Secure Mobile Networking
Fifth Quarterly Report

James R. Binkley and John McHugh
Portland State University

14 January 1997

Project Status Overview

Overall we feel we have made reasonable progress this quarter.

The two most important items accomplished were deployment of the ad hoc sub-system, and first cut work on IPSEC/ESP bound to routes for Mobile-IP.

We have finished debugging, and deployed our ad hoc link-layer replacement for ARP on our Mobile wireless infrastructure at PSU. All users and almost all agents (barring one left behind for mode testing between ARP and ad hoc) are running the authenticated link-layer ad hoc system.

The IPSEC work we intended to do turned out to be too large of a bite for one quarter, and as a result we divided the project up into two parts. In the first part, which is almost complete, we have constructed a kernel route binding for IPSEC/ESP (and partially for IPSEC/AH). We wanted to make a smaller pass over a first project, so that we could gain some assurance regarding "proof of concept". The route binding has involved much kernel routing work. At this point we have statically tested the Mobile-IP topologies that we described in the last report. We are currently in the middle of modifying our Mobile-IP routing daemons (applications) to take advantage of the new kernel mechanisms. We should have a simple form of "Mobile-IP meets IPSEC" under test in a few weeks. In the second part, which we will complete this quarter, we intend to finish our kernel IPSEC/route mechanism and carry that work to our Mobile-IP daemons.

Accomplishments for Fall 1996

1. Jim Binkley has written a paper on the ad hoc protocol entitled: "Authenticated Ad Hoc Routing at the Link Layer for Mobile Systems". The paper is currently a PSU Computer Science technical report, TR 96-3. It has been submitted to a special ACM journal on mobility for possible publication.

2. We have debugged and deployed the ad hoc routing protocol. We hope to release this code soon (probably in January).

3. We have created a kernel route binding to IPSEC, including a route API in the `route(8)` and `arp(8)` commands. We have extensively tested the NRL IPSEC

ipv4/socket port into FreeBSD and feel that it is sound at this point. We are in the process of teaching our MIP (Mobile-IP) daemons how to use IPSEC/routing. We will discuss this work in greater detail below.

4. David Reeder has attended Mobicom/IETF meetings and has kept the group here up to date on events at those meetings.

5. Over the course of the quarter, we have made several internal improvements to our Mobile-IP source base. These improvements were either due to bug fixes, IPSEC work, internal security improvements, or as an attempt to make the system more portable.

6. In spite of the fact that growing our Mobile-IP/wireless infrastructure is NOT a current focus, we still have deployed three more agents. We have deployed two Foreign Agents, one in the PCAT central engineering building and one at the Oregon Graduate Institute. The latter has led to some interesting "cross-routing-domain" experiments, which we will discuss briefly later in the report. We intend to put a new Foreign Agent in another engineering building next quarter (U.S. West). We also finally deployed a second "test" Home Agent, which has proved invaluable in the final stages of ad hoc testing, and is currently being used for IPSEC testing.

7. We have made some progress with redundancy theory. This work is primarily being driven by graduate students. We are in the process of completing a specification for multi-hop ad hoc routing. We have started specification work on a redundant Home Agent system (two HAs per Mobile Node subnet). We expect this work to bear fruit either by the end of winter or spring quarters.

8. Jon Inouye, a Ph.D candidate at Oregon Graduate Institute has ported our Mobile-IP into his link-layer redundancy system. Jon's Ph.D work is focused on teaching a Mobile Node how to retain network connectivity when link-layer devices are switched. For example, one might start with WaveLAN (which is now available at OGI), and switch to a PCMCIA Ethernet device. Jon has made available to us his kernel PCMCIA infrastructure work, which gives us hope of eventually being able to use a second simultaneous PCMCIA card, along with the WaveLAN card, which is a permanent fixture in our laptops.

Winter 97 Plan

In this section, we are going to briefly discuss our winter term goals and planning. Our planning is primarily personnel based; that is, person X works on project Y. We will present current people and the projects they are associated with.

Winter goals

In summary, we will make a couple of code releases, and continue our current focus on IPSEC/routing this quarter. We also are turning more to our redundancy work as two graduate students are making good progress in that area.

1. A Mobile-IP release. We intend to release our ad hoc version of Mobile-IP soon. We need to finish some release engineering and write some final documentation. (There is no IPSEC in this release). This release will replace our Mobile-IP release of last summer.

   Major features made available in the release include:

   (a) improved FreeBSD kernel features for Mobile-IP. These are all enhancements to the BSD routing mechanisms.
      i. use of Bill Fenner's RECVIF function, which allows routing daemons to determine the interface a packet came in on.
      ii. a revamped tunnel driver (discussed more below, however there are no IPSEC features in this version).
      iii. we fixed and revamped the mechanism our routing daemons use to tell the kernel how to bind a route to an interface. All of the above changes were useful and were almost all bug fixes as well as being part of an attempt "to get it right".

   (b) of course, the authenticated ad hoc routing mechanism is included.

   (c) We implemented per MIP entity MIP authentication. For example, all of our Mobile Nodes can now use a per MN Home Agent key. This was an important security improvement.

   (d) The release will include the fast and redundant handoff work we did last summer.

   (e) xmnstat, our visual X/tcl-based interface for the Mobile Node daemon will be included.

2. ipsec kernel. We want to release our current IPSEC/FreeBSD 2.1 kernel to the FreeBSD security community. A number of people including engineers at TIS and firewall companies have expressed interest. This release will ONLY include the kernel and various applications, including the `PF_KEY` `key(8)` utility, test applications developed by us, and our first cut ESP/route binding. It will also of course include the original NRL IPSEC/transport mechanisms including the crypto algorithms of last summer. For example, AH includes md5, hmac-md5, sha, hmac-sha, and "dummy" (the NULL algorithm). It will not include Mobile-IP daemons, since that work is not complete. The kernel will however contain our Mobile-IP routing modifications. This is important as our IPIP tunnel driver, originally developed for Mobile-IP has become a central part of our IPSEC work. We would have liked to have gotten this our earlier, but the ITAR-specific code has been held up in transit elsewhere. It would be extremely nice to have other FreeBSD people assisting us in this work.

3. We intend to finish our first cut IPSEC/Mobile-IP route binding that uses only ESP/DES in various routing topologies. We have tested transport IPSEC for all the AH and ESP transforms. We have implemented a first cut routing API. We are in the process of taking advantage of our current kernel mechanism dynamically in our Mobile-IP daemons. We intend to finish the routing API (we know what we want at this point) and carry the new mechanism (primarily AH) into our Mobile-IP daemons. We should be able to finish this work this quarter. We also hope to do a

bit of tunnel testing, since we think that AH may be useful in tunnels between Home Agents/Foreign Agents/and Mobile Nodes.

4. We will finish our multi-hop ad hoc specification and redundant Home Agent specification. We may be able to get the HA mechanism implemented and under test, although we will need to do some work to setup a reasonable testbed for redundant HA testing. It is likely that multi-hop ad hoc coding and test will stretch into spring quarter as that work is both theoretically complex and difficult to test.

Optional goals

1. as a group we hope to write a technical report on our Mobile-IP/routing security work. This paper will focus on the route/IPSEC binding.

2. we intend to do a little more investigation of CERT/firewall/Mobile-IP ingress filtering problems. We can do this because of the Foreign Agent we have deployed at OGI. OGI is outside of the NorthWestNet border router. As a result, PSU-based IP subnet addresses are filtered out when packets from a PSU laptop at OGI try to cross into the PSU domain. (The irony level here is high. When at OGI, we can talk to any Internet system except our own).

Personnel

Jim Binkley acts as project manager. He supervises David Reeder, and the graduate students. Jim is usually involved in design, code release, review, specification review, and kernel work.

John McHugh has recently resigned as Computer Science Department chair which will allow him to focus more on DARPA research. John will eventually drive Fortezza work in this project (See the Fortezza note below).

David Reeder is our full-time employee. David worked last quarter on IPSEC kernel routing issues and is currently completing the Mobile-IP application/daemon work in that area. In winter quarter, David will finish the IPSEC/routing API, rework the routing daemons in IPSEC phase #2. When this work is complete, David will begin work on ISAKMP and how it might be tied into Mobile-IP.

Xu Hao is a M.S. graduate student who will graduate in winter quarter. Last summer he worked on the first cut ad hoc protocol, and during fall (and winter), he has been working on the specification of our multi-hop ad hoc protocol.

Bjorn Chambless is a M.S. graduate student who has just started his M.S. work as of last summer. He has completed xmnstat for us and has recently started work on Home Agent redundancy.

Jennie Ye is a new M.S. student and has just joined the project this quarter. She will probably spend most of this quarter coming up to speed. We intend to deploy Jennie to work on redundancy as soon as we can. We need someone to carry on Xu Hao's work and also to begin work on Foreign Agent redundancy.

Mathew Mead is currently a Teaching Assistant, but we intend to hire him in spring quarter to replace Xu Hao. Both Mathew and Jennie are currently taking a reading class from Jim Binkley that focuses on TCP/IP, network security, and Mobile-IP. We intend to have Mathew work under John McHugh's direction in spring on Fortezza.

Fortezza-note

Due to lack of time, people, and a kernel infrastructure stumbling block, we have not yet been able to do anything with Fortezza cards. Of course, we have just gotten an IPSEC infrastructure in place and tested in the last two quarters. Presumably Fortezza would somehow fit into that infrastructure. A major kernel infrastructure problem has been that our current FreeBSD 2.1 kernel does not support more than one PCMCIA driver at a time. By definition, to use Fortezza, we feel we must use our WaveLAN PCMCIA cards simultaneously. We originally intended to wait for FreeBSD 2.2 in the hopes that it might contain a general PCMCIA kernel infrastructure. However, due to Jon Inouye's recent work, we now have a source of 2.1 kernel code that includes our Mobile-IP routing infrastructure and includes a reasonable PCMCIA infrastructure. We hope to be able to "digest" that code into our current source base over winter quarter. If we cannot, it still gives Mathew Mead and John McHugh a starting base in spring. It would seem that our OGI association has born fruit.

IPSEC && Mobile-IP

In this section, we are going to discuss our major focus of the quarter, in four sub-sections:

1. IPSEC-Mobile-IP architecture.
2. The MVIF tunnel driver.
3. OGI/Firewall experimentation.
4. Conclusions and Observations.

This work is still under progress and will continue into this quarter and possibly beyond (if we include ISAKMP. We expect to talk to the DARPA project manager about ISAKMP when she visits us in January).

We decided to break the IPSEC work up into two phases in order to test our most basic routing notions as soon as possible. The first phase consists of a complete Mobile-IP implementation that would dynamically use IPSEC-based ESP/DES routes only. The Mobile-IP focus in the first phase is primarily on routing topologies ONLY between a Mobile Node and a Home Agent. The second phase would finish the routing work and include AH, AH/ESP in combination, possibly a statically deployed FA/MN relationship (we are not sure yet if that is really useful).

We can divide the first-phase project into the following phases:

1. initial port of NRL/IPSEC to FreeBSD 2.1 (done last summer by Jim Binkley).
2. development of test code for testing (and understanding) the NRL transport-based IPSEC code. (done this quarter by a graduate student and Jim Binkley).

3. testing of NRL/transport with all algorithms AH and ESP (done this quarter).
4. definition and implementation of a route/based API for the BSD kernel for binding a NRL security association (SA) in the kernel to a BSD route table entry. (started this quarter by David Reeder/Jim Binkley. We will finish this early in the winter quarter).
5. static testing of route API. (David Reeder. done).
6. modification of MIP tunnel driver to support IPSEC/routing. Jim Binkley. (done)
7. static testing and debug of various (3) route topologies of interest to Mobile-IP by Jim Binkley (done).
8. modifications of Mobile-IP daemon applications to dynamically use IPSEC/routing (primarily HA/MN away or at home). David is currently doing this work.
9. modification of xmnstat to support simply IPSEC policy choices. To be done winter quarter by Bjorn Chambliss.
10. internal deployment and test.

In our second phase, we will finish the kernel/IPSEC routing work, and teach our Mobile-IP implementation how to do AH and AH/ESP simultaneously. We also need to decide what if anything to do about the FA/MN relationship. We are of mixed minds about that problem. Something useful can probably be done with static keys and FAs that belong to a given organization's routing domain. In particular, an authenticated tunnel between Home Agents and Foreign Agents may be interesting. It may be useful for a given administrative domain to setup all of its tunnels with authentication (in order to protect itself from the outside world). On the other hand, interaction between a Mobile Node and another domain's Foreign Agent in terms of setting up secure routes requires dynamic negotiation of authentication via certificates. An interior domain might of course with to use certificates for authentication too. We wish to point out that authentication should be considered not only for the current MIP relationships but also for tunnel endpoints.

IPSEC-Mobile-IP architecture

In this section, we will try to briefly explain the gist of our Mobile-IP/IPSEC architecture. The goals for the architecture may be summarized as follows:

1. route-based IPSEC. Our Mobile-IP implementation will bind IPSEC/ESP/DES (and eventually AH with various transforms) to routes; i.e., the security is done with tunnels at the network layer between entities such as Mobile Nodes and Home Agents.
2. We are for now only using static key distribution. For example, we assume that both Mobile-IP and IPSEC secret keys are installed at an MN and a HA before the MN "leaves home".

3. Our Mobile Node daemon application dynamically requests, and sets up secure (or insecure) routes in a number of routing topologies according to Mobile-IP topological information (secure enclaves). All routes are always two-way in terms of security. The MN to HA route and HA to MN route either both use IPSEC (or do not).

4. A MN user may dynamically or statically specify IPSEC security in terms of a simple policy interface. As a consequence, routes are installed dynamically and bound to pre-installed key material. This is done with the xmnstat controller (and may be done by rebooting mnd and changing its boot-time config file as well). For example, in our first phase, a user will be able to specify two modes of security, "none" or "esp/des" to a HA, either at home (link connection to HA as default router) or "away" (two-way security tunnels between the MN and HA over the FA without the FA knowing about it). In summary, the MN (and hence the user) will control whether ESP tunnels or no tunnels are setup to the HA.

BSD(4.4) IPSEC/routing API

It is probably easiest to simply present a routing API in terms of the switches/flags we currently use with the `route(8)` command. We have 4 possible IPSEC tuples that may ALL be used together in a few rare cases or more likely used as ones or twos in normal routing cases:

1. ESP on a normal route
   ```
   # route add <destination> <gateway> -esp -spi N -itsrc <ip src> -itdst <ip dst>
   ```
2. AH on a normal route
   ```
   # route add <destination> <gateway> -ah -spi N -itsrc <ip src> -itdst <ip dst>
   ```
3. ESP at the front of a tunnel route packet (special case)
   ```
   # route add -tunnel <destination> <COA> -esptunnel -spi N -itsrc <ip src> -itdst
   <ip dst>
   ```
4. AH at the front of a tunnel route packet (special case)
   ```
   # route add -tunnel <destination> <COA> -ahtunnel -spi N -itsrc <ip src> -itdst
   <ip dst>
   ```

The first two route bindings may be used with most normal routes. They are of generic interest in terms of routes and Virtual Private Networks. The latter two are only for Mobile-IP and apply to the virtual tunnel device that is typically used at the HA to tunnel to the FA.

We assume that the SA is loaded via `key(8)` and it specifies a Security Association in terms of (ah — esp, spi, IP src address and IP dst address). The security associations have been loaded via the `PF_KEY` socket by `key(8)`.

Note that each route type above specifies a 4-tuple; for example, (esp, spi, ip src, ip dst). When a route is "added", the routing code makes sure that it can find the SA and binds itself to it in the kernel. The 4-tuple is used as an SA signature. In general, the order of operations for routes and keys is:

1. add a key via PF_KEY and `key(8)`
2. add a route and bind to the key
3. delete the route
4. delete the key

On output, the IP layer simply uses the SA route pointer to do the IPSEC transformation.

It is possible that a given route might have more than one of the bindings associated with it. E.g., a given route might have both ESP and AH bound to it. In that case, AH would be performed first in ip output before ESP.

The `-itsrc` and `-itdst` switches are of special interest. The idea is relatively simple. These addresses must match the ip src and ip dst addresses specified in the /etc/keys security association. They represent an IP tunnel from this host to the destination. We assume that all routes are IP tunnels and that this relationship always holds. The semantics here are very interesting as routes become essentially double jointed with 4 IP hosts involved. The itdst and gateway may or may not be co-located. If not co-located, the tunnel endpoint obviously goes over the routing destination, which will simply forward the packets to the itdst router and is not involved in the security association. We might draw the following picture:

```
host(itsrc) ----> gateway ----> itdst ---> destination
     =========================> (security tunnel)
```

Routing is by definition one-way. Packets are forwarded for the destination to the next hop gateway. The security tunnel however is between the itsrc and itdst. On output, IP will (for example) take the IP datagram that reached it and append an IP — ESP header pair. So if we have say a UDP packet, we would have the following output for route binding #1:

IP outer — ESP — IP inner — UDP — data payload

We call this an "ipsec tunnel". It may take the following forms:

1. IP — AH — IP,
2. IP — ESP — IP, or
3. IP — AH — ESP — IP.

The `-ipsrc` and `-ipdst` values are used in the IP outer header as ip src and ip dst. ESP's next protocol is 4; i.e., IPIP. On arrival at the ipsec tunnel destination, the packet is fed first to ESP and if decryption works, it is then handed to our mvif tunnel driver for IPIP processing. ESP essentially removes itself from the header and hands the packet to the IPIP mvif driver. The mvif driver removes the outer IP header and forwards the inner datagram to `ip_output`; i.e., it forwards (routes) the packet.

The first two kinds of bindings MAY be bound to tunnel routes as well. For example, we might use binding #1 with ESP on a tunnel route (at a HA to a FA),

```
# route add -tunnel <destination> <gateway> -esp -spi X -itsrc <S> -itdst <D>
```

This route allows the HA to "ipsec tunnel" packets to the MN; i.e., send packets using DES to the MN where the security association is only between the HA and MN. This is probably the most important routing mechanism trick for Mobile-IP.

On the other hand, there might be a security association at the HA in terms of mechanism between it and the FA that the MN currently is visiting. And we might also have the HA/MN security association in place as well. In this case, the latter two route bindings (#3, and #4) could be applied as variations that ONLY apply to MIP tunnel routes (routes bound to mvif). As a result, one might produce the following example packet:

```
ip outer        |  AH     | ip inner      | ESP     | real IP datagram
ip src = HA     |  HA/FA  | ip src = HA   | HA/MN   |
ip dst = FA     |         | ip dst = MN   |         |
```

Assume we have bound both `-ahtunnel` (#4) and `-esp` (#1) to a tunnel route. The ip destination for the tunnel route is the MN. The ip gateway is the FA's "Care-Of Address". On output, first ip output would append an "ipsec tunnel" between the HA and MN. This would prepend the "ip inner" and ESP portions. Then the tunnel driver itself would note that the `-ahtunnel` association existed and instead of performing its normal operation (appending just an IPIP outer IP header), it would append the IP header AND perform an AH operation over the packet. In this case, all four bindings could be performed on one packet. If this router was also an end system, 6 IPSEC headers could be inserted if we add in two transport headers.

BSD route types and the IPSEC route binding

It should be noted that routes in a routing table perform different functions and their semantics are complex. Our Mobile Nodes typically set their default routes to an agent, HA or FA. They must also set a "link-layer" or ARP route to point to the agent as well. (In 4.4 BSD, the arp table has been merged with the traditional routing table, and "arp" routes simply have MAC addresses instead of IP addresses for the gateway portion). Our Home Agents set link-layer routes to MNs and tunnel routes for MNs that are "AWAY" at Foreign Agents. A Foreign Agent will set a link-layer route for a MN that is resident on a given FA link. In addition to default routes, and link-layer routes, there are other kinds of routes. We can distinguish at least the following kinds of routes in the BSD routing table:

1. default routes. The default route has a cloning feature and destination addresses that match it will cause the insertion of a specific host route that will time out if not used for 20 minutes.

2. link-layer routes (flag=LLINFO, gateway is MAC address)

3. per interface clone routes (typically used with ARP and closely tied to IP subnet semantics).

4. host routes (destination is a specific host) and the gateway is a next hop router.

5. net routes (destination is a specific network).

6. tunnel routes (new from us via the mvif virtual tunnel device - the tunnel driver simply appends an IP header and hands the packet to `ip_output` to be routed again).

7. multicast clone route - the multicast clone route is associated with a given interface. An attempt to multicast sans interface information, will cause a specific multicast route to appear in the routing table.

It should be pointed out that we have also modified the `arp(8)` command so that ESP and AH bindings may be associated with LLINFO (link-layer) routes. `arp(8)` and `route(8)` both use the BSD `route(4)` interface internally and are not dissimilar. We will not document the arp switches here at the present time. However this gives us needed mechanism so that a HA can "tunnel" with an ESP/route binding to a MN, when the MN is at home.

Our route bindings may potentially be applied to all of the above routing types. The semantics of the above set of "route types" are complex and we will not claim yet that all of the interactions between them and our bindings are clear or understood.

The "ipsec tunnel" when applied to the default route causes IPSEC security to be inserted, but does not apply to the default route gateway itself UNLESS of course the itdst is that of the gateway. In that case, the gateway's link-layer route must also have the same SA applied to it. Thus it is possible to tunnel over the next hop router or to tunnel directly to it.

One of the routing flags possible is called a CLONE flag (actually two, but never mind). Typically the cloning flag causes a more specific host route to be dynamically inserted in the routing table. E.g., a small 'little c' clone flag is always associated with the default route. If a specific ip destination, matches the default route, a host route is inserted in the routing table with a timeout. (These may be seen with `netstat -ran`. The clone routes will have a W flag associated with them.) We puzzled over what to do about clone routes in that particular case and decided the only correct response was for a clone to inherit the SA. If a user decides to use ESP to tunnel over a FA (default) to a HA. The user would want specific clones to carry over that attribute. There are other cases where cloning is used however, and in those cases we are not sure about proper behavior.

There are still however a number of semantic TBDs (To Be Determineds) that are lurking here. E.g.,

1. multicast routes. We have not had time to even play with these yet or think about it much.

2. per interface clone routes as used with ARP. In our link-layer ad hoc mode, we remove them. But a IPSEC binding might be used for some sort of defensive measure here.

3. cloning caused by IP redirects. A security association between one router cannot be blindly inherited by another router. We should make sure that SAs are not inherited by dynamic redirects. (More fundamentally, we need to have a per interface way to turn dynamic redirects off).

The semantics associated with the kernel routing table are NOT simple. We are still feeling our way here.

Static Routing Topologies

We have tested three different MIP-related routing topologies "by hand" at this point using primarily the ESP/binding. The topologies are:

1. MN to HA at home. The HA is the MN's default router. Both the default route and link-layer MN/HA routes at the MN point to the HA (gateway==itdst). The HA sets up a link-layer ESP tunnel to the MN.

2. MN to HA over FA. Since we assume tunnels are two-way, we of course setup a tunnel from the MN to the HA and from the HA to the MN. We will look at these tunnel types more below as they are extremely curious. Note however that the FA is not part of the equation. We tunnel "over" it. The MN sets the FA as its gateway and sets up a link-layer route to the FA as well. The itdst for the default route is set to the HA. The HA binds a `-esp` binding to the tunnel route with the `-itdst` set to the MN. Thus the routes are complimentary.

3. MN to MN. Given that we have a link-layer routing authentication protocol (ad hoc), it is reasonable to try and automatically setup secure routes between two MNs on the same link. Both MNs must insert link-layer routes to each other that for example, might be bound to ESP.

The most curious trick here is the MN to HA over FA topology. The MN to HA route is not so strange. We can simply set the gateway to the FA, and the itdst to the HA. Packets emitted from the MN to the default route will have an outer IP header where the ip dst is set to the HA, and ESP would be the next header.

As we have pointed out before, however the HA to MN packet is very interesting. If we assume no security association between the HA and FA, we have:

```
ip outer: | ip inner:      | ESP    |  ip datagram
ip src = HA              | ip src = HA   | HA/MN  |
ip dst = FA (COA)        | ip dst = MN   |        |
```

The packet has three ip headers in it. We might call it a "double tunnel". The innermost header is the original ip datagram, where the src may be any system. The destination of course, is the MN. Since we associated `-esp` with the tunnel route itself, ip output will append the "ip inner" header and the ESP header first to the original datagram. Here the `-itdst` == MN and `-itsrc` is the HA. That gives us the second IP header. The third one is appended naturally by the tunnel driver itself as a natural consequence of its operation. Please note again that it presents another opportunity for insertion of IPSEC as we could bind IPSEC between the HA and FA.

When the packet arrives at the FA, the outer IP header will be stripped and the inner part will be delivered to the MN (since the inner IP header has the MN ip address for its dst). When the packet arrives at the MN, it will process the inner IP and ESP headers. If we assume that the ESP portion is successfully decrypted, the mvif driver will deliver the packet to `ip_output` for forwarding, but since the real ip datagram is sent to the MN, the packet will be delivered (finally) to the MN, itself. Thus we are able to tunnel "over" the FA.

Tunnel Driver

Our MIP tunnel driver, called the "mvif" device, is integral to our IPSEC/MIP implementation. To paraphrase Churchill, it is "an enigma wrapped up in a mobius strip" and its operations may not be easy to understand.

Sans IPSEC, we may describe it as a virtual pseudo-device. A route is bound to it and the packet on input to the mvif driver, simply has an IP header prepended to it. The ip destination in the outer ip header is set to the gateway, which is typically the FA's ip address, or tunnel endpoint. That packet is then delivered to `ip_output` again for forwarding. (Note that we have an inner ip src address and an outer ip src address, since we have IPIP appended to the packet). On input at the tunnel endpoint, the mvif input function strips the outer IP header, and delivers the remaining IP datagram to `ip_output` for forwarding again. We typically use this for MN specific host routes between the HA and a FA where the MN is resident. The rule is that whatever mvif does, it always delivers the result to the ip output function so that the packet can be forwarded (routed) again. At a HA, on mvif output, a packet will typically be routed via the default route and head out to the FA via an Ethernet interface. We have described normal operation, but the reader should note that mvif tunnel routes may be used anywhere, including between a FA and another FA, or between a MN and agents at home.

The driver has 3 IP addresses of interest associated with it:

1. the normal IP address bound to an interface via ifconfig. We will call this the "ifconfig" address.

2. the inner ip header ip src address.

3. the outer ip header ip src address.

All 3 of these addresses are setable. In normal operation at a HA, we use a private IP address for the "ifconfig" address, as typically that address is not useful. We have modified the TCP/IP stack so that if it tries to choose the inner ip src and finds that the device is the mvif device, the driver itself will be asked what its IP src address should be. As a result, it can lie and will provide the setable #2, inner ip src. We typically set this to be the same as that of the Ethernet ip address on a HA. Thus the inner ip src is a "real" ip address. In normal HA operation, we typically do not set the outer ip address (which the driver itself can do), as the rest of the stack will set that according to the default route's real interface. The two inner IP addresses can be set via `ioctl(2)` calls to the driver.

Why so twisted? Because Mobile-IP itself is possibly a little strange since we are told that a Mobile Node might act as its own Foreign Agent if it can acquire a COA somehow (for example, via DHCP). We have some architectural flexibility because we can use the mvif device as a place to push the "original" MN ip address (home ip), since the COA will have to be bound to the real interface in use. This would allow the MN to at least recognize packets sent to its home ip address that arrive when it is abroad, and might enable the Mobile-IP MN as COA mode (although we have done nothing else to enable that aspect of Mobile-IP).

However the primary reason for this particular aspect of the mvif device is that it makes the driver very flexible in terms of border routers and firewalls that seek to toss packets arrived from abroad (exterior domain) where the ip src of the packet should be inside the firewall. Of course, this mechanism at a border router is an anti-spoofing function. We will talk about this aspect of the tunnel driver more below in a section on the MIP/Firewall problem.

In IPSEC terms, the mvif driver is playing the following functions:

1. it uses the NRL `ah_transport` and `esp_transport` functions and implements all output tunnel functions; i.e., it knows how to append a header and do AH/ESP. It is called at various times to do all four route bindings we have mentioned previously.

2. on input it is called after AH or ESP have gone off, as the next proto value for either of those headers is set to 4; i.e., IPIP.

Given that mvif is above AH/ESP on input, we have implemented two possible simple input filters. We have a filter for all input and one for IPSEC only. An administrator may choose to only allow mvif to be used for output, and any input will be discarded. This might be of use at a HA, where we do not expect any packets to be tunneled back. Or he/she may choose to allow only IPSEC processed packets; i.e., a packet must have successfully passed through AH or ESP before it reaches mvif and can be forwarded. Both switches may be set with `sysctl(8)`. This gives us some simple defenses against tunnel spoofing.

Firewalls and Tunnel Considerations

The Mobile-IP working group has recently discussed the problem of how Mobile-IP can work in terms of current CERT advisories that basically block external domain Mobile-IP connections. The CERT advisories in question are:

1. CA-95:01.IP.spoofing.

2. CA-96.21.`tcp_syn_flooding`.

The first basically calls for border routers to filter out IP packets trying to get into the local domain, based on an IP src that should belong to an interior domain. This is being called "ingress filtering".

The second expands on the first and calls for an additional filter on packets trying to leave a domain, where the packets have an ip src that is outside the domain. Logically it could be called "outgress filtering" (if that term makes any sense, but since when does routing terminology make sense?).

Filter #1 was probably deployed in 1995 on a number of border routers (routers that perform an Exterior Gateway Protocol such as BGP). It was apparently a response to external attacks on protocols that only use IP addresses as "authentication". For example, NFS, X, and the BSD rsh based utilities such as rsh, rlogin, and rcp. (We submit no one should ever use the term "IP address authentication" again, because it is an oxymoron as the authentication mechanism is so poor, it is a disgrace to the term "authentication").

The consequences of filter #1 for Mobile-IP are severe and ironical. If a Mobile-Node wanders out of its interior domain and tries to use a Foreign Agent found elsewhere, the following observations hold:

1. it will be able to register with the MIP UDP registration packets, because the FA acts as a proxy; i.e., the FA forwards the registration to the HA, hence the FA ip src address is used (and is legal). Hence the border router will not filter out the packet from the FA. Mobile-IP as a UDP-based routing protocol works.

2. the Mobile Node will be able to talk to any end systems that are not within its interior domain.

3. the Mobile Node will NOT be able to talk to any end systems that are within its interior domain. The most serious lack here is probably that it will lose its DNS service.

There is one simple way (barring the use of tunnels, which we will discuss later) out of this impasse. A remote FA could be considered as a "bastion host", and could provide proxy services for any visiting MNs via socks, proxy web servers or the like. There are trickier possibilities. One must observe that DNS should be used locally, and one probably should not be doing DNS over the Internet under these circumstances. There is much to be said for work here in general terms that would offer up services to a Mobile Node, possibly via DHCP. Of course that leads to another observation, a Mobile Node that could use DHCP under these circumstances, would not have the filtering problem, since its "interface" ip src address could be legal. At times, people in the Mobile-IP working group have wondered what the MN-COA mode of MIP is for. Possibly its greatest use should be in external domain situations, since the acquired COA would make packets legal in CERT terms.

We ran into filter #1 when we installed our FA at OGI. OGI is external to our NWNET border-router.

Filter #2 would additionally not allow a Mobile Node to send packets out of an interior domain because its own fixed MN ip src address should not be present locally.

We have performed a series of experiments at the OGI FA to see what would and would not work. We have not observed Filter #2 (yet), but we have observed that Filter #1 is deployed at the border router "entrance" to PSU. We have played a bit with tunnels and have performed two "interesting" (and scary) experiments that confirm that IPIP tunnels can be used to "solve" (in truth, cover up) the problem.

1. we setup two host tunnels from the OGI FA back to the only two PSU hosts likely to be of use for us; our own DNS server, and our compute server. As a result, any number of local PSU MNs at OGI could access PSU services.

2. we "borrowed" the Foreign Agents COA and setup tunnels directly from the MN in question to PSU. The two tunnels were host tunnels again to the PSU DNS and compute servers. The COA was used as the outer ip src address in the mvif driver.

We have always felt that in terms of topologies, there were at least three major Mobile-IP topologies that were of interest:

1. MIP within an interior domain. We are doing this now of course. We regard this as probably the most likely short-term use of MIP.

2. MIP across exterior domains. This is an interesting possibility, but it is hard to know at this point, how common it might be. Certainly there are interesting security

problems here that need to be solved (the firewall and HA/FA tunnel questions are two such problems). Possibly a solution might lie in the realm of expecting MNs to borrow COAs when abroad and use them "intelligently". We also firmly believe that trust negotiation between MNs/FAs and HAs/and firewalls are possibly and indeed likely components of a solution space in this topology.

3. MIP at home as a way to deploy more Internet-capable computers at home. A MIP box at home might fall into either the interior or exterior routing domain areas.

We intend to upgrade the FA at OGI to an ipsec-based kernel soon and try setting up AH based tunnels between it and a PSU agent. We would like to point out that agents are fundamentally a security exposure if they are capable of acting as tunnel endpoints, because they can be used to violate the CERT advised border router filters. We submit that tunnels should probably be tied down by AH and or AH/ESP. We also intend to see if we can setup AH-based tunnels directly from an MN abroad using the FA's COA back to "home".

IPSEC/Mobile-IP Security Considerations

1. route table type semantics are complicated, for example, cloning and routing redirect semantics. There is danger here at the Mobile Node, in that one might choose to tunnel to a local-link HA and use ESP on all packets, but a dynamic redirect might spirit packets away. At a minimum, MNs must be able to turn dynamic routing redirects OFF.

2. at a MN, if one chooses to use DES via the default route to the HA, it is always possible that a more specific non-DES host route might somehow be inserted (especially on a multi-user system). We claim that our laptops are single-user, but this possibility should be kept in mind. Furthermore, host routes might already exist and thus not be overridden by the default route. The first two points here can be summarized by simply pointing out that our route binding is a tool, but we must still study long-term how to use it.

3. by definition, routing means network layer security. This is a useful tool in some situations, but it is not end to end security which is something only app or transport layer security can accomplish. IPSEC/routing does not rule out IPSEC/transport OR ssh for that matter.

4. traffic analysis leading to cryptanalysis by an evil MN at a remote FA is always a possibility. Bellovin has stated that it is desirable for a node to always meet an encrypted packet with an encrypted packet, or plaintext with plaintext. Ideally a node should not respond to encryption with plaintext or vice versa. For example, imagine that EVILBART is pinging MNALICE. With our routing based encryption, EVILBART could notice that MNALICE is at a given FA, and try and ping MNALICE. The packets would go to MNALICE, who might choose to enable a chosen plaintext attach by encrypting EVILBART's packets. EVILBART could use various routing techniques to make sure that the packets were delivered through MNALICE's HA

and hence would go through the ESP tunnel. A MN should be able to filter traffic by end systems. By definition, ALICE might only talk to end systems, X, Y, Z, and EVILBART would not be amongst their number. Normal ip firewall mechanisms at MNALICE could be useful. We might also need more dynamic mechanisms so that ALICE's software systems might be informed that packets from an "as of yet unknown" end system had arrived and could make a policy decision.

5. The ability for a FA to act as the tunnel endpoint means that any FAs available in a local routing domain can be used by external spoofers to defeat current CERT "ingress filtering". At a minimum, we suggest that tunnels be able to turn off input (not possible at a FA by definition, but possible at a HA), and/or use IPSEC AH (at least) and insist that any incoming tunnel packets must be like this:

   `IP | AH | IP (etc).`

   In other words, HA to FA tunnels that cross enterprise domain boundaries should use IPSEC as mechanism. This suggests that tunnels themselves might need to be negotiated by a higher level trust mechanism across domains. Firewall filters can filter out IPIP packets and presumably can be taught to allow IP — AH or IP — ESP. Possibly they may allow packets to only certain IP src addresses (known FAs designed as bastion hosts). Without IPSEC (and higher dynamic key negotiation mechanisms), it is hard to see how Mobile-IP sans IPSEC can be anything other than a "cert'ifed" spoofing threat.

Redundancy

We are going to briefly touch on the current theory behind our redundancy work. We have "theory" in place for the following areas:

1. multi-hop ad hoc routing
2. home agent redundancy
3. foreign agent redundancy.

In addition, in working with Jon Inouye, we have realized that redundancy could also focus on allowing a Mobile Node to bind more than one interface at a time to a route (especially the default route). We do not intend to currently pursue that area, but the notion is interesting and could be useful.

We are currently in the final stages of working out a theory for multi-hop ad hoc routing. We intend to explore "redundancy" aspects of such a routing protocol and will base it on authenticated source routing via multicast. We will go into more details in the next report. For now however we only wish to raise a few points:

1. we are architecting a source-rate based protocol that will be based on top of our current ad hoc protocol. The current protocol is authenticated and establishes link-layer reachability. It is subnet-free. The next generation will also be authenticated and subnet-free. We hope to be able to explore multi-path aspects; i.e., give this protocol a redundancy orientation.

2. we have encountered a difficult kernel routing implementation problem. Basically we need an "input" (besides an IMCP host unreachable message, which may not arrive) from the routing code in the network layer in order to drive the "on demand" source route query in the MN routing daemon. The problem is that if a default route is present, it always matches any destination lookup where there is not a more precise answer (a specific host route). If the default route is not present, the kernel will generate a "`route_miss`'' message and send it upstream to readers of the `route(4)` socket (routing daemons). It is not satisfactory to simply remove the default route, since for MNs that can hear an agent, the default route will be set to the agent. We have experimented with adding a function to the routing code of the kernel that will cause the kernel to generate a "`route_clone`'' message whenever cloning occurs. This will allow us to learn that the kernel has generated a "host route" when the default route is used, and we can use this as an input to drive the multicast source query mechanism at a Mobile Node.

Home Agent Redundancy

The HA redundancy specification work is started. For a change, we do not need any kernel work. We think that HA redundancy might take two flavors: 1. "serial" HA redundancy or 2. "parallel" HA redundancy. "Serial" means that a MN would have a list of two HAs and would only use one HA at a time, but would be able to switch between them, if the current HA failed to respond to a Mobile IP UDP request. Parallel would mean the use of two HAs at once. We prefer the former, since it would lighten the Internet load and seems more practical.

The gist of the idea here is that the HAs will use normal unicast IP routing and act as unicast routers to the MN subnet, which will be partitioned by definition (i.e., the wavelan subnets on the other side of the HAs will not be co-located). In other words, we intend to use normal unicast IP routing for reachability to the Mobile IP subnet. The important point here is that at any time, an individual packet from a host sent to a MN might go to either one of the HAs. As a result, the HAs must keep each other up to date about the location of MNs.

We intend to configure the two cooperating HAs so that know about each other and keep a TCP connection open between the HA pairs. They will use that connection to exchange information about MN status as they must keep a parallel routing table setup for all cooperating MNs. A TCP-based application routing protocol will be defined that allows the HAs to check that their peer HA is up, and exchange MN location information. The HAs will install tunnels in parallel for MNs at remote FAs. If an MN is actually "home" at one HA, the peer HA will install a tunnel to the other HA for packets that are naturally routed to it.

One idea that we should consider here is whether or not MNs could be dynamically informed of a new HA. This would allow for a smooth handoff in case one HA was taken down and replaced by another that did not share the same IP address for the HA. This is a feature that could be built on top of the "serial HA redundancy" system. On the other hand, this may be a feature that is not useful in the short term.

Foreign Agent Redundancy

We have already implemented one possible aspect of FA redundancy; that is, we have a "faster" handoff algorithm that allows us to switch between FAs and gives us some ability to deal with FAs that are partitioned from our current HA. We hope to soon (probably in spring) begin an experimental implementation that will allow a MN to talk to two FAs at the same time.

The fundamental feature point needed here is that we must teach the routing code in the kernel to allow us to load a second gateway. Logically we want to be able to bind a list of gateways to a given destination. IP output would send each packet to every gateway in the list. In terms of the routing table, we might have:

```
Destination IP Gateway 1/if1 Gateway 2/if2
1.2.3.4 2.2.2.2/ed0 3.3.3.3/wlp0
```

When IP sends a packet to destination 1.2.3.4, we would send two packets. One would go the next hop router at 2.2.2.2 via the Ethernet ed0 interface. The other would go the next hop router at 3.3.3.3 via the wavelan interface. (We would probably use the same interface in both cases, but the capability to use two interfaces is interesting). We are simply trying to do what we can to improve the odds that a given packet might make it. On the other hand, the cost to the network is obviously twice as much. Still this may be useful in extreme cases, and as is often the case, a little mechanism in the kernel may give a lot of flexibility at the application layer. We intend to implement the mechanism in the next quarter and think about how to use it in the long term.

FA redundancy would use this mechanism as follows:

1. at the HA, the HA could setup a tunnel route to the MN, and bind two FA gateway/COAs to it.
2. at the MN, a MN using WaveLAN currently has a list of agents (FAs) sorted by signal strength. Instead of using the top single agent, we would use the top two.

We expect this dual gateway mechanism would make intermittent connectivity "better". Possibly it should only be used in the case of intermittent connectivity to the top two FAs. How to measure any cost and advantage here remains an open question.

Rough Measurements of IPSEC and Finnish ssh over WaveLAN

We have made some crude measurements of our current IPSEC/ESP route binding and the Finnish ssh (which has other encryption algorithms besides DES) on a few pentium machines, over WaveLAN, and over "localhost". "Localhost" means we run the client and server on the same machine, thus we get a rough idea of a machine's compute power. The goal here was simply to get a feeling about IPSEC versus ssh in terms of algorithms and to get a feeling about the cost of IPSEC encryption over the WaveLAN radio modem. The ostensible goal was not to measure crypto speed on a pentium machine, although there are certainly some inferences in that regard that can be made. These are rough benchmarks and it is important to point out that the conclusions are rough as well.

We will first characterize the test environment in terms of hardware and software tools, then present the numbers, and finish with a few conclusions.

- Hardware/hosts:
  1. IBM thinkpad laptop, 75mhz pentium
  2. Home agent running on 166mhz pentium machine. This is not a pentium pro.

- Hardware/WaveLAN:

  On paper, WaveLAN is advertised as having a 2mbit speed. we typically see speeds between 1mbit and 2mbit depending on collisions and on timeout errors that are built-in (unfortunately) to the Intel 82586 lan controller used in the ISA WaveLAN system. The testing here was done when the local WaveLAN link was otherwise unused. We were able to verify this by using a new BSD tool called "trafshow" that is a simple "real-time" network analysis tool that shows local link traffic. As a specific example, we typically see speeds of 100k bytes per second with ftp transfers over WaveLAN.

- Software/IPSEC route binding:

  Using the `arp(8)` command, we installed routes between the two hosts over the WaveLAN links that were bound to a DES (or dummy) crypto algorithm. We used ftp to transfer data and compared to this ssh transfers using DES (or other algorithms). It should be noted that the IPSEC/NRL DES algorithm is coded in assembler and is more efficient than the C-based algorithm used in our current ssh code.

- Software/tcpclient + tcpserver:

  As part of our IPSEC verification effort, we created a tcpclient/tcpserver application pair that allow us to send various sized tcp packets over a TCP connection. We used this to test the NRL/IPSEC transport mechanism. Tcpclient sends packets to tcpserver that discards them.

- Software/ssh:

  Ssh is the Finnish tool that allows RSA-based authentication, followed by use of encryption across a TCP-based connection. Scp can be used to replace FTP or rcp. Slogin can be used to replace telnet or rlogin. By default, ssh uses idea with 128 bit keys for encryption but can be told to use DES or 3DES as well. Ssh is entirely an application layer application. It does not use IPSEC and might well be considered as a competitor for IPSEC. (Frankly, our laptop users are making a great deal of use of ssh at this time).

- Software/ncftp:

  Ncftp is a replacement ftp client for the traditional BSD ftp client. Observation has shown that it is more efficient in a number of transfer situations than the traditional ftp. It also has a better user interface. The reason why it is more efficient is that it typically does 30k byte reads or writes and hence has a tendency to fill any TCP window that is available. As a result, the TCP window is often *not* the bottleneck where ncftp is concerned.

Note that all of our applications here were TCP-based. Our numbers are presented in kbytes per second. We only ran our tests a few times in each case and present a rough average. For reasons of comparison, note that maximum Ethernet bandwidth is 1280 kbytes per second.

We have not yet benchmarked IPSEC/ESP/DES over Ethernet and hope to be able to do that this quarter (we need to purchase another PCMCIA Ethernet card or two).

Actual Measurements

Using WaveLAN between the two hosts:

- ncftp with no security in place: 95 kbytes
- scp with only plaintext encryption (-c none): 100 kbytes
- scp with crypto:
  - -c des: 103 kbytes

  - -c idea: 110k kbytes
  - -c 3des: 100 kbytes

- using IPSEC/ESP/DES/route binding over WaveLAN (with ncftp): 100 kbytes
- using IPSEC/ESP/dummy (ncftp): 100 kbytes

Localhost testing on bridgeport:

- tcpclient/tcpserver using IPSEC/transport:
  - IPSEC/ESP/DES: 800 kbytes
  - IPSEC/ESP/dummy: 3,200 kbytes
  - no IPSEC (just plaintext with no ESP): 10,000 kbytes

scp testing over Ethernet

This was done between a second 90 mhz pentium desktop machine and the 166mhz desktop pentium. The Ethernet cards in both machines are 100mbit/10mbit SMC Ethernet cards used on 10mbit Ethernet. These cards are 32 bit PCI-bus cards and are probably among the fastest Ethernet cards available for PCs.

- scp with plaintext (-c none): 478 kbytes
- scp -c des (56): 313 kbytes
- scp -c idea (128 bit key): 262 kbytes
- scp -c 3des (3 * 56): 159 kbytes

ncftp testing over Ethernet - plaintext only

- 1076 kbytes (80+

Benchmark Conclusions

Our primary conclusion is that the computationally intensive cryptographic algorithms DES, 3DES, and IDEA are still fast enough on a pentium machine so that WaveLAN (at roughly 1/10 of Ethernet) speed becomes the bottleneck. All of the presumed scp encryption algorithms that have bigger key sizes than DES were slower than DES over Ethernet, but ran at the same speed over WaveLAN. There was no measureable difference between IPSEC/DES and scp/DES. It is possible that faster LAN radio systems may come onto the market in the next few years, but most likely compute speed will go up as well, and probably will be able to keep up with the encryption cost in the near future. We have tried a few encryption benchmarks on very slow 486/33mhz laptops. On those systems the speed of DES encryption begins to interfere with transmission over WaveLAN. A few of us have used slogin/3des over WaveLAN and have not noticed any apparent difference. This result explains why. In summary, WaveLAN simply isn't fast enough at the moment to get in the way.

On the other hand, a Home Agent acting as the tunnel endpoint for N remote Mobile Nodes using DES/tunnels back to the HA, will need all the compute power it can get. We will attempt in the next quarter or two to measure simultaneous DES stress on the Home Agent (if we can figure out a meaningful test case).

The IPSEC/DES algorithm that we are using was done in assembler and appears to be much more efficient than the C-based ssh DES algorithm. On a fast pentium machine, over localhost, it runs at a rate that is still less than Ethernet bandwidth, but is not bad (800 versus 1260 kbytes).

Certainly over 100mbit fast Ethernet, any of the encryption algorithms would become a bottleneck. On the other hand, our measurements show that ncftp is twice as fast over an unused Ethernet link than scp in plaintext mode. Scp could probably stand some application code tuning as well as use computer specific encryption code.

The tcpclient/tcpserver test that used IPSEC/DES versus IPSEC/dummy in transport mode on the same pentium server is interesting. It points out that there are two costs involved with IPSEC and both are very real. Of course, there is a compute cost involved in DES encryption and decryption. The dummy transform simply carries out mbuf (message buffer) manipulations in the kernel and inserts the ESP header in front of the plaintext TCP and data portions. One can see that there is a heavy cost involved in just the empty transform. As NRL has pointed out, there may be room there for future tuning.

Outreach

John McHugh and Jim Binkley paid a brief visit to BBN in late September to compare notes with Steve Kent and John Zao. At this point we have an idea as to what each group is up to in terms of security and mobile networking. Up to now, we have been working on low-level mechanism, and they have been working on high-level signature work.

John McHugh gave a presentation on the project as part of a panel at the National Computer security Conference in Baltimore.

David Reeder has attended recent IETF and Mobicom meetings and made various useful contacts in the mobility research community. We are hoping that we might be able to obtain a copy of FTP Inc's Mobile-IP release for interoperability testing. David will go to Boston for their announced interop testing before the Memphis IETF. Jim might and David will attend the next IETF meeting.

Jim Binkley plans on giving a talk in the next quarter at OGI and later in February at Intel to their mobility group on our current work. John will continue to attend various PI meetings and give briefings on our current status.

In winter quarter, at least 3 professors (Binkley/Daasch/Schubert) used mobile laptops to give software demos to their classes in the PCAT building. Professor Daasch of Electrical Engineering was able to demo Mentor CAD tools available on sparc systems elsewhere for his EE class and pronounced the radio system extremely useful. There is certainly much opportunity for deployment of mobile radio lans in campus teaching environments.