Internet Engineering Task Force
Internet Draft
Expiration Date: June 2003

P. Savola
CSC/FUNET

December 2002

Security of IPv6 Routing Header and Home Address Options

draft-savola-ipv6-rh-ha-security-03.txt

Status of this Memo

Abstract

All IPv6 nodes must be able to process Routing Header [IPV6] and Home Address [MIPV6] Options. With these, packet filter access lists can be tricked (among other things) as the destination and source addresses, respectively, are being rewritten as the packet traverses the network. Some of the security considerations of these features are analyzed, and a few possible solutions presented. It will be shown that with the current architecture, the network-based security does not seem to scale to the requirements of Mobile IPv6; it seems possible that unless security is taken seriously when implementing the nodes, the new Mobile IPv6 requirements might not be allowed to be used at all in some circumstances.

Table of Contents

1. Introduction

All IPv6 nodes must be able to process Routing Header [IPV6] and Home Address [MIPV6] Options. With these, packet filter access lists can be tricked (among other things) as the destination and source addresses, respectively, are being rewritten as the packet traverses the network.

Some of the security considerations of these features are analyzed, and a few possible solutions presented.

For both Routing Header and Home Address options, basically two approaches to enhance security are put forward. It will be shown that with the current architecture, the network-based security will not seem to scale to the requirements of Mobile IPv6; it seems possible that unless security is taken seriously when implementing the nodes, the new Mobile IPv6 requirements might not be allowed to

be used at all in some circumstances.

In many cases, ingress and egress filtering [FILTERING] are being performed. Unfortunately, the filtering is not being done everywhere; the existance of it cannot be relied on. Routing Header and especially Home Address options are very harmful if at least ingress filtering is not being performed, but to some extent, they can still be used quite effectively if filtering is in place too.

Routing Header and Home Address Option can be used in helping to hide the traces of DoS attacks from certain tracing methods. Here, ICMP Traceback [ITRACE] and Reverse ICMP Traceback [REVITRACE] are used as examples; the issues most probably affect other mechanisms as well.

A lot of discussion here is based on the fact that in the real world, packet filtering is a practical requirement for safe operation; almost everyone uses it. Therefore, it is important that it can be performed in a predictable fashion.

The security of Binding Updates is another very crucial issue but that is already discussed in other proposals, including [BUSEC].

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

1.1. Terminology

Ingress filtering

> Filtering on source addresses from the direction of a site to the Internet. This is often done to keep the site from using wrong source addresses.

Egress filtering

> Filtering on source addresses from the direction of the Internet to the site. This is often done to keep packets with IP addresses belonging to the site from arriving in the site from the Internet

Reflected attacks

> Attacks that are launched by the attacker, using a third party as a proxy, against the victim. Here, the term is used in a more generic sense than in, for example, [PAXSON].

2. Routing Header

All IPv6 nodes must be able to process Routing Headers. It is implied, even though not clearly stated, that all nodes (including hosts) must also have that processing enabled.

Here, "Routing Header" is used as a synonym for "Type 0 Routing Header" unless explicitly stated otherwise as it is the only type defined at the moment.

There are several problems with Routing Headers:

- Getting around access controls if done with destination address
- The behaviour is difficult to disable if one wants to support Mobile IPv6
- Can be used in reflected Denial of Service attacks
- Can be used to make tracing the path of DoS attacks with iTrace [ITRACE] more difficult.

These are discussed at more length below.

2.1. The Traffic Filtering Problem

Because destination address is replaced at every Routing Header processing point, it's impossible to perform traffic filtering based on destination addresses. An example:

```
        host1 --- rtr1 - Internet - fwrtr2 -+- webserver
                                           |
                                           +- host2
```

Assume that fwrtr2 is performing packet filtering on the internal interface. The rules could be like:

```
      [...]
      allow proto tcp from any to webserver port 80
      deny  proto tcp from any to any
      [...]
```

Here, malicious host1 would write packets as follows:

```
      src=host1
      dst=webserver
      rtheader=host2, segments left=1
      payload proto=tcp, dport=80
```

It would pass the packet filter checks fine, and be processed at webserver. When being forwarded directly from there to host2, the packet would look like:

```
src=host1
dst=host2
rtheader=webserver, segments left=0
payload proto=tcp, dport=80
```

If the packet had been sent directly to host2 without Routing Header, it would have been denied in the packet filter access lists.

Even though webserver is configured as a Host, it will forward packets with Routing Header. This breaks the principle of least surprise, and as any node can be used as a traffic reflector, the network is very difficult to secure.

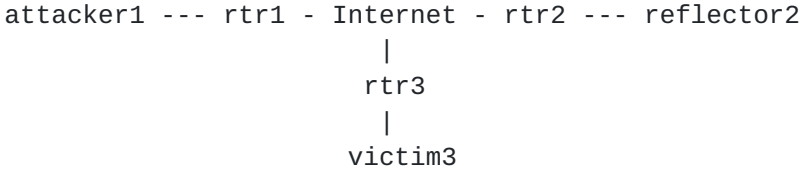
The same naturally applies to Routers too, but they usually don't have many publically available services, so they aren't optimal for this kind of "access list avoidance via a reflector" attack.

2.2. The Denial of Service Reflector Problem

This attack only makes sense if source addresses can be spoofed. Some issues about this are also covered under "Home Address Option" sections. One can avoid being used in this reflecting attack by performing proper ingress filtering at the "reflector" site.

So, this method can be used to route Denial of Service attacks through intermediary reflectors, to make it more difficult to trace them.

Assume the scenario:



Now, assuming that ingress filtering is not done at rtr1 (from the direction of attacker1) and rtr2 (from the direction of reflector2), one could send packets that would be reflected anonymously to victim3 via reflector2:

```
src=spoofedX (attacker1)
dst=reflector2
rtheader=victim3, segments left=1
```

This way, if victim3 wants to investigate from where the packets are coming from, first the source would appear to be spoofedX; it may or may not be obvious to victim3 whether the address is spoofed. Then, investigating Routing Header, the trails would lead to reflector2, where they would disappear.

However, it should be noted that reliance on Routing Header trails is very chancy at best. The attacker could very well construct a packet with false routing header entries, like:

```
src=spoofedX (attacker1)
dst=reflector2
rtheader=someoneA,victim3, segments left=1
```

At reflectror2, the packets are now directed straight to victim3 (because segments left is one smaller than it normally should be) and one node (someoneA) would only be inserted for false trails. Segments left could have been 0 at the sender, for all the destination knows. So, in consequence, the packets can be made to look like they went through someoneA but really didn't.

One should note that even if rtr2 would be performing ingress filtering, rtr2 itself, if the filtering is not carefully implemented, is also susceptible to this.

It should be noted that one can never be safe from this, but if one would be able to easily disable routing header processing (as it should be in Hosts at the very least), the damage might be more limited and controlling easier.

2.3. The ICMP Traceback Avoidance Problem

ICMP Traceback [ITRACE] specifies a new method where intermediate routers may send ICMP Traceback messages probabilistically along the path of the packets to the destination address. Among others, this can be used to trace the origin of Denial of Service attacks with forged source addresses.

If the attacker inserts a Routing Header in the spoofed packets, the only iTrace messages to reach the final destination will be those that came from between the last Routing Header entry and the final destination. To illustrate a simple scenario:

```
attacker1 -- Internet -- rtr2 -- Internet -- victim3
             (20 hops)          (10 hops)
```

Here, iTrace messages from between attacker1 and rtr2 (the really interesting data) would be sent to rtr2, and from between rtr2 and victim3 to victim3. Rtr2 would probably just discard the messages, having no knowledge what to do with them. Victim3 would receive messages only from 10 last hops which would be next to unusable.

Naturally, it would be possible to circulate the traffic via many different routers, possibly adding a few in the list for confusion (that is, using smaller segments left than the number of routers, as described above).

One could revise the iTrace specification so that it'll send messages to every destination listed in the routing header, but it might be more appropriate just to acknowledge the problem; after all, victim3 would know from the packets it receives that rtr2 should know the full path, and contact its adminitration.

2.4. Some Requirements for Routing Header Use

One should not limit the flexibility of Routing Header usage too much by too strict constraints. One could assume that it might be used for at least:

- Mobile IPv6: packets are sent to the mobile node with the care-of address as destination address and Home Address as the last Routing Header entry. These should be assigned on the same node. This way packets can be transmitted transparently between stable addresses.

- Traffic engineering or multihoming: for example, one could want to choose the ISP dynamically by some fine-grained criteria (e.g. TCP destination port number) with an automatic use of Routing Header. With this, the Routing Header processing nodes would often be publicly known routers (identifiable by specific configuration or an anycast address, for example) in a topologically important location.

2.5. Solutions

Two node-based approaches are defined. It is expected that only one would be chosen when it becomes more clear which one is best.

2.5.1. Node-based Approach

Traffic engineering requirements are not difficult to meet; one just has to assume that most routers do have Routing Header processing enabled. This does not create new significant security considerations, unless site's internal routers were to also process Routing Headers. It can be assumed that responsible administrators turn off the feature on critical routers; Security Considerations section also discusses this issue.

Mobile IPv6 is more difficult, as it requires that all mobile nodes are able to completely process Routing Header. However, with MIPv6, the maximum segments left value used is 1 when it reaches the destination network. Taking this into consideration, we would be able to deduce additional new rules for processing Routing Headers (we're assuming here that Home Address would be assigned on the appropriate interface; please also see Security Considerations about this):

If a node would have to process the Routing Header (that is, destination address equals the node and segments left > 0), it SHOULD check whether segments left equals 1, and if both the current destination address and to-be-swapped destination address in the Routing Header are both assigned to the same interface of the node and are both of the same zone [ADDRSCOPE], the Routing Header is here referred to as "Interface-local Routing Header".

On Routers, further processing of Routing Headers SHOULD be configurable and SHOULD be enabled by default.

On Hosts, further processing of Routing Headers MAY be configurable and MUST be disabled by default.

Regardless of the general Routing Header processing setting, all nodes SHOULD still process "Interface-local" Routing Headers. Disabling this exception MAY also be configurable.

2.5.2. Second Node-based Approach

This approach just tackles the current Routing Header use requirements (Mobility), and leaves the rest undefined; as the processing rule change would be rather simple, this might be the best way forward until it becomes clear for what else "Interface-local" Routing Headers would be needed for. The revision would be as follows:

On Hosts, Routing Header processing MUST be supported, but the processing MUST NOT be enabled by default. The enabling of the

processing SHOULD be configurable.

Regardless of the processing setting above, routing headers destined to the host are further processed if the following applies:

type == 0 and segments left == 1 and,
if the to-be-swapped address in Routing Header is either:

1. a Home Address assigned to the node,

   the Routing Header MUST be processed in full.

2. or, an address which has been authorized in some way to be a valid target of routing header processing,

   the Routing Header MAY be processed in full.

2.5.3. Network-based Approach

If the packet filters supported parsing Routing Header and performing checks for it, one could argue that there might not be need to change the processing in nodes.

The problem is that intermediate nodes, for example packet filters, cannot distinguish mobile nodes from stationary nodes. The distinguishing would only be possible if the node was located in such a way (usually a single point of failure) that every Binding Update would go through the intermediate node and it would keep state of mobile nodes in the internal network. It might also be possible to keep record of the state via some AAA mechanism.

If one assumes keeping state reliably cannot be done, one could perform filtering in general case if and only if either is known:

- there are no mobile nodes in the internal network, either foreign (Home Agent is not in this network) or local.
- the only mobile nodes in the internal network are foreign, only roaming in the internal network.

The first scenario is trivial, but in the long run, probably too restrictive.

In the second scenario one would have to require that the packets with Routing Header must not have the to-be-swapped address topologically in the internal network ("must bounce out").

One should note that if this "bouncing out" is allowed, anyone can use any node in the internal network as a traffic reflector if

ingress filtering isn't being used.

On the other hand, if one assumes the state of Mobile Bindings could be kept, one could derive rules on packets from outside to inside, for example:

- Packets with Routing Header going to mobile node care-of Addresses are allowed if and only if they are "Interface-local".
- If packet with Routing Header bounces inside the internal network, deny it by default (local policy issue).
- If packet with Routing Header bounces out of the internal network, deny it by default (mostly testing scenarios)
- If packet with Routing Header bounces out of the internal network, so that the final destination and source addresses are equal, allow it by default (e.g. advanced traceroute, a special case from above)

2.5.4. New Routing Header Type

A more radical approach might be to define a new Routing Header type. This type would be syntactically identical to Type 0 Routing Header, except it would have the requirement of "Interface-local" property, as discussed above in 2.3.1, for the last-to-be-swapped addresses.

The "Interface-local" property would be a MUST to be observed at the receiving side; otherwise the packet would be discarded (or alternatively, delivered to the second-last address; the exact behaviour is TBD).

This way, nodes communicating with mobile nodes would use this new Routing Header type, which could be easily identifiable by intermediate nodes (and passed through without too big worries about security), and the processing of Type 0 Routing Header could remain intact. Of course, it would still be allowed to send "Interface-local" Routing Headers as Type 0, but practically the probability of it getting through might be smaller.

A new Routing Header type would be of great help for both Node and Network-based approaches.

An apparent problem of new Routing Header type is that all participating nodes should be able to recognize and process it. However, as MIPv6 is still work in progress, this might not be such a big obstacle after all. One should note, however, that this might effectively hinder combining MIPv6-like Routing Header use with e.g. certain traffic engineering solutions, as the participating participating routers would have to be able to understand the new Routing Header type.

3. Home Address Option

Home Address option of Mobile IPv6 must be processed in every node whether mobile or not. The source address of a packet is replaced with the address in the Home Address option. The packets don't need to be authenticated in any way.

The problem with Home Address is that it allows trivial unidirectional spoofing (good for simple exploits, DoS attacks); destinatation network's internal addresses can also be spoofed (which could normally be prevented in destinatin network's border router by egress filtering).

As a general note on both spoofing attacks, one could argue that there will always exist operators that do not perform filtering, and as the packets are not in general authenticated, the source address of an unauthenticated packet should not be trusted anyway.

It is true that source address alone is not enough for authentication, but it still should be enough for some rudimentary form of identification.

3.1. The Source Address Spoofing Problem

Sites and operators perform ingress filtering to keep nodes from spoofing their source address. With Home Address option, anyone can work around these checks. An example:

```
host1 --- fwrtr1 - rtrISP - Internet - fwrtr2 ---host2
```

Assume packets are originated from host1 with real IPv6 address 3ffe:ffff:0::1/64. Now fwrtr1 could perform ingress filtering from the direction of host1 as follows:

```
[...]
allow ip from 3ffe:ffff:0:0::/64 to any
deny ip from any to any
[...]
```

And even if fwrtr1 fails to do that, rtrISP would probably perform ingress filtering as well, with 3ffe:ffff:0::/48.

Now, malicious host1 would write packets as follows:

```
        src=host1 (or some spoofed address from 3ffe:ffff::/64)
        dst=host2
        home address=3ffe:fff0::1 (or whatever)
```

The packet would have been dropped if 3ffe:fff0::1 had been put in the source address itself. Now, however, the packet passes through to host2, which MUST replace src=host1 with src=3ffe:fff0::1 based on the Home Address option.

A more dangerous variation is being able to circumvent egress filtering of the destination site; with topology:

```
          host1 ---  rtr1  - Internet - fwrtr2 ---host2
               3ffe:ffff:1::/48       3ffe:ffff:2::/48
```

Assume that fwrtr2 aims to protect the site by filtering all source addresses belonging to the site that come from the direction of the Internet, like:

```
        [...]
        deny ip from 3ffe:ffff:2::/48 to any interface from_internet
        allow ip from any to 3ffe:ffff:2::/48 interface from_internet
        [...]
```

Now, the attacking host1 could write packets as follows:

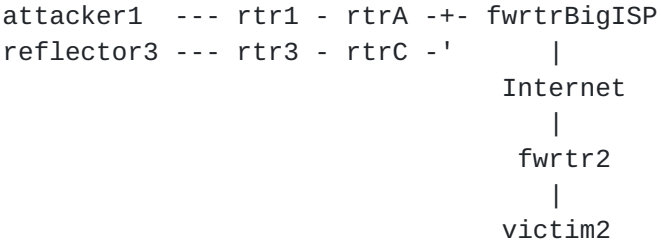
```
        src=<spoofed> (or something from 3ffe:ffff:1::/48 if rtr1 is
filtering)
        dst=host2
        home address=3ffe:ffff:2::2 (some trusted node in site2)
```

One could argue, that as the original source address is still in the header (whether spoofed or not), it is not usable for spoofing. Assuming the address could not be used, this would be only true to a certain extent. Consider a user from some foreign dial-up service looking for vulnerabilities. The account is possibly stolen, or the local authorities do not care even if the IP address was recorded in conjunction of the probing. However, with Home Address option, you might be able to check whether the scanning with some other IP addresses, e.g. a privileged address from the destination network, would yield more interesting results.

3.2. The Double Spoofing Problem

This is an extension of a reflector attack with Routing Header from section 2.2. The main purpose is to perform a denial of service, or similar, attack, with an additional goal of blaming someone (or two someones) else for it.

"Double Spoofing" is also possible if ingress filtering is only being done very far in the upstream. Combined with Routing Header, this could be used to throw blame on others. Assume the scenario:



Now malicious attacker1 could write:

```
src=reflector3
dst=reflector3
rtheader=victim2, segments left=1
home address=thepresident.whitehouse.com
```

Even though victim2 noticed something strange was going on, everything would point at reflector3, even the incoming packet trail (note: there would still be a Routing Header with reflector3 in it, so one might be able to notice something was not right); reflector3 would be too far away from attacker1 to pose a "threat" to attacker1.

Instead of reflector3, one could of course also use some unsuspecting router on the other side of the globe.

Now, let's see what the packets would actually look like at victim2's access logs. Casully checked, it would be like:

```
destination=victim2
source=thepresident.whitehouse.com
```

A more detailed analysis (a packet dump from the wire with a tool that is able to parse home address) would indicate:

```
destination=victim2
source=reflector3 (or some spoofed address)
home address=thepresident.whitehouse.com
```

And complete data would be:

```
destination=victim2
source=reflector3 (or some spoofed address)
routing header=reflector3
home address=thepresident.whitehouse.com
```

As noted, a casual observer would probably blame "thepresident.whitehouse.com" immediately. As of this writing, there are no IPv6 packet filters or system-level loggers that would also log the original source address; thus causing a very high probabily for these false trails to go unnoticed.

3.3. The Protocol Reflection and Untraceability Problem

ICMP Traceback [REVITRACE] suggest modifying iTrace so that it would also be possible to send ICMP Traceback messages to the source of the packets as well. There are certain issues with this, as the source addresses are most probably spoofed, but here only the interaction with Home Address Option is considered.

Here, with 'protocol reflector', we mean a more specific case of packet reflection, as introduced in [PAXSON]. (For example, src sends TCP SYN to protocol reflector, and reflector sends SYN+ACK to the victim). Consider:

```
src = attacker (gets by ingress filtering) or some other node
dst = protocol reflector
home address = victim
```

The attacker could use Home Option to send these Reverse iTrace packets to itself or some node that it believes will not react to them. Now, Reverse iTrace messages would go to the attacker. Home Address Option in protocol reflection is especially harmful, as once reflected, the packet will look like:

```
src = reflector
dst = victim
```

And every trace of the message is lost, including Reverse iTrace traceability. However, the path from the attacker to protocol reflector would still be traceable with Forward iTrace; the issues with this are discussed under "The ICMP Traceback Avoidance Problem" section.

3.4. Some Requirements for Home Address Option Use

Home Address option only makes sense when it's used by mobile nodes. The intent is to make the use of care-of addresses transparent, and it apparently should only be used when there exists, or is being formed, a binding between the communicating two nodes.

3.5. Solutions

3.5.1. Node-based Approach

The most important thing is that Home Address option can be trusted to do the right thing in the end node (so that there would be no need to limit it in the network): Home Address must not be allowed to be used by completely untrusted and unauthenticated users.

How this is achieved is beside the point; one way to get to this would be to revise the processing rules as follows:

Home Address option MUST NOT be processed unless either of the two applies:

- the packet, including Home Address option, is authenticated, and the authentication is verified to be correct.

- there exists a binding between the source address and Home Address in the destination node, and the binding (usually, but not necessary, one created with a Binding Update) was created securely.

One should note that the latter might require some additional changes in Mobile IPv6, as it is required that all Binding Updates MUST also include a Home Address option, thus creating a chicken-and-egg problem. One should use authentication for these, though.

It should be noted that this approach makes the so-called triangle routing between MN, CN and HA impossible.

3.5.2. Network-based Approach

The discussion of Network-based Approach under Routing Header applies to Home Address as well; without state, and assuming there are mobile nodes in the network, it's impossible to create generic rules on which would be an allowed Home Address and which not.

Without keeping state, filtering could be performed only if:

- local mobile nodes never roam outside of the internal network performing filtering.

4. Conclusions

It's very difficult to securely control, above all Mobile IPv6 -related, use of Home Address and Routing Header in the network. Even if the packet filters would support rule-based matching of Home Address and Routing Header fields, it would be practically impossible to create any kind of general rules on what should be allowed and what not.

Therefore, improving the node-based security should definitely be taken into consideration. Additional, local policies can also be be made in very advanced packet filters, but the existence of this technology must not be relied on.

To summarize, the author believes the following should be done:

- Routing Header processing be disabled on all hosts except for Interface-local Routing Headers, or a new Routing Header type taken to use for MIPv6 purposes, and

- Home Address option processing in the correspondent nodes revised in such a way that HA option will only be processed if it is provably secure or the binding that will use it was created securely.

- It should be considered whether the use of Routing Headers or Home Address Option makes traceback mechanisms too non-effective, and whether the mechanisms should be revised.

5. Security Considerations

This draft discusses security considerations; only some of the presented issues (mostly acknowledged problems) are presented here.

The widespread use of Routing Header or Home Address Options may make traceback mechanisms at least partially non-effective.

It was suggested that Routers should by default process Routing Header packets. It is assumed that site internal routers should have this feature disabled (or controlled) by the administration. If this is not done, the routers could be used to reflect packets and possibly avoid access controls as outlined in section 2.1.

One should note that note that "Interface-local" property of Routing Headers is important to be, indeed interface-specific, not node-

specific. Consider a node with two interfaces, one public, and one completely private. Even if a private service would be bound to the private interface only, one should not be able to use this private service from anywhere in the Internet.

Even the interface-locality is not bulletproof, as some addresses assigned to the interface may be more private and public than others; at least this way the potential harm can be minimized.

It is clear that interface-locality is a must in certain scenarios (e.g. a security gateway, or a secure node), but in a general case, it is debateable whether a significant loss of security would occur if the restriction would be lifted from normal hosts.

The same zone requirement is also important (how this is observed is implementation dependent; the check doesn't need to be explicit in Routing Header processing) so one cannot reach e.g. link- or site-local addresses through Internet.

More fine-grained control of the two addresses could be obtained via access lists if they support Routing Header processing.

6. Acknowledgements

Discussions with Francis Dupont led to the writing of this draft. Valuable comments were received from Alexandru Petrescu. Erik Nordmark provided extensive commentary and brought up the interaction with Reverse iTrace. Jarmo Molsa reviewed a later version of the draft.

7. References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[FILTERING] Killalea, T., "Recommended Internet Service Provider Security Services and Procedures", BCP 46, RFC 3013, Nov 2000.

[PAXSON] Paxson, V., "An Analysis of Using Reflectors for Distributed Denial-of-Service Attacks", Jun 2001, http://www.aciri.org/vern/papers/reflectors.CCR.01/.

[IPV6] Deering, S., Hinden, R., "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[MIPV6] Johnson, D., Perkins, C., "Mobility Support in IPv6", draft-ietf-mobileip-ipv6-16.txt (work in progress).

[BUSEC] Arkko, J., "Issues in Protecting MIPv6 Binding Updates", draft-arkko-mipv6-bu-security-01.txt (work in progress).

[ADDRSCOPE] Deering, S., et al. "IPv6 Scoped Address Architecture", draft-ietf-ipngwg-scoping-arch-04.txt (work in progress).

[ITRACE] Bellovin, S., Leech, M., Taylor, T., "ICMP Traceback Messages", draft-ietf-itrace-01.txt (work in progress).

[REVITRACE] Barros, C., "A Proposal for ICMP Traceback Messages", http://www.research.att.com/lists/ietf-itrace/2000/09/msg00044.html.

Author's Address

Pekka Savola
CSC/FUNET
Espoo, Finland
EMail: psavola@funet.fi