Experimental Performance Evaluation of Mobile IPv6 Handovers over Wireless LAN

Jong-Hyouk Lee and Tai-Myoung Chung
Internet Management Technology Laboratory,
Dept. of Computer Engineering, Sungkyunkwan University,
300 Cheoncheon-dong, Jangan-gu, Suwon-si, Gyeonggi-do, 440-746, Korea
{jhlee, tmchung}@imtl.skku.ac.kr

Abstract— **Over the past several years there has been increasing the number of mobile devices. Mobile IPv6 enables mobile devices to communicate with each other while moving and has been implemented by several organizations. Though Mobile IPv6 implementations has been confirmed, the performance evaluation is one of the important next step to expect deployment in real environments. In this paper, we present experimental results on the performance of Mobile IPv6 handovers with the test parameters such as RTT, TCP and UDP throughput.**

I. Introduction

With the rapid growth of mobile technologies and demands of mobile users for ubiquitous service, Mobile IPv6 (MIPv6) has been become more important. MIPv6 proposed by the Internet Engineering Task Force (IETF) allows a mobile node (MN) to change its location without restarting its applications and without disrupting any ongoing communications [1]. In MIPv6, When a MN changes its point of attachment to the network, it moves from one network to another new network. This process is known as handover [1], [2]. During handover, the MN usually has disconnected from the old network before connecting to the new network and thus there is a time when the MN has lost connectivity to the network. During this period it cannot send or receive data to the existing application sessions. While many TCP applications are designed to cope with intermittent loss of connectivity by retransmitting unacknowledged packets, UDP applications will not be able to recover such losses. Furthermore, both TCP and UDP applications that rely on timely packet delivery within certain acceptable thresholds (e.g. VoIP and audio/video streaming applications) will be sensitive to the length of time the MN loses connectivity while performing handover [3].

We cannot expect from people to use an implementation as long as it has not been proved that it is complying with the MIPv6 specification and competitive enough to support high loads. Due to those reasons, we do an experimental performance evaluation of handover on implementation of MIPv6. With this performance evaluation, we can know how long time a MN loses the connectivity with the network.

The rest of this paper is organized as follows. In section 2, we describe Mobile IPv6 and MIPL. In section 3, we introduce our network topology, equipments, test tools, and test parameters to obtain the performance evaluation results. In section 4, we define the scenarios and evaluate the performance of MIPv6 handovers with the scenarios and then we finally give our conclusions and future work in section 5.

II. Background and Motivation

A. Mobile IPv6

MIPv6 defines a method that allows nodes to change their point of attachment to the network while not breaking existing application sessions. This is achieved primarily through the MN always being reachable at its home address (HoA) via its home agent (HA). The problem of triangular routing can be eliminated in Mobile IPv6 with the route optimization and MIPv6 does not need Foreign Agent (FA), which is used for Mobile IPv4 (MIPv4) [4]. In route optimization, the MN sends a Binding Update (BU) to its Correspondent Node (CN) to inform the CN of its current location. Once the BU has been acknowledged by the CN, communication can continue on a direct path without the triangular routing problem [1], [2], [5].

B. MIPL

There are few implementation projects of Mobile IPv6 such as MIPL, Kame, and Monarch project [6], [7], [8]. We choose MIPL due to MIPL has been released under GPL and it is available to anyone for free [9]. Moreover, this implementation is for Linux has lots of useful performance evaluation tools and user groups. MIPL stands for Mobile IPv6 for Linux. Mobility support allows a MN to be tracked as it migrates between networks or even ISPs, allowing packets to be forwarded to where the MN is currently located. Since MIPL release 2.0.1, it works under Linux kernel 2.6.x [10]. So our test-beds were installed Linux kernel 2.6.15. The main features of MIPL release 2.0.1 has following:

- MIPL supports MIPv6 specification and IPsec as specified in RFC3775 (Mobility Support in IPv6) and RFC3776 (Using IPsec to protect MIPv6 signaling between MNs and HAs)
- A router advertisement daemon (radvd) is included.
- Binding request works find and with different MIPv6 implementations.
- Home Agents list can be added in the MN and the HA.

Fig. 1. The network topology of Mobile IPv6 test-bed

- Dynamic Home Agent Address Discovery supported in the MN and the HA.
- ICMP destination unreachable messages are taken into account in Binding Cache and Binding Update List.
- Tunneling from previous CoA if previous router had the H-bit on in its router advertisements.
- The package consists of a kernel patch for a particular kernel version and user space programs. It means MIPL has only absolutely necessary infrastructural support for MIPv6 in the kernel and moved most functionality to user space.

III. Mobile IPv6 Test-bed

In this section, we introduce our MIPv6 test-bed based on network topology, equipments, testing tools, and test parameters.

A. Network Topology

Fig. 1 shows our MIPv6 test-bed used for performing the handover tests. A MN running MIPL v2.0 is away from home network (HN) and can attach to one of two networks represented by the SSIDs, IMTL_AP1 and IMTL_AP2, respectively. There are two networks. Vessel and valkyrie exist in the same link (2001:0220:1404:2::/64), which is foreign network (FN). Vessel running MIPL v2.0 is the access router (AR) of valkyrie, which is one of CN. In 2001:0220:1404:3::/64, siege-tank running MIPL v2.0 and vulture are exist. Siege-tank does as the AR or the HA of marine which is the MN. Valkyrie and vulture are normal fixed nodes and can do as CNs in their links. Vessel and siege-tank have connected the same link (2001:0220:1404::/128). Finally, we have the MN (marine) in 2001:0220:1404:3::/64.

B. Equipments

For simplicity, all ARs are installed HA module and radvd daemon of MIPL v2.0 on Linux kernel 2.6.15 and all fixed nodes are installed CN module of MIPL v2.0 on different Linux kernels. All ARs and fixed nodes have 100Base-T (Fast-Ethernet) card and they are connected 100 Mbps links. Also, our MN have MN module on Linux kernel 2.6.15. To make wireless communication, we deploy two access points (APs). Each AP is connected the AR in each link. Therefore, the MN can receive the Router Advertisement (RA) message of ARs through the APs.

C. Test Tools

In order to run and analyze MIPv6 features, test tools are needed that allow to generate network traffic and to monitor the throughput performance of the network. Tools like IPv6 enabled FTP and ping6 have basically a problem to do the network performance measurements. For instance, they include in their measurements the buffering at the user terminals, therefore including the end terminal performance in the total measurement [11]. So we must need to introduce external dedicated elements to avoid this inconvenient before doing the network performance measurements, even though we use ping6 to measure the simple test such as RTT.

To get precise measurement, we use Network Protocol Independent Performance Evaluation (NetPIPE) tool [12]. This tool focuses on application-oriented performance measurement. With the application performance in mind, what one wants to know in regard to network communication in not the available throughput but, for instance, the optimal message size that gives the best application performance [11]. Another performance evaluation objective could be to observe the network congestion level with a particular message size.

D. Test Parameters

The performance comparison quantitatively evaluates the overhead added by MIPL implementation. In this paper, performances are based on the following parameters:

- Round Trip Time (RTT): This is a measure of the delay on a network between two nodes. In this test, lower RTT means better processing time on the nodes in the routing path.
- UDP and TCP throughput: In this test, these are the highest throughput that a mobile node can reach by forwarding UDP datagrams or TCP packets higher throughput means better performance.
- Handover Latency: With this measurement, we can know how long time a MN loses the connectivity with the network.

IV. Results of Experimental Performance Evaluation

To measure the experimental performance of MIPv6 implementation, operating scenarios are required to analyze various aspects separately.

A. Scenario 1 (Mobility Disabled)

All nodes are fixed nodes. When the MN (marine) is on its Home Network (HN), it uses conventional routing functions to exchange IP packets with its CN (valkyrie). Therefore, while the MN is linked on 2001:0220:1404:3::/64, it behaves like any fixed node.

Round Trip Time

The aim of this test is to compare the Round Trip Time (RTT) between the MN (marine) and the CN (valkyrie) when the MN is acting as a fixed node. To calculate the RTT, we use the ping6 command on the MN with the CN

TABLE I

AVERAGE RTT WITH DIFFERENT DATA SIZE

64 bytes	128 bytes	1024 bytes	2048 bytes
4.579 (msec)	3.147 (msec)	6.320 (msec)	10.289 (msec)

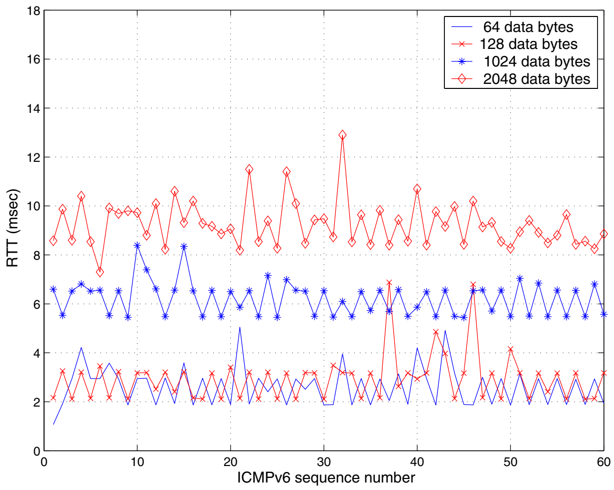

Fig. 2. RTT with different data size for scenario 1

as the destination. We increased step-by-step the payload size (with the -s option of ping6) until we exceeded the Path MTU (Maximum Transmission Unit). The average time of the RTT shows in Table I. Each case, the MN transmitted 60 packets and received 60 packets. There was no packet loss.

As we can see Table I and Fig. 2, when the MN sends ICMP messages of 128 data bytes, the average time shows the best performance. Moreover, we can see the round trip time increases when the message size set to 1024 bytes or above.

TCP throughput

In this test, we measure the TCP throughput with different window size and observe the TCP throughput for 360 seconds between the MN (marine) and the CN (valkyrie). In one of this test, different TCP window size was configured. All results are illustrated in Figure 3 (a). When the TCP window size was set to 128 Kbytes, the MN could send the maximum TCP throughput about 4.9 Mbps. In another test, we can see the TCP throughput with the default window size (16 Kbytes) for 360 seconds. All results are illustrated in Figure 3 (b). The MN transfers TCP packets of 192 MBytes to the CN on 4.48 Mbps.

UDP throughput

In this test, we measure the UDP throughput. Firstly, to know the real UDP throughput, we are sent the UDP datagrams from MN (marine) to CN (valkyrie). We thus increase step-by-step the throughput of the UDP flow sent from the MN to the CN until we reach the maximum limit. The limit is known when the rate used to send the UDP datagrams from

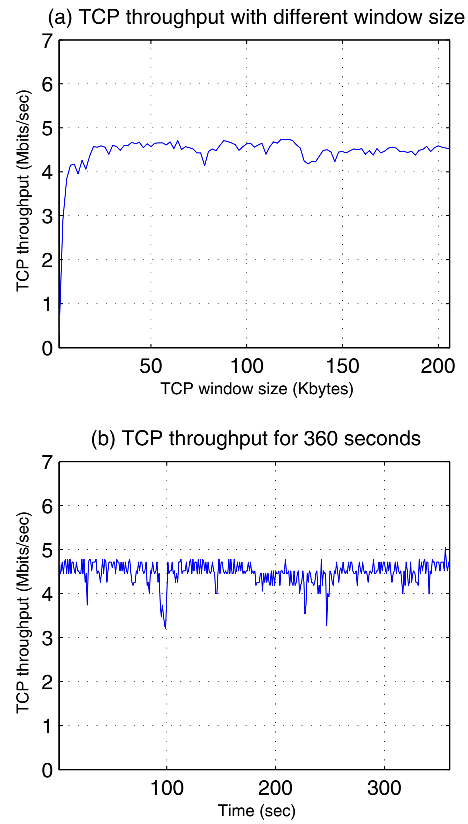

Fig. 3. TCP throughput for scenario 1

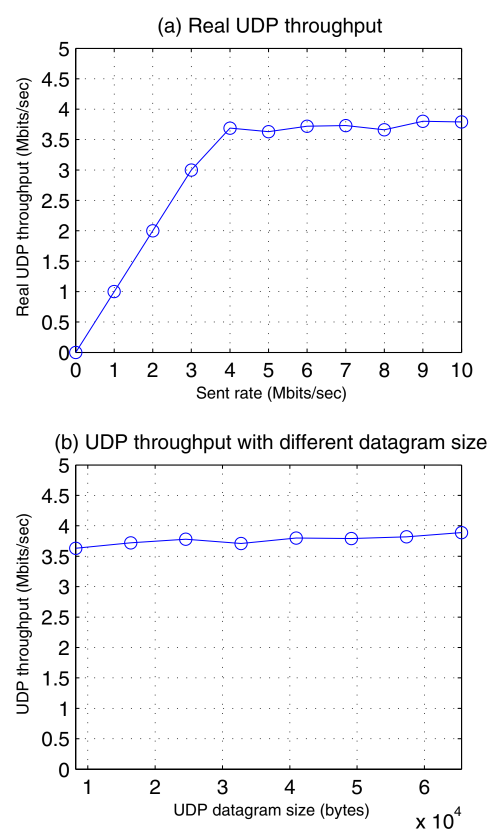

Fig. 4. UDP throughput for scenario 1

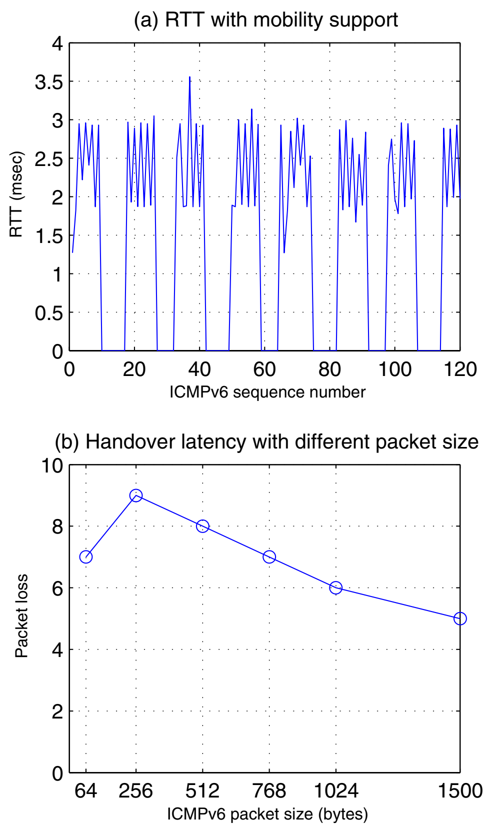

Fig. 5. Handover latency for scenario 2

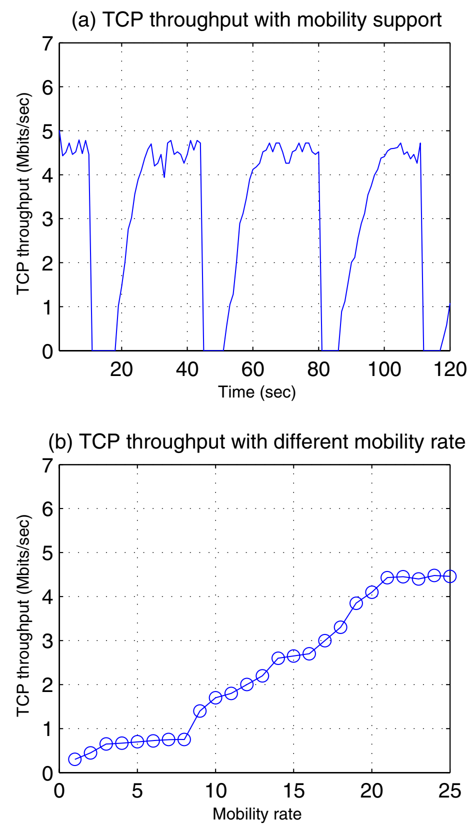

Fig. 6. TCP throughput for scenario 2

the NetPIPE client is higher than the rate at which those datagrams are received on the NetPIPE server. The result shows in Fig. 4 (a). As we can see in Fig. 4 (a), the maximum UDP throughput is under 4Mbps even though the maximum TCP throughput is about 4.9 Mbps. Fig. 4 (b) shows us the result of the UDP throughput with different datagram size. We can see the different datagram size does not have an effect much on the UDP throughput.

B. Scenario 2 (Mobility Enabled)

The configuration is the same as that of scenario 1, except that the MN (marine) is moved between the HN and the FN.

Handover Latency

The goal of this test is to know the length of time the MN (marine) loses connectivity while performing handover. The packet loss was evaluated by ICMPv6 packets. Fig. 5 (a) illustrates that about 6-8 packets were lost during the handover with the 64 bytes payload size. By default, the ping6 waits for one second between each packet, enabling the system handover latency to be measured. In Fig. 5 (b), we can see the result of handover latency with different ICMPv6 packet size. We increased step-by-step the payload size until 1500 Kbytes. The lost packets measured 7, 9, 8, 7, 6, and 5, respectively. Thus, increasing the packet size in MIPL can improve the performance of the handover latency.

TCP throughput

In this test, we measure the TCP throughput for 120 seconds when the MN (marine) performs the handover and observe the TCP throughput with different mobility rate. Fig. 6 (a) illustrates the TCP throughput when the MN moves between the HN and the FN. As we can see, the TCP throughput stabilizes slowly after 10-20 seconds when the MN performs the handover. It means that TCP congestion control mechanism needs the adaptations for mobile environments. In Fig. 6 (b), we can see the result of the TCP throughput with different mobility rate. The TCP throughput was increased when the mobility rate between 9 and 21. Especially, the TCP throughput stabilized at about 4.8 Mbps when the mobility rate was set to 21. Therefore, the throughput of MIPL can be improved to 4.8 Mbps when the time period between movements is set to 21 or above. In practical case, however, the time period required to reach TCP throughput stabilization seems to take long time.

UDP throughput

In this test, we measure the UDP throughput for 120 seconds when the MN (marine) performs the handover and observe the TCP throughput with different mobility rate as like the test of the TCP throughput for scenario 2. However, the results of this test is slightly different from the test of the TCP throughput due to the characteristic of TCP and

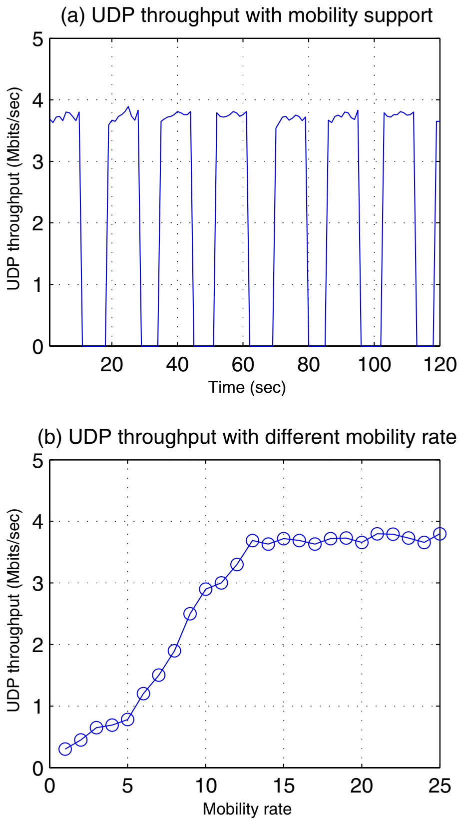

Fig. 7. UDP throughput for scenario 2

UDP. Fig. 7 (a) illustrates the UDP throughput when the MN moves between the HN and the FN. As we can see the UDP throughput stabilized after 3-7 seconds in a different way from the UDP throughput in the handover process. Also, Fig. 7 (b) shows the result of the TCP throughput with different mobility rate. The UDP throughput was increased when the mobility rate between 5 and 13. Especially, the TCP throughput stabilized at about 3.85 Mbps when the mobility rate was set to 13. So, the UDP throughput stabilizes faster than the TCP throughput. It means we should choose UDP protocol when we want to transport the short amount of data in performing frequent handovers.

V. Conclusions and Future work

Mobile IPv6 is becoming a major component in next generation networks and its performance evaluation is one of the important next step to expect deployment in real environments. This paper presents the tests, which are mainly evaluated the handover performance, performed against MIPL and analyzes the results. In order to improve and develop those results, more tests need to be done in the future, such as QoS guarantees using RSVP with location registers and AAA service in mobile environments.

VI. Acknowledgments

This study was supported by a grant of the Korea Health 21 R&D Project, Ministry of Health & Welfare, Republic of Korea. (02-PJ3-PG6-EV08-0001)

References

[1] D. Johnson, C. Perkins, and J. Arkko, "Mobility Support in IPv6", RFC 3775, June 2004.
[2] Jong-Hyouk Lee and Tai-Myoung Chung, "Distributing Multiple Home Agents in MIPv6 Networks", High Performance Computing and Communications (HPCC) 2005, LNCS 3726, pp. 991-999, September 2005.
[3] M. Dunmore, "Mobile IPv6 Handovers: Performance Analysis and Evaluation", WP4 Report in 6NET, June 2005.
[4] C. Perkins, "Mobility Support in IPv4", RFC 3344, August 2002.
[5] Jong-Hyouk Lee, Young-Ju Han, Hyung-Jin Lim and Tai-Myung Chung, "New Binding Update Method Using GDMHA in Hierarchical Mobile IPv6", International Conference on Computational Science and its Applications (ICCSA) 2005, LNCS 3480, pp. 146-155, May 2005.
[6] MIPL Project, http://www.mobile-ipv6.org, Accessed on April 17 2006.
[7] Kame Project, http://www.kame.net, Accessed on April 17 2006.
[8] Monarch Project, http://www.monarch.cs.cmu.edu, Accessed on April 17 2006.
[9] GNU General Public License, http://www.gnu.org/copyleft/gpl.html, Accessed on April 17 2006.
[10] The Linux Kernel Archives, http://www.kernel.org, Accessed on April 17 2006.
[11] M. Hassan, and R. Jain, "High Performance TCP/IP Networking: Cencepts, Issues, and Solutions", Prentice Hall, October 2003.
[12] The official site for NetPIPE, http://www.scl.ameslab.gov/netpipe, Accessed on April 17 2006.