

Simultaneous Mobility in MIPv6

K. Daniel Wong, Malaysia University of Science and Technology and Telcordia Technologies

Ashutosh Dutta, Telcordia Technologies

dwong@must.edu.my, adutta@telcordia.com

Abstract—Mobile IP for IPv6 (MIPv6) includes a procedure for route optimization, which allows packets from the correspondent host to go directly to the mobile host. However, the way route optimization is implemented makes it vulnerable to problems with simultaneous mobility, which can occur when two mobile hosts are communicating with each other. In this paper, we suggest ways to change MIPv6 to reduce its vulnerability to the simultaneous mobility problem.

I. INTRODUCTION

Mobile IP (MIPv4, [1]) is the dominant mobility-handling protocol for host mobility on the Internet. However, with the rise of IPv6, a new mobility-handling protocol was needed for host mobility on IPv6 networks. The new protocol, proposed by the Internet Engineering Task Force (IETF) is Mobile IPv6 (MIPv6, [2]). Among the new features introduced with MIPv6 is route optimization (MIPv4 also had a way to do route optimization, but route optimization for MIPv4 was always a “draft-stage” add-on and never part of the main MIPv4 specifications). Route optimization has its benefits, but a major problem is that it makes MIPv6 vulnerable to the simultaneous mobility problem. Since 3G wireless specifications groups are moving towards IPv6, and beyond-3G systems are even more likely to use IPv6, the vulnerability of MIPv6 to the simultaneous mobility problem becomes a critical issue.

Simultaneous mobility is the case when both end hosts are mobile and they move at about the same time. The simultaneous mobility problem is when mobility signaling is lost during simultaneous mobility. This problem may arise because a particular mobility protocol was designed with the implicit assumption that the correspondent host does not move when the mobile host moves. The figure below shows how the route optimization-related signaling can be sent to the old IP address on both sides, when two mobile hosts are moving at about the same time. This results in binding updates being lost, thus breaking communications between the

mobile hosts. The problem doesn't occur with MIPv4 because MIPv4 uses stationary HAs that don't move.

A related but different problem is the problem of *consecutive mobility* that is too quick. This would be when an MN (mobile node) moves to a new network and begins binding updates and registrations. However, before these procedures complete, the MN moves again (thus, the name “consecutive mobility”) to a new network. In this paper, we assume that consecutive movement between networks is not so rapid that any problems arise with consecutive mobility.

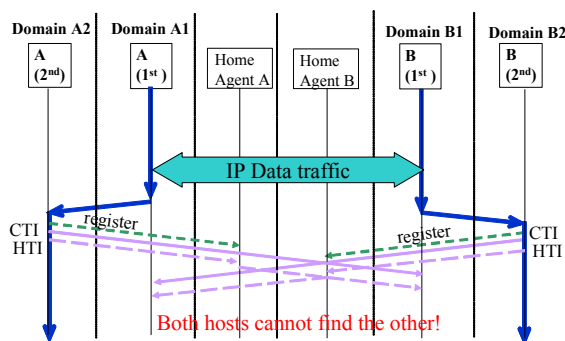


Figure 1: Mishandling of simultaneous mobility by MIPv6 with Route Optimization

We will analyze the simultaneous mobility problem for MIPv6, propose solutions and analyze them. Earlier work [3] on simultaneous mobility has focused only on solutions for other mobility protocols, namely SIP mobility management and MIP-LR (Mobile IP with Location Registers). A simple probabilistic analysis has shown that the probability of simultaneous mobility can be unacceptably high in ordinary situations [3].

This paper is organized as follows: in Section II, we lay out the pertinent details of MIPv6 that will be needed for the analysis of the effects of simultaneous mobility on MIPv6. These details include the sequence of events when an MN moves to a new foreign network, and the return routability procedure. Based on such details, in Section III we analyze the effects of simultaneous mobility on MIPv6, for a number of important cases. In Section IV, we then present our proposed solution.

II. RELEVANT ASPECTS OF MIPv6

The direct binding updates from the MN to the CNs pose a security problem. In the case of the home agent, it is reasonable to stipulate that the MN and home agent should have an IPsec security association, but it is not reasonable to stipulate the same between the MN and every potential CN. How, then, can a MN send binding updates to its CN securely? The solution adopted in MIPv6 is to use the so-called Return Routability procedure. This allows the MN and CN to set up a shared key in a “reasonably” secure manner.

Since timing is so important in analyzing simultaneous mobility, we need to review the exact sequence of events that happens whenever an MN moves to a new foreign network. The typical sequence of events is as follows:

1. (if handing off from a previous network) MN stops being reachable at the previous network
2. MN obtains a new care-of-address at the new foreign network
3. MN initiates registration of its new care-of-address with its home agent
4. MN performs correspondent registration procedure with all its CNs (except those from which it may wish to keep its topographical location private); correspondent registration consists of two sequential parts:
 - o Return routability procedure
 - o Sending authenticated binding update using the shared key generated during the latest return routability procedure

We note that the MN must initiate registration of its new care-of-address with its home agent (by sending a Binding Update to the home agent) before it may perform any correspondent registration procedures related to the new care-of-address. However, it does not need to wait to see if the registration with the home agent is successful before performing correspondent registrations.

We also note that the correspondent registration procedure, as specified by RFC 3775 [2], *always* includes a return routability procedure before sending of binding updates to a CN. This may seem to imply that an MN that moves fast between foreign networks would have to go through a long correspondent registration procedure with each of its CNs every time it moves. This could be fairly disruptive to the packet flows from CN to MN. However, RFC 3775 does specify an optional expedited return routability procedure. We now summarize both the standard and expedited return routability procedures.

A. Return Routability Procedure

The MN sends two messages to the CN; these are the Home Test Init and the Care-of Test Init messages. The Home Test Init is reversed tunneled to the home agent from the MN, and then forwarded by the home agent to the CN. The Care-of Test Init is sent directly to the CN.

The CN responds by sending a Care-of Test message directly to the MN, addressed to its care-of address, and by sending a Home Test message indirectly to the MN, addressed to its home address. The Care-of Test message contains the *care-of keygen token*, and the Home Test message contains the *home keygen token*.

The care-of keygen token and home keygen token are hashed together to form the *binding management key*, a shared secret key that can be used by the MN to send a binding update to the CN. The CN is the only node that can generate the care-of and home keygen tokens, since the generation of these tokens involves the *node key* of the CN, where the node key is a secret key never shared with anybody, and whose value changes from time to time. Because the tokens are sent to the MN by two paths, it is most likely only the MN that would also know both tokens¹. Thus, only the CN and MN can generate the correct binding management key, which is used to authenticate binding updates from the MN to CN.

B. Expedited Return Routability Procedure

RFC 3775 does not allow an MN to perform return routability once, generate the binding management key, and then re-use the key for subsequent binding updates (and furthermore, the correspondent registration procedure for each CN proceeds independently, each with their own binding management key). Instead, a new binding management key must be derived each time an MN moves. However, RFC 3775 allows for an expedited process for deriving the key – a recent home keygen token can be re-used, with a new care-of keygen token.

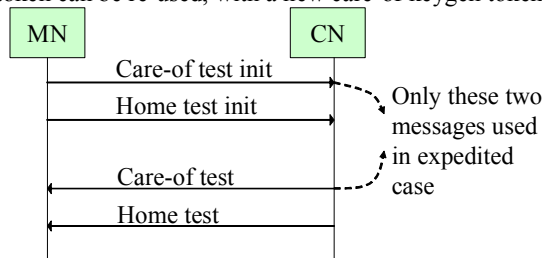


Figure 2: Return Routability

Thus, only a new care-of keygen token needs to be obtained, so the return routability procedure then consists of a Care-of Test Init message from MN to CN, and the care-of keygen token being sent in response directly to the new care-of-address, in the Care-of Test message. This is expected to reduce correspondent registration latency in some cases, especially where the roundtrip through the home agent (for the Home Test Init and

¹ Herein lies one of the controversial features of route optimization for MIPv6 – it only provides “reasonable” security – it is unlikely that an attacker would listen in on both paths and capture both tokens, but it is possible. Further discussion of this security weakness is beyond the scope of this paper. However, the problem is addressed in other papers.

Home Test messages) may be significantly larger than the roundtrip directly between MN and CN.

III. ANALYSIS OF SIMULTANEOUS MOBILITY PROBLEMS

What happens when both the MN and CN are both mobile, and they both move at around the same time? When an MN moves to a new foreign network (in Section II), we see that up to, and including, step 3 (initiation of registration with the home agent), any movement of the other node does not matter. However, when it comes to step 4 (correspondent registration), the situation changes. Correspondent registration depends for its success upon messages reaching the other node. If the other node moves any time before receiving the binding update, one or more messages may be lost and the correspondent registration procedure may therefore fail. The messages lost may be during the return routability phase or after it. We consider the two cases in turn, in sections III.A and III.B. In these sections, we first assume that expedited return routability is not implemented. Later, in section III.C, we extend our explorations to the implications and impact of the use of expedited return routability.

A. Messages Lost During Return Routability Procedure

Let's call the two nodes A and B. A moves, initiates home agent registration, and then initiates the return routability procedure with B. One or both of the Home Test Init and Care-of Test Init messages may be lost because of simultaneous movement by B. We do not consider the case that the Home Test or Care-of Test messages are lost, though, because that would be a problem with overly rapid consecutive mobility by A (consecutive mobility as defined in Section I).

Now consider what happens to the correspondent registrations that B wishes to initiate with its correspondent nodes. Without receiving both the Care-of Test Init and Home Test Init from A, B will not update A's address in its cache, so B will send its Care-of Test Init and Home Test Init messages to A's old care-of-address. More precisely, B will send its Care-of Test Init message directly to A's old care-of-address, and reverse tunnel its Home Test Init to its home agent which will then forward it to A's old care-of-address. Thus, B will be unable to proceed with its own return routability procedure.

Basically, the scenario is that one of the nodes moves first (say, A), but by the time B moves a little while later, one or both of the Home Test Init messages has not yet reached B, so A's return routability procedure never completes, which implies that B will also be unable to complete its return routability procedure. This scenario is the one pictured in Figure 1.

B. Messages Lost After Return Routability

Now suppose that B did not move so soon after A moved, but only moved after A's return routability procedure has completed. Then A and B would have the binding management key for A to send an authenticated binding update to B. However, because of B's movement, A's binding update goes to B's old care-of-address and is lost. Meanwhile, B is trying to initiate B's return routability procedure, but both the Home Test Init and Care-of Test Init messages go to A's old care-of-address, and so are lost as well. This scenario is pictured in Figure 3.

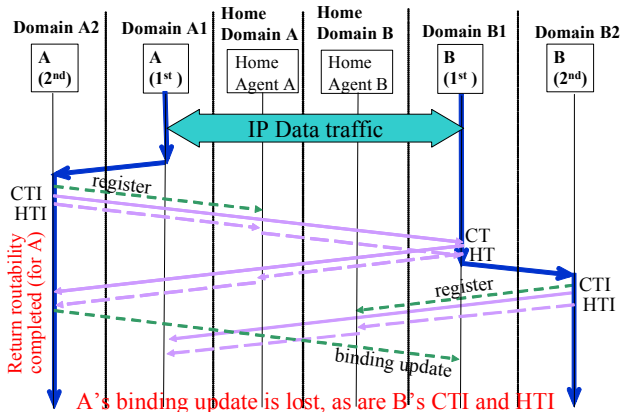


Figure 3: Scenario where one side completes return routability procedure but the other does not

C. The Use of Expedited Return Routability

When a standard return routability procedure has been recently performed, an MN can re-use the home keygen token sent by a CN. Thus, only the Care-of Test Init and Care-of Test messages need be exchanged in the expedited return routability procedure, as discussed in Section II.B. In Section III.A, we considered the case where both sides are unable to complete their respective return routability procedures. An analogous scenario exists when one or both of A and B are attempting an expedited return routability procedure.

Basically, the scenario is that one of the nodes moves first (say, A), but by the time B moves a little while later, the Care-of Test Init message has not yet reached B, so A's expedited return routability procedure never completes. This implies that B will also be unable to complete its return routability (whether expedited or not).

IV. PROPOSED SOLUTIONS

In our previous work [3], we introduced a framework for handling simultaneous mobility problems. We briefly summarize the framework in Section IV.B. We then propose a new solution that is more suitable for MIPv6 in particular, in Section IV.C.

A. Forwarding Mechanisms from Previous Network

Since the lost packets in simultaneous mobility situations are going to the previous network (associated with the old care-of-address), a natural idea is to install a forwarding mechanism in the previous network. The forwarding mechanism would intercept packets destined to the old care-of-address and forward them to the new care-of-address.

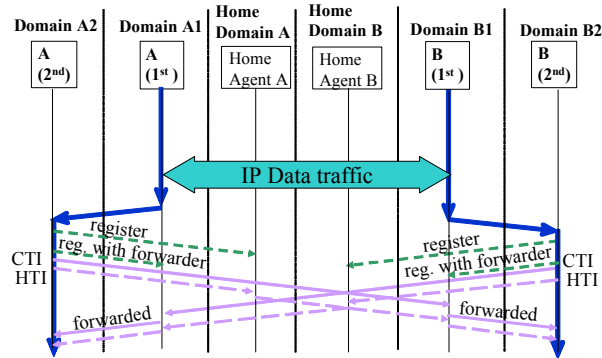


Figure 4: Use of Forwarding Mechanism in Previous Network

Figure 4 shows the scenario from Figure 1 again, except that this time there are forwarders in the previous networks. Thus, the CTI and HTI messages from both sides are successfully conveyed to the correct location. After this (not shown), return routability would be completed. The binding updates would be forwarded in a similar way (also not shown).

Unlike with MIPv4, there are no Foreign Agents in MIPv6, so it is inconvenient to add a forwarding mechanism from the previous network. Either a new network element is needed, or the local router needs to take on this functionality.

B. Stationary Proxies

We have previously introduced a generalized solution for the simultaneous mobility problem, and applied it to SIP mobility and Mobile IP with Location Registers [3]. The solution introduced two abstract concepts, namely *stationary binding update proxies* and *stationary location proxies*. Each MN has both these proxies. The stationary binding update proxy is responsible for obtaining the latest location of the MN's CNs. It has an advantage over the MN in doing this, in that it is at a fixed location. The stationary location proxy, on the other hand, is the dual of the stationary binding update proxy. It is responsible for maintaining the latest location of the MN itself, and delivering such information to the MN's CNs (or the stationary binding update proxies of the MN's CNs). The stationary proxies are abstract in the sense that they need not be separate network elements, but are roles that can be assumed by existing network elements.

The generalized solution was designed for mobility protocols where the process of informing the CN of a new IP address was simply through the sending of an update message, e.g., SIP re-INVITE or MIP-LR registration. This is simpler than with MIPv6, which requires a return routability procedure before the binding update can be sent. Furthermore, to be effective, it needs the stationary binding update proxy to be able to pro-actively issue binding updates to CNs as soon as it obtains the latest location information from the CNs' stationary location proxies. In the case of MIPv6, the binding update must be authenticated, so we cannot simply put the stationary binding update proxy at the home agent and expect it to send home agent-initiated binding updates to CNs.

Thus, in order to use stationary proxies with MIPv6, additional modifications are needed to MIPv6. For example, as shown in Figure 5, A may need to send its Care-of Test Init (or binding update later) to B through A's home agent (step 1). The home agent forwards the message to B (step 2), but keeps a copy of it for a short time, T_h . A's home agent, acting as A's stationary binding update proxy, then queries B's home agent (which is B's stationary location proxy). B's home agent sends A the latest address it has for B, and waits for a short time, T_p . There are two cases; case (a): during this waiting period, B registers a new care-of-address with B's home agent (optional step 4). Acting like the faithful stationary location proxy that it is, B's home agent updates A's home agent with the latest care-of-address (optional step 5); case (b): during the waiting period, B's home agent receives no new updates from B. In either case, A's home agent then has B's latest address, and if it is new, it can send to that address the message that it had kept (optional step 6; of course, the destination address would be changed to the new address).

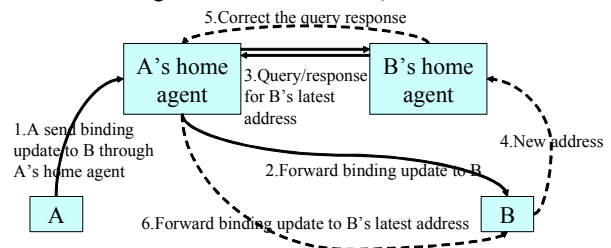


Figure 5: Solution With Stationary Proxies

The selection of the waiting periods, T_h and T_p , should be based on reasonable estimates of the appropriate signaling and computational delays for the network in question. Clearly also, we need $T_h > T_p$, otherwise by the time B's home agent informs A's home agent about B's latest address, A's home agent may no longer have the binding update to send.

C. Signaling to Home Address

We now introduce a more efficient solution. When A is trying to send an update to B, we have seen how A's home agent can be A's stationary binding update proxy for query B's home agent (which acts as B's stationary location proxy) for B's location. However, the query/response functionality that would be added would be a significant addition to home agent functionality. In their normal operations, home agents are not queried for the location of MNs they serve – they only forward packets to the latest known address of those MNs. Thus, in this section, we propose a solution that is more MIPv6-centric and does not involve query/response messages between home agents.

Unlike the solution with stationary proxies, A's home agent does not have to be involved. A just sends control messages directly to B's *home address* (not care-of address, which may be out-dated in the case of simultaneous mobility). The home test init, care-of test init and binding update messages would be the affected messages. This is step 1 in Figure 6, which shows binding update, but the same applies also to home test init and care-of test init. Thus, B's home agent will take care of forwarding the binding update to B; this is anyway the home agent's normal function (step 2). However, it is possible that B may move before the binding update arrives. B then updates its home agent with its new address (step 3, part of the normal operation of an MN). The difference from regular MIPv6 is in step 4, where B's home agent re-forwards A's binding update, this time to the latest location of B. This smart re-forwarding does away with the simultaneous mobility problem.

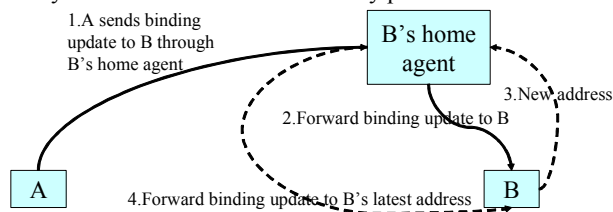


Figure 6: Solution with Signaling to Home Address

What modifications are needed to a home agent to support smart re-forwarding? It would need to not just blindly forward all traffic from CNs to B, but to scan the traffic for binding updates. Whenever a control message (binding update, home test init or care-of test init) arrives, the home agent keeps a temporary copy of it for a short period of time, T_b , after which it is discarded. The identity of the target node (B, in this case), is also stored. Whenever a binding update arrives at the home agent from one of its own MNs like B, it checks if there are any control messages in temporary storage that are destined to B. If there are (e.g., from A), then they can be forwarded

(step 4). What might be an appropriate value for T_b ? It should clearly be large enough to take care of the following case: A moves first, and the control message reaches *almost* all the way to B's care-of address before B moves; B then obtains a new care-of address at its new location and sends a binding update to its home agent. Thus, T_b should take into account the two-way latency from B's home agent to B, plus the time for B to move and obtain a new care-of address.

The modifications to the MNs are also simple. For control signaling only, where the messages (home test init, care-of test init, and binding update) would be sent to the last-known care-of address, they would be sent to the home address. Thus, a slight disadvantage is a possible minor increase in latency in the control signaling only.

Supposing only one of the two home agents implements this solution, it will still be useful. For example, if only A's home agent implements this feature, B's binding updates will always reach A correctly. A can then resend its own binding update to B's latest address, so both nodes will have the correct latest addresses.

We can think of this solution as keeping to the spirit of MIPv6 in that CNs don't have to know the current address of an MN; the MN's home agent will take care of the forwarding. The only problem is the small interval when the home agent does not yet have the latest care-of address of the MN. With just a slight modification to the home agent to deal with that problem, the stationary binding update proxy becomes unnecessary.

V. CONCLUSIONS

We have seen that MIPv6 has some serious problems with simultaneous mobility. MIPv4 did not have problems with simultaneous mobility, but when route optimization was introduced with MIPv6, it brought with it the unintended consequence of problems with simultaneous mobility. We introduced and discussed three possible solutions in this paper. In comparing the solutions, we found that the third solution, "signaling to the home address" is the best. It is efficient and keeps the spirit and flavor of MIPv6, requiring only small modifications to home agents and MNs.

REFERENCES

- [1] C. Perkins et al., "IP Mobility Support for IPv4," IETF RFC 3344, August 2002.
- [2] D. Johnson, C. Perkins and J. Arkko, "Mobility Support in IPv6," IETF RFC 3775, June 2004.
- [3] K.D. Wong, A. Dutta, K. Young and H. Schulzrinne. On Handling Simultaneous Mobility. IEEE Milcom 2003, Boston, MA, USA, October 2003.