in the figure should be ignored. They are going to disappear.

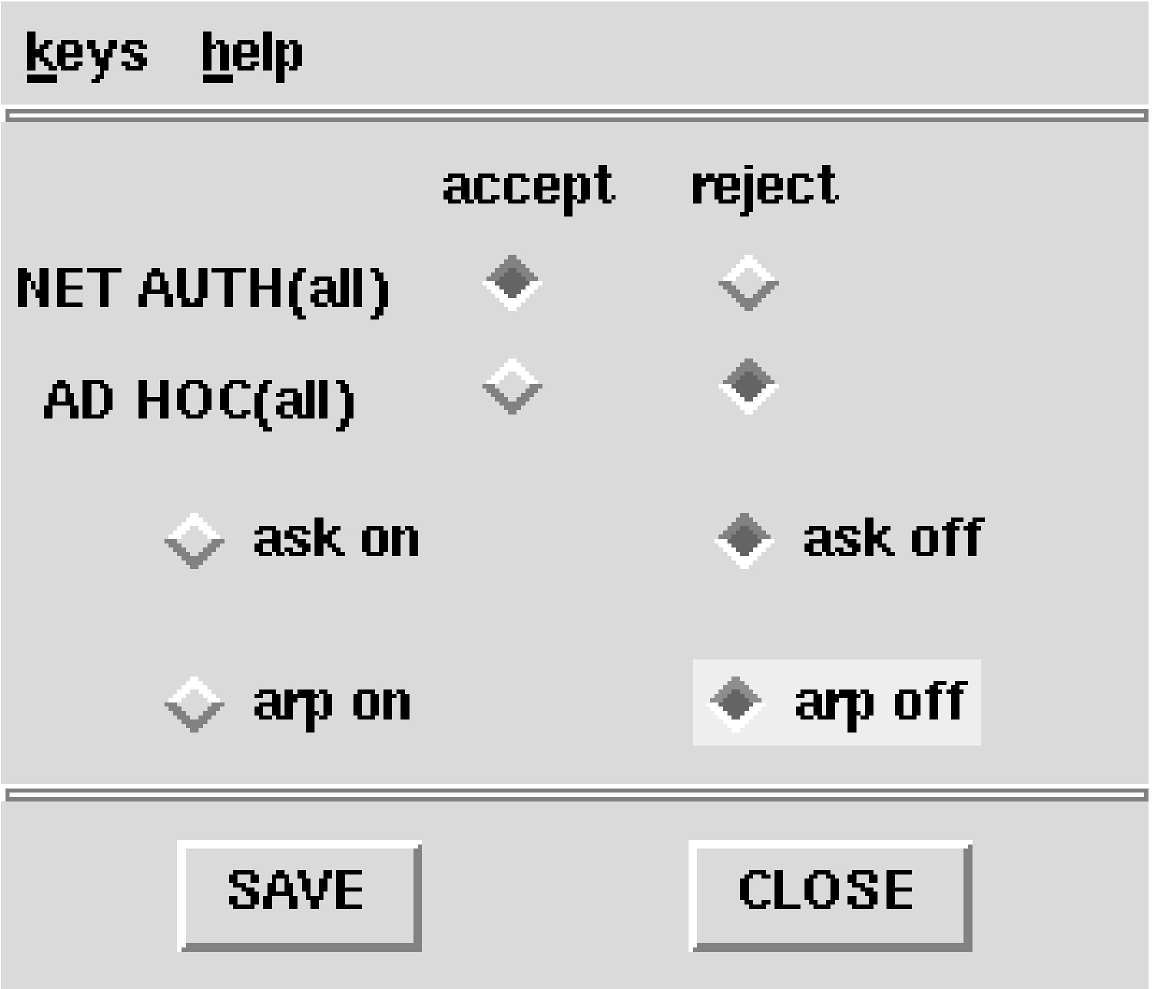

Figure 2: Mobile-IP Link Authorization

Mobile IP and IPSEC

We are going to be brief here as this will constitute the bulk of our work next quarter (fall) and our fall report will cover our work in more detail. We have ported the NRL (June) netbsd IPSEC code into freebsd and are testing it as given. As of this date, we know that the key socket works, and that we have not damaged normal non-IPSEC operation. The most difficult part of the port was that the IPSEC code is not cleanly differentiated from the IPV6 code, and we have no interest in the latter and have not compiled it into the kernel. (For example, the mods for IPV6 would also modify IPv4 code. We want to minimize IPv4 changes since it would introduce unnecessary risk and force us to completely retest the IPv4 stack). We are testing the AH-transport and ESP-tranport socket-level mechanisms.

We have decided what we want to do to support routing and how that will be combined with Mobile-IP. The work to teach the kernel to bind routes to IPSEC is now under way. We include some discussion of high-level assumptions here.

As we outlined in previous reports, we seek to support a number of Mobile-IP toplogies and these toplogies will change dynamically as a system moves from HA to FA. We will need a few extensions to the Mobile-IP UDP registration messages. Simply put, the MN will tell the HA (or a FA with which is shares a out-of-band a priori key relationship) what security it desires, e.g., none, AH, or ESP (the default). (We are assuming that NRL will eventually supply the IPSEC community with an up to date ESP that includes authentication. For now, we will use what we have been given, which is ESP according to RFC 1827). Our user interface will inform the user as to which policy choices have been made and allow a certain amount of dynamic switching. Basic topologies of interest include:

1. MN to default router (HA or FA with which a priori key relationship exists).

2. MN to HA when away (to HA and back across untrusted or any FA).

3. MN to MN (for ad hoc situations).

All route level bindings will assume an outer IPSEC security tunnel. Spi parameters, and ipsec src and dst will be supplied to the route mechanism and used to determine the a priori loaded key material in the NRL key table. It appears that we can make fairly straightforward use of the NRL key socket and kernel lookup mechanisms.

A very important general question here is: “how does IPSEC interface with the os”? A more focused question along those lines is: “how do we bend IP policy on output to combine route-based IPSEC with NRL socket-based IPSEC?”.

Very roughly we intend to change ip_output as follows:

```
if (!forwarding and there is a socket ipsec binding)
        do ESP transport
        do AH transport
else
        if there is an IPSEC binding to a route
                do ESP (with the ipsec tunnel)
                do AH (with the ipsec tunnel).
```

The above is an over simplification. The first part of the if clause roughly represents what NRL does now, and the second else part represents what we will add.

The upshot of this is that socket-level IPSEC bindings which are end to end will override any route level bindings which are probably not end to end. Socket-level bindings will not be associated with tunnels. All route bindings will have a tunnel slapped on the end according to the route in question (and decided by a route daemon). For example, for a MN at home, it could choose to bind AH or ESP flags to its default route, and the packet coming out of it would have a structure as follows:

```
ipsec tunnel ip header | ESP mn/ha | ip original | transport
ip src = MN            |           |             |
ip dst = HA            |           |             |
```

We suspect we should also implement a setsockopt that would allow a socket to proclaim that it desired NO ipsec route binding be applied to it. This would allow applications like slogin which uses TCP/IDEA to avoid having a DES outer binding applied. The question of exactly how should IPSEC bind to a kernel is very interesting and the above represents an incremental improvement but is by no means a final answer. We do not state that it would be wrong to have both AH/ESP due to routes piled on top of AH/ESP due to socket, but we do not want to implement it or test it. Possibly IPSEC mechanisms in the kernel should be tied to firewall mechanisms as well and/or to ports in a general sense. There is room for further research here.

Redundancy

In this section, we are going to briefly discuss some preliminary redundancy work we did this summer (including one curious and bizarre radio link situation we ran into), and also present a few principles of how we think HA and FA redundancy should work. Our thinking along those lines has crystalized over the summer.

Of course, the authenticated ad hoc beaconing mechanism proposed and implemented above is at heart a routing redundancy mechanism. We choose to not use subnetting and ARP. MNs can talk to each other without routers present. Certain (bizarre) problems with Mobile-IP and ARP that are primarily due to subnet assumptions built into routing tables are cleanly solved as well. At some point, we would like to build a higher-level routing protocol that would use signal strength as a metric, and would allow > 1 hop host to host routing based on an IPSEC tunnel (say ESP). We need to consider how such a mechanism would both interface with our link-layer routing and interface with the kernel routing table.

We enhanced our Mobile-IP handoff algorithm so that in the case of the Wavelan interface only, instead of basing handoff on agent starvation, handoff is now based on best signal strength with a bit of hysteresis. The PCMCIA Wavelan driver keeps a signal strength cache indexed by MAC address. Each cache entry consists of (MAC, ip, various Wavelan specific parameters including raw signal strength). The cache is only updated when a broadcast or multicast IP packet is received, thus only beacons update the cache. The mobile node daemon can read this information and it sorts its possible list of agents by signal strength. The algorithm is more complex than we will present here, but in summary it can be said to consist of a running sum of N samples where N is set to the last 10..15 beacons. Thus our handoff is not fast, but it is fast enough for current uses. We can walk around our square building with agents in the corners and watch handoff occur cleanly as we walk by. More important, if a FA (or the HA's Wavelan interface) goes down, an agent switch does not take long. Thus we can cleanly overlap agent cells and deal with the loss of an agent's Wavelan interface. This list will also be useful later on for FA redundancy work since we will choose to use the top N (likely 2) agents for simultaneous registration.

We also implemented a redundancy modification to the UDP registration cycle. When an MN attempts to send a registration packet to the HA, it enters a registration cycle that consists of N retries, with a backoff mechanism built in. When we are in the presence of multiple agents (FAs), we short-circuit the registration retries (say halfway). Thus if registration via one FA does not seem to be successful, we try another FA as long as its signal strength is above the low-water signal strength mark (10%). The FA currently deemed "useless" will be retried later on (a few minutes) since its signal strength is higher. This allows us to attempt to use a possibly different end to end (MN to HA) route and will effectively deal with a nearby and short-term routing partition.

When we began to test the improved handoff algorithm, we found a very bizarre short-term situation that would manifest itself in our building with the 3 overlapping agent cells (2 FAs, 1 HA). For a period of time, typically 5-15 seconds, agent signal strengths would mysteriously rise and fall so that even for static MNs, an agent within a yard could lose its signal strength, and a normally weak agent at the other end of the building would appear stronger. There are only two possibilities at this point: 1. either we have a bug in the Wavelan driver in terms of how the information is cached (but we doubt it)., or 2. this is a real phenomenon (not reported elsewhere as far as we know) and might be due to multipath or some as yet unknown radio interference. We adjusted our hysteresis mechanism to compensate for this.

We believe that our current work has nicely aimed us towards HA and FA redundancy mechanisms. We already have some capabilities in the latter direction since we can switch between available overlapping FAs now, if one should go away. Our ad hoc routing mechanism will also help us deal with certain partitioning problems inherent in subnetting. This will actually have applicability to HA redundancy since if we have two HAs on different IP networks (as we desire), the HA subnet itself may be partitioned (physically split to two places) and that is toplogical case we feel we can now deal with.

We intend to modify the fundamental BSD radix routing mechanism so that a gateway can logically be list of gateways; i.e., one destination can have multiple gateways. In point of fact, we will probably limit this to two. This is a profound change. The ip output function would take a packet and cycle through the list of gateways sending a packet to each gateway in turn. Of course, this means we are trading redundancy for bandwidth. This mechanism will be used both at HAs and MNs since a HA might have two COAs for two different FAs and an MN might have two different FAs associated with its default route. An interesting variation on the latter idea would be to bind a multicast address to the default route and thus multicast default route packets to agents.

We believe that HA redundancy should have the following attributes:

1. at least two HAs should be supported and they should ideally be on different subnets so that the failure of one router does not eliminate both HAs. This can be done with conventional unicast IP routing. However it means that any packet from a peer host to an MN might go to either HA at any time according to unicast IP routing. It should also be possible if desired to put both HAs on the same link.

2. We believe the HAs should have a protocol (say based on TCP so that the messages are reliable) so that they can keep each other informed about MN location. They need to tell each other about MNs at HOME or AWAY. If an MN is at HOME, the peer HA must be told so that it can setup a tunnel with the other HA as the COA. Note that there are at least 2 HOMEs. If an MN is AWAY, both HAs must setup tunnel routes with the FA/s in question as COAs since packets may be header towards either HA at any time.

3. The MN will have a list of HAs, and will not try to use them both at the same time, but will merely switch from one to another when it deems that HA #1 is no longer available; i.e., UDP replies do not show up during one complete registration cycle. We choose to have the HAs communicate MN state between themselves as opposed to having the MN keep both HAs up to date as this would minimize potential cross-internal traffic. In general HAs will be within an enterprise.

We hope to begin this work when we have completed the IPSEC/Mobile-IP work.

Visit to BBN

John McHugh and Jim Binkley on Sept 27 1996 made a one day visit to BBN in Cambridge in order to compare project directions with John Zao of BBN. We met with John Zao, Joshua Gehm, and Steve Kent. Jim Binkley and John Zao made overviews about their respective Mobile projects. BBN is focused on key management issues, i.e., how to tie certificates and session keys into the Mobile-IP protocol, e.g., between a Mobile Node and its HA. They are also interested in the notion of hierarchical Foreign Agents and how certificate exchange between Mobile Node and

Home Agent might be minimized. PSU is interested in fundamental integration of IPSEC and Mobile-IP, and redundancy and survivability issues including ad hoc routing, toleration of multiple agents and the elimination of ARP spoofing. The one point in common we have is the desire to use NRL's IPSEC code. In that regard, we (PSU) are currently ahead of BBN as they have apparently focused on their certificate work first. We would be happy to share any work with them along these lines. We, in turn, will adopt BBN results in the areas of certificate management and Fortezza integration.

Plans for Fall Quarter

In brief, our plans are as follows:

1. the primary focus is the design, implementation, and testing of binding our Mobile-IP to NRL's IPSEC. We need to make o.s. changes to FreeBSD so that routes can be bound to their key table, change our Mobile-IP daemons to install those routes, etc.

2. write a test plan and test the authenticated ad hoc mechanism, and also get the X user interface policy windows fleshed out and real so that it actually controls the MN's behavior.

3. commence formal modeling of the implemented security protocols.

4. develop a cumulative acronym list for the quarterly reports as requested by RADC.

5. begin integration of the Fortezza cards with the system.

6. occupy new laboratory space.

In general, we hope to get our fundamental security work done this quarter, and begin HA/FA (and further ad hoc) redundancy efforts next quarter. At some point, possibly next quarter we should make various measurements of IPSEC transforms over Wavelan links.

Within the next few weeks, we hope to occupy laboratory and temporary office space in a building recently acquired by PSU. The building itself will become a major facility in our experimental plan as it's layout will accommodate a variety of mobile-ip configurations.