Handoff Delay Analysis and Measurement for SIP based mobility in IPv6

Nobuyasu Nakajima
Toshiba America Research, Inc.
P.O.Box 136
Convent Station, NJ 07961, USA

Ashutosh Dutta and Subir Das
Telcordia Technologies, Inc.
445 South Street
Morristown, NJ 07960, USA

Henning Schulzrinne
Columbia University
450 Computer Science Building
New York, NY 10027, USA

Abstract— **This paper describes the Session Initiation Protocol (SIP) based mobility in IPv6 and its performance in our laboratory testbed. For real-time mobile multimedia communication, we use SIP for signaling protocol as well as for supporting terminal mobility. While performance study for real-time mobile communication refers to several factors and their measurements, we analyze here only the handoff delay due to a node mobility. In particular, we are interested to examine the delay incurred when a mobile node moves to a new network and perform the Duplicate Address Detection (DAD) and router selection. We also analyze the delay in each cases. We notice that the IPv6 Linux implementation provides substantial amount of delay during node movement which severely affects the performance of real-time applications. However, we modify the Linux kernel and compare with the unaltered one. Finally, we conclude that a faster handoff is achievable. with intelligent modifications to Linux kernel.**

I. INTRODUCTION

The proliferation of wireless devices along with the rapid growth of the Internet is demanding the Internet community to move from Internet Protocol version 4 (IPv4) [1] to Internet Protocol version 6 (IPv6) [2]. The major motivation behind this is the limitation of IPv4 address space. Although Network Address Translation (NAT) is widely used to circumvent the address space problem, it fails to provide the global routability. IPv6, on the other hand, is designed to solve such problems. Expanded address space of IPv6 will enable us to assign global routable IP addresses to every possible device willing to connect to the Internet. Moreover, simplification of the IPv6 header with respect to options, explicit labeling of IPv6 traffic flows for special handling, and better support for authentication, data integrity and confidentiality features make IPv6 more attractive.

Commercial and non-commercial IPv6-based Internet services are becoming popular. Many standard bodies are considering IPv6 for their next generation networks and services. For example, 3GPP mandates IPv6 for IP Multimedia Subsystem (IMS) including Voice over IP (VoIP) in Release 5 [3]. Operating systems, routers, and other network elements are starting to support IPv6. It is anticipated that all future wireless devices will have a built-in IPv6 stack. Therefore, wireless service providers are interested in IPv6 based services. Apart from data communication, such as file transfer and web browsing, providers are more interested in offering real-time services, such as voice and video. However, careful investigations are needed regarding the IPv6 performance issues for real-time communication since it has different features than IPv4. This paper addresses such issues and measure the performance of real-time multimedia communication in our IPv6 laboratory testbed.

Although several documents such as [4] have already mentioned drawback of Duplicate Address Detection (DAD) [5] in the handoff delay, performance measurement is hardly available. Flykt and Alakoski [6] discuss an IPv6-based mobile communication testbed using the Session Initiation Protocol (SIP) [7] without providing much analysis and performance measurements. For the purpose of performance measurement, we built our wireless IPv6 testbed in the laboratory. We integrate both SIP and Mobile IP to measure the performance of real-time mobile communication. Our experiments include measurement of various delays, disruptions, packet loss, and media quality. In this paper, however, we give emphasis only on handoff delay due to Duplicate Address Detection (DAD), and router selection performed by the mobile node while visiting a new network and using SIP for mobility.

The rest of the paper is organized as follows: Section II presents the SIP based terminal mobility and Section III describes our testbed configuration. While Section IV deals with the handoff delay caused by DAD and router selection, Section V gives the overview of SIP mobility implementation in IPv6. Section VI presents the analysis and measurements of handoff delay and Section VII concludes the paper.

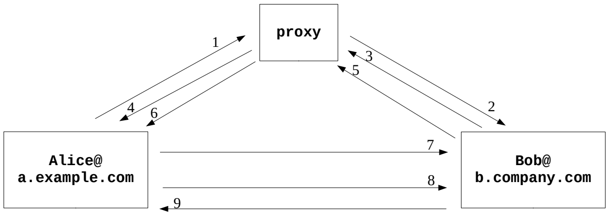
II. TERMINAL MOBILITY USING SIP

SIP is a protocol for establishing and tearing down multimedia sessions. SIP can also support various types of mobility such as terminal mobility, session mobility, personal mobility, and service mobility [8]. Since our focus in this paper is on terminal mobility we describe the SIP terminal mobility after briefly mentioning the SIP basics in the following subsection.

A. SIP Basics

SIP is a protocol for establishing a multimedia session. Fig. 1 shows a basic procedure of session initiation and disconnection using SIP.

In the example shown in Fig. 1, a call is initiated by `Alice` at `a.example.com` to `Bob` at `b.company.com`. Suppose `Bob`'s SIP URL is `Bob@company.com` and proxy



```
1. INVITE sip:Bob@company.com
2. INVITE sip:Bob@b.company.com
3. 180 Ringing
4. 180 Ringing
5. 200 OK
6. 200 OK
7. ACK sip:Bob@b.company.com
8. BYE sip:Bob@b.company.com
9. 200 OK
```

Fig. 1. SIP basic procedure

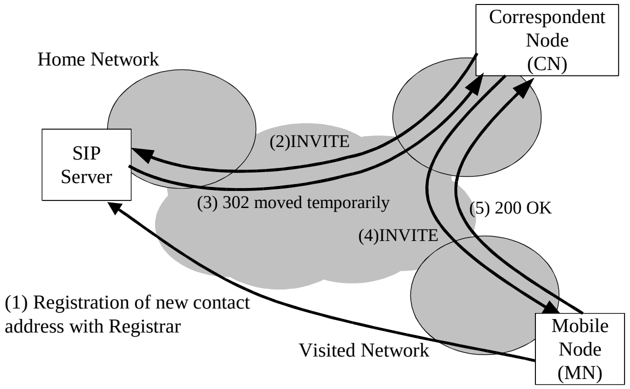


Fig. 2. An example of pre-call mobility by SIP

in this figure has a location information for `Bob`, which is `b.company.com`. A session is initiated by using this proxy as shown in Fig. 1; `Alice` sends an `INVITE` request to `proxy` (1), and the `proxy` relays the `INVITE` request to `Bob` at `b.company.com` (2). While `Alice` is waiting for `Bob` to answer to the call, `180 Ringing` is sent back to `Alice` via proxy (3,4). After `Bob` answers to the call, `200 OK` response is sent back to `Alice` to notify that `Bob` accepts the call (5,6). Then `Alice` sends `ACK` (7) to `Bob` to confirm a session establishment. Now `Alice` knows the contact address of `Bob`, `Bob@b.company.com`, via `200 OK` and `Alice` can send `ACK` to `Bob` directly without going through the `proxy`. When `Alice` wants to terminate the call, `Alice` sends a `BYE` request to `Bob` (8) which is confirmed by another `200 OK` (9) issued from `Bob`.

B. *SIP Terminal Mobility in IPv6*

Reference [8] discussed how SIP can be used to support terminal mobility and its advantages over other mobility protocols. It is important to note that SIP-based terminal mobility does not add extra bytes to base SIP in order to support mobility. For the completeness of the paper, we illustrate here briefly the pre-call mobility and mid-call mobility with Figures 2 and 3.

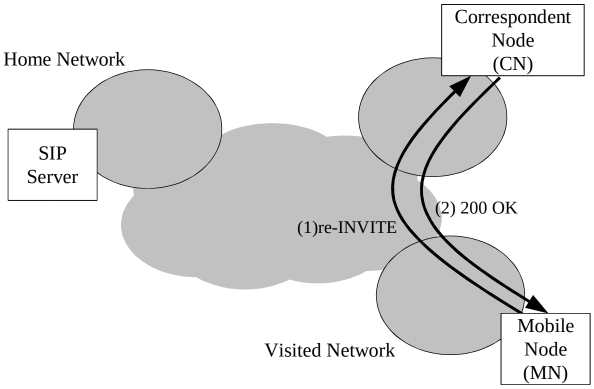


Fig. 3. An example of mid-call mobility by SIP

Pre-call mobility guarantees that a Correspondent Node (CN) can reach a Mobile Node (MN). Fig. 2 shows an example of how pre-call mobility is supported by SIP. In this example, an MN registers its new contact address with the SIP Server (1). When a CN sends an `INVITE` request to the SIP Server (2), SIP Server notifies MN's new contact address (3) to CN. CN then sends an `INVITE` request to the MN directly (4) and receives a `200 OK` from MN (5).

Mid-call mobility allows a node to continue an ongoing session with its peer during handoff. Fig. 3 shows an example of how mid-call mobility is supported by SIP. In this example, an MN sends a re-`INVITE` request with new IP address to CN (1), and CN directly sends packets to MN at the new point of attachment to the network (2). In this paper, we measure the handoff delay during mid-call mobility only.

III. TESTBED FOR REAL-TIME MOBILE COMMUNICATION USING IPV6

We extended our existing IPv4-based multimedia testbed [9] to support IPv6-based mobile multimedia communication. Based on IEEE 802.11b wireless LAN, our IPv6 testbed presently supports stateless address autoconfiguration [5], SIP, and Mobile IPv6.

Fig. 4 shows the configuration of our experimental IPv6 testbed. It consists of two routers (Router A and B), an MN, and a CN. Router A has two ethernet segments: one for the home network and another for the Router B. Router B has three ethernet segments; two for visited networks and one for the connection to Router A.

We use Linux 2.4.9 kernel in our testbed. Additionally, a patch for better conformance with IPv6 specification developed by USAGI projects [10] is applied to the kernel. An MN uses stateless address autoconfiguration for its IPv6 address configuration.

Both MN and CN use SIP as their signaling protocol when they establish a voice over IP session between them. We use Columbia University's SIP User Agent (UA) implementation [11] in our testbed and modified it to comply with IPv6 specification. We also use RAT [12], which enables voice communication over IPv4 and IPv6.

IV. HANDOFF DELAY IN MID-CALL SIP MOBILITY

A. *Components of Handoff Delay*

Fig. 5 depicts the message flow during handoff process and its delay components in detail. We define handoff delay (D)

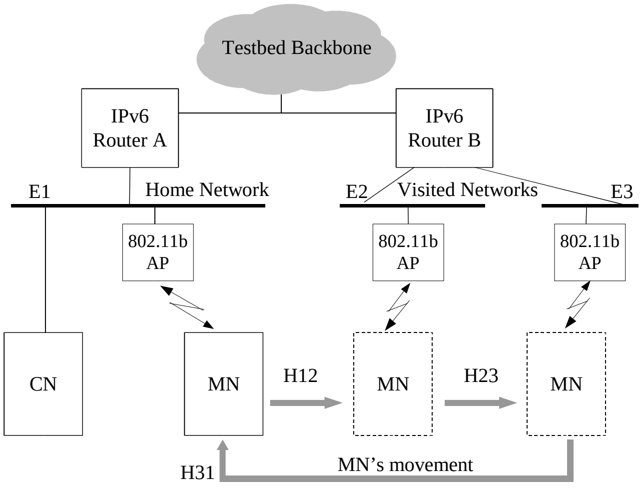


Fig. 4. Experimental testbed configuration

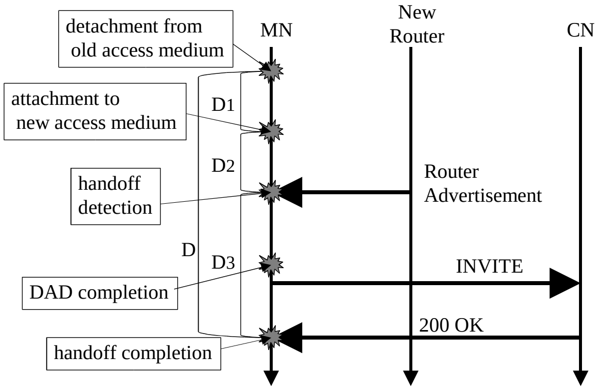


Fig. 5. handoff flow

as the delay when an MN changes its location and attaches to a new subnet so that it is capable of communicating with a CN. It essentially consists of three components: delay for switching lower layer medium to access network (D_1), delay for detecting a new router and a new IP subnet (D_2), and SIP mobility delay (D_3).

In our testbed, D_1 is 802.11b access point switching delay. In IPv6, after switching the access point, an MN can detect a new access router by listening to Router Advertisement (RA). An access router periodically sends unsolicited RAs as well as sends solicited RAs when it receives Router Solicitation (RS). In our testbed, an MN waits for periodical RA. By listening to a new RA, an MN determines that the IP subnet has changed. The maximum value of D_2 can be one RA interval in our testbed. After receiving a new RA, an MN configures its interface with the new IPv6 address using stateless address autoconfiguration. D_3 is the time between a re-`INVITE` message sent by an MN and `200 OK` response sent by the CN.

Although D_1 and D_2 are equally important to the delay performance in a real-time mobile communication, we concentrate our attention on D_3 in this paper. We believe that D_1 is specific to link layer technology and can be reduced. For example, D_1 can be considered zero for link layer technologies supporting soft handoff. D_2 could be large in a bandwidth constraint environment where MN depends upon RAs for subnet detection. On the other hand, if an MN is equipped with a technique to send RS which is triggered by the lower layer, D_2 can also be reduced. In that case, D_2 is the time between an RS sent by an MN and an RA sent by a new access router. In our testbed, however, D_2 is the average interval of RAs, because such trigger is not yet implemented in our environment.

There are two main factors which contribute to delay D_3; Duplicate Address Detection (DAD) and router selection. DAD imposes delay between receiving an RA and sending packets out of the interface with new IPv6 address. Router selection also adds delay before selecting proper access router. Following subsections describe these methods and estimate the delay.

B. *Duplicate Address Detection*

The purpose of Duplicate Address Detection (DAD) is to confirm the uniqueness of the IPv6 address on the link. When an MN uses stateless address autoconfiguration to assign a new IPv6 address to its interface after getting a new RA, it sends a Neighbor Solicitation (NS) on the local link in order to verify whether any other node on the link has the same address. When the pre-determined time elapses and the MN does not receive any reply, the MN assumes that no other node on the link has this address and finally assigns this to its interface as a valid address. This new address is called a tentative address during the DAD process.

According to [5], a tentative address is not allowed to be used by a node. This means that the MN cannot send packets with a tentative address as a source IPv6 address and has to discard all inbound packets during DAD phase. This causes additional delay in sending a re-`INVITE` in case of SIP.

Fig. 6 depicts the delay caused by DAD. A random delay D_{rand} between 0 to a certain maximum value ($D_{rand,max}$) has been chosen and is applied before sending the NS. We also denote the number of transmissions of NS and the interval of the transmission of the two consecutive NSes as N and D_{ret}, respectively. These variables are configurable kernel parameters. The average delay caused by DAD is therefore

$$\overline{D_{DAD}} = \overline{D_{rand}} + N \cdot D_{ret}, \qquad (1)$$

in which $\overline{D_{DAD}}$ and $\overline{D_{rand}}$ denote the average of D_{DAD} and D_{rand}, respectively. In our testbed, $D_{rand,max}$ is 1000 ms, N is 1, and D_{ret} is 1000 ms, which are also default values in [5]. Since D_{rand} is uniformly distributed,

$$\overline{D_{DAD}} = 1500 \text{ ms}. \qquad (2)$$

We can reduce the average handoff delay by 1500 ms if we avoid DAD process.

C. *Router Selection*

Router selection is an important component for faster handoff. According to RFC 2461 [13], a host needs to perform

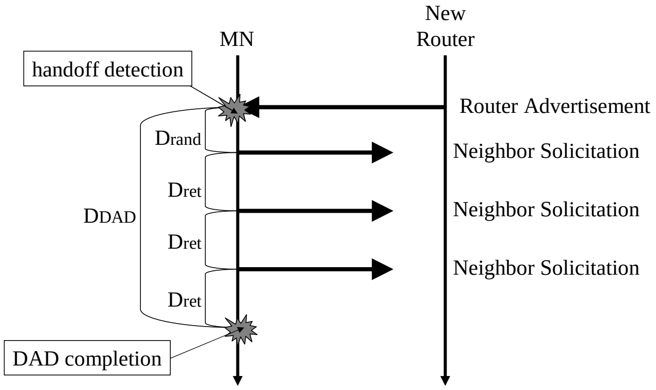


Fig. 6. DAD

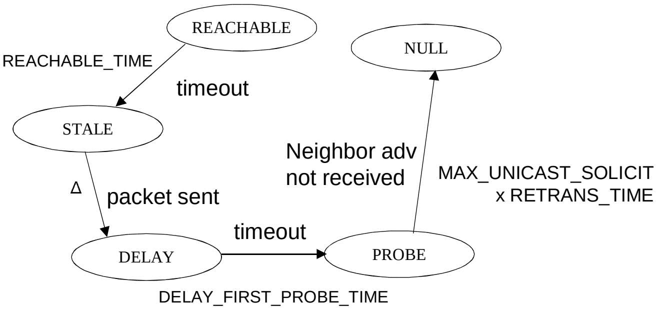


Fig. 7. Neighbor's State Transition

certain steps before switching to another access router. Following paragraph describes one such procedure, Neighbor Unreachability Detection (NUD) and estimates the delay.

1) Neighbor Unreachability Detection: In IPv6, before a host decides that its neighbor is unreachable, host uses Neighbor Unreachability Detection (NUD) to confirm the unreachability of its neighbor according to RFC 2461 [13]. In IPv6, neighbor is another IPv6 host on the same link, and an access router is also an IPv6 neighbor. A host has to start the NUD process to confirm if the old access router is reachable or not during handoff time. Once the host confirms unreachability of the old access router, it switches to a new access router.

In NUD, each neighbor has a reachability state. Part of the state transition is shown in Fig. 7. When a host confirms that a neighbor is reachable, the reachability state of that neighbor is called REACHABLE. Then the host waits for REACHABLE_TIME ms before the state goes to STALE. By receiving an RA it can also go to the STALE state. When the interval of RA is shorter than REACHABLE_TIME, access router's reachability state goes to STALE even if REACHABLE_TIME does not elapse. During the STALE state nothing happens until the host sends new packets. After a host sends a packet, active reachability confirmation starts. During this cycle, the host waits for another DELAY_FIRST_PROBE_TIME seconds and uses Neighbor Solicitation (NS) to confirm reachability with predefined number of retransmissions (MAX_UNICAST_SOLICIT). Once this is over, the reachability state of the neighbor goes to NULL.

The amount of delay introduced by NUD process depends upon the time when an MN sends a packet to old access router and the time when an MN receives a new RA from new access router. This suggests that NUD operation may not introduce any additional delay to D_3 if unreachability is confirmed before the reception of RA, because a transmitted packet triggers the reachability state transition from STALE to DELAY. This can be the case if, for example, an MN keeps sending RTP packets during handoff via the old access router. This old router's reachability state changes from STALE to DELAY and thereby reduces the additional delay caused by NUD. However, if an MN sends a packet after receiving a new RA from new access router, then NUD contributes additional delay to D_3. Fig. 8 shows the latter case.

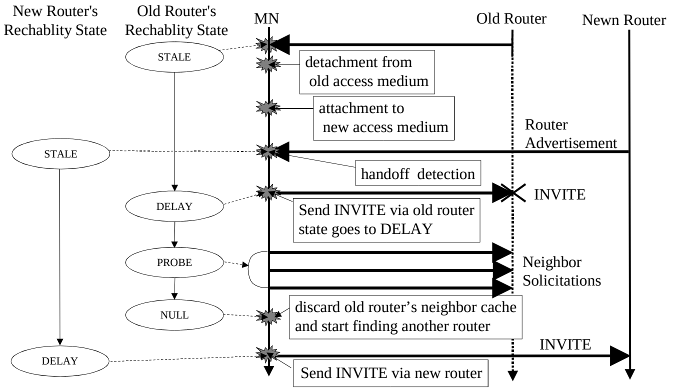


Fig. 8. NUD and handoff delay

The maximum delay introduced by NUD, $D_{nud,max}$, is

$$\begin{aligned} D_{nud,max} \quad \geq \quad & \Delta + \text{DELAY\_FIRST\_PROBE\_TIME} \\ & +\text{MAX\_UNICAST\_SOLICIT} \\ & \quad \times \text{RETRANS\_TIMER} \end{aligned} \tag{3}$$

where Δ shows an additional delay between the time the reachability state goes into STALE and the time a host sends a packet. With default value shown in [13], $D_{nud,max}$ is

$$\Delta + 5\text{s} + 3 \times 1s = \Delta + 8\text{s}. \tag{4}$$

This study shows that NUD with default values may impose more than 8 seconds delay. As seen above, for real-time multimedia such as voice and video, NUD is too conservative.

2) Routing Table Update: To perform the rapid handoff, hosts in an IPv6 environment should attach to the new access router whose RA is most recent. However, commonly used Linux hosts do not always select the new access router. If routing table has other routes, a host may select a different old router. As a result, host performs the NUD to a wrong neighbor (i.e., old router) which eventually adds more delay to the overall selection process.

For this reason, MIPL [14] implements the Mobile IPv6 in different way and provides faster handoff. We investigated the MIPL Mobile IPv6 code and understood the following: although MIPL Mobile IPv6 is a kernel module and works with standard Linux IPv6 stack, MIPL Mobile IPv6 has two features about router administration. i) MIPL Mobile IPv6 has its own router list. This list is not the standard IPv6 routing table and is updated independently. It contains the current access router as well as routers which are not currently selected and whose lifetime do not expire. This lifetime is independent of reachability in the NUD context. ii) The other feature is that MIPL Mobile IPv6 forcibly updates routing table after receiving router advertisement from new access router, and deletes all routing information which does not have the same prefix as the current one. Due to these two features, MIPL Mobile IPv6 always selects new router.

In order to achieve such faster handoff as MIPL Mobile IPv6 provides, we incorporate the movement detection part of MIPL Mobile IPv6 in our implementation as Aggressive Router Selection module. This module enables fast router switching in our testbed.

V. IMPLEMENTATION OF SIP TERMINAL MOBILITY

In order to integrate SIP terminal mobility in IPv6, we modified Columbia SIP UA in MN and CN accordingly. Since handoff is triggered by reception of Router Advertisement (RA) in our current implementation, MN's SIP UA is now equipped with the module for detection of RAs. In addition, MN's SIP UA is modified to send re-`INVITE` message to carry new IP address information to CNs when a SIP UA attaches to a new access router. This is necessary to support the mid-call mobility.

Additionally, CN's SIP UA has to process the re-`INVITE` message from MN, in oder to cope with MN's new address. We modify the SIP UA so that it passes the new IP address information of MN to media program (i.e., RAT in our testbed).

As discussed earlier, MN experiences a substantial amount of delay between detection of handoff and transmission of re-`INVITE` message. In order to keep this delay small, we also modify the Linux kernel and skip the DAD process, reduce the NUD's timer value, and select the new router through aggressive router selection mechanism.

VI. HANDOFF DELAY MEASUREMENT

We have measured the handoff delay of SIP terminal mobility in our IPv6 testbed. Two different scenarios have been considered: (a) SIP mobility without kernel modification; (b) SIP mobility with kernel modification. While IEEE 802.11b access points are already installed in our testbed, we do not use wireless LAN during this measurement.

Table I and Table II show the handoff delay for each mobility scheme and each handoff case, which is labeled as shown in Fig. 4. Although delay of SIP mobility without kernel modification is the average of two times measurement for each handoff case, the other one is the average delay measured 10 times for each handoff case.

Table I shows the signaling delay measured at Mobile Node. In SIP Mobility, this delay is defined as the time from getting new router advertisement to getting `200 OK` from the other party. Table II shows the voice packet delay also measured at Mobile Node. In SIP Mobility, this delay is defined as the time from getting new router advertisement to getting media UDP packet from the other party.

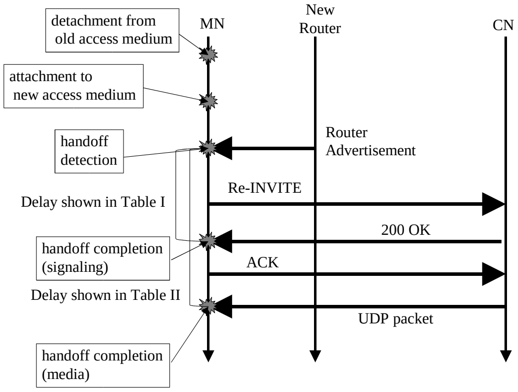


Fig. 9. Handoff flow of SIP mobility in IPv6

TABLE I

HANDOFF DELAY OF SIGNALING

handoff case	(a)	(b)
H12	38290.9 ms	171.4 ms
H23	3932.2 ms	161.6 ms
H31	1934.7 ms	161.1 ms

TABLE II

HANDOFF DELAY OF MEDIA UDP PACKET

handoff case	(a)	(b)
H12	38546.3 ms	420.8 ms
H23	4187.7 ms	418.6 ms
H31	1949.4 ms	408.4 ms

As we can see in Table I and Table II, modified kernel has reduced handoff delay by 1500 ms to over 30 s for some cases.

We understand that even with the modified kernel, the delay figure is not acceptable for real-time multimedia communications. For the completeness of our study, we then integrated MIPL [14] Mobile IPv6 in our testbed and performed the same experiment as we did for SIP mobility. In case of Mobile IPv6, while the handoff delay for signaling is measured as the time between receiving a new router advertisement from the new access router and receiving the binding acknowledgment from the Home Agent, the definition of the handoff delay for media UDP packet is the same as that in case of SIP mobility. We observed that the handoff delay for signaling is about 2 ms and the handoff delay for media UDP packet is less than 31 ms. Obviously, Mobile IPv6 with modified kernel outperforms the SIP mobility with modified kernel. However, we believe that an application layer mobility, such as SIP mobility, is a potential candidate to support real-time applications. Although we want to investigate in depth regarding our measurements and delay differences, a quick survey states that this is due to two different implementations. Mobile IPv6 is a kernel implementation whereas our current SIP implementation is a user space program and based on Tcl/Tk. Our understanding is that Tcl/Tk takes longer time to process messages. Therefore, we believe that with proper implementation, SIP terminal mobility can provide significantly low handoff delay for real-time application.

VII. CONCLUSION

In this paper, we have described the handoff delay analysis and measurement for SIP mobility in IPv6 laboratory testbed. In our performance study we concentrate on the delay incurred due to Duplicate Address Detection (DAD), and router selection when an MN moves to a new network and attaches to a new access router. We notice that DAD and router selection processes in Linux implementation offer substantial amount of delay during handoff. As a part of our integration we modify the Linux kernel to avoid the DAD. We also reduce the timer values of NUD, and select new router more aggressively.

Handoff delay is measured for SIP mobility with and without modification of Linux kernel. With our modified kernel, we observe the handoff delay improved a lot for SIP mobility. Our results reflect that handoff delays without the kernel modification varies from 2 to 40 s, whereas with kernel modification they are within 450 ms. This is a significant improvement in terms of real-time operation. Although we agree that this may be too high for some real-time applications, our experiments prove that application layer mobility, such as SIP mobility, can be a potential alternative to other mobility management schemes.

As a future work, we would like to investigate the issues and impacts in the network due to removing the DAD, reducing the NUD timer values, and choosing aggressive router selection mechanisms. As pointed out earlier, we would also like to experiment our SIP mobility with Mobile IPv6 in detail. In addition, we would like to investigate and measure other delay components.

ACKNOWLEDGMENTS

Authors are grateful to other ITSUMO members who participated in the interesting discussions and helped to create the testbed.

REFERENCES

[1] Jon Postel (Editor), “Internet Protocol”, RFC791, September 1981

[2] S. Deering and R. Hinden, “Internet Protocol, version 6 (IPv6) specification”, RFC2460, December 1998

[3] 3rd Generation Partnership Project, Technical specification group services and system aspects, 3rd Generation mobile system Release 5 specifications, 3GPP TS 21.103, work in progress

[4] D. B. Johnson and C. E. Perkins, J. Arrko, “Mobility support in IPv6”, draft-ietf-mobileip-ipv6-18.txt, June 2002, work in progress

[5] S. Thomson and T. Narten, “IPv6 stateless address autoconfiguration”, RFC2462, December 1998

[6] P. Flykt and T. Alakoski, “SIP Services and Interworking with IPv6”, Second International Conference on 3G Mobile Communication Technologies, March 2001, pp.186-190

[7] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler, “SIP: Session Initiation Protocol”, RFC3261, June 2002

[8] H. Schulzrinne and E. Wedland, “Application-layer mobility using SIP”, ACM SIGMOBILE Mobile Computing and Communications Review, Vol. 4, Number 3, July 2000, pp.47-57

[9] A. Dutta, J.-C. Chen, S. Das, M. Elaud, D. Famolari, S. Madhani, A. McAuley, M. Tauil, S. Baba, T. Maeda, N. Nakajima, Y.Ohba, and H. Schulzrinne, “Implementing a testbed for mobile multimedia”, Proceedings of IEEE Globecom 2001, San Antonio, Texas, USA, November 2001

[10] http://www.linux-ipv6.org

[11] http://www.cs.columbia.edu/˜hgs/software/sipc

[12] V. Hardman, M. A. Sasse, M. Handley, A. Watson, “Reliable audio for use over the Internet”, Proceedings of INET’95, Honolulu, Hawaii, June 1995

[13] T. Narten, E. Nordmark, and W. Simpson, “Neighbor Discovery for IP version 6 (IPv6)”, RFC2461, December 1998

[14] A. J. Tuominen and H. Petander, “MIPL Mobile IPv6 for Linux in HUT campus network MediaPoli”, Proceedings of Ottawa Linux Symposium 2001, Ottawa, Canada, June 2001