SIP Terminal Mobility for both IPv4 and IPv6

Che-Hua Yeh[1], Quincy Wu[2], and Yi-Bing Lin[1]
[1] *Department of Computer Science & Information Engineering,*
National Chiao Tung University
{chyei, liny}@csie.nctu.edu.tw
[2] *Graduate Institute of Communication Engineering,*
National Chi Nan University
solomon@ipv6.club.tw

Abstract

Session Initiation Protocol (SIP) supports application layer mobility during a session. In this paper the architecture design on the protocol stack implementation of SIP terminal mobility is described, and the performance of SIP user agents developed with open-source libraries are measured from empirical experiments. The experiments are performed in both IPv4 and IPv6 environment. In the best case, the delay of SIP mobility only takes 38ms in SIP signaling exchange, for both IPv4 and IPv6. Therefore, SIP mobility is suitable for supporting seamless handover in VoIP communications.

Keyword: IPv4, IPv6, Handover, SIP Mobility, VoIP

1. Introduction

Session Initiation Protocol (SIP) is an application-layer signaling protocol for Internet multimedia session establishment, modification, and termination [1]. SIP supports four types of mobility, i.e. terminal mobility, session mobility, personal mobility, and service mobility [2]. *Terminal mobility* is the capability to keep a session alive after the terminal device moves to a different IP subnet. *Session mobility* is the capability to maintain a session while the user is changing the terminal device. *Personal mobility* allows a user to become reachable at different terminal devices by the same logical address. *Service mobility* is the capability to access the user's services (e.g. address book, speed dialing, buddy lists) while the user is moving or changing devices and network service providers. In this paper we focus on SIP terminal mobility and propose a software architecture to implement this capability in a SIP User Agent. The empirical measurements (in both IPv4 and IPv6) show the SIP mobility performance for real-time multimedia applications, especially voice over IP (VoIP).

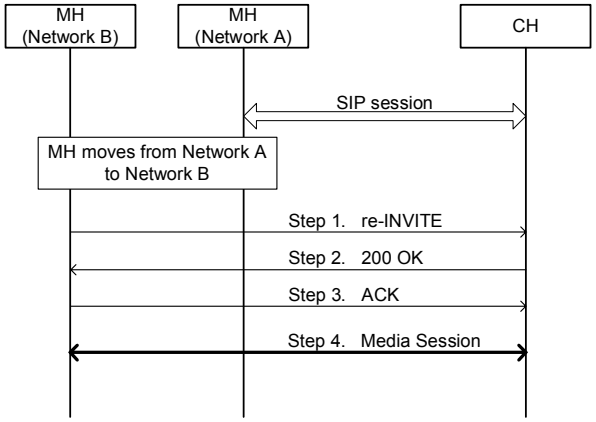

Figure 1. SIP terminal mobility procedure

Figure 1 illustrates the flow of SIP terminal mobility procedure. In the figure, the Mobile Host (MH) and the Correspondent Host (CH) are SIP User Agents (UAs). The MH will move from Network A to Network B. Suppose that Network A assigns an IP address 140.113.214.108 to the MH. Then a SIP multimedia session can be established between the MH and the CH, by following the standard SIP call setup procedure (INVITE/ 200 OK/ ACK) as described in [1]. In this procedure an INVITE message is sent between the MH and the CH. The INVITE message with the Session Description Protocol (SDP) content is shown in Table 1(a). Now, let us consider the scenario with mobility support. During the SIP multimedia session, the MH moves from Network A to Network B and acquires a new IP address 140.113.131.72 at Network B. Then the following steps are executed.

Table 1. SIP INVITE message and re-INVITE message

(a) INVITE

```
INVITE sip:944021117@sip.ipv6.club.tw SIP/2.0
Via: SIP/2.0/UDP 140.113.214.108:5060;branch=z9hG4bK6608
From: <sip:chyei@sip.ipv6.club.tw>;tag=8653
To: <sip:944021117@sip.ipv6.club.tw>
Call-ID: 7557@140.113.214.108                          (1)
CSeq: 20 INVITE
Contact: <sip:chyei@140.113.214.108:5060>              (2)
Max-Forwards: 5
User-Agent: Lab117-PoC-VoIP-UA/0.0.1
Subject: test
Expires: 120
Allow: INVITE, ACK, CANCEL, BYE, OPTIONS, REFER,
            SUBSCRIBE, NOTIFY, MESSAGE
Content-Type: application/sdp
Content-Length:   217

v=0
o=userX 20000001 20000001 IN IP4 140.113.214.108
s=A call
c=IN IP4 140.113.214.108                               (3)
t=0 0
m=audio 9000 RTP/AVP 0 8 18 3
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:18 G729/8000
a=rtpmap:3 GSM/8000
```

SIP headers: INVITE sip:944021117@sip.ipv6.club.tw SIP/2.0 … Content-Length: 217

SDP content: v=0 … a=rtpmap:3 GSM/8000

(b) re-INVITE

```
INVITE sip:944021117@sip.ipv6.club.tw SIP/2.0
Via: SIP/2.0/UDP 140.113.131.72:5060;branch=z9hG4bK41
From: <sip:chyei@sip.ipv6.club.tw>;tag=8653
To: <sip:944021117@sip.ipv6.club.tw>;tag=10651
Call-ID: 7557@140.113.214.108                          (1)
CSeq: 21 INVITE
Contact: <sip:chyei@140.113.131.72:5060>               (2)
Max-Forwards: 5
User-Agent: Lab117-PoC-VoIP-UA/0.0.1
Subject: test
Content-Type: application/sdp
Content-Length:   217

v=0
o=userX 20000001 20000002 IN IP4 140.113.131.72
s=A call
c=IN IP4 140.113.131.72                                (3)
t=0 0
m=audio 9000 RTP/AVP 0 8 18 3
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:18 G729/8000
a=rtpmap:3 GSM/8000
```

SIP headers: INVITE sip:944021117@sip.ipv6.club.tw SIP/2.0 … Content-Length: 217

SDP content: v=0 … a=rtpmap:3 GSM/8000

Step 1. The MH sends a SIP re-INVITE request to the CH. The format of the re-INVITE message is shown in Table 1(b). Note that the headers "From", "To", and "Call-ID" must be the same as those in the INVITE message (see Table 1 (1)). The Contact header field is updated to the MH's new IP address (from 140.113.214.108 to 140.113.131.72; see Table 1 (2)). The Contact header field provides the address that can be used to contact the SIP UA for subsequent requests. A new SDP content will be included in the re-INVITE message. Specifically, the connection address field ("c=") is updated to the MH's new IP address (from 140.113.214.108 to 140.113.131.72; see Table 1 (3)).

Step 2. When the CH receives the re-INVITE request, it replies a SIP 200 OK response to indicate that the CH recognizes the IP address change of the MH.

Step 3. The MH replies with a SIP ACK message to notify the CH that it has received the SIP 200 OK response.

Step 4. The CH modifies the session parameters according to the new connection address in the SDP content, and then the media data transmission is re-established between the CH and the MH.

In the above procedure, SIP re-INVITE message is utilized to notify the calling party to change media transmission parameters. Table 1 shows that the format of the SIP re-INVITE is exactly the same as SIP INVITE. Therefore SIP terminal mobility does not need to modify the SIP protocol or create a new SIP method. As indicated in [3], SIP terminal mobility supports fast handoff, low latency, and high bandwidth utilization.

In the remainder of this paper we will elaborate the architectural design on the protocol stack implementation of SIP terminal mobility and compare the performance results measured from empirical experiments.

2. Software Architecture of a SIP User Agent

This section describes the implementation of SIP terminal mobility. Figure 2 illustrates the software architecture of the SIP UA which is designed and implemented in National Chiao Tung University (NCTU). Figure 2 (1), (2), and (3) are open-source libraries and application programming interfaces (APIs). In the **SIP Core module**, eXtended osip (eXosip [4]) based on the GNU osip [5] library is utilized to implement the SIP protocol stack. The SIP Core module supports SIP communication with other SIP UAs. This module is invoked by the **Call Control module** to execute the call setup or teardown procedure following the standard SIP protocol. The **RTP Core module** utilizes the oRTP library [6] to implement Real-time Transport Protocol (RTP) stack under GNU Lesser General Public License (LGPL). This module builds RTP sessions between SIP UAs. The **User Interface module** supports interfaces for interactions between a user and the SIP UA. The Call Control module instructs other NCTU SIP UA modules to handle call related activities such as call answer, call rejection, and make an outgoing call, etc. The **Multimedia Control module** generates audible tones such as the ringing tone, the ringback tone, and

the busy tone to notify the user of various call status. When a call is established, this module plays the voice data it received from the TCP/IP module, or converts the voice from the user and delivers it over the TCP/IP module. The details of the above modules can be found in [7].

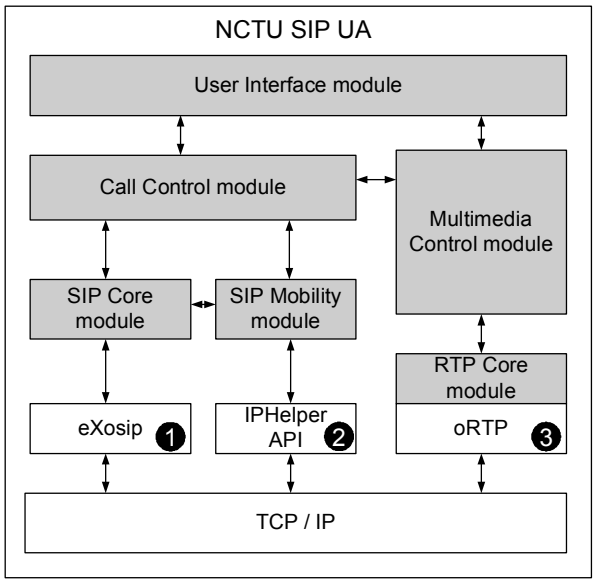

Figure 2. Architecture of NCTU SIP UA

The **SIP Mobility module** utilizes Windows IPHelper API with a function WSAIoctl() (WSA prefix for Winsock API) to monitor the status of local IP addresses of the SIP UA and control the whole process of the SIP terminal mobility procedure. Whenever the SIP UA detects that its IP addresses are modified, added, or deleted, the callback function IPChangeHandler() in SIP Mobility module will be invoked. The detailed operations will be elaborated in the next section.

3. Implementation of SIP Terminal Mobility

Figure 3 illustrates the interaction between modules of the NCTU SIP UA during the SIP terminal mobility procedure. The steps in Figure 3 are described as follows.

Step 1. As the SIP UA moves to a new network, the modification of IP addresses causes IPHelper API to trigger the event which activates the callback function in SIP Mobility module. Through IPChangeHandler(), SIP Mobility module retrieves IP addresses on local host by function GetAdaptersAddresses() in IPHelper API, and detects that it is assigned a new IP addresses from the network. Therefore SIP terminal mobility procedure will be activated.

Step 2. At the startup of the NCTU SIP UA, the eXosip library will create a UDP socket for SIP signaling [4]. The socket is bound to the local IP address. After the IP address is changed, the SIP mobility module will modify this socket with the new IP address through the function eXosip_modify_ip() provided by the SIP Core module.

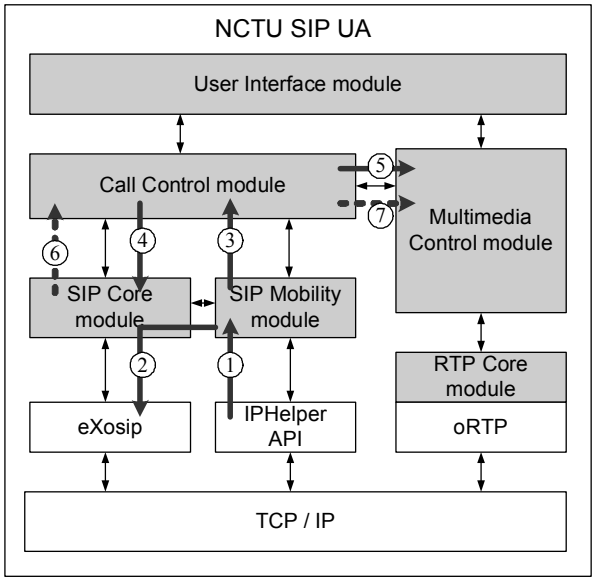

Figure 3. The process flow of SIP terminal mobility

Step 3. After the socket in eXosip is modified, the SIP mobility module generates an event *SIPCore_IPCHANGE_NEWIP_NOTIFY* to notify the Call Control module to execute SIP terminal mobility procedure.

Step 4. Because the UA is involved in an ongoing session, the Call Control module will send a SIP re-INVITE request through the SIP Core module to the CH.

Step 5. Meanwhile, the Call Control module instructs the Multimedia Control module to suspend the RTP session.

Step 6. After the UA has received the SIP 200 OK response from the CH, it follows the standard SIP procedure to send SIP ACK to the CH.

Step 7. The Call Control module instructs the Multimedia Control module to resume the RTP session at the new connection address as described in Section

1. The RTP session between the UA and the CH is re-established.

4. Performance Comparison

In [8], the delay of a SIP UA mobility was empirically measured. Figure 4 illustrates the delay for SIP terminal mobility procedure, which can be divided into the following parts: D1 is the delay for switching from one AP to another. D2 is the delay of detecting a new router and a new IP subnet after switching AP, where MH can detect that it has moved to a new subnet by listening to IPv6 Router Advertisement. D3 is the delay between when the MH activates the SIP terminal mobility procedure and when it receives the SIP 200 OK response for SIP re-INVITE request. D4 is the delay between when the MH activates the SIP terminal mobility procedure and when the RTP session is resumed again. We adopt the same notations of D1, D2, D3, D4 as in [8] for performance measurement.

D1 and D2 are the link-layer delays, which are not the focus of SIP terminal mobility. This paper evaluates D3 and D4 as in [8]. In D3 and D4 under IPv6 environment, there is a delay from Duplicate Address Detection (DAD) process. The details of DAD are elaborated below:

4.1 IPv6 with Duplicate Address Detection

If the MH configures its IPv6 address via stateless address autoconfiguration [9], the stateless mechanism allows the MH to generate its own IPv6 addresses (referred to as the tentative address). For example, the MH has an Ethernet card with MAC address 00:11:5B:3A:71:E8, when it moves to a subnet with prefix 2001:238::/64, then its 48-bit MAC address will be converted by EUI-64 process [9] to a 64-bit interface identifier 0211:5BFF:FE3A:71E8. The IPv6 address 2001:238::0211:5BFF:FE3A:71E8 is (temporarily) assigned to it. Since the tentative address may be duplicated, the Duplicate Address Detection (DAD) algorithm is activated before the MH can send packets via this IPv6 address.

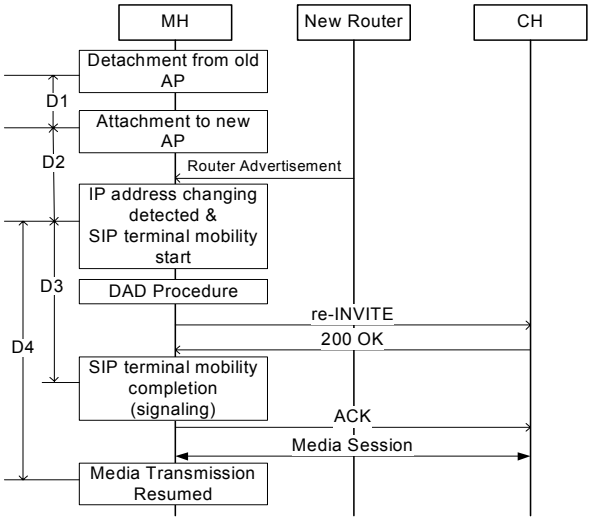

Figure 4. The delay of SIP terminal mobility procedure

After generating the IPv6 address according to the received IPv6 subnet prefix, the MH sends a Neighbor Solicitation (NS) message on the local link. If there is no response until timeout of a pre-determined timer, the address will be assigned to the Ethernet card. Otherwise a duplicate IP address is detected. Before the DAD procedure completes, the host can not send packet via the tentative address. Thus the DAD procedure introduces additional delay before sending the SIP re-INVITE request. According to [9], the delay of DAD is 1.5 seconds in average.

Table 2 shows the delays for D3 and D4. The values in the first row are the results of NCTU SIP UA, and the values of the second row are the results for the SIP UA developed by N. Nakajima, A. Dutta, S. Das and H. Schulzrinne (NDDS) SIP UA [8].

Table 2. Delay of SIP mobility procedure in IPv6 environment with DAD

Devices Under Test	D3 (ms)	D4 (ms)
NCTU SIP UA	1742.1	1822.6
NDDS SIP UA	3932.2	4187.7

The delay D4 from NCTU SIP UA takes 1882.6ms. The results from NDDS SIP UA show longer delay, 4187.7ms. The possible reason which causes the longer delay is that NDDS SIP UA was implemented with Tcl/Tk [10], which is a higher level programming language such that more overhead might be involved. For the 1822.6ms delay measured in our experiment, the DAD process takes approximately 1500ms while the SIP terminal mobility mechanism only takes about 300ms. This clearly shows that DAD process is the

bottleneck which causes the delay of SIP terminal mobility. In next section we will show the results which exclude the DAD process and compare them with the results in IPv4 environment.

4.2 IPv4 and IPv6 without Duplicate Address Detection

In the previous section, we compared the SIP terminal mobility delay for NCTU SIP UA and NDDS SIP UA under IPv6 environment. The time analysis of each step in the mobility mechanism shows that the DAD is the bottleneck of the whole process. To solve this problem, it was proposed in [8] to modify the Linux kernel to remove the DAD mechanism in IPv6 address autoconfiguration process. In our experiments, we adopted a simpler approach by hardwiring the IPv6 address assignment in our program rather than using autoconfiguration. This approach also demonstrates well to exclude the DAD mechanism. In both approaches, the total delay of SIP terminal mobility is significantly shortened. The experimental results are shown in Table 3.

Table 3. Delay of SIP mobility procedure in IPv6 environment without DAD

Devices Under Test	D3 (ms)	D4 (ms)
NCTU SIP UA	38.8	217.9
NDDS SIP UA	161.6	418.6

This table shows that the delay of D4 is reduced to 217.9ms in NCTU SIP UA and 418.6ms in NDDS SIP UA. Compared with Table 2, the delay is decreased significantly. This experiment clearly shows that the DAD procedure is the bottleneck which increases the delay during the SIP terminal mobility procedure. If we remove the DAD procedure, the delay of SIP terminal mobility (38.8ms in NCTU SIP UA, and 161.6ms in NDDS SIP UA) is short enough to support VoIP communication with seamless handover [11], where shorter than 50ms of interruption in handoff is desired for VoIP communications. However, as noted in [8], another address configuration method must be provided to replace the DAD procedure after it is removed. Otherwise we can not make sure whether there is any duplicate address existing in the same subnet.

A comparison of the performance we measured in IPv4 and IPv6 environments also shows interesting results. Table 4 shows the delay of SIP terminal mobility in IPv4 network and the results in IPv6 environment excluding DAD procedure (the same as the first row in Table 3). In IPv4 network, there is no DAD procedure, so the delay is short (214.4ms). Moreover, the delay time of SIP mobility procedure in IPv4 and IPv6 (with DAD process excluded) is very close, either in signal completion or media resumption.

Table 4. Delay of SIP mobility procedure in IPv4/IPv6 environment

Devices Under Test	D3 (ms)	D4 (ms)
NCTU SIP UA (IPv4)	38.2	214.4
NCTU SIP UA (IPv6)	38.8	217.9

5. Interoperability

In previous sections, the experiment is conducted between the same NCTU SIP UA software. In this section we shall show the experimental results with other SIP UAs. We shall keep using NCTU SIP UA as the MH, and select one SIP UA to be the CH. We have tested several SIP UAs including softphones and hardphones. Table 5 shows those SIP UAs and their experimental results. NCTU SIP UA, Windows Messenger, and X-Lite UA are sofphones. Snom200, Cisco 7940, InnoMedia video phone, and Pingtel are hardphones. Notice that even though the MH is always NCTU SIP UA, the delay of D3 is quite different. The reason is that after receiving a re-INVITE request, each SIP UA requires different time to handle the request and then generates the SIP 200 OK response.

From Table 5, the results of NCTU SIP UA and Windows Messenger 5.1 are almost the same, and the X-Lite UA is a little longer (about 11% for D4). In the results of hardphones, the delay is obviously longer than sofphones. We have consulted the engineers in InnoMedia Corporation, and they believe that it is caused by the time spent on SDP parsing in the protocol stack. Because InnoMedia video phone includes video transmission during SIP conversation, the SDP contains video media description. Therefore this extra complexity increases the delay slightly. We also perform the same experiment on CISCO 7940 SIP hardphones with different firmware version. In firmware version 7.5, the delay of sending 200 OK response is more than that in firmware version 6.3. The reason is that it applies more rigorous rules in checking the SDP contents. For example, in firmware version 7.5 when the re-INVITE is received, the SDP version in session identifier field (the third field in the "o=" line) must be verified to see whether it is incremented by 1. Certainly this increases the delay, too.

Table 5. Delay of SIP terminal mobility between NCTU SIP UA and other SIP UAs

Devices Under Test	D3 (ms)	D4 (ms)	Media resumption delay (D4) compared to NCTU SIP UA
NCTU SIP UA (IPv4) Ver1.1	38.2	214.4	100%
Windows Messenger Ver5.1.0680	38.2	214.3	99.95%
X-Lite UA Ver2.0 build 1103	50.2	238.4	111.19%
Snom 200 (hardphone) Ver1.16x 4904	94.8	270.9	126.35%
Cisco 7940 (hardphone) Ver5.3	151.3	340.2	158.68%
Cisco 7940 (hardphone) Ver7.5	230.2	404.4	188.62%
InnoMedia (hardphone) Ver2.4.17	173.1	356.1	166.09%
Pingtel Ver2.1.11.24	195.0	370.6	172.85%

6. Summary

In this paper the protocol architecture design and implementation of the SIP terminal mobility are illustrated. Empirical measurements show that, the delay of the SIP terminal mobility in IPv6 environment is about 1822.6ms. However, if DAD is excluded, the delay is significantly reduced to 217.9ms. We also perform the experiments in IPv4 environment. Comparing the two results, it is interesting to notice that the delay of SIP terminal mobility in IPv6 environment without the DAD process is close to that in IPv4 environment. Obviously the delay from the DAD process is the bottleneck in the SIP terminal mobility procedure.

Moreover, the interoperability testing of terminal mobility among SIP UAs is demonstrated. It can be seen that the delay of SIP terminal mobility does not depend only on the MH, but also on the CH. Because each SIP UA sends a SIP 200 OK response after receiving a re-INVITE request, the delay time differs divergently. According to the information provided by the manufacturer, one of the major factors is the complexity required in processing the SIP header and SDP contents.

7. References

[1] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler , "SIP: Session Initiation Protocol", IETF RFC 3261 , June 2002.

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

[3] E. Wedlund and H. Schulzrinne, "Mobility support using SIP", in Second ACM/IEEE International Conference on Wireless and Mobile Multimedia (WoWMoM'99), August 1999.

[4] The eXtended osip library, http://savannah.nongnu.org/ projects/exosip/

[5] GNU oSIP library, http://www.gnu.org/software/osip/

[6] oRTP – a Real-time Transport Protocol stack under LGPL, http://linphone.org/ortp/

[7] L.-Y. Wu, M.-H. Tsai, Y.-B. Lin, and R.-S. Yang, "A Client-Side Design and Implementation for Push to Talk over Cellular Service". Accepted and to appear in Wireless Communications and Mobile Computing Journal.

[8] N. Nakajima, A. Dutta, S. Das, and H. Schulzrinne, "Handoff Delay Analysis and Measurement for SIP based Mobility in IPv6", IEEE International Conference on Computers and Communications, 2003.

[9] S. Thomson and T. Narten, "IPv6 stateless address autoconfiguration", IEEE RFC 2462, December 1998.

[10] Tcl/Tk, http://www.tcl.tk/

[11] S. Shin, A. G. Forte, A. S. Rawat, and H. Schulzrinne, "Reducing MAC layer handoff latency in IEEE 802.11 wireless LANs", International Conference on Mobile Computing and Networking, 2004.