Project report
Integration of MIP in an IMS environment
by
Alexandre Boursier boursier@kom.aau.dk
Thior Santander Marin thior@kom.aau.dk
Axel Adenet axela@kom.aau.dk

MSc Mobile Communications
Aalborg University
Center for PersonKommuniKation

Supervisors:
Thibaut Renier tr@kom.aau.dk
Hans Peter Schwefel hps@kom.aau.dk

Submission date: 2nd June 2005
ABSTRACT

The IP Multimedia Subsystem (IMS) is a framework designed by the Third Generation Partnership Project (3GPP) in order to provide operators with a packet-switched core network that manages IP-based multimedia services.

It uses the Session Initiation Protocol (SIP) for signaling and as a mobility management tool. However, SIP is a high-layer protocol so it cannot deal with macro-mobility effectively. Indeed, an IP address change brings down existing sessions and forces to re-register to the IMS.

As a network layer solution for managing mobility, Mobile IP (MIP) is a good candidate to address the inefficiencies of the IMS in this domain. Nevertheless, the hidden IP address changes for which MIP is praised collapse the system because of the QoS requirements in the IMS such as access control.

This project presents some solutions in order to successfully integrate MIP into the IMS, by modifying key IMS components (P-CSCF, SBLP...) so they get additional information from MIP mechanisms. Several experimental setups have been deployed in order to quantify the possible benefits that stem from the conceptual work.

Undeniably the time to perform a handover with MIP is faster than with SIP. But thanks to SIP mechanisms, some data packets can be saved from being dropped. Our implementation shows the way to quantify exactly this balance, nevertheless it shows some advantages of using MIP.
# Contents

1 Introduction ............................................. 7  
  1.1 Overview ............................................ 7  
  1.2 Basic handover scenario .............................. 8  
  1.3 Thesis outline ........................................ 8  

2 Background ................................................. 11  
  2.1 IP Multimedia Subsystem (IMS) ....................... 11  
    2.1.1 IMS Functionalities & Architecture ............. 11  
    2.1.2 Session Initiation Protocol (SIP) ............... 12  
    2.1.3 IMS entities .................................... 17  
    2.1.4 Quality of Service ................................ 18  
    2.1.5 Mobility related concepts ....................... 23  
  2.2 Mobile IP ............................................. 24  
    2.2.1 Comparison between MIPv4 and MIPv6 .......... 25  
    2.2.2 MIPv6 operation .................................. 27  
    2.2.3 Modifications to the IPv6 protocol ............ 29  

3 Analysis .................................................... 34  
  3.1 Traffic flows ......................................... 34  
    3.1.1 Registration .................................... 34  
    3.1.2 Session flows - serving to serving procedure .. 35  
    3.1.3 Session flows - Data sending establishment .... 37  
    3.1.4 Session flows - Handover ....................... 37  
  3.2 Integration issues .................................... 37  
    3.2.1 Switch-on ...................................... 41  
    3.2.2 Communication with a correspondent node ...... 41  
    3.2.3 Handover ....................................... 43  

4 Solution Proposals ........................................ 44  
  4.1 Issue A: MIP Binding Updates to the HA ............ 44  
  4.2 Issue B: Registration to the P-CSCF ................. 45  
  4.3 Issue C: Tunneling through HA ....................... 46  
  4.4 Issue D: Binding unknown in route optimization .... 46
CONTENTS

4.5 Issue E: Transfer of QoS parameters ........................................ 47
4.6 Issue F: Unused resources in previous network ............................ 48

5 Experimental work ........................................................................ 49
  5.1 IMS/SIP operations ...................................................................... 49
    5.1.1 Presentation of the new SIP client ....................................... 49
    5.1.2 Architecture of the new SIP client ....................................... 49
    5.1.3 Laboratory configuration .................................................... 52
    5.1.4 Problem & solutions due to the present IMS system .......... 54
    5.1.5 Problem & solution due to the handover procedure .......... 56
    5.1.6 SIP Experimental flow ....................................................... 57
  5.2 MIPv6 implementation ................................................................. 60
    5.2.1 Presentation of the new implementation ............................. 60
    5.2.2 Architecture & Configuration of MIPv6 ............................. 60
    5.2.3 Experimental flow .............................................................. 62
    5.2.4 Limitations of the implementation ...................................... 63
  5.3 MIPv6/SIP integration ................................................................. 64
    5.3.1 Problem ............................................................................ 64
    5.3.2 Solutions ........................................................................... 65
    5.3.3 Description of the chosen solution ..................................... 67

6 Experimental measurements .......................................................... 69
  6.1 Soft handover ............................................................................. 69
    6.1.1 Test configuration ............................................................... 69
    6.1.2 Results .............................................................................. 70
  6.2 Hard handover ............................................................................ 72
    6.2.1 Test configuration ............................................................... 72
    6.2.2 Different scenarii ............................................................... 73
    6.2.3 Results .............................................................................. 75
    6.2.4 Analysis ............................................................................ 76

7 Conclusion & Future Work ............................................................... 82
  7.1 Conclusion .................................................................................. 82
  7.2 Future Work ............................................................................... 83

A Abbreviations ................................................................................. 84

B SIP details ....................................................................................... 86
  B.1 SIP responses ............................................................................ 86
  B.2 SIP headers .............................................................................. 88

C Experimental setup migration from IPv4 to IPv6 ........................... 91
  C.1 MIPv4 existing implementation ............................................... 91
  C.2 SIP existing implementation .................................................... 91
  C.3 MIPv6 new implementation ..................................................... 91
  C.3.1 Software .............................................................................. 91
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Project Scenario</td>
<td>9</td>
</tr>
<tr>
<td>2.1</td>
<td>IMS - Overview</td>
<td>12</td>
</tr>
<tr>
<td>2.2</td>
<td>Example of SIP architecture</td>
<td>14</td>
</tr>
<tr>
<td>2.3</td>
<td>QoS establishment between GGSN and PDF during session setup</td>
<td>21</td>
</tr>
<tr>
<td>2.4</td>
<td>QoS establishment between P-CSCF and PDF during session setup</td>
<td>24</td>
</tr>
<tr>
<td>2.5</td>
<td>CoA registration to the HA in MIPv4</td>
<td>25</td>
</tr>
<tr>
<td>2.6</td>
<td>Triangle routing in MIPv4</td>
<td>26</td>
</tr>
<tr>
<td>2.7</td>
<td>IP-in-IP encapsulation</td>
<td>26</td>
</tr>
<tr>
<td>2.8</td>
<td>Binding Update to the HA in MIPv6</td>
<td>27</td>
</tr>
<tr>
<td>2.9</td>
<td>Bidirectional tunneling in MIPv6</td>
<td>28</td>
</tr>
<tr>
<td>2.10</td>
<td>Return routability procedure</td>
<td>29</td>
</tr>
<tr>
<td>2.11</td>
<td>Binding Update operation to a CN</td>
<td>30</td>
</tr>
<tr>
<td>2.12</td>
<td>Route optimization in MIPv6</td>
<td>30</td>
</tr>
<tr>
<td>2.13</td>
<td>Mobility Header format</td>
<td>30</td>
</tr>
<tr>
<td>2.14</td>
<td>Format of a Binding Update message</td>
<td>31</td>
</tr>
<tr>
<td>2.15</td>
<td>Format of a Binding Acknowledgement message</td>
<td>31</td>
</tr>
<tr>
<td>2.16</td>
<td>Format of a Home Address option within its IPv6 header</td>
<td>32</td>
</tr>
<tr>
<td>2.17</td>
<td>Format of a Type 2 Routing Header</td>
<td>32</td>
</tr>
<tr>
<td>2.18</td>
<td>Dynamic Home Agent Address Discovery procedure</td>
<td>32</td>
</tr>
<tr>
<td>2.19</td>
<td>Mobile Prefix Discovery mechanisms</td>
<td>33</td>
</tr>
<tr>
<td>3.1</td>
<td>IMS - Registration flow</td>
<td>35</td>
</tr>
<tr>
<td>3.2</td>
<td>Serving to serving procedure - different operators</td>
<td>36</td>
</tr>
<tr>
<td>3.3</td>
<td>Message flow from the connection to the IP-CAN until the first packet sending</td>
<td>38</td>
</tr>
<tr>
<td>3.4</td>
<td>Handover in IMS context</td>
<td>39</td>
</tr>
<tr>
<td>3.5</td>
<td>Network architecture retained for the analysis</td>
<td>40</td>
</tr>
<tr>
<td>3.6</td>
<td>Filtering of IP packets at the GGSN</td>
<td>42</td>
</tr>
<tr>
<td>3.7</td>
<td>Filtering issues with MIP packets</td>
<td>42</td>
</tr>
<tr>
<td>4.1</td>
<td>HA validation from the GGSN</td>
<td>45</td>
</tr>
<tr>
<td>4.2</td>
<td>BU to the P-CSCF</td>
<td>46</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

4.3 GGSN analysis of packets tunneled through the HA .......................... 47
4.4 Informing the SBLP of the CoA: double entries ............................... 47
4.5 Transfer of QoS parameters ...................................................... 48

5.1 Architecture of the new sip client .............................................. 50
5.2 Experimental configuration of the SIP system ................................. 53
5.3 Filter architecture ..................................................................... 56
5.4 Experimental flows until an opened SIP session ............................... 58
5.5 MIPv6 architecture .................................................................... 61
5.6 MIP - Experimental flow - Route Optimization ............................... 62
5.7 MIP - Experimental flow - Bi Tunnelling ....................................... 63
5.8 SIP & MIP integration ................................................................. 66
5.9 Translator processing ................................................................. 68

6.1 Linear relation between packet loss and IP obtention ....................... 73
6.2 Handover - MIP+SIP w/o renegotiation ........................................ 76
6.3 Confidence Intervals - Handover delay ......................................... 77
6.4 Packet loss - MIP+SIP w/o renegotiation ....................................... 78
6.5 Confidence Intervals - Packet loss ................................................ 79

C.1 Lab Architecture & Equipment .................................................... 92
C.2 TestBed Architecture ................................................................. 93
List of Tables

6.1 Test measurement - sip establishment2 ............................. 71
D.1 Programming files - programmes ..................................... 96
Chapter 1

Introduction

1.1 Overview

The IP-based Multimedia Subsystem (IMS) is a framework designed by the 3GPP (Third Generation Partnership Project) that allows mobile operators to offer services based on Internet applications independently from the actual connectivity, whether it is GPRS, UMTS, WLAN, or else – although it was at first an extension to UMTS. The IMS provides session and connection control as well as an application services framework.

In order to achieve that, all entities in an IMS network use the Session Initiation Protocol (SIP) to communicate with one another. SIP provides session to a user equipment (UE) connecting to the IMS. Using SIP, the IMS offers services such as User registration, Instant messaging or Presence Services. The IMS also supposes that services can be deployed externally from the IMS framework, letting PLMNs and other third parties develop their own services through Application Servers (AS).

On a financial level, the IMS is interesting for providers because it allows them to provide new services to counter the downward trend in average revenue per user from voice services; this is done without changing the access network. They can then have a cost effective new revenue stream that can have long-term growth and profitability.

The IMS uses SIP which manages multimedia sessions and mobility. One of the main advantages of SIP is the simplicity if its syntax: commands and parameters are sent as plain text to be interpreted by the recipient. This protocol begins to be very popular [1].

However, using SIP does not provide so good a performance regarding mobility management [3], especially in a macro-mobility case where a change of IP address takes place. Such a change can typically occur when the end user’s terminal changes its access network (e.g. UMTS to/from WLAN) and gets into a new network domain. As SIP is a protocol from the OSI session layer (OSI L5), it is supported by lower layer protocols such as IP; thus an IP address change brings down existing SIP sessions and induces long delays because of session re-establishment.

To reduce the delays inferred by the handover process, it is possible to work on lower layers. By hiding the mobility problem in these layers, the SIP session could stay opened.
1.2 Basic handover scenario

Although solutions have been designed on layers 2 to 4, a strong possibility is the network layer (L3) for several reasons; the undeniable dominance of IP as a convergence protocol on this layer is perhaps the most obvious one. Hence, the layers above could use the same fixed IP address, even if the mobile node moves in another network and changes its IP address. Mobile IP (MIP) is a likely candidate, because of its performance potential, its standardization by the IETF, and its seamless integration into IPv6. In addition, it begins to be deployed in a standard fashion.

IMS and MIP implementations already exist separately. However, combining them directly has not been achieved. On one hand, what happens at layers 3 and below should be hidden completely by MIP. On the other hand, some of the IMS functionalities such as access control or quality of service (QoS) control require some knowledge about the access network. For example, access control supposes that communications fit a certain profile whereas these may be different depending on how the L3 protocol is implemented. When implementing MIP in the IP Multimedia Subsystem, we need to care about all these functionalities, and make sure that the modifications induced by MIP do not prevent communications from matching IMS requirements; IP packets intended to the end user must be delivered accurately and effectively.

1.2 Basic handover scenario

The work is carried out with the objective to perform a handover in an IMS/MIP hybrid system. However, there are several types of handover [4], from which we chose the “reactive” type: a MIP-compliant mobile node (also user end, i.e. UE, in the IMS nomenclature) communicates with a fixed correspondent node (CN) via the IMS from network WLAN1, then abruptly loses IP connectivity from WLAN1 and gains IP connectivity from network WLAN2 (see Figure 1.1).

Indeed, this handover type will trigger the assignment of a new IP address to the UE. In addition, it is assumed that no effort is to made to keep connectivity on both networks. This macro-mobility situation leads to long handover delays that damage user satisfaction. The goal is to use Mobile IP mechanisms to keep the same IP address on the SIP layer when experiencing such a handover, so as to keep on-going sessions open. This may help reduce handover delays, and thus enhance the performance of the IMS. An experimental testbed will be developed to measure the performance of such a specifically designed IMS/MIP hybrid network.

In addition, as QoS available at one access network might change after handover to a new access network, Mobile IP integration only would not be enough to keep an end-to-end QoS, as MIP does not manage QoS aspects. Hence, some modifications have to be made with respect to the QoS specifications.

1.3 Thesis outline

The following section will provide the reader with some background on the IMS and its key concepts, with an emphasis on its entities and its use of SIP, as well as a clear description of
Figure 1.1: Project Scenario
1.3 Thesis outline

MIP mechanisms. An analysis will then expose the important issues, followed by a section pointing to potential solutions to address them. After experimental implementation setups are presented, experimental results will be exposed and analyzed. The final section will then bring conclusions to this work and will lay ideas for future developments.
Chapter 2

Background

The purpose of this chapter is to make the reader more familiar with the key concepts and notions considered during the course of this work. At first, the IMS will be presented with its functionalities and architecture, its SIP base and its specific entities, as well as QoS aspects to be taken into account. Then, Mobile IP mechanisms will be detailed so as to understand better how they could be combined with the requirements of an IMS network.

2.1 IP Multimedia Subsystem (IMS)

2.1.1 IMS Functionalities & Architecture

Functionalities

The IMS framework allows operators to manage IP-based multimedia services such as instant messaging applications, video/audio data streaming, Push-To-Talk (a service which enables subscribers to use their terminals as walkie-talkies) and network gaming, in an efficient and profitable fashion. Therefore, resource control and user charging are of prime concern.

Moreover, voice traffic as the traditional revenue source provides a less and less substantial share of total revenue compared with multimedia data traffic. Combined with the rise of Voice-over-IP (VoIP) solutions, this changing revenue pattern has put pressure on operators to solve the problem of voice and data convergence. The 3GPP has also designed the IMS with this intent.

These functionalities are supported by a specific architecture that is described in the following sub-section.

Architecture

The actual architecture of the IMS is illustrated in Figure 2.1; the involved entities will be described extensively in Section 2.1.3. Only the signalling path is represented, because the data flow follows a direct path between the User End (UE) and, for instance the...
2.1 IP Multimedia Subsystem (IMS)

Application Server (AS) with which it communicates. Data packets do not go through the IMS specific entities, as they only forward SIP signalling messages.

2.1.2 Session Initiation Protocol (SIP)

SIP is a protocol that allows to establish sessions among different users. Created in the middle of the 1990’s at Columbia University, it was then standardized by the Internet Engineering Task Force (IETF) in 2001. The IETF’s Request For Comments (RFC) 3261 is the original standard, and extensions have been added later to enhance some security and authentication aspects (from RFC 3262 to RFC 3265). Initially, SIP only establishes sessions, so it is supported by the Session Description Protocol (SDP) to manage sessions once they are initiated [6].

Even though the standardization of this protocol occurred later than its main competitors (H.323 and MGCP – Media Gateway Controller Protocol), it begins to be very popular and has been chosen in the IMS architecture. This success is due to its simplicity.
2.1 IP Multimedia Subsystem (IMS)

and its similarities with the famous Hyper-Text Transfer Protocol (HTTP). In the near future, it is going to be a common Internet protocol. For instance, Microsoft Corporation have already integrated SIP in their Windows XP\textsuperscript{TM} and Pocket PC\textsuperscript{TM} operating systems, as well as in its famous MSN Messenger\textsuperscript{TM} instant messaging application, which is actually based on SIP.

This section is organized as follows: a first part exposes the architecture of a standard IETF SIP system; then SIP messaging is described, before IMS specific aspects are considered.

**SIP Architecture**

Although SIP is a protocol built to establish sessions, it corresponds – like HTTP – to the session layer in the OSI model. Its main purpose is to provide management for mobility, data traffic and security.

![OSI model]

The SIP architecture is defined by 6 entities (see Figure 2.2):

- the **UAC** (User Agent Client) corresponds to the user who initiates the establishment of a session, and corresponds to the UE in the IMS;

- the **UAS** (User Agent Server) corresponds to the end terminal with which the UAC wants to establish the session. It can be another user or a server; this is the CN in the IMS;

- the **PS** (Proxy Server) is the entry point for the UAC/UAS in SIP architecture. Every UAC/UAS is attached to a PS. All SIP registration requests, and all session initiation requests sent from or received by the UAC/UAS flow through the PS, which is able to modify or add some fields in the headers of SIP requests;

- the **RS** (Redirect Server) provides information about the mobility of an end user: if a user wants to call a mobile end user, they need to find the attached proxy, which can have changed. Requests to the Redirect Server do not generally pass through the PS;

- the **LS** (Location Server) provides the location of UACs/UASs. It is used by both the PS and the RS;

- the **RG** (Registration Server) allows a UAC/UAS to register to the SIP infrastructure.
2.1 IP Multimedia Subsystem (IMS)

SIP Methods

For a UAC and a UAS to establish a multimedia session between themselves, they need to exchange requests and replies. In the standard SIP protocol, there are 6 main request methods. For each request, there is an associated response, to inform the originating side about the status of its requests.

Requests There are 6 main SIP requests, defined in the standard SIP protocol:

- **INVITE** is used by a UAC to initiate a session with a UAS;
- **ACK** is an acknowledgement of a positive response to an INVITE request (200 OK, see **Responses** below); the ACK request does not go through the PS;
- **CANCEL** is used to abort a previous request. However, this request can only be used before the final response of the UAS; the UAS then responds to the initial request with an error;
- **BYE** terminates a session;
- **REGISTER** permits to register contact information;
- **OPTIONS** is used to query servers about their capabilities, e.g. supported methods, content types, extensions, or codecs.

The first line of a SIP request contains the type of the request and must be formatted with the following syntax:

\[
\text{Method-SIP Request-URI SIP-Version CRLF}
\]

The method corresponds to one of those described above. The requested URI is a SIP URI (Uniform Resources Identifier) corresponding to the user to whom the request is destined. The SIP version is currently SIP/2.0. “CRLF” is the abbreviation for *Carriage Return Line Feed*.
2.1 IP Multimedia Subsystem (IMS)

Return Line Feed, the identifier for the character that codes the end of the line. All of these fields must not contain LWS (Linear WhiteSpace).

A SIP URI looks like a classic email address. First there is the name of the identified user, then the ’@’ character, and then the domain identifier.

Responses

SIP responses are used to inform the UAC about the statement of its request. So, for one request many responses can be generated: for example, to inform the UAC that the phone is still ringing.

The type of responses is identified by 3 digits. Then, all of them are classified in 6 main categories. The first digit permits to identify this category.

- **1xx** (from 100 to 199) contains all provisional responses: the request is still processing. For example, response 100 corresponds to the “trying” response, and 180 informs that the sending node is “ringing”.

- **2xx** (from 200 to 299) is used to indicate the success of the request. Until now, the only defined response is the number 200.

- **3xx** (from 300 to 399) informs the UAC that further actions need to be performed in order to complete the request. For example, error 300 indicates that there are multiple choices. Error 305 tells the user to use a PS in order to communicate with the UAS.

- **4xx** (from 400 to 499) indicates a UAC error. For example, error 400 is used for bad requests, and error 401 points to an unauthorized access. Error 404 means that the destination has not been found.

- **5xx** (from 500 to 599) is used when the request is valid, but the server is not able to complete it. For example, error 500 is used to specify an internal server error, and error 503 is used to inform the UAC that the service is not available.

- **6xx** (from 600 to 699) informs the user about a global failure, the request cannot be fulfilled by any server. For example, when the UAS does not want to participate in the session, it returns error 603.

The syntax of a SIP response is as following:

```
SIP-Version Status-Code Reason-Phrase CRLF
```

As for the request syntax, the SIP version currently used is also SIP/2.0. The status code corresponds to one of the categories described above. The reason phrase is a textual phrase used to describe the error. It is used by the programmers, but it does not have to be shown to the end user. The “CRLF” character symbolizes again the end of the line.
2.1 IP Multimedia Subsystem (IMS)

SIP Tags

By using different types of transactions, a UAC is able to establish a SIP session. To be able to communicate with the callee, the UAC has to negotiate some parameters, e.g. the encoding type, authentication or used route. To do so, the SIP protocol has defined some tags – fields in the request header, which describe different aspects of the transaction. The syntax of these tags is very simple:

```
Header-name : Header-value [, list of other values] CRLF
```

Some of the main headers of the standardized SIP protocol are listed below.

- **Call-ID**: is used to identify a unique session initiation, and should be the same for all subsequent registrations of a UAC. Using a specific algorithm, the key has to be unique;

- **Contact**: contains a URI which is interpreted in different ways, depending on the type of request;

- **Content-Length**: indicates the size of the message body;

- **Content-Type**: defines the media type used in the message body;

- **Cseq**: contains a single decimal number of 32 bits, used to order the transactions of a session;

- **From**: indicates the initiator of the request;

- **Max-Forwards**: limits the number of proxies or gateways that can forward the request. The default value is 70;

- **Record-Route**: permits the proxy to force future requests of the session to pass through itself;

- **Route**: is used to force a request to be routed through a list of proxies;

- **To**: specifies the logical recipient of the message;

- **Via**: indicates the path used by the request, so that the response uses the same path; this is imposed by the SIP protocol so as to process the response with the same servers as for the request.

IMS/SIP specifications

The IMS is implemented around the SIP protocol and uses it to provide and control IP access to 3GPP networks. However, in order to achieve this goal, some modifications have been made to the original SIP architecture. The most important ones are listed below.

- All SIP messages from/to the UAC have to flow through the Proxy, for security reasons.
2.1 IP Multimedia Subsystem (IMS)

- Some additional requests have been developed:
  - the UPDATE request carries SDP parameters from the UAC so as to reserve resources (e.g. bandwidth);
  - the PRACK request also carries SDP parameters, in order to validate QoS negotiation with the UAS; it is usually sent after an UPDATE request/response exchange.

2.1.3 IMS entities

This sub-section describes the entities presented in the IMS architecture in Figure 2.1

P-CSCF (Proxy-CSCF)

The Proxy Call State Control Function (P-CSCF) is the first contact point to the IMS for the UE. It behaves like a proxy (SIP PS) in that it accepts requests and processes them internally, or forwards them on towards their final recipient. The P-CSCF performs the following functions:

- Forward the SIP register request received to an I-CSCF determined using the home domain name, as provided by the UE.
- Forward SIP messages received from the UE to the SIP server (S-CSCF) whose name has been received as a result of the registration procedure.
- Forward the SIP request or response to the UE.
- Maintain a security association between itself and each UE in order to secure the SIP signalling.
- Perform SIP message compression/decompression to limit the load on the air interface.
- Authorize the use of bearer resources and manage the QoS.

S-CSCF (Serving-CSCF)

The S-CSCF performs the session control services for the UE, and maintains a session state needed for support of services. Its primary functions are described below.

- It performs session control for the registered UE’s sessions. After completion of registration, it shall reject IMS communications to/from users that are blacklisted.
- It may behave as a PS in that it accepts requests and processes them internally or forwards them on, possibly after a translation.
- It may behave as a User Agent in that it may terminate and independently generate SIP transactions.
- It provides endpoints with information related to service events such as notification of tones.
2.1 IP Multimedia Subsystem (IMS)

I-CSCF (Interrogating-CSCF)

The I-CSCF is the contact point for all connections going to or coming from an external network. It performs the following functions:

- **Registration**
  Assigning an S-CSCF to a user performing SIP registration; it is worth noticing that a given user is generally assigned the same S-CSCF, even if this user is roaming from an external network.

- **Signalling flows**
  - Obtaining the address of the S-CSCF from the HSS
  - Routing SIP requests received from another network towards the S-CSCF

HSS (Home Subscriber Server)

The HSS holds the subscriber data for a given identity, e.g. the actual location/authentication parameters. It is contacted by the P-CSCF for registration and setting QoS parameters.

2.1.4 Quality of Service

This section exposes the functionality of QoS in the IMS and its requirements on the core network. Then, there is a presentation of the logical entities involved in managing QoS, after which the signalling flows are detailed.

Description of functionalities

Quality of Service performs the following functions, from two different points of view:

*In the control plane*

- **Translation function** – This translates the QoS requirements of the communication into a set of QoS parameters for service establishment or modification. Translation is performed between external QoS requirements and the provided UMTS service, and between the QoS parameters of the different UMTS internal services.

- **Admission/Capability control** – This function maintains information about the available resources for a network entity. It determines for each service request or modification whether the requested QoS can be provided by this entity.

- **Subscription control** – The IMS framework must make sure the user requesting a service has previously subscribed to it, and that they are charged accordingly.

*In the user plane*

- **Mapping function** – This function changes the QoS parameter indication provided with a packet according to that indication relevant for the next service, at transition from one service to another.
2.1 IP Multimedia Subsystem (IMS)

- **Classification function** – This assigns packets to the established services, according to the related QoS parameters. The related QoS parameters are derived from the packet header, or are fixed for all packets of a specific UE service access point (if a physical or logical interface supports different QoS parameters without analysing each packet).

- **Resource Manager** – In the IP Connectivity Access Network (IP-CAN), this function supervises the resources for the Radio Access Bearer and allocates resources on request for services establishment or modification.

- **Traffic conditioners** – They maintain the signalling and user data traffic of an individual service, of aggregated services or of a whole network within certain limits. These limits are defined by specific QoS parameters. Services with different QoS parameter levels shall be supported by the traffic management functions. These functions ensure the provision of the QoS negotiated for a service.

**Requirements for IP Multimedia Core Network**

1. The UE shall be able to establish a dedicated signalling IP-CAN bearer for IMS related signalling or use a general-purpose IP-CAN bearer for IMS signalling traffic.

2. All messages from the UE that use a dedicated signalling IP-CAN bearer shall have their destination restricted to the P-CSCF assigned to this UE, and towards DHCP and DNS servers within the IMS operator’s domain where the P-CSCF is located.

**Logical entities managing QoS**

1. A **PDP (Packet Data Protocol) Context** is initially a logical association between a UE and an Access Point Name (APN) running across a GPRS network. As the UMTS network was the base for the IMS and is itself based on the GPRS architecture, the concept of a PDP Context is still fundamental to the IMS even though it is now supposed to be access network independent. This enables the access network to recognize which UE to send incoming data to.

   A secondary PDP context allows differentiated QoS under one APN. For example, a user can receive an audio streaming flow from a Web server while this user browses the latter’s website.

2. A **Traffic Flow Template (TFT)** is a packet filter, supplied by the UE, allowing the GGSN to classify packets received from the external network into the proper PDP context. Thus, it applies only on the downlink. TFTs control how traffic is mapped to Secondary PDP contexts; each TFT contains a combination of the following attributes:
   - Source Address and Subnet Mask
   - Destination Port Range
   - Source Port Range
2.1 IP Multimedia Subsystem (IMS)

- IPsec Security Parameter Index (SPI)
- Type of Service (TOS)(IPv4)/Traffic Class (IPv6) and mask
- Flow Label (IPv6)

3. The **Service Based Local Policy** (SBLP) manages access control and QoS parameters on both uplink and downlink. It defines what traffic the operator allows on its network and, as such, overrides the TFTs when it is applied. In case the SBLP cannot be used, the TFT will be applied. In every case, QoS parameters will be included in PDP contexts. The SBLP checks the following parameters:

- Destination IP address;
- Destination port number;
- Transport Protocol id;
- Media direction information;
- Direction of the source (needed as the SBLP is applied on both uplink and downlink);
- Media type information;
- Bandwidth parameters;
- Indication of forking/non-forking.

4. The **Policy Decision Function** (PDF) is a logical entity co-located with the P-CSCF which constitutes the decision point of the SBLP. The decisions are based on information obtained from the P-CSCF. The PDF enables coordination between events in the SIP session level and resource management in the bearer level (see GGSN below).

5. The last entity detailed here is typically the access router between the UE and the P-CSCF that constitutes the frontier to the IMS and performs the tasks assigned to the **Gateway GPRS Support Node** (GGSN) in a UMTS/IMS implementation. As the IMS originally derives from UMTS specifications, this entity will be referred to as “GGSN” in the following.

Even if the GGSN does not participate in SIP signalling, it is the SBLP Enforcement Point. In fact, the PDF has a protocol interface with the GGSN (Go interface) which supports the transfer of information and policy decisions between the policy decision point and the IP BS Manager in the GGSN (following the COPS framework).

**QoS in session setups**

This sub-section underlines the influence of QoS parameters in IMS session setups, by pointing out interactions between the PDF and its counterparts in SBLP related procedures, namely the GGSN, then the P-CSCF. For simplicity, only mobile initiated sessions are described.

**Interaction between the GGSN and the PDF**
2.1 IP Multimedia Subsystem (IMS)

Figure 2.3 shows an example of the QoS interactions implicating the GGSN during a session setup when SBLP is being applied. Because the focus is set on the originating side, the next IMS entities involved in the communication are not shown in the flow, but it is assumed that the S-CSCF or I-CSCF is the next entity after the P-CSCF in the signalling flow.

![MO Network Diagram]

**Figure 2.3:** QoS establishment between GGSN and PDF during session setup

This example is appropriate for a SIP session requesting the establishment of QoS preconditions, although only SBLP aspects are highlighted. It is assumed in this example that both the UAC and UAS have chosen to use the GPRS procedures to guarantee end-to-end QoS, which means both the UAC and UAS establish satisfactory PDP context
2.1 IP Multimedia Subsystem (IMS)

with their respective access infrastructures. SBLP decisions are taken by the PDF.

Key steps in performing the session setup are detailed below:

6. Authorise QoS resources - At the reception of the 183 (Session Progress) response at
   the P-CSCF, the P-CSCF obtains the media authorisation token from the PDF.

7. 183 (Session Progress) (P-CSCF to UE) - This message typically contains the P-Media
    Authorisation header, which holds the Media Authorisation Token. Upon receipt of
    the Media Authorisation Token, the UE generates a flow identifier which identifies
    an IP media flow associated with the SIP session. The Flow Identifiers are based
    on the sequence of media flows in the SDP. A Flow Identifier combined with the
    Authorisation Token is sufficient to uniquely identify an IP media flow.

12. GPRS: Active PDP Context (UE to SGSN) - The UE sends an Activate PDP Context
    message to the SGSN. The UE associates the PDP context to the session by includ-
    ing the media authorisation token information and the flow identifier(s) information.
    The PDP context is bi-directional.

13. GPRS: Create PDP Context (SGSN to GGSN) - The SGSN checks the user profile
    to authorise the requested QoS and also the available resource, if both are granted,
    it sends the corresponding Create PDP Context message to the GGSN. This mes-
    sage contains the media authorisation token information and the flow identifier(s)
    information.

14. COPS: REQ (GGSN to PDF) - When the Create PDP Context message is received
    in the GGSN containing the media authorisation token information and the flow
    identifier(s) information, the Policy Enforcement Point in the GGSN sends a COPS
    REQ message to the PDF. The PDF verifies that the media authorisation token
    information and the associated flow identifier(s) information are as expected.

16. COPS: RPTC (GGSN to PDF) - The GGSN sends a COPS RPT message back to
    the PDF, and includes an acknowledgement and/or an error response to the DEC
    message.

17. GPRS: Create PDP Context Resp (GGSN to SGSN) - The GGSN checks its own
    available resources and if enough resources are available, it sends a Create PDP
    Context Response message back to SGSN containing the negotiated value of the
    UMTS QoS information element.

18. GPRS: Active PDP Context Accept (SGSN to UE) - The SGSN sends an Activate
    PDP Context Accept message to UE containing the negotiated value of the UMTS
    QoS information element.

19. UPDATE request (UE to P-CSCF) - As the confirmation of the preconditions are
    requested in the 183 (Session Progress) response, when the UE finishes the QoS
    reservation for both the uplink and downlink directions, according to the GPRS
    procedures as indicated by the GPRS: Active PDP Context Accept message, it
2.1 IP Multimedia Subsystem (IMS)

sends the UPDATE request to the terminating endpoint, via the signalling path established by the INVITE request. The UPDATE request includes in the SDP the information about the successful QoS bi-directional mode, due to the successful bi-directional PDP context established. The SDP indicates that the QoS resource reservation for both send and receive mode was successful from the terminating endpoint side.

30. COPS: DEC (PDF to GGSN) - When the P-CSCF receives the 200 (OK) response to the INVITE request, the PDF sends a COPS DEC message to the GGSN to enable the use of the authorized QoS resources, i.e. to open the “gate”, and allow packet flows in both directions in accordance with the policy decision within the GGSN Policy Enforcement Point.

31. COPS: RPT (GGSN to PDF) - The GGSN receives the COPS DEV message and enables the use of the authorised QoS resources, i.e. opens the “gate” within the GGSN, and sends a COPS RPT message back to the PDF.

Interaction between the GGSN and the PDF

Figure 2.4 shows an example of P-CSCF and PDF interaction.

6. PDF generates token and authorize QoS resource – the authorize QoS resources procedure is triggered by the P-CSCF receiving a 183 (Session Progress) response. Based on the SDP information about this session, such as the end-points, bandwidth requirements, and the characteristics of the media exchange.

The PDF shall authorize the required QoS resources for the session and install the IP bearer level policy based on information from the P-CSCF. In order to ensure that the IP bearer flow correlates to the one approved during the SIP session establishment, the SIP extensions for media authorisation are used.

Based on local policy, QoS resources may also be enabled at the time they are authorised by the PDF.

The authorisation token is generated by the PDF and sent to the UE.

8. UMTS Bearer Setup - The UE uses that token to activate PDP Context from GGSN network. The PDF makes final decision to enforce GGSN network to accept or reject PDP Context activation based on SBLP.

10. Approval of QoS Commit - The Approval of QoS Commit procedure is triggered by the P-CSCF receiving a 200 (OK) message. The PDF will interact with GGSN network to open the ‘gate’ for the IP bearer.

2.1.5 Mobility related concepts

The following procedures are supported by a UE when accessing IMS:
2.2 Mobile IP

Mobile IP (MIP) is a protocol designed to manage host mobility on the Internet, at the network level. The two versions of IP have induced two versions of MIP, namely MIPv4

If a UE explicitly deactivates the IP-CAN bearer that is being used for IMS signalling, it shall first de-register from the IMS (while there is no IMS session in progress).

If a UE explicitly deactivates the IP-CAN bearer that is being used for IMS signalling while an IMS session is in progress, the UE must first release the session and de-register from the IMS and then deactivate the IP-CAN bearers.

If a UE acquires a new IP address, the UE shall re-register in the IMS by executing the IMS registration.

These concepts underline that a service from the IMS may have to be restarted when performing a handover with change of IP address.
2.2 Mobile IP

for IPv4 networks and MIPv6 for IPv6 networks. In a first step, the IPv4 version will be presented as a means to introduce concepts and show some of the evolution of Mobile IP. Then, the analysis focuses on MIPv6 and its specifications, as it will be used for the experimental part of this work.

2.2.1 Comparison between MIPv4 and MIPv6

MIPv4 – Basics

The main idea behind Mobile IP is to keep upper OSI layers unaware of mobility-induced IP address changes, as a mobile node (MN) communicates with a correspondent node (CN): the MN appears to its correspondents as always owning the same IP address that comes from what is called its home network. This was obtained mainly by adding 2 entities to the existing network infrastructure: the home agent (HA) in the MN’s home network and the foreign agent (FA) in the MN’s visited network.

When an MN with address $IP_1$ from its home network visits a new network, it is assigned a secondary IP address $CoA_1$, called a care-of address (CoA), from the FA. In order to complete the registration process, the HA has to be informed of the location of the MN. Depending on the implementation, the CoA can either be located in the FA - in which case it forwards the new CoA to the HA - or directly co-located in the MN, which leads to two possible registration schemes (see Figure 2.5).

The purpose of the HA is to tunnel all packets intended to the MN via $IP_1$ to its care-of address $CoA_1$ in the visited network (see Figure 2.6). Depending on the location of the CoA, the tunnel ends either at the FA or the MN. The tunneling is performed with IP-in-IP encapsulation (see Figure 2.7), or minimal encapsulation which prevents from repeating redundant information in encapsulated IP headers.

However, because the MN responds to the CN directly, Figure 2.6 also shows what is referred to as triangle routing: the MN sends packets from the visited network with its home address $IP_1$, with the CN replying to the same address via a tunnel through the
2.2 Mobile IP

Figure 2.6: Triangle routing in MIPv4

Figure 2.7: IP-in-IP encapsulation

HA. In practice, firewalls will often reject packets emitted from the MN, as techniques such as ingress filtering are more and more employed (e.g. to prevent IP spoofing).

To avoid triangle routing, solutions were proposed, such as reverse tunneling: the MN routes all of its requests and replies back through the HA, also with encapsulation; even then, all issues are not addressed as tunnels can be compromised, triggering man-in-the-middle attacks, and there is additional overhead. The concept of a “routing optimization” was also brought up in order to allow direct communication between the MN and the CN, but this solution was based on a set of non-standard extensions to IPv4.

Evolution to MIPv6

MIPv6 has substantial improvements compared to MIPv4, with benefits stemming in great part from the use of built-in IPv6 mechanisms [7]. As such, there is no more need for an FA: the correlation between the home address (HoA) and the current CoA – a binding – is sent by the MN directly to its HA.

In addition, route optimization is a fundamental part of the protocol. As it induces
2.2 Mobile IP

less overhead and less likely congestion in the home network, it is the desired behavior for communication to/from an MN in a visited network.

Moreover, the security issues that flawed MIPv4 have been addressed, with for instance IPSec protection of IP tunnels between the MN and the HA, support for ingress filtering, etc.

2.2.2 MIPv6 operation

When an MN is in its home network, MIPv6 requires nothing specific from any node involved, so the specifications for MIPv6 only apply when in a visited (or foreign) network. However, they can be split into two categories: the operations performed so as to maintain connectivity after an IP address change and those concerning the actual communication with a CN.

Maintaining connectivity

Firstly, the MN obtains a CoA through conventional IPv6 stateful or stateless mechanisms. Then it registers the new CoA with a router on its home network acting as HA: this is called the Binding Update (BU) operation. Every BU to the HA is protected by (preferably offline) pre-arranged IPSec associations between the MN and the HA. In addition, the HA verifies that the HoA of the MN requiring the BU is from a network domain which it actually performs as HA for. These preliminary precautions thus allow for a simple registration scheme, illustrated in Figure 2.8.

![Figure 2.8: Binding Update to the HA in MIPv6](image)

Communication with a correspondent node

When the MN is in a foreign network, it can exchange data with a CN following one out of two possible schemes. The default mode is referred to as bidirectional tunneling through
the HA and requires no MIP capabilities from the CN (see Figure 2.9). In bidirectional
tunneling, the CN sends data packets to the HoA of the MN. The HA intercepts these
packets, as the BU performed beforehand has made it aware of the MN’s not being
in its home network, then it forwards the data packets to the CoA of the MN via IPv6 encapsulation. Packets from the MN are tunneled back through the HA via the same IPv6 encapsulation mechanism, decapsulated by the HA, then forwarded to the CN using the HoA as Source Address. Hence, this method requires the MN and the HA to perform IPv6 encapsulation and decapsulation so as to tunnel data packets. These tunneled packets can be protected by IPSec between the HA and the MN in order to prevent attacks on bindings.

![Figure 2.9: Bidirectional tunneling in MIPv6](image)

The second communication mode requires the CN to be MIPv6-compliant so as to implement route optimization. A first phase for the CN consists in registering bindings through a two-step scheme. The first step, known as the return routability procedure, makes sure that the MN actually owns both the HoA and the CoA it claims to own. Figure 2.10 shows how: the MN initiates the procedure by sending the CN a Home Test Init message (HoTI) via its HoA and a Care-of Test Init message (CoTI) via its CoA; the CN then replies with a Home Test (HoT) message and a Care-of Test message (CoT) to the given HoA and CoA, respectively. These four messages contain parameters used for computing authentication keys necessary to proceed with the second step of binding registration, a binding update/acknowledgement scheme similar to that with the HA, shown in Figure 2.11.

After the binding registration to the CN, the second phase is a direct data exchange between the MN and the CN using the CoA (see Figure 2.12). However, specific headers/options for packets transmitted via route optimization provide with the HoA, so as to facilitate upper layer transparency (in case of multiple home addresses, for instance). These headers/options may also be used by administrators to perform more accurate access control.
2.2 Mobile IP

2.2.3 Modifications to the IPv6 protocol

Operations that are specific to MIPv6 need to be identified as such. Hence, the IETF has approved a set of additions and modifications to the standard IPv6 protocol which are presented in the following.

Mobility Header

Defined in [7], the Mobility Header is an extension header used by mobile nodes, correspondent nodes, and home agents in all messaging related to the creation and management of bindings. It is identified by a Next Header value of 135 in the immediately preceding header and has the format represented in Figure 2.13. There are 8 types of Mobility Headers, identified by the MH Type field and detailed in the following.

1. The Binding Refresh Request message (MH Type = 0) is sent by a CN, requesting an MN to update its mobility binding in order to avoid its expiration, which would imply a new BU to the HA, temporary bidirectional tunneling, i.e. unnecessary overhead.

2. The Home Test Init message (MH Type = 1) initiates the return routability procedure and starts the parameter exchange which secures registration to a CN, on the HoA path. It is sent from the MN to the CN.

3. The Care-of Test Init message (MH Type = 2) initiates the return routability procedure and starts the parameter exchange which secures registration to a CN, on the CoA path. It is sent from the MN to the CN.

---

Figure 2.10: Return routability procedure

![Diagram of Mobile IP processes]

- MN (Mobile Node)
- HA (Home Agent)
- CN (Correspondent Node)

- Home Test Init (HoTI)
- Care-of Test Init (CoTI)

- Home Test (HoT)
- Care-of Test (CoT)
4. The Home Test message (*MH Type = 3*) is a response to the Home Test Init message and is sent from the CN to the MN via its HoA.

5. The Care-of Test message (*MH Type = 4*) is a response to the Care-of Test Init message and is sent from the CN to the MN via its CoA.

6. The Binding Update message (*MH Type = 5*) is used by the MN to notify other nodes of a new CoA for itself. As seen in Figure 2.14, several fields are worth noting: the Acknowledge (A) bit is set by the sending mobile node to request a Binding Acknowledgement (BA) be returned upon receipt of the BU; the Home

<table>
<thead>
<tr>
<th>Payload Protocol</th>
<th>Header Length</th>
<th>MH Type</th>
<th>Reserved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Checksum</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Message Data

**Figure 2.13:** *Mobility Header format*
2.2 Mobile IP

Registration (H) bit is set by the sending MN to request that the receiving router should act as its HA; then different Mobility Options can be appended, such as the Binding Authorization Data option (that is mandatory in binding updates to a CN as it provides authentication parameters derived from the return routability procedure) or the Alternate Care-of Address option when an alternate CoA is owned by the MN.

<table>
<thead>
<tr>
<th>Payload Protocol</th>
<th>Header Length</th>
<th>MH Type = 5</th>
<th>Reserved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Checksum</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>H</td>
<td>L</td>
<td>K</td>
</tr>
</tbody>
</table>

Mobility Options

Figure 2.14: Format of a Binding Update message

7. The Binding Acknowledgement message (MH Type = 6) is used to acknowledge receipt of a Binding Update and is described in Figure 2.15. The Status field in the BA is used to indicate the disposition of the BU: values less than 128 indicate that the BU was accepted by the receiving node (e.g. 0 – Binding Update accepted), while values greater than or equal to 128 indicate that the BU was rejected (such as 128 – Reason unspecified or 133 – Not home agent for this mobile node).

<table>
<thead>
<tr>
<th>Payload Protocol</th>
<th>Header Length</th>
<th>MH Type = 6</th>
<th>Reserved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Checksum</td>
<td></td>
<td>Status</td>
<td>K</td>
</tr>
<tr>
<td>Sequence Number</td>
<td></td>
<td></td>
<td>Lifetime</td>
</tr>
</tbody>
</table>

Mobility Options

Figure 2.15: Format of a Binding Acknowledgement message

8. The Binding Error message (MH Type = 7) is used by the CN to signal an error related to mobility, such as an inappropriate attempt to use the Home Address destination option without an existing binding (see further).

Information about the HoA

The following additions to IPv6 carry information about the HoA of the MN involved, be it source or destination of the data packets. Firstly, the Home Address option is a new type (Option Type = 201) of the IPv6 Destination Option extension header (Next Header value = 60). It is sent by the MN while away from home, to inform the recipient of its HoA, and is the marked part of the IPv6 header represented in Figure 2.16. The use of this option does not compromise bindings, with the requirement that it be accepted by a CN only when a current binding is registered for the sending MN.
2.2 Mobile IP

Secondly, a new IPv6 Routing Header was defined, simply called Type 2 Routing Header (see Routing Type field in Figure 2.17). This allows packets to be routed directly from a correspondent to the MN on its CoA, inserted into the IPv6 Destination Address field. Once packets arrive at the CoA, the MN retrieves its HoA from the Type 2 Routing Header, and this is used as the final destination address for the packet.

**Figure 2.16: Format of a Home Address option within its IPv6 header**

<table>
<thead>
<tr>
<th>Next Header</th>
<th>Header Extension Length</th>
<th>Option Type = 201</th>
<th>Option Data Length = 16</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Home Address</td>
<td>HoA</td>
</tr>
</tbody>
</table>

**Figure 2.17: Format of a Type 2 Routing Header**

**Additions to ICMPv6**

The first addition fixes the problem that arises when the MN does not know the IP address of a potential HA. In this case, the Dynamic Home Agent Address Discovery procedure (see Figure 2.18) allows the MN to request a list of HA addresses from the home network’s anycast address reserved for MIPv6 Home Agents.

**Figure 2.18: Dynamic Home Agent Address Discovery procedure**

Moreover, the Mobile Prefix Discovery mechanisms (see Figure 2.19) allow the MN to
2.2 Mobile IP

be aware of a home network renumbering, while away from home – for instance enabling the MN to change its HoA even though it is in a foreign network. This Mobile Prefix Discovery can be initiated by the MN with a Mobile Prefix Solicitation message sent to the current HA; the latter replies with Mobile Prefix Advertisement message. In addition, the HA can send unsolicited Mobile Prefix Advertisements whenever prefix information on the home link changes.

![Diagram](image)

(a) With solicitation from the MN

(b) Unsolicited Advertisement from the HA

Figure 2.19: Mobile Prefix Discovery mechanisms

Then, some messages of the Neighbor Discovery protocol have been edited, such as Router Advertisements which enable nodes to know if the sending router can operate as HA, thanks to a Home Agent (H) bit taken from a Reserved field.
Chapter 3

Analysis

In this chapter, the actual integration of MIP in an IMS environment will be developed. At first, we will describe the traffic flows that derive from a mobile user’s roaming in an IMS implementation. Then, with this knowledge and that accumulated from Chapter 2, we will explore issues that come with combining MIP and the IMS architecture.

3.1 Traffic flows

This section will describe basic IMS procedures, though here occurring in a roaming situation. The first two procedures appear in a chronological order: user registration, communication between S-CSCFs on the originating and terminating sides of the session. Then, the initiation of data sending from the CN is detailed. Lastly, the specific handover case with an on-going session will be presented.

3.1.1 Registration

Once the UE is registered to the IP-CAN, it can start the application level registration. In Figure 3.1, the user is considered to be always roaming. If not, the home network shall perform the role of the visited network elements and the home network elements.

Here is a description of the traffic flows:

1-2. The UE sends the register information flow the the proxy: public user identity, private user identity, home network domain name, UE IP address.

3-4. The I-CSCF sends Cs-Query/Cx-Select-Pull information flow to the HSS which will check if the user is already registered and if he is allowed to do so in that P-CSCF network according to the user subscription and any operator limitations/restrictions. The HSS sends Cs-Query Resp/Cx-Select-Pull Resp back to the I-CSCF. It shall contain the S-CSCF name/capabilities or it shall reject the registration attempt.

6-7. The S-CSCF sends Cs-Put/Cx-Pull (public user identity, private user identity, S-CSCF name) to the HSS, which stores the S-CSCF name for that user and returns
3.1 Traffic flows

Figure 3.1: IMS - Registration flow

the information flow Cx-Put Resp/Cx-Pull Resp to the S-CSCF, with names/addresses information to access the platform(s) used for service control.

8. The S-CSCF sends registering information to the service control platform and performs any service control procedure that is appropriate.

9-11. S-CSCF returns the 200 OK information flow to the UE via the I-CSCF and the P-CSCF.

3.1.2 Session flows - serving to serving procedure

Once the registration to the IMS is done, different procedures may occur. Figure 3.2 is an example of a serving to serving procedure – communication between two S-CSCFs – assuming different operators. Only key steps are detailed afterwards.

The following procedures occur in this case:

1. SIP INVITE request is sent from the UE to S-CSCF#1. This message should contain the initial media description offer in the SDP.

2. S-CSCF#1 invokes any service logic is appropriate for this session attempt.

9. The media stream capabilities of the destination are returned along the signalling path.

16-17. S-CSCF#2 forwards the offered SDP to the terminating endpoint. The terminating endpoint acknowledges the offer with answered SDP.
3.1 Traffic flows

Figure 3.2: Serving to serving procedure - different operators
3.2 Integration issues

3.1.3 Session flows - Data sending establishment

This section describes the procedures leading to data sending from the CN (see Figure 3.3). The description starts upon the UE’s obtaining its IP address.

The description starts upon the UE’s obtaining its IP address:

1-5. IP address attribution and Primary PDP context setup.

6-16. IMS registration.


3.1.4 Session flows - Handover

This section focuses on a macro-mobility case: a UE is communicating with a CN when it experiences a handover to a new IP-CAN. Figure 3.4 shows the IMS flows associated with such a situation.

These are the main steps of the procedure:

1. An on-going session is assumed, the user is then already registered in the IMS.

2. The UE moves from the first IP-CAN coverage area to a second one.

3-12. Network initiated session release.

13-17. IP address attribution and Primary PDP context setup.

18-28. IMS registration.

3.2 Integration issues

The following section is aimed at exposing the issues that stem from the actual integration of MIP mechanisms in an IMS environment. Although the final objective is to study the macro-mobility case during a handover, problems occur even in a static environment, as soon as the UE launches MIP specific procedures in a roaming situation.

The analysis was conducted in a sequential manner, with issues arising in increasingly complex scenarios, as integration issues are encountered even with no handover consideration. The network architecture is presented in Figure 3.5.

The architecture described in Figure 3.5 is to be interpreted with the following assumptions:

- The UE/MN is away from its home network from an MIP point-of-view
- The UE/MN stays within its operator’s administrative domain, although potentially moving between different subnetworks such as, for instance, a UMTS cell and WiFi hotspot.
3.2 Integration issues

Figure 3.3: Message flow from the connection to the IP-CAN until the first packet sending
3.2 Integration issues

Figure 3.4: Handover in IMS context
3.2 Integration issues

Figure 3.5: Network architecture retained for the analysis
3.2 Integration issues

- The topology of the operator’s subnetworks is not investigated. Hence, the GGSN, the P-CSCF and the HA are reachable from one another, with no concern of exactly how this may happen.

3.2.1 Switch-on

The issues described in this section are encountered when the user terminal is switched on in a foreign network.

**Issue A: MIP Binding Updates to the HA**

After acquiring its new CoA, the MN must perform a BU with its HA prior to any communication with a correspondent node, including the new P-CSCF in the visited network. Thus, it is very important to be able to contact the HA as soon the new CoA is acquired. However, until a Secondary PDP context is assigned, the UE can communicate only with the P-CSCF, so the BU to the HA cannot be performed.

**Issue B: Registration to the P-CSCF**

If prior communication between the UE/MN and the HA is impossible, the UE/MN cannot communicate with any other node. Moreover, even if the previous issue is addressed, the P-CSCF checks the Source Address in the signalling flow coming from the UE against mismatches between the SIP and IP layers. As MIP hides the mobility from the SIP layer, the P-CSCF can only know of the HoA on this layer, whereas on the IP layer it can receive either the HoA or the CoA depending on the operation mode. This could cause IMS registration to fail.

In addition, as the P-CSCF is the IMS entry point for the UE, a bidirectional-tunnelled connection (via the HA) – which is the default behavior for MIPv6 – would prove costly on the long haul with respect to performance.

3.2.2 Communication with a correspondent node

The final recipient is the CN, not the HA. The end-to-end QoS through the operator’s infrastructure is not considered so as to keep a manageable degree of complexity, hence only access control issues have been fully assessed. Access control is managed by the SBLP and TFT filters following the method presented in Figure 3.6. In the downlink, they operate sequentially: the SBLP takes care of the operator policy, while the TFTs link incoming IP flows with PDP contexts.

Similarly to the P-CSCF Source IP Address check, the filtering entities are only aware of the HoA of the UE and the IP address of the CN. Hence, Figure 3.7 shows there are several filter mismatches (issues C, D and E) impeding the communication between an MIP-compliant UE and its correspondent, in both bi-tunnelling and route optimization modes. The symmetry of the diagram can be used to model both cases for the UE, whether it is a mobile node or communicating with one.

**Issue C: Tunneling through the HA**
3.2 Integration issues

**Figure 3.6:** Filtering of IP packets at the GGSN

**Figure 3.7:** Filtering issues with MIP packets
3.2 Integration issues

- Mismatch 1: SBLP Destination Address in uplink (packets destined to the HA whereas the filter expects the CN)
- Mismatch 2: TFT Source Address in downlink (packets from the HA whereas the filter expects the CN)
- Mismatch 3: SBLP Destination Address in downlink (packets destined to the CoA whereas the filter expects the HoA)

**Issue D: Binding unknown in route optimization**

- Mismatch 4: TFT Source Address in downlink on receiving side for CoTI (packets from the CoA whereas the filter expects the HoA)
- Mismatch 5: SBLP Destination Address in uplink on receiving side for CoT (packets destined to the CoA whereas the filter expects the HoA)
- Mismatch 6: SBLP Destination Address in downlink on originating side for CoT (packets destined to the CoA whereas the filter expects the HoA)

3.2.3 Handover

**Issue E: Transfer of QoS parameters**

The new sub-network in the operators domain may not offer the same QoS guarantees (e.g. WiFi and UMTS), so the QoS parameters from the previous network have to be retrieved and compared to those of the new network, if a session was established with a correspondent node.

**Issue F: Unused resources in previous network**

- The “old” P-CSCF and GGSN have reserved resources for a now absent UE. These resources should be released as soon as possible when the UE no longer needs them.
- As the “old” P-CSCF was a CN (on the signaling path), the UE could keep trying to reach it; this has to be prevented.
Chapter 4

Solution Proposals

This chapter contains the proposed solutions to the issues described in Section 3.2. As the problems have appeared sequentially, the solutions are designed with full consequences of those preceding. For instance, the solution proposed in Section 4.3 benefit from those designed in Section 4.1 and Section 4.2.

4.1 Issue A: MIP Binding Updates to the HA

The proposed solution is based on letting the BU to the HA pass through the Primary PDP Context. Therefore, the GGSN is made fully aware of MIPv6 specifications and should be able to recognize this type of communication with the HA:

- The operator maintains a HA Anycast Address List based on network topology, i.e. for each IP subnetwork with a unique prefix there is a HA Anycast Address entry. As such, this list is considered to be static, so it can be forwarded to GGSNs over the operator’s domain with specific management tools which are not explicitly defined in this work.

- The GGSN dynamically maintains a HA List based on the Dynamic HA Address Discovery procedure performed by UEs that do not know the IP address of their HA, or try to use a HA which is not in the HA List (see Figure 4.1):
  - If the UE knows a HA, it tries to perform a BU. If the this HA is in its HA List, the GGSN forwards the packet (a). Otherwise, the GGSN silently discards the packet.
  - If the UE does not know a HA or its BU has timed out (e.g. because the GGSN discarded the packet) the UE will then initiate Dynamic HA Discovery with its home networks anycast address (b) and will be able to perform its BU afterwards.

- The GGSN then checks if the packet destined to the HA is a BU, with the Home Registration bit and the Mobility Header Type field inside the Mobility Header; another check is performed on the reply which must be a BA coming from the HA.
4.2 Issue B: Registration to the P-CSCF

Any packet is silently discarded if a problem occurs: this prevents any abusive use of the special treatment granted to packets to/from the HA regarding the Primary PDP context.

Note: It is indispensable to use the Authentication Header as IPSec protocol instead of ESP, otherwise the payload is encrypted end-to-end between the MN and the HA, thus the GGSN cannot check the Mobility Header.

4.2 Issue B: Registration to the P-CSCF

The idea is to make the P-CSCF aware of the binding on the SIP level (see Figure 4.2):

- The P-CSCF is made fully compliant to MIPv6 specifications regarding CNs
4.3 Issue C: Tunneling through HA

- The UE must perform a BU to the P-CSCF as soon as possible; slight modifications to MIP procedures may be necessary to prioritize the P-CSCF.
- The protocol stack of the P-CSCF must be modified, especially the MIP socket interface, so as to associate the CoA to the HoA on the SIP level.

![Figure 4.2: BU to the P-CSCF](image)

The knowledge of the Binding must not go further than the P-CSCF on the SIP level, otherwise no real benefit can be drawn from using network layer mobility. This solution also provides with a performance enhancement as the UE and the P-CSCF can then have direct communication, not delayed by tunneling through the HA.

4.3 Issue C: Tunneling through HA

The GGSN must decapsulate packets to and from the HA for access control, as described in Figure 4.3:

- The HA is known to the GGSN from its HA List, so all packets sent to or received from the HA can be processed accurately. As these packets go through a Secondary PDP Context, this processing does not affect the treatment of the BU to the HA in the Primary PDP Context.
- For packets (whose outer headers have the HA as source/destination) which do not correspond to any filters AND have a Next Header value corresponding to IPv6 encapsulation, the GGSN must filter with the encapsulated packet instead.
- The GGSN discards the entire packet if the inner packet still results in a mismatch.

4.4 Issue D: Binding unknown in route optimization

The P-CSCF (PDF) provides the SBLP with a CoA/HoA correlation on the originating side to counter mismatch 6 (see Figure 3.7). In a simple scheme, double SBLP entries could be used, duplicating all fields from the original entry corresponding to the HoA into a second one with identical fields except for the Destination Address (see Figure 4.4).

The two issues encountered on the receiving side cannot be fixed effectively. Depending on the choice of the correspondents operator (see below), one of two problems remains.
4.5 Issue E: Transfer of QoS parameters

**Figure 4.3:** GGSN analysis of packets tunneled through the HA

**Figure 4.4:** Informing the SBLP of the CoA: double entries

a. **Route optimization is allowed:** A BU has to be delivered to the CN as the first step, so the RR procedure has to be performed. As there is no way to know of the Binding from the MN beforehand, all CoTI/CoT must be allowed to/from the CN no matter what. Hence, this opens a security hole for flooding attacks on all nodes expected to operate as a CN in MIPv6 route optimization.

b. **Only bidirectional tunneling is allowed:** Traffic to/from any of this operators CNs is routed through a HA when in communication with an MN. Even though the system works safely, MIPv6 is never used at its best with respect to reducing overhead.

### 4.5 Issue E: Transfer of QoS parameters

A new procedure is designed so as to make the new P-CSCF (P-CSCF2) aware of the QoS parameters used in the previous one (P-CSCF1) and is presented in Figure 4.5. A simple request/reply scheme is used: P-CSCF2 sends the new message type *QoS_Param Request* to P-CSCF1, which responds with the requested parameters (e.g. SBLP parameters) via the corresponding new type *QoS_Param Reply*. If these parameters are acceptable for P-CSCF2, they are used “as is” to program the involved entities (e.g. PDF, GGSN).
4.6 Issue F: Unused resources in previous network

If the parameters are refused by P-CSCF2, then it tells the UE (with a new message type, QoS_Param Reject) to perform a QoS renegotiation with its correspondent, via SIP UPDATE/PRACK messages. If this negotiation fails, P-CSCF2 initiates session termination.

4.6 Issue F: Unused resources in previous network

Under the assumption that the UE shall not perform a handover to its previous network, P-CSCF2 can send P-CSCF1 a QoS_Param Ack upon reception of the QoS_Param Reply. Hence, the unused resources in the previous network can be freed. This function is described in figure 4.5.

**Figure 4.5:** Transfer of QoS parameters
Chapter 5

Experimental work

5.1 IMS/SIP operations

5.1.1 Presentation of the new SIP client

In the laboratory, an implementation of SIP is already available. Provided by Siemens Germany, this implementation permits to establish SIP sessions, through a P-CSCF, an I-CSCF and two S-CSCF; to use it, UAC/UAS program is available, in order to test the system.

The main purpose of the existing UAC/UAS implementation is to read SIP requests from an input file. The sent requests are read as well as the received requests. There is no interaction: for example, if a request fails, the next one will be sent, even if there is no sense to do it.

Because we need to implement a handover scenario, we need to have more interactivity. And because we cannot predict the exact time of the occurrence of a handover, we are not able to use this implementation. So we have re-written completely a new SIP client, in order to achieve our goal.

The new implementation uses multiple threads (see [8]), in order to listen the incoming packets, to send SIP messages (according to the RFC 3261: a request can be sent several times, if the response does not arrive) or to check IP address changes. Using simple interface functions, we make the SIP protocol become transparent for the application.

5.1.2 Architecture of the new SIP client

By using the UDP protocol, our implementation allows an application to open multiple SIP sessions at the same time. The main architecture of our implementation is represented in Figure 5.1.

This architecture is composed of main 4 different blocks:
5.1 IMS/SIP operations

![Architecture of the new sip client](image)

**Figure 5.1: Architecture of the new sip client**

- **Application interface**: composed by 6 functions, and 5 call-back functions (_ready, _received, _terminated, _invitation, _negotiation), this block is the link between any application and our implementation.

- **Send packets**: used to send packets through the network, it is composed of three main kinds of packet. The SIP packets correspond to the packets sent to IMS. The data packets are directly sent to the correspondent node. The SIP special packets are an exception due to the actual IMS implementation. More details are given in the Section 5.1.4. When sending a SIP packet, the procedure is to send it several times, until the response arrives.

- **Receive packets**: it corresponds to some threads listening for incoming packets. There are data packets, and SIP packets. In the SIP category, there is a distinction between a request and a response, in order to stop a request being sent if the corresponding response has been received.

- **Request treatment**: There are three functions to treat a type of request. For exam-
5.1 IMS/SIP operations

For example, for the INVITE request, the initiate function is used when the sip client invites someone. The response function is called when the system receives a response to its INVITE request. The receive function, is used when the system receives an INVITE function (someone else invites us). There are different treatments for each request: REGISTER, INVITE, PRACK, UPDATE, CANCEL, BYE, OPTIONS, ACK.

In Figure 5.1, we present the main interactions, among each entity. But each entity uses different threads, in order to listen to the incoming packets on different ports, or to check IP address changes. Threads are not processes, and they use the same memory space. To protect data, the sipclient also use multiple mutex.

Below, we explain the different interactions represented in the Figure 5.1:

1. When the SIP system is started, a new thread is created, in order to listen the SIP packets incoming. The StartSip function is blocked: to exit the system, the application has to call the CloseSip function.

2. When the initialization is over, the StartSip function calls the call-back function _ready(), in order to warn the application. The StartSip function will return a value only when the CloseSip function will be called.

3. After the initialization of the system, the sipclient implementation is prepares the REGISTER request.

4. When the application is closes the system, an exit signal is sent to the received packet threads, in order to unblock the StartSip function.

5. To open a new SIP session is a long process, as described in the background. By calling this function, the user just prepares the INVITE request. Then the function finishes, and all opening processes continue to be performed. The evolution of it can be checked with the StateSessionSip function.

6. As for opening a session, a BYE request is sent when the application needs to close a session.

7. This arrow corresponds to a sending of a new request.

8. This arrow corresponds also to a sending of a new request. It is a special case, due to the actual SIP implementation (explained later). This case is used to send the PRACK, UPDATE and BYE requests.

9. When a SIP message comes in, there is a distinction made between response and request. If it is a response, the system checks in its buffer, and tries to find the corresponding request. In its memory, the system has also put the call-back response function.

10. If the packet is a request, then the predefined function for the type of received request is called.
5.1 IMS/SIP operations

11. When a new request has been received, the system has to send a response. Depending on the type of request, the response is sent normally or with the special path.

12. When receiving the Session in Progress response for the INVITE request, or the response for the UPDATE request, the system call the `_negotiation` function, in order to allow the application to deal with the SDP parameters.

13. If the system is not the initiator of the request, the negotiation is done when receiving the INVITE and UPDATE requests.

14. If the system receives an INVITE request, the `_invitation` function is called, in order to ask the application if the new session is accepted.

15. If the system receives a BYE request, the `_terminated` function is called, in order to warn the application that the session is over.

16. The SendViaSessionSip allows the application to send data packets directly to the correspondent node.

17. The thread listens the data packets created during the SIP session. Then all data packets incoming are transmitted to the `_received` call-back function.

18. When SIP is started, a thread is created in order to check continually the IP address. Then, if this one changes, the application is able to perform a new REGISTER, and send a REINVITE for all opened session.

5.1.3 Laboratory configuration

As shown in Figure 5.2 the present laboratory configuration is using 4 PCs. It is possible to reduce this number to three, by having the UAC and the UAS in the same PC. But the final goal of our project is to perform an handover. By separating the UAC and the UAS, we can perform the handover only on the UAC.

**UAC/UAS**

The UAS (User Agent Server) is the entity waiting for incoming INVITE request. The UAS accepts INVITE requests, establishes sessions, and send data. It stops to send data when the correspondent node sends the BYE request. The interval between 2 data packets is randomized between 0 and 50 ms.

The UAC (User Agent Client) is the entity performing the invitation. It initiates the INVITE request, and waits for reception of data. Necessary, the registration of the UAS has to be performed before the registration of the UAC. Otherwise, the IMS will send back a 404 error for the INVITE request.

The UAC and the UAS can be located in the same machine. In this case, they can use both the port number 5060. In the laboratory configuration, the UAC is located in the
5.1 IMS/SIP operations

Figure 5.2: Experimental configuration of the SIP system

machine 192.168.2.4, and the UAS in the machine 192.168.2.3. So, in this case, they can both use the port 5060. But for resource allocation, it is preferable to use a different port numbers for every connexion. In our example, the UAC used the port 32870 as SIP port, and the port 32871 as data port. The UAS used the port number 1074 for SIP messages, and the port 1075 for data. But for next sessions, these ports wouldn’t be necessary the same.

**P-CSCF**

The P-CSCF is the entry point of all SIP messages. Because the standard SIP port number is 5060, the P-CSCF uses this one. The IP address of the machine where the P-CSCF is located is 192.168.2.1.

**S-CSCF**

The S-CSCF is located in a dedicated machine, and associated to the port number
5.1 IMS/SIP operations

5061. The IP address of the machine is 192.168.2.2.

**I-CSCF**

The I-CSCF is located in the same machine as the P-CSCF (192.168.2.1). So, it cannot use the same port number as the P-CSCF: it uses the port number 5062.

**Filter**

We have built the filter program, in order to make our implementation work with the present SIP system, more details are given in the Section 5.1.4. Our filter uses a predefined port number 5051. It has been arbitrarily chosen, and has to be known by the UAC and the UAS.

**DNS**

Like the filter, the DNS program permits to make our implementation work. It solves a problem due to the handover scenario. More details are given in Section 5.1.5. We have arbitrary chosen the port number 5052. This one has to be known by the UAC and the UAS before any registration.

### 5.1.4 Problem & solutions due to the present IMS system

**Problem**

In the new sipclient implementation, the IMS flow described in the background section is respected. The requests PRACK and UPDATE are used, in order to negotiate the SDP parameters and to activate the secondary PDP context.

But the IMS system available in the laboratory is a basic implementation. It allows to connect directly a user to the P-CSCF, but there is no GO interface. Moreover, the system is old, and cannot interpret the PRACK and UPDATE requests. When sending this type of requests, the system returns an error 483, corresponding to a "Call/Leg Transaction Does Not exist".

By using only our implementation, and the present SIP system, it is not possible to simulate IMS flow.

**Different solutions**

To solve this problem, we have seted different solutions:

1. To look in the SIP system code, and enable the PRACK and UPDATE requests by changing the code. But this implementation is very large, and it would take too much time to analyse the code.

2. To use different names for these specific requests. Instead of using the PRACK name, we can just use another name, already known by the SIP system. This solution is very simple, but we need at least two different other requests. In fact, in our implementation there is a distinction between PRACK and UPDATE. On another side, using other requests can affect inexpertly the SIP system behaviour. For these two reasons, we dismissed this solution.
5.1 IMS/SIP operations

3. To send the PRACK and UPDATE requests directly to the corresponding nodes. By doing this, the request does not go through the SIP system (the implementation is not able to understand them). If the UAC/UAS sends the request directly to the correspondent node, they need at least the IP address and the port number. The UAS can find the UAC port number by analysing the contact field of the INVITE request. But the UAC can never find the SIP port number of the UAS: the UAS replies by copying all the field of the request.

4. To create a filter in the same machine as the P-CSCF. Instead of sending the SIP packets to the port number 5060 of the P-CSCF, we send the packet to another port (5051 for example). Then, the filter analyses the packet. If it is a PRACK or UPDATE requests (or response), it forwards the packet directly to the corresponding node. If not the packet is forwarded to the port number 5060 (SIP port number).

The main disadvantage of this solution concerns the IP layer. By sending a packet from the filter to the P-CSCF, we change the IP address and the port number in the corresponding layers. When the packet is going to be analysed by the P-CSCF, this entity would get a wrong IP address, and a wrong port number from the TCP/IP layer.

The chosen solution

Our final solution is a mix between the third and the fourth. The main idea is to build a filter. But only the PRACK and UPDATE requests will be sent there.

By doing this, the normal SIP requests are sent directly to the IMS: the IP layer of the message will contain the correct address. The PRACK and UPDATE requests never go through the IMS, and are sent to the filter, which forward them directly to the correspondent node. The architecture of the filter is very simple (see Figure 5.3).

In the filter, there are two memory cases, one for each corresponding nodes. Each node has to register in the filter by using a special SIP request called "SPECIAL_REQUEST". To register, the node uses its Call-ID (for registration). Then, if the node changes its IP address, it can register again, by using the same Call-ID.

When the filter receives a packet, it first looks where it comes from, and then sends it to the other node. If the packet does not come from one of the two registered nodes, it is discarded.

Using the filter make our sipclient implementation working. But the filter can not manage more than two nodes. So, it must be restarted again before every registration. So, the SIP session establishment can be done with only two nodes, even if these two nodes can open multiple sessions, as our implementation did.
5.1 IMS/SIP operations

5.1.5 Problem & solution due to the handover procedure

**Problem**

The experimental IMS implementation does not manage the mobility. In another way, our IMS system can allow a user to be registered with two different IP addresses. If a user tries to register itself with two different IP addresses, the IMS system will forward a conflict error.

Now, the actual IMS supports mobility and allows user to be registered in different IP addresses. But we are working with an old IMS version, and we have to implement the mobility, by avoiding as much as possible to change the IMS code.

**Solution**

In the experimental IMS, a user can register by using its IP address, or the hostname
of its machine. Then, the P-CSCF analyzes the SIP message, in order to obtain the IP 
address, or the hostname.

In IMS it is not possible to deregister a user: no existing request does it. However, it 
is possible to fix a life time for a registration. Usually, it is 3600 seconds (1 hour). This 
parameter can be fixed with the register request by using the Expires tag. But, it is not 
possible to determine the exact time of the handover. So we cannot use this option to 
solve our problem.

A solution is to not register the user with his IP address, but with the hostname of 
its machine. By using the IP address, the P-CSCF will compare two different strings: 
one containing the IP address before the handover, and the other one containing the new 
one. The P-CSCF will find a conflict. But if the user registers with the hostname of its 
machine, the P-CSCF is going to have the identical string: the hostname of the machine 
does not change after the handover.

Then, the P-CSCF will send SIP messages using the hostname of the user. To do 
it, it will use a local DNS (program “named”). The disadvantage of this solution is that 
the user must be sure that the DNS is updated after a handover: it is not possible to 
know the new IP address before the handover occurs. Even if it was possible, the DNS 
cannot have two times the same hostname with two different IP addresses: the user has 
to perform a DNS update.

To update the DNS, we have created a new program (as the filter), which will listen 
to the port 5052. When a user needs to perform a DNS update, it sends his hostname at 
this port (in the same machine as the P-CSCF). Our program updates the configuration 
file and restarts the DNS. In the end, the program sends a response to the user, in order 
to let him continue his transaction. The DNS update has to be done before sending any 
SIP message, otherwise the P-CSCF will not be able to find the IP address of the user.

In term of performance, the DSN update is very fast, and does not influence too much 
the final measurements: it takes less than 5 ms, when the total registration time takes 
more than 70 ms.

5.1.6 SIP Experimental flow

In Figure 5.4, there is a presentation of the global experimental message flow. The 
main differences compared to the background section concern the PRACK and UPDATE 
requests, the special register and the DNS registration.

The UAC and the UAS use also the same entry point. So the traffic flow in the 
P-CSCF is bigger than the one described in the background section (it supports two 
correspondent nodes). In our experimental setup, only two correspondent nodes are con-
sidered. And in reality, a P-CSCF supports more than one connection/user.

The main steps of the message flow are described below:

1. The UAS makes the DNS update, in order to inform the P-CSCF about its IP 
address.
5.1 IMS/SIP operations

Figure 5.4: Experimental flows until an opened SIP session
5.1 IMS/SIP operations

2. The UAS registers first. It sends the SPECIAL_REGISTER to the filter in order to be registered.

3. The UAS proceed to the REGISTER step. It sends the REGISTER to the SIP system, and expects to receive a 200 OK response.

4. The UAC makes the DNS update, in order to inform the P-CSCF about its IP address.

5. The UAC registers after the UAS. Then it sends the SPECIAL_REGISTER to the filter.

6. Then, the UAC proceeds to the normal registration.

7. The UAC tries to establish a SIP session with the UAS by sending the INVITE request.

8. When the UAS receives the INVITE request, it immediately sends a Trying, to warn the UAC that the request has been received and will be treated.

9. When the UAS accepts the invitation, it sends back a "Session in Progress", in order to specify new SDP parameters, and continue the opening procedure.

10. The UAC sends the PRACK request to confirm the SDP parameters, and the UAS replies back. The request is sent to the filter, and does not go through the SIP system.

11. The UAC sends the UPDATE request to activate the context, and the UAS replies back. The request is sent to the filter.

12. The UAC sends the PRACK request to confirm the SDP parameters, and the UAS replies back. The request is sent to the filter.

13. The UAS sends another response to the INVITE request in order to prevent the UAC that it is ringing.

14. The UAS sends a 200 OK to the invitation, to warn the UAC that the data transmission can begin.

15. The UAC sends the ACK request to confirm the session establishment to the UAS.

Compared to the standard IMS flows, two messages have been added: the DNS update, and the special register:

- The first one is sent and a response is expected. In fact, before sending any message to the P-CSCF, the DNS server has to be updated. The processing time is very fast: change a file and restart the DNS server. It does not influence too much the global performance.
5.2 MIPv6 implementation

- It is almost better for the special register. When sending the normal register, the sipclient has to wait for the IMS response before opening any session. The special register is sent at the same time as the normal register, and its processing time is much faster than for the normal register: so it is finished first. The only disadvantage concerns network resource consumption.

However, the measured time for opening a session does not correspond to a real IMS system. Because our IMS implementation does not support the PRACK and UPDATE requests, they are sent directly to the correspondent node. They do not go through all the IMS entity, and the system does not take time to allocate all the resources.

5.2 MIPv6 implementation

5.2.1 Presentation of the new implementation

An MIP implementation was available in the laboratory. However, this implementation is IPv4 compliant only, which prevent from the possibility of mechanisms like Route Optimization as well as from the benefit of an IPv6 network regarding the use of an up-to-date technology in general (see Section 2.2).

Even though the MIPv6 version we used is still under development (cf. appendix for more details), the bonus of working with IPv6 brings more advantage than inconvenient.

Hence, a completely new MIPv6 implementation has been used. This new implementation is meant to support Route Optimization, parts of IPsec mechanisms, and is fully IPv6 compliant.

5.2.2 Architecture & Configuration of MIPv6

Architecture

The architecture implemented in the laboratory is described in Figure 5.5. It follows the description done in Section 2.2.

Two sub-networks are created in order to proceed to a handover. The Home Agent is located outside of those two, which is more similar to an UMTS architecture, where the HA would be in the core network. In order to proceed to the handover, the Mobile Node will move from one Visited Network to another. In that scenario, the Home Network is only conceptual: the Mobile Node will never be in, and it will never have his Home Address as real address.

Routers represent GGSNs and are physically separated in order to be able to implement some of their filtering functionalities like TFT or SBLP.

Configuration

The new implementation is composed with 4 computers: 1 Home Agent (HA), 2 Routers and 1 Mobile Node (MN).
5.2 MIPv6 implementation

![MIPv6 architecture](image)

**Figure 5.5: MIPv6 architecture**

- **Home Agent**

  The HA is set up with a fixed IPv6 address. Packet forwarding is turned on because it uses normal routing to deliver packets captured from a physical interface to the virtual tunnel interface.

  The MIPv6 daemon has to be started. The main option activates an IPsec connection between the HA and the MN.

  In this configuration, the HA uses two interfaces: one corresponding to the Home network, the other to the external interface (toward "Internet"). Hence, the HA it set also as a router. However, as the MN is not supposed to be directly connected to the Home network, the former interface is not connected anywhere either.

  In the case of the Home Network is not only conceptual (the MN can be connected to it) the HA should be able to send specific router advertisements to enable the MN to know this specific location.

- **Router**

  The routers broadcast Router Advertisements regularly in order to fulfil their GGSN role. Indeed, one of their functions is to provide an IP address to the MN.

  Routing tables are set in order to be able to reach other networks.

  Two Access points are directly connected to the routers, in order to permit the use of the air interface.
5.2 MIPv6 implementation

Figure 5.6: MIP - Experimental flow - Route Optimization

Packet forwarding is turned on and all auto-configuration related parameters are turned off.

Routers do not need to have the MIPv6 daemon running.

- Mobile Node

Parameters for the mobile node are set as follow:
- Packet forwarding is turned off and all auto-configuration parameters are turned on.
- The MIPv6 daemon has to be started.
- Route Optimization between MN and CN is activated.
- The use of IPsec between the MN and HA is turned off.
- An acknowledgement for the binding update with the CN is requested.
- The MN home link is set: Home Address and Home Agent Address are declared.
- All the routing link is set: ICMP and UDP protocol are enabled.

5.2.3 Experimental flow

In the Figures 5.6 and 5.7, we summarize the MIP message flow described in 2.2.3 in a case of a handover.

According to our solution proposal, we want to try with a route optimization configuration: more delay after a handover (more messages) but the IP packets are sent directly to the mobile node.
5.2 MIPv6 implementation

But the MIP implementation used is an experimental one. Even if we have configured the system in order to use route optimization, the implementation is able to work only in Bi-Tunnelling.

So, in our experimental setup, MIP is working as described in the Figure 5.7, with a bi-tunnelling configuration.

5.2.4 Limitations of the implementation

As the implementation used for these experiments is still a work in progress - a new version is being released by the time those lines are written - some limitations occurred along with the tests.

Class of addresses to be used

Even if using a site local address type should work; it appears that it is not the case with the implementation available at the time of the project. With a site local address, everyone who use this address type has the capability to use the given 16 bits for a maximum number of 65536 subnets (comparable with the 10.0.0.0/8 in IPv4).

However, the use of this kind of address incurs the absence of Binding Updates between the MN and the HA.

In order to solve this problem, global address type replaces site local address type.

Missing features

Some features were not implemented still at the time we did the experiments. Those features are described below:

- Mobile Prefix Discovery.
- IPsec protection of HoTI/HoT messages and tunneled payload.
- IPsec between MN and CN.
5.3 MIPv6/SIP integration

Test features

The version we implemented MIPv6 with has a pre-version of IPsec, which works between the HA and the MN. In this one, it seems that only ESP protocol works. However, as described in 4.1, our solution proposal asks for AH protocol.

This limitation in the MIPv6 implementation induced that we had to disable IPsec between the HA and the MN, even though this is supposed to be mandatory in MIPv6 specifications.

As a result, the real processing time of MIPv6 is slightly diminished for the experimentation.

New release of MIPv6 implementation

To finish with the limitation of this implementation, and more as a remark, a new MIPv6 implementation has been released a week before the end of this project. This one called MIPL Mobile IPv6 version 2.0 RC2 should be "mostly feature complete".

Concerning the use of IPsec, people who works around the GO/Core project suggested to use this RC2, as it should work much better with it. Unfortunately, time constraint holds us from trying it in this project.

5.3 MIPv6/SIP integration

5.3.1 Problem

On one hand, there is an IMS system working, with 2 users (one for the UAC and the other for the UAS) and the IMS entities. On the other hand, there is a MIP network, with two access nodes, permitting to test wireless handover.

For integrating these systems, there is a major problem. One is using IPv4, and the other one used IPv6. It was not possible to use the IPv4 version of MIP, because it doesn’t use Route Optimization as we do in the solution proposal part.

To make these system inter working, a new entity is needed, in order to translate from IPv4 protocol to IPv6 and the other way.

But that’s not all. The sipclient implementation has been built for an IPv4 node. In order to make it working with IPv6, the code has to be changed. In Linux, the APIs for IPv6 and IPv4 application are almost similar: the same functions are called. There are mainly three major differences:

- When calling the socket() function, the type of network must be specified. In IPv6, the network is identified by the constant PF_INET6, opposite to the PF_INET constant for IPv4 protocol.
- To convert a string containing an IP address into a in_addr structure, the function inet_ntoa() is used in IPv4 (inet_aton() for the other way). In IPv6, the functions are inet_pton() and inet_ntop(), and they convert the string into a in6_addr structure.
5.3 MIPv6/SIP integration

- In IPv4, the IP address and the port number of a node are contained in the sock-
  addr_in structure. This one contains a in_addr structure, which corresponds to a
  4 bytes integer. In IPv6, the correspondent structure is sockaddr_in6, containing a
  in6_addr structure, which is a 16 char array.

In order to make the integration, we need first to change the sipclient implementation,
and second to build the translator.

5.3.2 Solutions

This sections does not involve the sipclient implementation, because it just corresponds
to some change in the actual implementation, and more details are given in Appendix.
This section is describing the two possible ways to implement a IPv4/IPv6 translator.

For both solutions, we need to install a new entity, between the IPv4 and IPv6 net-
works, as shown in Figure 5.8.

The first solution consists to work in the IP layer. Receiving a IP packet in its
interface, the translator acts as a router, but by making a correspondence between an
IPv6 address and an IPv4 address. There are already existing implementations on Internet,
found under the name NAT-PT. The IETF has also published a RFC 2766.

The main disadvantage of this solution is the used of DNS functionalities. It is done
to almost map an IPv6 address to an IPv4 one. As described in previous sections our
implementation is already using a DNS. Even if it should work, it is still dangerous to use
a same system for two distinguished functionalities.

Whatever, it was still possible to program our own NAT-PT, without using DNS
functionality. The number of entities in our experimental setup is reasonable. But we
need to be careful too: we are not using IPv6, but MIPv6. At some point, the mobile node
is sending binding updates to the correspondent nodes, in order to use route optimization.
As ICMP, a binding update works in the IP layer. In this case the translator is going
to translate all the binding updates. But the correspondent node is IPv4, and no MIP
daemon can be run on it. Any way, the information has to be stored in the translator,
because this entity builds the IPv6 packets.

The second solution is to make a translator more elaborated, working in upper layer.
This kind of translator will be able to work only with our experimental setup.

Typically, the location of an application is made with the port number and the IP
address. The purpose of this specific translator is to open the same number of port as the
number of application. Then, instead of locating an application with the port number
and the IP address, it can be done with only the port number. Then, the translator will
be able to build the IP packet with the right IP address and port number.

This solution is more elaborated, because it will hide the real P-CSCF to the mobile
node and the mobile node to the P-CSCF, by substituting the IP addresses. Moreover,
the translator needs to open the SIP packets in order to change the SIP port numbers.
5.3 MIPv6/SIP integration

Figure 5.8: SIP & MIP integration
5.3 MIPv6/SIP integration

5.3.3 Description of the chosen solution

We have chosen to implement the second solution: the MIPv6 daemon is not ours, and it will take us a long time to modify the already existing code to make binding updates working with the IPv4 network. Moreover, the classes for manipulating SIP packets are already written, so the treatments of the SIP packets can be easily done.

In order to work, the translator distinguishes 3 kinds of messages:

- Messages sent to the proxy. These messages are sent through the port number 5060 (for normal SIP requests), 5051 (for the filter), and 5052 for DNS update. When receiving a packet on one of these ports, the translator forwards it to the P-CSCF in the corresponding port. If it is the first message of the mobile node, the translator allocates a new port number, in order to identify all messages for this user. By changing the port number in the SIP messages, the P-CSCF will always answer to the translator at this port. The DNS update is done, by using the IP address of the translator.

- Messages sent to the mobile node, coming from the proxy. They are the most easy messages to forward. When the user registers for the first time, the translator identifies the user (IP address, and port number), and creates a new port number in order to communicate with the P-CSCF. All messages from IMS go through this port number. The translator has only to forward to the user all the packets coming in its dedicated port number.

- Data packets send directly to the correspondent node. The mobile node uses IMS to establish a session with a correspondent node. After the session establishment, the two final users exchange directly data packets. The correspondent node is in the IPv4 network. All data packets need to go through the translator. So, before that the correspondent node sends the first data packets, the translator has to open 2 port number (one for the correspondent node, and one for the mobile node), in order to let them exchange data. It can be done, only by looking at the SIP messages. We have added one field (named ”Infos”), which contains the IP address and the port number for data packets. The translator has to look at this field, and change both (Ip address and port number) in order to allocate 2 new port numbers for data transmission.

Figure 5.9 describes the behaviour of the translator for the different types of request. The figure shows the different port numbers used, and the processing done in the translator.
5.3 MIPv6/SIP integration

Figure 5.9: Translator processing
Experimental measurements

The first step of the experimental part of the project was to build a new sipclient. It is not the easiest one, and in order to test it, we have made some intermediate stages. Before to perform hard handover, we have tested our IMS implementation with soft handover, in an IPv4 network only. Then, we have integrated MIPv6, and did some measurements with different scenarii using access points.

6.1 Soft handover

6.1.1 Test configuration

Protocol

There are two kinds of packets exchange:

- SIP traffic : used by the UAC/UAS to communicate with IMS, in order to register or open a session with a correspondent node.
- data traffic : used by the UAS to send directly data packets to the UAC.

The UDP protocol is used in both case.

The SIP protocol uses requests and responses with normal transactions. Thus, there is almost acknowledgements for SIP request (except for the ACK). So, it is not necessary to have another acknowledgement mechanism in the under layers. Then, by using UDP instead of TCP, we can save resources, by avoiding overhead packets.

For data packets it was possible to use the TCP protocol. But the purpose of our prototype is to have a time performance evaluation, but also to evaluate the packet loss. Hence, because of TCP’s acknowledgement mechanism, we choose to use the UDP protocol instead of TCP. Moreover, in the hard handover scenarii, we simulate a streaming video data flow, and this type of application uses the UDP protocol.
6.1 Soft handover

**Bit rate**

The main purpose of the soft handover was to test our new client, and to have an idea about the SIP delays. Thus, we did not choose a fix bit rate: the UAS sends data packets with a random interval between 1 and 50 ms.

The size of the packets is also variable, but in average very small. We just send ASCII text in order to print it out on the client side.

**Handovers**

In the general case, a soft handover can be done when a user is able to connect to two access points at the same time. For example, a user is connected to an UMTS network, and then finds a WLAN network. Its wireless card connects to this new network, and when the user obtains his new IP address, the equipment can switch from the UMTS to the WLAN network.

We do not use different technologies to test a soft handover: we only use a wired network, and we change the IP address manually. The known command `ifconfig` is used in order to simulate the soft handover.

6.1.2 Results

The measurements have been made within the UAC. The UAS is run first (it needs to be registered first), and then the UAC is started. Just after its registration, the UAC tries to establish one session with the UAS, and when it is done, the UAS sends data packets.

We have measured the registration and the invitation time:

- The registration time corresponds to the beginning of the registration process until the REGISTER response. In the experimental flow, it corresponds to the steps 5 and 6 (see Section 5.1.6).

- The invitation time corresponds to the time of the INVITE request until the ACK request are sent. In the experimental flow, it corresponds from the step 7 to the step 14 (see Section 5.1.6).

The time performance has been measured on a Pentium III 933MHz. The Operating System was Fedora.

We have made 20 measurements, and put them in the table 6.1.

The TOTAL duration does not correspond exactly to the sum of the REGISTER and INVITE duration, because there is a certain amount of time, between the end of the registration and the beginning of the invitation, when the client is still processing.
## 6.1 Soft handover

<table>
<thead>
<tr>
<th>Test n*1</th>
<th>open session</th>
<th>REGISTER duration</th>
<th>INVITE duration</th>
<th>TOTAL duration</th>
<th>LOSS packets</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.10348</td>
<td>0.278794</td>
<td>0.384154</td>
<td></td>
<td></td>
</tr>
<tr>
<td>handover procedure</td>
<td>0.072473</td>
<td>0.156173</td>
<td>0.230273</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>Test n*2</td>
<td>open session</td>
<td>0.074205</td>
<td>0.30848</td>
<td>0.384576</td>
<td></td>
</tr>
<tr>
<td>handover procedure</td>
<td>0.072204</td>
<td>0.171767</td>
<td>0.246027</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>Test n*3</td>
<td>open session</td>
<td>0.072049</td>
<td>0.287906</td>
<td>0.361851</td>
<td></td>
</tr>
<tr>
<td>handover procedure</td>
<td>0.071409</td>
<td>0.144331</td>
<td>0.243334</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>Test n*4</td>
<td>open session</td>
<td>0.087606</td>
<td>0.148375</td>
<td>0.242827</td>
<td></td>
</tr>
<tr>
<td>handover procedure</td>
<td>0.071855</td>
<td>0.122554</td>
<td>0.196331</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>Test n*5</td>
<td>open session</td>
<td>0.071219</td>
<td>0.344698</td>
<td>0.417795</td>
<td></td>
</tr>
<tr>
<td>handover procedure</td>
<td>0.073664</td>
<td>0.176857</td>
<td>0.252419</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>Test n*6</td>
<td>open session</td>
<td>0.071891</td>
<td>0.286599</td>
<td>0.360344</td>
<td></td>
</tr>
<tr>
<td>handover procedure</td>
<td>0.073156</td>
<td>0.189275</td>
<td>0.26432</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>Test n*7</td>
<td>open session</td>
<td>0.089748</td>
<td>0.161417</td>
<td>0.364989</td>
<td></td>
</tr>
<tr>
<td>handover procedure</td>
<td>0.074552</td>
<td>0.159101</td>
<td>0.235688</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>Test n*8</td>
<td>open session</td>
<td>0.071097</td>
<td>0.250066</td>
<td>0.323151</td>
<td></td>
</tr>
<tr>
<td>handover procedure</td>
<td>0.073046</td>
<td>0.169421</td>
<td>0.244513</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>Test n*9</td>
<td>open session</td>
<td>0.07316</td>
<td>0.285702</td>
<td>0.361089</td>
<td></td>
</tr>
<tr>
<td>handover procedure</td>
<td>0.0743</td>
<td>0.185975</td>
<td>0.262034</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>Test n*10</td>
<td>open session</td>
<td>0.073011</td>
<td>0.272984</td>
<td>0.456846</td>
<td></td>
</tr>
<tr>
<td>handover procedure</td>
<td>0.072548</td>
<td>0.165028</td>
<td>0.239299</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>Test n*11</td>
<td>open session</td>
<td>0.080169</td>
<td>0.148867</td>
<td>0.348359</td>
<td></td>
</tr>
<tr>
<td>handover procedure</td>
<td>0.072012</td>
<td>0.188187</td>
<td>0.262189</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>Test n*12</td>
<td>open session</td>
<td>0.073014</td>
<td>0.148054</td>
<td>0.32642</td>
<td></td>
</tr>
<tr>
<td>handover procedure</td>
<td>0.072548</td>
<td>0.18365</td>
<td>0.257786</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>Test n*13</td>
<td>open session</td>
<td>0.07252</td>
<td>0.404306</td>
<td>0.478842</td>
<td></td>
</tr>
<tr>
<td>handover procedure</td>
<td>0.073235</td>
<td>0.308147</td>
<td>0.4076</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>Test n*14</td>
<td>open session</td>
<td>0.073083</td>
<td>0.151141</td>
<td>0.332936</td>
<td></td>
</tr>
<tr>
<td>handover procedure</td>
<td>0.071899</td>
<td>0.158267</td>
<td>0.232021</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td>Test n*15</td>
<td>open session</td>
<td>0.071331</td>
<td>0.270489</td>
<td>0.343754</td>
<td></td>
</tr>
<tr>
<td>handover procedure</td>
<td>0.072278</td>
<td>0.165493</td>
<td>0.239636</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>Test n*16</td>
<td>open session</td>
<td>0.07946</td>
<td>0.150574</td>
<td>0.440411</td>
<td></td>
</tr>
<tr>
<td>handover procedure</td>
<td>0.072473</td>
<td>0.158776</td>
<td>0.233238</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>Test n*17</td>
<td>open session</td>
<td>0.071916</td>
<td>0.159138</td>
<td>0.347147</td>
<td></td>
</tr>
<tr>
<td>handover procedure</td>
<td>0.07332</td>
<td>0.150082</td>
<td>0.257175</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>Test n*18</td>
<td>open session</td>
<td>0.073223</td>
<td>0.14833</td>
<td>0.339743</td>
<td></td>
</tr>
<tr>
<td>handover procedure</td>
<td>0.073735</td>
<td>0.268776</td>
<td>0.379915</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>Test n*19</td>
<td>open session</td>
<td>0.071721</td>
<td>0.158212</td>
<td>0.44876</td>
<td></td>
</tr>
<tr>
<td>handover procedure</td>
<td>0.075026</td>
<td>0.16394</td>
<td>0.240899</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>Test n*20</td>
<td>open session</td>
<td>0.074144</td>
<td>0.278489</td>
<td>0.35296</td>
<td></td>
</tr>
<tr>
<td>handover procedure</td>
<td>0.073047</td>
<td>0.181815</td>
<td>0.256767</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td>open session</td>
<td>0.07640235</td>
<td>0.23213105</td>
<td>0.3708613</td>
<td></td>
</tr>
<tr>
<td>handover procedure</td>
<td>0.072939</td>
<td>0.17838075</td>
<td>0.25907315</td>
<td>16.4</td>
<td></td>
</tr>
</tbody>
</table>

### Table 6.1: Test measurement - sip establishment
6.2 Hard handover

6.2.1 Test configuration

Protocol

In order to exchange SIP traffic or data packets, the UAC and the UAS use the UDP protocol for the same reason as in the soft handover section.

Bit rate

In our prototype, we have simulated a video streaming data flow. The UAS sends data packets of 700 bytes, every 40 ms.

Indeed, 700 bytes corresponds to 5.6 kbits. Sending packets every 40 ms corresponds to sending 25 packets per second. Then the total bit rate is about 140 kbits per second, which corresponds to a small video. Indeed, since we consider IMS, the end user equipment will most likely be a mobile phone, and this data rate is enough to play a video on a mobile screen.

Handovers

As explained earlier, soft-handovers can be done only when the node is able to connect to two networks at the same time. However, in our experiments, we use a mobile node with one wireless interface. Furthermore, the handover is done by changing the SSID between the two networks at our disposal.

Hence, a hard-handover was the natural choice for our experiments.

The measurements of the experimentation showed that the time needed to obtain an IP address does not depend on SIP nor MIP mechanisms only. It mainly depends on the frequency of sending router advertisements, presently a fixed interval of 10s in our configuration.

So, we can show that there is a linear relation between the amount of lost packets and the time needed to obtain an IP address (cf. figure 6.1 – the same kind of figure is obtained with every other scenario).

Hence, in order to measure specifically the influence of MIP and SIP mechanisms on handover delays, we do not have to take into account the time needed to obtain the IP address. So we have to estimate the amount of lost packets during this time, in order not to take them into account in our results. By doing this, we focus on SIP/MIP mechanisms.
6.2 Hard handover

6.2.2 Different scenarii

We did not have the time to implement all the functionalities needed in order to test our solution. The main part missing is the filtering functions in the GGSN, represented by the routers. However, in order to evaluate the performance of our prototype, we have built 4 different scenarii, and we have tried to emulate the missing functionalities.

**MIP only**

In the first scenario, we measured the global performance of the MIP implementation. On one side of the translator, there is an MIPv6 implementation, and on the other side, there is an IMS (Ipv4), as described in section 5.2. We just use the IMS to establish a session with the correspondent node, then the UAS sends data packets to the UAC.

By using simple scripts, the mobile node changes its network from one access point to the other every 20 seconds. The mobile node obtains a new IP address, thanks to router solicitations, and sends a binding updates to the home agent. Then the home agent will forward all data packets from the correspondent node (represented by the translator in our configuration) to the new care of address of the mobile node.
6.2 Hard handover

In this case, only MIP is working because there is no filtering in the routers/GGSN to drop data packets. In the mobile node, we just disable the re-invite and register procedures after a handover, because the mobility is managed only by MIP: IMS does not need to be aware of this change of address in this scenario.

**SIP only**

The second scenario concerns the performance of an IMS handover.

To do this handover, we have reactivated in the client the ability to re-invite and to register after a handover. We have also switched the kernel of the mobile node from the kernel 2.6.8.1 to the kernel 2.6.9.1 in order to disable the MIP daemon. We have also completely disconnected the home agent.

Nevertheless this configuration shows two major problems:

- After getting a new IP address, the mobile node sends one DNS update to the P-CSCF. The request goes through the translator and the P-CSCF answers. But when the translator sends back the response to the mobile node, this one never gets the packet. There is a certain amount of time after getting its new IP address, when the mobile node is not able to receive any packets. If the mobile node does not receive the DNS response, it will not continue the registration.

- The translator creates a new SIP port, because it considers the new IP address as a new correspondent node. By changing the SIP request, the translator makes the mobile node register with a new port number. The present IMS implementation does not allow it, and will send back a conflict error.

To solve the first problem, we have changed the code of the mobile node in order to send a DNS update every 40 ms until a response arrives. Then, the translator forwards only the first DNS update to the P-CSCF, and answers to the other ones only if the P-CSCF has already answered. No more than one DNS updates should be sent to the P-CSCF. When the mobile node will send the REGISTER request, the DNS will still be restarted (by another DNS update), and IMS will not be able to find the corresponding IP address (DNS will not work).

In order to deal with the second problem, we have considered only one mobile node. Then, the translator is able to be aware that the 2 IP addresses belong to the same mobile node and uses the same SIP port for both of them. This hypothesis induces a lot of changes in the translator code.

**MIP+SIP without negotiation**

In the third scenario, we integrated MIP and IMS. We emulated a QoS transfer from one GGSN to the other one (AR in the experiment), and we assumed that the new GGSN accepts the QoS parameters.

After a hard handover, the client needs to be registered in IMS, in order to warn its former GGSN to transfer the QoS. So, we activate the registration procedure after a handover in the UAC implementation, but not the re-invite procedure.
6.2 Hard handover

DNS update are still sent with a loop, because the problem of the previous scenario is the same in this case.

We also emulate some filtering functionalities by dropping some data packets, which, otherwise would be sent to the mobile node via MIP. Those functionalities are implemented in the translator. When this one receives the first DNS update, all following data packets are dropped until the end of the registration. In a real scenario, those packets would be dropped by the GGSN, because the QoS transfer would not not be over.

In order to emulate the QoS transfer, the translator also wait a random time after receiving the REGISTER OK. We chose an interval between 50 and 90 ms, which corresponds roughly to the average of a REGISTER request (72 ms). After that, the translator allows the data packets to go through again.

**MIP+SIP with negotiation**

In the last scenario, we have done almost the same as the previous one, but in this case the new GGSN needs to renegotiate the QoS parameters. We assume that the UAS accepts the new parameters.

The configuration is almost the same: the data packets are still dropped by the translator when this one receives the DNS update, but there are now allowed when the translator receives the PRACK response.

We also changed the re-invite procedure, and the client just sends an UPDATE and a PRACK requests in order to renegotiate the SDP parameters with the correspondent node.

### 6.2.3 Results

In this section, we will present the results of the measurements: first the handover delay and then the packet loss.

To underline the performances obtained with the different scenarii, measurements of handovers have been made. Those measurements do not take into account the time to get a new IP address, but show the duration of signalling messages and data re-establishment.

For every scenario, hundreds of handovers are generated automatically through a shell script. The quantitative measurements are done at the application level (through a C-program), and the qualitative part is done through tcpdump/ethereal.

- Handover delay

Figure 6.2 shows one of the handover delay distribution obtained, with a superimposed fitted normal density.

Figure 6.3 shows the confidence intervals for the true mean of the handover delays in every scenario.
6.2 Hard handover

- Packet loss

Figure 6.4 shows one of the packet loss distribution obtained, with a superimposed fitted normal density.

Figure 6.5 shows the confidence intervals for the true mean of the amount of lost packet in every scenario?

6.2.4 Analysis

We will discuss here the results presented in the previous section, drawing our first conclusions on the experiments’ outcome.

Handover delay comparison between MIP and SIP scenarii

The confidence intervals of the mean show a noticeable difference among the experimental scenarii when they do not overlap.

We can state from here that MIP has the fastest response to a hard handover. This was to be expected according to [9].
6.2 Hard handover

![Confidence Intervals Comparison](image)

**Figure 6.3**: Confidence Intervals - Handover delay
6.2 Hard handover

Figure 6.4: Packet loss - MIP+SIP w/o renegotiation

From there, we see that SIP is noticeably slower than MIP to complete the handover procedure. When we look at the experimental traffic flows described in Section 5.1.6 and Section 5.2.3, we can note that the difference between MIP and SIP handover delay, which goes roughly between 70ms and 200ms is rather small.

Indeed, from the Section 6.1, we can see that a SIP soft handover takes around 300ms in a wired environment. Thus, it would mean that MIP would take at least 100ms in the wireless environment. This value for a MIP handover delay is rather large when compared to literature values (cf. [10] [11]). In fact, it is reasonable to expect a bigger difference between MIP and SIP handover.

A look to ethereal shows that MIP handover delay is in fact around 400ms, which is a very large value compared to what we could expect. Two factors may explain it:

- the MIP implementation is still in development, and may not have yet an optimal behaviour. Processing time in the HA may be the cause for such long delays with MIP (400ms between sending the binding update and receiving the binding acknowledgment).

- in the SIP only scenario, the translator drops the data packets at the beginning of the handover procedure, which alleviates the air medium. However, this is not
6.2 Hard handover

![Confidence Intervals Comparison](image)

**Figure 6.5:** Confidence Intervals - Packet loss
true for the MIP scenario: the air interface bears more load which can slow the transmission.

**Lost packets comparison between MIP and SIP scenarii**

Concerning the packet loss, our experiment does not allow us to conclude about any difference between MIP and SIP. As far as our experiment went, the confidence intervals of the mean of the amount of packet lost are quite similar.

In a normal case, it is true that we would expect to have more drops from the SIP experiment than from the MIP one. But in our case, the MIP registration takes at least 400 ms. This value is larger than a simple SIP registration (110 ms, see appendix F).

In our SIP implementation the UAS will stop sending data packets when it will receive the re-INVITE request. In term of delay, the SIP scenario is longer than the MIP one because the re-INVITE transaction has to be performed. But in term of lost packets the SIP scenario should save data packets, because the UAS stops sending them after receiving the re-INVITE request, meaning a little more than the REGISTER duration, but not as much as 400 ms.

Our hypothesis is about the difference of layers. MIP is a protocol of the network layer, and SIP is in an upper layer. So, the MIP registration may start before the SIP registration: all the intermediate layers have to be reactivated after the handover. The first lost data packets occured when we change the access point, then, the network layer will be re-established. In the MIP case, it will take a long time (400 ms). All intermediate layers will be reactivated and the the SIP procedure begun. Roughly it is possible to have the same amount of lost data packets.

**MIP and MIP+SIP scenarii**

We will now compare those two primary analysis with the MIP/IMS hybrid, for the two different scenarii we described in Section 6.2.2.

As we pointed out that the MIP handover delay was biased, any comparison between the SIP only situation and the MIP/IMS hybrid would be biased also. Thus, the only relevant comparison to make is between MIP only case and the hybrid ones.

A significant difference appears with the handover delay between the MIP scenario and the MIP/IMS hybrid. Two parameters take part in this behaviour.

- The fact that we use bi-tunnelling instead of route optimization increases the transmission distance between the MN and any correspondant (we have to go through the HA). In the MIP scenario only the data packets were using the bi-tunnelling. In the hybrid scenarios all the SIP signalling as weel as the data traffic are bi-tunneled, increasing the time delay.

- SIP and MIP do not work in the same layers, this induce a delay to reactivate all the intermediate layers.
6.2 Hard handover

The difference between the two hybrid scenarii is more expected. Indeed, the only change is the emulation of a QoS negotiation. This negotiation corresponds to the addition of two messages compared to the previous case: one UPDATE and one PRACK.

We can notice here that packets lost and handover delay are roughly proportional.
Chapter 7

Conclusion & Future Work

7.1 Conclusion

The IMS, a recently designed framework, is a very attractive infrastructure for mobile operators. Its main advantage is the possibility to offer all kinds of multimedia services to potential subscribers (e.g. Push-To-Talk, video or audio streaming, instant messaging). The IMS is based on the SIP protocol, which should become a standard Internet protocol in the near future.

During this project, we had the opportunity to work with a system dependent on many different technologies and concepts such as the Go interface, GGSN filtering functionalities, Quality of Service, etc. All these made us familiar with such diverse literature as the IETF Request For Comments or the 3GPP Technical Specifications. Eventually, we were able to work with a system which we did not know.

In order to improve the handover performance of this system, we wanted to use MIP mechanisms in such a handover scenario. However, because of the IMS requirements with respect to QoS control procedures and particularly access control, we encountered unexpected problems, just to connect a MIP-compliant terminal to the IMS infrastructure.

Hence we have designed a solution so as to associate MIP mechanisms with GGSN filtering functions, which are administered by the IMS. We have dealt with the IP addresses provided by MIP in order to match them with the ones expected in the filters.

To evaluate the performance of this integration, we have implemented a prototype. Firstly, we developed a SIP client for the existing IMS implementation; we used network programming and the thread technology. In a second part we configured the MIPv6 implementation from Helsinki University. The integration of these two systems now allows us to work with an IMS/MIP prototype.

The obtained results are little relevant, mainly because of the MIP implementation. In fact, the time to perform a MIP handover is too long compared to usual values found in literature. Moreover, two functionalities defined in the standard are not implemented: IPSec tunneling and route optimization.
7.2 Future Work

However, from the analysis of the experimental results, we can conclude that the main interest of integrating MIP in an IMS infrastructure is to reduce the handover delay. On the other hand, the IMS can lose less data packets than expected from the handover delay, if the application server stops sending data packets after receiving a re-INVITE.

7.2 Future Work

There may actually be a quite much more done on this subject, particularly because we lacked time to perform truly conclusive experiments in relation to the conceptual work. At first, the filtering functions of the access routers can be implemented; the Go interface can then be emulated with a traffic flow only similar to the standard COPS mechanisms. Moreover, the QoS transfer can undergo the same procedure and be emulated with a traffic flow seemingly identical to what is expected from the conceptual solution proposals.

In addition, the IPv6 transition of the nodes used in our experiments can be carried through, with a true IPv6 IMS implementation. Furthermore, the current MIPv6 setup is still in a development phase and a new version was published just days before the end of this work: the development team have said that the latest implementation is the last one before the one with full MIPv6 features.

Then, new measurements would be much more realistic. The obtained results would leave little doubt about whether our conceptual solutions are viable.

In a second approach, there is also plenty to work from on conceptual aspects. The model could be more detailed, for instance, and include more IMS entities in QoS transfer in a handover situation.

Lastly, the scope of the analysis should be broadened so as to consider handovers across networks from different operators.
Appendix A

Abbreviations

• 3GPP: Third Generation Partnership Project
• APN: Access Point Name
• AR: Acess Router
• AS: Application Server
• BS: Bearer Service
• BGCF: Breakout Gateway Control Function
• CN: Correspondent Node
• COPS: Common Open Policy Service protocol
• CS: Circuit Switched
• CSCF: Call State Control Function
• GGSN: Gateway General Support Node
• HSS: Home Subscriber Server
• HTTP: HyperText Transfer Protocol
• IETF: Internet Engineering Task Force
• IMS: IP Multimedia Subsystem
• IP-CAN: IP Connectivity Access Network
• ISIM: IMS SIM
• LS: Location Server
• LWS: Linear WhiteSpace
- MGCP: Media Gateway Controller Protocol
- MN: Mobile Node
- NAI: Network Access Identifier
- OSI: Open System Interconnection
- PCF: Policy Control Function
- PDP: Packet Data Protocol
- PS: Proxy Server
- PSI: Public Service Identity
- PSTN: Public Switched Telephone Network
- QoS: Quality Of Service
- RFC: Request For Comments
- RG: Registration Server
- RS: Redirect Server
- SBLP: Service-Based Local Policy
- SDP: Session Description Protocol
- SIP: Session Initiation Protocol
- SPI: Security Parameter Index
- TFT: Traffic Flow Template
- UAC: User Agent Client
- UAS: User Agent Server
- UE: User Equipment
- URI: Uniform Resources Identifier
## B.1 SIP responses

The following table lists the main response codes in the SIP protocol:

<table>
<thead>
<tr>
<th>response code</th>
<th>status</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>1xx</td>
<td>Trying: indicates that the response has been received and is going to be processed</td>
</tr>
<tr>
<td>180</td>
<td>1xx</td>
<td>Ringing: The INVITE has been received and the equipment is alerting the user</td>
</tr>
<tr>
<td>181</td>
<td>1xx</td>
<td>Call is being forwarded: indicates that the call is forwarded to different set of destinations</td>
</tr>
<tr>
<td>182</td>
<td>1xx</td>
<td>Queued: The callee is not available, but the server has decided to queue this call</td>
</tr>
<tr>
<td>183</td>
<td>1xx</td>
<td>Session Progress: report information about the progress of the call</td>
</tr>
<tr>
<td>200</td>
<td>2xx</td>
<td>OK: indicates that the request has succeeded</td>
</tr>
<tr>
<td>300</td>
<td>3xx</td>
<td>Multiple Choices: indicates that the address resolution provides several locations, and ask the UAC to choose the correct one</td>
</tr>
<tr>
<td>301</td>
<td>3xx</td>
<td>Moved Permanently: the user cannot be find again at the address and the UAC has to use the one in the Contact tag</td>
</tr>
<tr>
<td>302</td>
<td>3xx</td>
<td>Moved Temporarily: As the error 301, and the validity of the new address is given in the Expires tag</td>
</tr>
<tr>
<td>305</td>
<td>3xx</td>
<td>Use Proxy: The resource has to be contacted through the proxy specified in the Contact tag</td>
</tr>
<tr>
<td>380</td>
<td>3xx</td>
<td>Alternative Service: The call was unsuccessful, but alternative services are available</td>
</tr>
<tr>
<td>400</td>
<td>4xx</td>
<td>Bad Request: the request has not been understood</td>
</tr>
<tr>
<td>401</td>
<td>4xx</td>
<td>Unauthorized: the request requires user authentication</td>
</tr>
</tbody>
</table>

continued on the next page
## B.1 SIP responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Reason Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>402</td>
<td>Payment Required for future use</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden the user is not allowed to use this request</td>
</tr>
<tr>
<td>404</td>
<td>Not Found the final destination doesn’t exist at the specified domain</td>
</tr>
<tr>
<td>405</td>
<td>Method Not Allowed the method is not allowed by the Request-URI</td>
</tr>
<tr>
<td>406</td>
<td>Not Acceptable response entities have no acceptable content characteristics according to the header fields of the request</td>
</tr>
<tr>
<td>407</td>
<td>Proxy Authentication Required As the error 401, but the client has to authenticate itself with the proxy</td>
</tr>
<tr>
<td>408</td>
<td>Request Timeout the request must repeat the exact request later (for example the location is not yet determined by the server)</td>
</tr>
<tr>
<td>410</td>
<td>Gone the requested resource is no longer available</td>
</tr>
<tr>
<td>413</td>
<td>Request Entity Too Large the request entity body is larger than the server is able to process</td>
</tr>
<tr>
<td>414</td>
<td>Request-URI Too Long the Request-URI is too long to be interpreted by the server</td>
</tr>
<tr>
<td>415</td>
<td>Unsupported Media Type The message body is in a format unsupported by the server</td>
</tr>
<tr>
<td>416</td>
<td>Unsupported URI Scheme the Requested-URI scheme is unknown by the server</td>
</tr>
<tr>
<td>420</td>
<td>Bad Extension the server doesn’t understand the extension of the protocol</td>
</tr>
<tr>
<td>421</td>
<td>Extension Required The UAS needs a particular extension to process the request</td>
</tr>
<tr>
<td>423</td>
<td>Interval Too Brief the extension time of the resource refreshes by the request is too short</td>
</tr>
<tr>
<td>480</td>
<td>Temporarily Unavailable the callee equipment has been reached, but the user is not available. This error indicates the time to retry the request.</td>
</tr>
<tr>
<td>481</td>
<td>Call/Transaction Does Not Exist the request didn’t match any existing dialog or transaction</td>
</tr>
<tr>
<td>482</td>
<td>Loop Detected a server has detected a loop</td>
</tr>
<tr>
<td>483</td>
<td>Too Many Hops the number of authorized hops has been expired</td>
</tr>
<tr>
<td>484</td>
<td>Address Incomplete the Request-URI is incomplete</td>
</tr>
<tr>
<td>485</td>
<td>Ambiguous the Request-URI is ambiguous. A listing of possible ambiguous addresses may be sent</td>
</tr>
<tr>
<td>486</td>
<td>Busy Here the user is not able to take additional calls. It may indicates a better time to call again</td>
</tr>
<tr>
<td>487</td>
<td>Request Terminated indicate that the request has been finished by a BYE or a CANCEL</td>
</tr>
<tr>
<td>488</td>
<td>Not Acceptable Here As the error 606, but applied only fo the specific resource addressed</td>
</tr>
</tbody>
</table>

*continued on the next page*
B.2 SIP headers

<table>
<thead>
<tr>
<th>Code</th>
<th>Status Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>491</td>
<td>Request Pending</td>
<td>a request with the same dialog is already in the queue</td>
</tr>
<tr>
<td>493</td>
<td>Undecipherable</td>
<td>the UAS uses an encrypted MIME body, that the UAC doesn’t possess. The public key has to be sent to it.</td>
</tr>
</tbody>
</table>

**responses 5xx (Server Error)**

<table>
<thead>
<tr>
<th>Code</th>
<th>Status Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>Server Internal Error</td>
<td>an exception occurs in the server, and it is not able to fulfill the request</td>
</tr>
<tr>
<td>501</td>
<td>Not Implemented</td>
<td>the server doesn’t support the functionality required</td>
</tr>
<tr>
<td>502</td>
<td>Bad Gateway</td>
<td>the server acting as a gateway didn’t receive a correct response from the UAS</td>
</tr>
<tr>
<td>503</td>
<td>Service Unavailable</td>
<td>the server is actually overloading or in maintenance</td>
</tr>
<tr>
<td>504</td>
<td>Server Time-Out</td>
<td>the server didn’t received a timely response from an external server it tried to access</td>
</tr>
<tr>
<td>505</td>
<td>Version Not Supported</td>
<td>the server doesn’t support the SIP version</td>
</tr>
<tr>
<td>513</td>
<td>Message Too Large</td>
<td>the message length exceeds the capabilities of the server</td>
</tr>
</tbody>
</table>

**responses 6xx (Global Failure)**

<table>
<thead>
<tr>
<th>Code</th>
<th>Status Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>600</td>
<td>Busy Everywhere</td>
<td>the callee doesn’t wish to take the call at this time. The response indicates a better time to call back</td>
</tr>
<tr>
<td>603</td>
<td>Decline</td>
<td>the user declines or is not able to participate to the session</td>
</tr>
<tr>
<td>604</td>
<td>Does Not Exist Anywhere</td>
<td>the server has authoritative information about the non existance of the Request-URI</td>
</tr>
<tr>
<td>606</td>
<td>Not Acceptable</td>
<td>some aspects of the session are not supported by the end user, for example media or bandwith</td>
</tr>
</tbody>
</table>

*continued on the next page*

B.2 SIP headers

The main headers of the standardized SIP protocol are listed below:

- **Accept** : defines the accepted format for the transaction (body of the request). If it’s no specified, the server should assume a default value of application/sdp.

- **Accept-Encoding** : specifies the content coding accepted. For example : gzip. The default value of encoding is none.

- **Accept-Language** : defines the human language which can be used in the reason phrases, the session description, or the status responses.

- **Alert-Info** : suggests an alternative ring to alert the end user

- **Allow** : lists all the method supported by the User Agent (UA), like INVITE, CANCEL, ACK, BYE, OPTIONS...

- **Authentiﬁcation-Info** : provides mutual authentiﬁcation
B.2 SIP headers

- **Authorization**: contains authentication credentials of a UA.
- **Call-ID**: is used to identify a unique invitation, or all registrations of a UAC. Using specific algorithm, the key has to be unique.
- **Call-Info**: provides additional informations about the caller or the callee.
- **Contact**: contains a URI. This URI is interpreted in different way according to the type of request.
- **Content-Disposition**: tells how the message body has to be interpreted.
- **Content-Encoding**: describes what additional content encoding have been added.
- **Content-Language**: specify the human language used.
- **Content-Length**: indicates the size of the message body.
- **Content-Type**: defines the media type used in the message body.
- **Cseq**: contains a single decimal number of 32 bits, used to order the transactions of a session.
- **Date**: contains the date and time at which the request has been sent.
- **Error-Info**: provides a pointer to additional information about the error: for example, to get a formatted message to show to the user.
- **Expires**: gives the relative time after which the message will expire.
- **From**: indicates the initiator of the request.
- **Max-Forwards**: limits the number of proxies or gateways that can forward the request. The default value is 70.
- **Min-Expires**: gives the minimum refresh interval of the server.
- **MIME-Version**: describes the MIME (Multipurpose Internet Mail Extension) version used.
- **Organization**: contains the organization name which the SIP element issuing the request.
- **Priority**: indicates the urgency of the request.
- **Proxy-Authenticate**: contains an authentication challenge.
- **Proxy-Authorization**: allows a client to identify itself with a proxy.
- **Proxy-Require**: indicates some features that may be supported by the proxy.
- **Record-Route**: permits to proxy to force the future requests of the session to flow through itself.
B.2 SIP headers

- **Reply-To**: contains a logical return URI
- **Require**: lists the options that the UAC is expected to use. So the UAS has to support them.
- **Retry-After**: specifies the delay after which the UAC can retry the request.
- **Route**: is used to force a request to be routed through the list of proxies.
- **Server**: contains information about the software used by the server.
- **Subject**: provides a summary to indicate the nature of the call.
- **Supported**: lists all the extensions supported by the UAC or the UAS.
- **Timestamp**: describes when the UAC sent the request.
- **To**: specifies the logical recipient of the message.
- **Unsupported**: lists all the unsupported features of the UAS.
- **User-Agent**: contains information about the UAC originating the request.
- **Via**: indicates the path used by the request, and then the path use for the response.
- **Warning**: contains additional information about the status of a response.
- **WWW-Authenticate**: contains an authentication challenge.

In IMS, the implementation of the SIP protocol is not exactly the same. Some headers have been added to ensure user authentication, or to specify 3GPP architectures. Next, there is a list of some headers found in IMS:

- **Events** : - not ready.
- **P-Access-Network-Info** : - not ready.
- **P-Asserted-Identity** : - not ready.
- **P-Associated-URI** : - not ready.
- **P-Charging-Vector** : - not ready.
- **Privacy** : - not ready.
- **Security-Client** : - not ready.
- **Security-Server** : - not ready.
- **Security-Verify** : - not ready.
- **Service-Route** : - not ready.
Experimental setup migration from IPv4 to IPv6

C.1 MIPv4 existing implementation

The implementation available in our lab is described in figure C.1:

<table>
<thead>
<tr>
<th>Equipment</th>
<th>Version</th>
<th>Software</th>
<th>Names</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco Routers</td>
<td>3620</td>
<td>c3620-i-mz.122-5d.bin</td>
<td>Aalborg, Frankfurt and Delft</td>
</tr>
<tr>
<td></td>
<td>2631</td>
<td>c3631-telco-mz.122-8.T5.bin</td>
<td>Tokyo and San Francisco</td>
</tr>
<tr>
<td>Cisco Switches</td>
<td>2950</td>
<td>c2950-i6q4l2-mz.121-11.EA1.bin</td>
<td>Toronto and Shanghai</td>
</tr>
<tr>
<td>Cisco APs</td>
<td>350</td>
<td></td>
<td>Istanbul and Dhaka</td>
</tr>
<tr>
<td></td>
<td>1200</td>
<td></td>
<td>Sydney</td>
</tr>
<tr>
<td>Bluetooth AP</td>
<td></td>
<td></td>
<td>Ericsson BlipNode L1</td>
</tr>
</tbody>
</table>

Mobile IPv4 is implemented on:

- 3 Pentium pro 166 MHz, 64 MB RAM, 2GB hard disk, Linux Red Hat 7.3 platform.
- 1 Pentium 4, Linux Red Hat 9.0 platform (10.10.1.254) WARNING: Check, it should be 10.1.1.136.

The most frequently used computers are the 3 Pentium pro computers, where: 10.10.1.254 acts as the home-agent 10.10.2.254 and 10.10.3.254 acts as the foreign-agents.

C.2 SIP existing implementation

C.3 MIPv6 new implementation

C.3.1 Software

There are currently two Mobile IPv6 Linux implementations available. The Lancaster University in the United Kingdom has an implementation. The latest kernel supported
C.3 MIPv6 new implementation

Figure C.1: Lab Architecture & Equipment
C.3 MIPv6 new implementation

is 2.1.90. The code and website has not been updated since 1998, so it is considered obsolete.

The other implementation, which is up-to-date, is Helsinki University of Technology’s MIPL project. The latest supported kernel is 2.4.22, and they have patches for the upcoming 2.6 kernel.

MIPL (Mobile IPv6 for Linux) is an implementation of Mobility support following IPv6 Internet Draft. It was originally developed in the Helsinki University of Technology (HUT). Then, further development was taken up by the GO/Core project at HUT Laboratory for Theoretical Computer Science.

MIPL Mobile IPv6 for Linux has been released under GPL and is available to anyone for free.

Version 1.1 was the last release for the 2.4 kernel series. All development effort is now focused on finishing MIPL 2.0 for 2.6 kernel series and user space implementation.

Version 2.0 RC1 is the 13th public release and is the first for the 2.6 kernel series. It supports IPsec protection of the home registration signalling, but is still missing proper tunnel mode protection of the HoTI/HoT (Home Test Init) signalling and tunneled payload data, as well as MPS/MPA support.

C.3.2 TestBed

The TestBed needed to evaluate MIPv6 is described in figure C.2:

![TestBed Architecture](image)

Figure C.2: TestBed Architecture

Tools used in this implementation are:

- tcpdump: dump traffic on a network
- mipdiag: To verify binding update procedures
- ping6
C.3 MIPv6 new implementation

- route: view/modify IP routing table
- ...

Project report - Axel Adenet - Alexandre Bousier - Thior Santander Marín
SIP Implementation

D.1 introduction

To stimulate the IMS system, we have written different programs:

- **sipclient**: this program permits to initiate SIP transaction, in order to receive data packets from the correspondent node.
- **sipclient6**: the same as before, but working with an IPv6 network.
- **sipserver**: this program permits to register the user in IMS, and then wait for clients to open session. Then, after a SIP session establishment, it sends data packet to the correspondent node. The main difference with the sipclient concerns the application behavior.
- **sipserver6**: same as before, but working with IPv6 network.
- **filter**: this program is used to forward special request as PRACk or UPDATE, that IMS is not able to forward.
- **DNS**: the purpose of this program is to update the DNS configuration, when a user change its IP address.
- **translator**: it permits to have SIP transaction with a IPv6 node, and the IMS system in IPv4

Most of these programs use common files. In order to have an idea of the complexity of the programming architecture, we have listed all of this files in the table D.1.
### D.1 introduction

<table>
<thead>
<tr>
<th></th>
<th>sipclient</th>
<th>sipclient6</th>
<th>sipserver</th>
<th>sipserver6</th>
<th>filter</th>
<th>DNS</th>
<th>translator</th>
</tr>
</thead>
<tbody>
<tr>
<td>common.h</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>(69 lines)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>common.cpp</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>(209 lines)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IPAddress</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(3 lines)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIP_message.h</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>(78 lines)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIP_message.cpp</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>(1007 lines)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIP_env.h</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(104 lines)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIP.h</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(78 lines)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIP.cpp (IPv4)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(2950 lines)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SIP.cpp (IPv6)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(2959 lines)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>main.cpp (client)</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(102 lines)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>main.cpp (server)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(127 lines)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>main.cpp (filter)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(174 lines)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>restartDNS</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(4 lines)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>main.cpp (DNS)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(169 lines)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>main.cpp (trans-</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lator)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(1003 lines)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table D.1:** Programming files - programs
D.2 SIP client

The main purpose of the new SIP client is to provide a simple user interface, for SIP transactions. The flow uses in this implementation is based on IMS flow. This implementation is only done for the end user, and all additional mechanisms initiates by IMS entities (like P-CSCF, S-CSCF, I-CSCF) are not implemented.

The code is written in C++, and used the thread mechanisms. The main architecture of the program can be simply described with the next diagram:

There are four main blocks in the program:

- **Application interface**: provides some simple C functions for the programmer. These functions allow him to use SIP transactions.
- **Send**: permits to send SIP signalling as well as data packet.
- **Receive**: listens different defined port and redirect the packet when one arrived.
- **Request treatment**: is the main part of the program, because it defines the client (or server) behaviour.

D.3 Function interface

We have defined 6 main functions to manage the SIP sessions. And because playing with SIP implies an interactive exchange, the programmer of the application has to define also 6 other functions. The name of these call-back functions are given during the initialisation.

**2 functions to manage SIP**

The first function is the one which permits to initiate the SIP context: network information (IP-address, P-CSCF, machine name), send the REGISTER, start the RECEIVE block.

```c
bool StartSip(int port, char* sip_uri, char* card,
              char* machine, bool (* __asking) (int, SIP_PACKET*,
              CALLBACK_FUNCTIONS*, char*), void* (* __ready) (void*));
```

This function takes 6 arguments:

- **port**: indicates the using port for the SIP signalling
- **sip_uri**: corresponds to the URL of the user connected
- **card**: corresponds to the ethernet card used (to check IP addresses change)
- **machine**: corresponds to the name of the machine used for the transactions
D.3 Function interface

- **__asking**: is a call-back function, used when the program received an
- **__ready**: this call-back function is called when the registration is over INVITE request

The function returns false if SIP can’t be started. Otherwise, the function doesn’t return true, until the function CloseSip() is called. StartSip() has to be called in the main function. It’s to prevent an unexpected end of the program, which will kill all the created thread, and interrupt improperly the SIP system.

The second function is used to completely finish SIP. It frees all memory, and closes all the SIP sessions still open

```c
int CloseSip();
```

### 4 functions to manage SIP sessions

OpenSessionSip() permits to open a new SIP session. An application is allow opening different session at the same time. The maximum number of possible opened sessions is defined in the file SIP_env.h.

```c
int OpenSessionSip(char* target, SIP_PACKET* pkt, void (*_received)(int, SIP_PACKET*), bool (*_negotia)(int, SIP_PACKET*), void (*_terminate)(int));
```

This function takes 5 parameters:
- **target**: URL of the correspond node
- **pkt**: packet to send with the INVITE. For more details about the SIP_PACKET type, see the data paragraph.
- **_received**: call-back function used to transmit the received data packet to the application
- **_negotia**: call-back function used to renegotiate the SDP parameter
- **_terminate**: call-back function used to alert the application that the SIP session is going to be closed.

The function returns -1 if no new session are available. Otherwise, it returns an id for the session. Because it takes time to open a session, the session is still opening, when the function returned the id.

The next function is used to simply close a session.

```c
int CloseSessionSip(int id);
```

The function takes only one parameter corresponding to the id of the session. It returns 0, if success, otherwise, it returns an error code, corresponding to one of them defined in the file SIP.h.

A session can have different state: ready, opening, closing, registering. The next session find the state of a SIP session.
D.3 Function interface

The function takes only one parameter corresponding to the id of the session. It returns a code, corresponding to one of them defined in the file SIP.h.

The last defined function for the SIP session permits to send data packet.

The function takes 2 arguments:

- **id**: defined the id of the session
- **packet**: corresponds to the packet to send

It returns 0, if success, otherwise, it returns an error code, corresponding to one of them defined in the file SIP.h.

5 call-back functions to manage SIP sessions

There are 6 call-back functions, which need to be defined by the application. The first one asking, is defined via the StartSip() function, and is called every time the application is invited to participate to a new SIP session.

The function takes 4 parameters:

- **int**: is corresponding to the new id for the new SIP session
- **SIP_PACKET**: is the SDP packet received. The function can change it, if the SDP parameters need to be renegotiate
- **CALLBACK_FUNCTIONS**: is a structure where the four other call-back functions need to be defined
- **char**: contains the URL of the person who sends the invitation

If the invitation is accepted, then the function needs to return true. Otherwise returns false.

The CALLBACK_FUNCTIONS structure is defined as follow:

```
typedef struct callback_functions {
    void (* _received) (int, SIP_PACKET*);
    bool (* _negotiation)(int, SIP_PACKET*);
    void (* _terminate) (int);
    bool (* _ringing)(int);
} CALLBACK_FUNCTIONS;
```

The _received function is used when the SIP session received a data packet. This packet is transmit to the application by using this function.
D.3 Function interface

```c
void (*__received)(int, SIP_PACKET*);
```

The function takes 2 arguments:

- `int`: is the id of the SIP session
- `SIP_PACKET*`: is the data packet

The `__negotiation` function is called every time the application can renegotiate the SDP parameter.

```c
bool (*__negotiation)(int, SIP_PACKET*);
```

The function takes also 2 parameters:

- `int`: is the id of the SIP session
- `SIP_PACKET*`: is the SDP packet. It has to be changed if the application needs to renegotiate the SDP parameters

If the new SDP parameters satisfy the application, this one has to return true. Otherwise, it returns false, and the transaction stops.

The `__terminate` function indicates to the application that the SIP session is going to be finished.

```c
void (*__terminate)(int);
```

The function uses only one parameter; correspond to the id of the session.

The `__ringing` function is used to prevent the final user of the new invitation.

```c
bool (*__ringing)(int);
```

The function uses only the id of the SIP session. If the invitation is accepted by the end user, then the function has to return true, otherwise it returns false.

The `__ready` function is used when all the SIP system is ready for opening session.

```c
void (*__ready)();
```

The function does not take any argument, and does not need to return something.
D.4 Send & receive

The send and receive blocks are the interface with the C++ code and the network (without consideration of the under layer code).

**Send block**

In our implementation, we have defined 3 kinds of sending:

1. **data sending**

   It is used to send data directly to the other application. Then the ports and the IP addresses used are the one defined by the function InitiationSendViaSip().

2. **SIP signalling without expecting response**

   This time, the packet is send to the P-CSCF, using the SIP port (defined in the StartSip() function). The packet is sent only one time, and no response is expected (for example, the ACK request, or simply a response of a received request).

3. **SIP signalling with expecting request**

   As for the previous case, the request is sent to the P-CSCF using the SIP port. But this time, because a response is expected, the request has to be safe in a buffer. And, according to the RFC, the request has to be sent several times, unless the response doesn’t arrive. The interval time of retransmission has to be multiply by 2, at each send.

The buffer of the sent packet is as follow:

```c
typedef struct sip_sent {
    SIPMESSAGE message ;
    int nextsend ;
    int idsession ;
    bool (* __func) (SIPMESSAGE*, SIPMESSAGE*, int);
    bool answered ;
    bool treated ;
    struct sip_sent* next ;
} SIP_SENT ;
```

There are 7 variables:

- **message**: corresponds to the message sent (to understand the difference between SIP_MESSAGE and SIP_PACKET, see the section 4).

- **nextsend**: is the interval time when the request has to be resent. (The minimal interval and maximal interval are defined in the file SIP_env.h)
D.5 Data

- **idsession**: contains the id of the SIP session of this request

- **_func**: is an internal call-back function. This function is going to be called when a response is going to arrive.

- **answered**: permits to know if a response has been received.

- **treated**: make the difference between a provisional response, and a final one. If this variable is true, then the message can be deleted from the buffer.

- **next**: is just a pointer to the next sent message. The buffer is allocated dynamically.

**Receive block**

All receive blocks corresponds to one launched threads. There are two type of receive blocks:

1. **data receive block**

   There are one data receive thread for each SIP session. The thread is launched, when the InitiationSendViaSession() function is called. Then, when a data packet is received, it is redirect to the corresponding call-back function.

2. **SIP signalling receive block**

   There is only one thread for the SIP signalling receive block, and for all the SIP sessions. It is launched at the beginning, when the StartSip() function is called. If the packet received is a request, it is redirect to a predefined function. If the packet is a response, it tries to find the corresponding request in the buffer, and then, the internal call-back function is called.

   Because of the RFC, a request can be sent many times. So the SIP receive block contains another buffer (with a defined variable - see the file SIP_env.h), which contains the last received request. When a received request is the same as an older one, it is discarded.

**D.5 Data**

**SIP_message vs SIP_packet**

We use 2 type of packet in the implementation. They are defined in the file SIP_message.h.

SIP_packet is the simplest one. It corresponds to a basic C++ class, containing a buffer (char*) and an integer, corresponding to the size of the buffer. All standard functions are defined, in order to manage the memory properly. This class is used, because
when sending a data packet, this one can contain the character \'0\'. Then the strlen() function will not be able to calculate the length of the message.

SIP_message is another class, which read SIP packets. It provides some simple functions in order to access/add/delete/modify some fields in a SIP message. The class is filled from a SIP_packet, and then it checks the SIP syntax. All the SIP manipulations are done in one class, providing robustness to the implementation.

**SIP environnement data**

The implementation use threads, in order to share the same memory space. In one hand, it permits to decrease the used memory (compare to process), but in the other hand we have to be careful: 2 threads can write in the same memory space at the same time. There are 2 types of data: one related to the global SIP environment, and one for each SIP session.

All the global environment variables are listed bellow:
typedef struct sip_context {
    pthread_mutex_t bolt_general;
    pthread_mutex_t bolt_sent;
    pthread_mutex_t bolt_received;
    pthread_mutex_t bolt_socket;
    char port[6];
    char localIP[50];
    char sip_uri[100];
    char uri_proxy[100];
    char proxyIP[50];
    char uri_machine[100];
    char domain_name[100];
    char call_id_reg[100];
    long cseq;
    SIP_SESSION session[MAX_SIP_SESSION];
    SIP_SENT sent;
    SIP_PACKET received[MAX_RECEIVED_MESSAGE];
    int next_received_position;
    int UDP_socket;
    bool registering;
    bool closing;
    bool finishing;
    bool (*__asking)(int, SIP_PACKET*, CALLBACK_FUNCTIONS*, char*)
};

typedef struct sip_context {
    pthread_mutex_t bolt_general;
    pthread_mutex_t bolt_sent;
    pthread_mutex_t bolt_received;
    pthread_mutex_t bolt_socket;
    char port[6];
    char localIP[50];
    char sip_uri[100];
    char uri_proxy[100];
    char proxyIP[50];
    char uri_machine[100];
    char domain_name[100];
    char call_id_reg[100];
    long cseq;
    SIP_SESSION session[MAX_SIP_SESSION];
    SIP_SENT sent;
    SIP_PACKET received[MAX_RECEIVED_MESSAGE];
    int next_received_position;
    int UDP_socket;
    bool registering;
    bool closing;
    bool finishing;
    bool (*__asking)(int, SIP_PACKET*, CALLBACK_FUNCTIONS*, char*)
};

• bolt_general: this mutex is used when a thread needs to change a global variable
• bolt_sent: this one is used when the buffer for send message has to be check or modify
• bolt_received: it is the same as before, but for received packet
• bolt_socket: because send and receive blocks use the same socket, it is done to prevent an utilisation at the same time
• port: contain the SIP port
• localIP: contain the IP address of the user
• sip_uri: it is the URL of the user connected
• uri_proxy/proxyIP: it is the URL/IP address of the proxy used to communicate with IMS
D.5 Data

- **uri_machine**: corresponds to the name of the machine used by the user
- **domain_name**: is the domain name of the machine
- **call_id_reg/cseq**: they are id/counter for REGISTER request
- **session**: it is the table containing all session parameters
- **sent**: it is the buffer for sent packet
- **received/next_received_position**: permits to manage the received packet buffer
- **UDP_socket**: socket used for SIP signalling packet
- **registering**: indicates that SIP is still registering
- **closing**: indicates that SIP is closing
- **finishing**: this one is used to ask all listening threads to finish
- **_asking**: this function is used when SIP received an INVITE request
- **bolt_file**: (optionnaly) used when all received/sent packet has to be print in a file

*SIP session data*

The session variable in the global environment corresponds to a table containing all the SIP sessions variable. The size of this table can be changed by using the variable MAX_SIP_SESSION, defined in the file SIP_env.h.

All SIP session variables are listed bellow:
typedef struct sip_session {
  bool busy;
  pthread_mutex_t bolt;
  pthread_mutex_t bolt_socket;
  pthread_t id_create;
  int UDP_socket;
  char uri_target[100];
  char targetIP[50];
  char port_to_send[50];
  char port_to_receive[50];
  char call_id[100];
  long cseq;
  SIP_MESSAGE invite;
  SIP_MESSAGE model;
  int state;
  bool received_update;
  bool invite_final_response;
  void (*)_received (int, SIP_PACKET*);
  bool (*)_negotiation(int, SIP_PACKET*);
  bool (*)_ringing(int);
  void (*)_terminate (int);
} SIP_SESSION;

- **busy**: indicates if the session is already used or not
- **bolt**: it is used to be sure that only one thread is running for a session at one time
- **bolt_socket**: the received/sent blocks for data packet used the same socket, so the implementation has to be careful to not use it in the same time
- **id_create**: contains the id of the listening thread. It is used, to wait the end of this thread before closing.
- **UDP_socket**: socket for data packet
- **uri_target/targetIP**: URL/IP address of the destination
- **port_to_send/port_to_receive**: ports of the destination/machine used to sent and received data packets
- **call_id/cseq**: id/counter of the SIP signalling session
- **invite**: when SIP received an INVITE request, it keeps it in its memory, because the final response has to be sent later
- **model**: contains all the basic fields for this SIP session (including Route header)
- **state**: contains the state of the SIP session. All states are defined in the file SIP.h.
D.5 Data

- **received_update**: permits to know if the UPDATE request has already been received.

- **invite_final_response**: permits to know if the final response has been received. If not, and the session has to be close, the sent request must be a CANCEL and not a BYE.

- **received/negotiation/ringing/terminate**: contains all the functions defined in previous part.
Appendix E

SIP Transaction (open a session)

In order to follow the SIP transaction between the UAC and the UAS, every received and sent messages have been saved in some files. In this appendix, there is a transaction example, from the UAC side and then from the UAS side.

After a complete opening session, an handover occurs from the UAC side. Without MIP integration, the UAC registers again, and open the session again, in order to renegotiate the SDP parameters.

E.1 Client side

*************** SENT DNS ***************

REGISTER sip:domain.com SIP/2.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:ls001@domain.com>
To: <sip:ls001@domain.com>
Call-ID: 98762515-8ad6-48e4-f2b9-18f49f36301c@sipclient.domain.com
CSeq: 1 REGISTER
Expires: 3600
Contact: <sip:ls001@sipclient.domain.com:32869>
Content-Length: 0

*************** SPECIAL SENT *************** 2005-05-03 13:49:11.653
SPECIAL_REGISTER sip:domain.com SIP/2.0
Call-ID: 98762515-8ad6-48e4-f2b9-18f49f36301c@sipclient.domain.com
E.1 Client side

*****************************************************************************

SIP/2.0 200 OK
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s001@domain.com>;tag=975711567
Call-ID: 98762515-8ad6-48e4-f2b9-18f49f36301c@sipclient.domain.com
CSeq: 1 REGISTER
Expires: 3600
Content-Length: 0
Max-Forwards: 68
Contact: <sip:1s001@sipclient.domain.com:32869>
*****************************************************************************

*****************************************************************************

INVITE sip:1s002@domain.com SIP/2.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 1 INVITE
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147
Infos: 192.168.2.25:32870

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000
*****************************************************************************

*****************************************************************************

SIP/2.0 100 Trying
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 1 INVITE
E.1 Client side

Max-Forwards: 69
Content-Type: application/sdp
Content-Length: 147
Infos: 192.168.2.25:32870

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

SIP/2.0 183 Session in progress
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@domain.com>
CSeq: 1 INVITE
Max-Forwards: 65
Content-Type: application/sdp
Content-Length: 147
Infos: 192.168.2.4:1075

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

*************** SPECIAL SENT *************** 2005-05-03 13:49:11.932
PRACK sip:1s002@domain.com SIP/2.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 2 PRACK
Max-Forwards: 70
E.1 Client side

Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

******************************* RECEIVED *******************************
SIP/2.0 200 OK
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 2 PRACK
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

******************************* SPECIAL SENT *******************************
UPDATE sip:1s002@domain.com SIP/2.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 3 UPDATE
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserName 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

SIP/2.0 200 OK
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 3 UPDATE
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserName 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

*************** SPECIAL SENT *************** 2005-05-03 13:49:11.950
PRACK sip:1s002@domain.com SIP/2.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 4 PRACK
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
E.1 Client side

```
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

SIP/2.0 200 OK
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 4 PRACK
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

SIP/2.0 180 Ringing
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 1 INVITE
Max-Forwards: 65
Content-Type: application/sdp
Content-Length: 147
Infos: 192.168.2.25:32870

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
```
E.1 Client side

c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

clearfix

************** RECEIVED ************** 2005-05-03 13:49:12.001
SIP/2.0 200 OK
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:ls001@domain.com>
To: <sip:ls002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:ls001@domain.com>
CSeq: 1 INVITE
Max-Forwards: 65
Content-Type: application/sdp
Content-Length: 147
Infos: 192.168.2.25:32870

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

clearfix

************** SENT ************** 2005-05-03 13:49:12.003
ACK sip:ls002@domain.com SIP/2.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:ls001@domain.com>
To: <sip:ls002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:ls001@sipclient.domain.com>
CSeq: 5 ACK
Content-Length: 0

clearfix

************** SENT DNS **************

REGISTER sip:domain.com SIP/2.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:ls001@domain.com>
E.1 Client side

To: <sip:1s001@domain.com>
Call-ID: 98762515-8ad6-48e4-f2b9-18f49f36301c@sipclient.domain.com
CSeq: 2 REGISTER
Expires: 3600
Contact: <sip:1s001@sipclient.domain.com:32869>
Content-Length: 0

******************** SPECIAL SENT *******************
SPECIAL_REGISTER sip:domain.com SIP/2.0
Call-ID: 98762515-8ad6-48e4-f2b9-18f49f36301c@sipclient.domain.com

******************** RECEIVED *********************
SIP/2.0 200 OK
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s001@domain.com>;tag=975711567
Call-ID: 98762515-8ad6-48e4-f2b9-18f49f36301c@sipclient.domain.com
CSeq: 2 REGISTER
Expires: 3600
Content-Length: 0
Max-Forwards: 68
Contact: <sip:1s001@sipclient.domain.com:32869>

******************** SENT ***********************
INVITE sip:1s002@domain.com SIP/2.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 6 INVITE
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147
Infos: 192.168.2.26:32870

v=0
c=IN IP4 100.101.102.103
m=audio 49170 RTP/AVP 0
E.1 Client side

```
E.1 Client side

a=rtpmap:0 PCMU/8000
*******************************

SIP/2.0 100 Trying
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 6 INVITE
Max-Forwards: 69
Content-Type: application/sdp
Content-Length: 147
Infos: 192.168.2.26:32870

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000
*******************************

SIP/2.0 183 Session in progress
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 6 INVITE
Max-Forwards: 65
Content-Type: application/sdp
Content-Length: 147
Infos: 192.168.2.4:1075

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```
E.1 Client side

********************************************************************************
*************** SPECIAL SENT *************** 2005-05-03 13:49:17.963
PRACK sip:1s002@domain.com SIP/2.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 7 PRACK
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

********************************************************************************
SIP/2.0 200 OK
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 7 PRACK
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

********************************************************************************
*************** SPECIAL SENT *************** 2005-05-03 13:49:18.000
E.1 Client side

UPDATE sip:1s002@domain.com SIP/2.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 8 UPDATE
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

SIP/2.0 200 OK
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 8 UPDATE
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

********************** SPECIAL SENT ****************** 2005-05-03 13:49:18.009
PRACK sip:1s002@domain.com SIP/2.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
E.1 Client side

To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 9 PRACK
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

SIP/2.0 200 OK
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 9 PRACK
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

SIP/2.0 200 OK
Via: SIP/2.0/UDP p-cscf.domain.com:5060;branch=z9hG4bK8a71af6c.1
Via: SIP/2.0/UDP s-cscf.domain.com:5061;branch=z9hG4bK936c9724.1
Via: SIP/2.0/UDP i-cscf.domain.com:5062;branch=z9hG4bKa4f8b444.0
Via: SIP/2.0/UDP s-cscf.domain.com:5061;branch=z9hG4bK486c8672.0
Via: SIP/2.0/UDP p-cscf.domain.com:5060;branch=z9hG4bK7b10986d.0

Project report - Axel Adenet - Alexandre Boursier - Thior Santander Marin
E.2 Server side

Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@domain.com>
CSeq: 6 INVITE
Max-Forwards: 65
Content-Type: application/sdp
Content-Length: 147
Infos: 192.168.2.26:32870

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

************** SENT DNS **************

ACK sip:1s002@domain.com SIP/2.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@client.domain.com>
CSeq: 10 ACK
Content-Length: 0

E.2 Server side

************** SENT DNS **************

************** SENT ************** 2005-05-03 14:06:00.097
REGISTER sip:domain.com SIP/2.0
Via: SIP/2.0/UDP sipserver.domain.com:1074
From: <sip:1s002@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 7a542cb1-06e4-497f-e741-5504a80f6c6d@sipserver.domain.com
CSeq: 1 REGISTER
Expires: 3600
Contact: <sip:1s002@sipserver.domain.com:1074>
Content-Length: 0

************** SPECIAL SENT ************** 2005-05-03 14:06:00.098
SPECIAL REGISTER sip:domain.com SIP/2.0
Call-ID: 7a542cb1-06e4-497f-e741-5504a80f6c6d@sipserver.domain.com

************** RECEIVED ************** 2005-05-03 14:06:00.177
SIP/2.0 200 OK
Via: SIP/2.0/UDP sipserver.domain.com:1074
From: <sip:1s002@domain.com>
To: <sip:1s002@domain.com>;tag=975711567
Call-ID: 7a542cb1-06e4-497f-e741-5504a80f6c6d@sipserver.domain.com
CSeq: 1 REGISTER
Expires: 3600
Content-Length: 0
Max-Forwards: 68
Contact: <sip:1s002@sipserver.domain.com:1074>

************** RECEIVED ************** 2005-05-03 14:06:04.928
INVITE sip:1s002@sipserver.domain.com:1074 SIP/2.0
Via: SIP/2.0/UDP p-cscf.domain.com:5060;branch=z9hG4bKb55c51a6.1
Via: SIP/2.0/UDP s-cscf.domain.com:5061;branch=z9hG4bKdd09a6d6.1
Via: SIP/2.0/UDP i-cscf.domain.com:5062;branch=z9hG4bKfb426d81.0
Via: SIP/2.0/UDP s-cscf.domain.com:5061;branch=z9hG4bK9df89f.0
Via: SIP/2.0/UDP p-cscf.domain.com:5060;branch=z9hG4bK7b10986a.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@domain.com>
CSeq: 1 INVITE
Max-Forwards: 65
Content-Type: application/sdp
Content-Length: 147
Infos: 192.168.2.25:32870

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
E.2 Server side

t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

*******************************

************** SENT ************** 2005-05-03 14:06:04.929
SIP/2.0 100 Trying
Via: SIP/2.0/UDP p-cscf.domain.com:5060;branch=z9hG4bKb55c51a6.1
Via: SIP/2.0/UDP s-cscf.domain.com:5061;branch=z9hG4bKdd09a6d6.1
Via: SIP/2.0/UDP i-cscf.domain.com:5062;branch=z9hG4bKfb426d81.0
Via: SIP/2.0/UDP s-cscf.domain.com:5061;branch=z9hG4bKfb426d81.0
Via: SIP/2.0/UDP p-cscf.domain.com:5060;branch=z9hG4bK7b10986a.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@domain.com>
CSeq: 1 INVITE
Max-Forwards: 65
Content-Type: application/sdp
Content-Length: 147
Infos: 192.168.2.25:32870

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

*******************************

************** SENT ************** 2005-05-03 14:06:04.930
SIP/2.0 183 Session in progress
Via: SIP/2.0/UDP p-cscf.domain.com:5060;branch=z9hG4bKb55c51a6.1
Via: SIP/2.0/UDP s-cscf.domain.com:5061;branch=z9hG4bKdd09a6d6.1
Via: SIP/2.0/UDP i-cscf.domain.com:5062;branch=z9hG4bKfb426d81.0
Via: SIP/2.0/UDP s-cscf.domain.com:5061;branch=z9hG4bKfb426d81.0
Via: SIP/2.0/UDP p-cscf.domain.com:5060;branch=z9hG4bK7b10986a.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@domain.com>
CSeq: 1 INVITE
E.2 Server side

Max-Forwards: 65
Content-Type: application/sdp
Content-Length: 147
Infos: 192.168.2.4:1075

v=0
o=UserID 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

*************** RECEIVED *************** 2005-05-03 14:06:04.955
PRACK sip:1s002@domain.com SIP/2.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 2 PRACK
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserID 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

*************** SPECIAL SENT *************** 2005-05-03 14:06:04.956
SIP/2.0 200 OK
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 2 PRACK
Max-Forwards: 70
Content-Type: application/sdp
E.2 Server side

Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

***************************** RECEIVED ********************* 2005-05-03 14:06:04.963
UPDATE sip:1s002@domain.com SIP/2.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 3 UPDATE
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

***************************** SPECIAL SENT ********************* 2005-05-03 14:06:04.964
SIP/2.0 200 OK
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 3 UPDATE
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
E.2 Server side

```
---
.
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

*************** RECEIVED *************** 2005-05-03 14:06:04.971
PRACK sip:1s002@domain.com SIP/2.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 4 PRACK
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

*************** SPECIAL SENT *************** 2005-05-03 14:06:04.972
SIP/2.0 200 OK
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 4 PRACK
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
```
E.2 Server side

```
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

************** SENT ************** 2005-05-03 14:06:04.973
SIP/2.0 180 Ringing
Via: SIP/2.0/UDP p-cscf.domain.com:5060;branch=z9hG4bKb55c51a6.1
Via: SIP/2.0/UDP s-cscf.domain.com:5061;branch=z9hG4bKdd09a6d6.1
Via: SIP/2.0/UDP i-cscf.domain.com:5062;branch=z9hG4bKfb426d81.0
Via: SIP/2.0/UDP s-cscf.domain.com:5061;branch=z9hG4bK9df4e89f.0
Via: SIP/2.0/UDP p-cscf.domain.com:5060;branch=z9hG4bK7b10986a.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@domain.com>
CSeq: 1 INVITE
Max-Forwards: 65
Content-Type: application/sdp
Content-Length: 147
Infos: 192.168.2.25:32870

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

************** SENT ************** 2005-05-03 14:06:04.973
SIP/2.0 200 OK
Via: SIP/2.0/UDP p-cscf.domain.com:5060;branch=z9hG4bKb55c51a6.1
Via: SIP/2.0/UDP s-cscf.domain.com:5061;branch=z9hG4bKdd09a6d6.1
Via: SIP/2.0/UDP i-cscf.domain.com:5062;branch=z9hG4bKfb426d81.0
Via: SIP/2.0/UDP s-cscf.domain.com:5061;branch=z9hG4bK9df4e89f.0
Via: SIP/2.0/UDP p-cscf.domain.com:5060;branch=z9hG4bK7b10986a.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@domain.com>
CSeq: 1 INVITE
```
Max-Forwards: 65
Content-Type: application/sdp
Content-Length: 147
Infos: 192.168.2.25:32870

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

******************************* RECEIVED ************** 2005-05-03 14:06:10.974
INVITE sip:1s002@sipserver.domain.com:1074 SIP/2.0
Via: SIP/2.0/UDP p-cscf.domain.com:5060;branch=z9hG4bK8a71af6c.1
Via: SIP/2.0/UDP s-cscf.domain.com:5061;branch=z9hG4bK936c9724.1
Via: SIP/2.0/UDP i-cscf.domain.com:5062;branch=z9hG4bKa4f8b444.0
Via: SIP/2.0/UDP s-cscf.domain.com:5061;branch=z9hG4bK486c8672.0
Via: SIP/2.0/UDP p-cscf.domain.com:5060;branch=z9hG4bK7b10986d.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@domain.com>
CSeq: 6 INVITE
Max-Forwards: 65
Content-Type: application/sdp
Content-Length: 147
Infos: 192.168.2.26:32870

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

******************************* SENT ************** 2005-05-03 14:06:10.975
SIP/2.0 100 Trying
Via: SIP/2.0/UDP p-cscf.domain.com:5060;branch=z9hG4bK8a71af6c.1
Via: SIP/2.0/UDP s-cscf.domain.com:5061;branch=z9hG4bK936c9724.1
E.2 Server side

Via: SIP/2.0/UDP i-cscf.domain.com:5062;branch=z9hG4bKa4f8b444.0
Via: SIP/2.0/UDP s-cscf.domain.com:5061;branch=z9hG4bK486c8672.0
Via: SIP/2.0/UDP p-cscf.domain.com:5060;branch=z9hG4bK7b10986d.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:ls001@domain.com>
To: <sip:ls002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:ls001@domain.com>
CSeq: 6 INVITE
Max-Forwards: 65
Content-Type: application/sdp
Content-Length: 147
Infos: 192.168.2.26:32870

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

**********************************************************************

************** SENT ************** 2005-05-03 14:06:10.976
SIP/2.0 183 Session in progress
Via: SIP/2.0/UDP p-cscf.domain.com:5060;branch=z9hG4bK8a71af6c.1
Via: SIP/2.0/UDP s-cscf.domain.com:5061;branch=z9hG4bK936c9724.1
Via: SIP/2.0/UDP i-cscf.domain.com:5062;branch=z9hG4bKa4f8b444.0
Via: SIP/2.0/UDP s-cscf.domain.com:5061;branch=z9hG4bK486c8672.0
Via: SIP/2.0/UDP p-cscf.domain.com:5060;branch=z9hG4bK7b10986d.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:ls001@domain.com>
To: <sip:ls002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:ls001@domain.com>
CSeq: 6 INVITE
Max-Forwards: 65
Content-Type: application/sdp
Content-Length: 147
Infos: 192.168.2.4:1075

v=0
o=ls001 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
E.2 Server side

t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

************** RECEIVED ************** 2005-05-03 14:06:11.030
PRACK sip:1s002@domain.com SIP/2.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 7 PRACK
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

************** SPECIAL SENT ************** 2005-05-03 14:06:11.030
SIP/2.0 200 OK
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 7 PRACK
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000
E.2 Server side

******************** RECEIVED ******************* 2005-05-03 14:06:11.038
UPDATE sip:1s002@domain.com SIP/2.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 8 UPDATE
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

******************** SPECIAL SENT ******************* 2005-05-03 14:06:11.039
SIP/2.0 200 OK
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 8 UPDATE
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

******************** RECEIVED ******************* 2005-05-03 14:06:11.047

--- Project report - Axel Adenet - Alexandre Boursier - Thior Santander Marin ---
E.2 Server side

PRACK sip:1s002@domain.com SIP/2.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 9 PRACK
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

****************************** SPECIAL SENT ************** 2005-05-03 14:06:11.047
SIP/2.0 200 OK
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@sipclient.domain.com>
CSeq: 9 PRACK
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 147

v=0
o=UserA 2890844526 2890844526 IN IP4 here.com
s=Session SDP
c=IN IP4 100.101.102.103
t=0 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000

****************************** SPECIAL SENT ************** 2005-05-03 14:06:11.047
SIP/2.0 200 OK
Via: SIP/2.0/UDP p-cscf.domain.com:5060
Via: SIP/2.0/UDP s-cscf.domain.com:5061

E.2 Server side

Via: SIP/2.0/UDP i-cscf.domain.com:5062;branch=z9hG4bKa4f8b444.0
Via: SIP/2.0/UDP s-cscf.domain.com:5061;branch=z9hG4bK486c8672.0
Via: SIP/2.0/UDP p-cscf.domain.com:5060;branch=z9hG4bK7b10986d.0
Via: SIP/2.0/UDP sipclient.domain.com:32869
From: <sip:1s001@domain.com>
To: <sip:1s002@domain.com>
Call-ID: 6074ed8f-9390-98bb-53b4-560569ae74ed@sipclient.domain.com
Contact: <sip:1s001@domain.com>
CSeq: 6 INVITE
Max-Forwards: 65
Content-Type: application/sdp
Content-Length: 147
Infos: 192.168.2.26:32870

v=0
o=UserName 2890844526 2890844526 IN IP4 here.com
s=Session SDP
a=rtpmap:0 PCMU/8000
******************************************
Test measurements

We have done different tests with our prototype, in order to evaluate the performance of the integration of MIP in IMS.

The tests have been according the configuration described in the report.

We have measured different time intervals:

- "time to get IP address": time interval between the script to change network starts and the mobile node gets its new IP address
- "arrival time of the 1st data packet": time interval between the new IP address and the first data packet arrived in the mobile node
- "number of lost packets": number of lost packets during all the handover processing
- "estimated lost packets": estimation of lost packets after the mobile node gets its new IP address. The data rate is 1 packet of 700 bytes every 40ms.

F.1 MIP only

First we have measured the performance of Mobile IPv6. After opening a SIP session using IMS, the correspondent node sends data packets (a data packet of 700 bytes every 40 ms, corresponding to a data rate of 140 kbits/s). Then, an handover is done through the two access points.

In this case, the mobility is solved by MIP only: there is no SIP exchange after the handover, and the UAC doesn’t send any packets, except the binding updates or the router solicitations.
### F.1 MIP only

<table>
<thead>
<tr>
<th>time to get IP address</th>
<th>arrival time of the 1st data packet</th>
<th>number of lost packets</th>
<th>estimated lost packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>5,065396</td>
<td>2,065156</td>
<td>171</td>
<td>44,3651</td>
</tr>
<tr>
<td>7,791746</td>
<td>1,947507</td>
<td>233</td>
<td>38,20635</td>
</tr>
<tr>
<td>1,375859</td>
<td>1,67392</td>
<td>73</td>
<td>38,603525</td>
</tr>
<tr>
<td>1,365978</td>
<td>2,065099</td>
<td>83</td>
<td>48,85055</td>
</tr>
<tr>
<td>7,978259</td>
<td>2,101134</td>
<td>242</td>
<td>42,543525</td>
</tr>
<tr>
<td>1,180912</td>
<td>1,881208</td>
<td>74</td>
<td>44,4772</td>
</tr>
<tr>
<td>5,067983</td>
<td>1,428185</td>
<td>155</td>
<td>28,300425</td>
</tr>
<tr>
<td>5,637234</td>
<td>2,046037</td>
<td>183</td>
<td>42,06915</td>
</tr>
<tr>
<td>1,179861</td>
<td>2,157691</td>
<td>80</td>
<td>50,503475</td>
</tr>
<tr>
<td>3,686306</td>
<td>1,758155</td>
<td>130</td>
<td>37,84235</td>
</tr>
<tr>
<td>1,130082</td>
<td>1,997076</td>
<td>76</td>
<td>47,74795</td>
</tr>
<tr>
<td>1,409932</td>
<td>1,932844</td>
<td>80</td>
<td>44,7517</td>
</tr>
<tr>
<td>1,138858</td>
<td>1,714204</td>
<td>69</td>
<td>40,52855</td>
</tr>
<tr>
<td>4,788455</td>
<td>2,049345</td>
<td>165</td>
<td>45,288625</td>
</tr>
<tr>
<td>4,232444</td>
<td>1,475031</td>
<td>136</td>
<td>30,1889</td>
</tr>
<tr>
<td>1,339727</td>
<td>1,382731</td>
<td>65</td>
<td>31,506825</td>
</tr>
<tr>
<td>9,088801</td>
<td>1,93629</td>
<td>264</td>
<td>36,779975</td>
</tr>
<tr>
<td>7,282226</td>
<td>1,653406</td>
<td>215</td>
<td>32,94435</td>
</tr>
<tr>
<td>1,079916</td>
<td>1,797873</td>
<td>69</td>
<td>42,0021</td>
</tr>
<tr>
<td>1,014911</td>
<td>1,774902</td>
<td>67</td>
<td>41,627225</td>
</tr>
<tr>
<td>8,054212</td>
<td>2,140709</td>
<td>244</td>
<td>42,6447</td>
</tr>
<tr>
<td>1,141124</td>
<td>2,020997</td>
<td>77</td>
<td>48,4719</td>
</tr>
<tr>
<td>5,814261</td>
<td>1,364778</td>
<td>172</td>
<td>26,643475</td>
</tr>
<tr>
<td>1,2627</td>
<td>1,523855</td>
<td>67</td>
<td>35,4325</td>
</tr>
<tr>
<td>1,344811</td>
<td>1,980027</td>
<td>80</td>
<td>46,379725</td>
</tr>
<tr>
<td>1,185793</td>
<td>1,612674</td>
<td>67</td>
<td>37,355175</td>
</tr>
<tr>
<td>1,076142</td>
<td>1,869657</td>
<td>71</td>
<td>44,09645</td>
</tr>
<tr>
<td>1,419038</td>
<td>1,343313</td>
<td>66</td>
<td>30,52405</td>
</tr>
<tr>
<td>1,078076</td>
<td>1,524825</td>
<td>63</td>
<td>36,0481</td>
</tr>
<tr>
<td>5,208444</td>
<td>2,178728</td>
<td>177</td>
<td>46,7889</td>
</tr>
<tr>
<td>6,00702</td>
<td>1,860091</td>
<td>189</td>
<td>38,8245</td>
</tr>
<tr>
<td>9,552258</td>
<td>1,990437</td>
<td>276</td>
<td>37,19355</td>
</tr>
<tr>
<td>9,447615</td>
<td>1,804375</td>
<td>271</td>
<td>34,809625</td>
</tr>
<tr>
<td>1,422094</td>
<td>2,099875</td>
<td>85</td>
<td>49,44765</td>
</tr>
<tr>
<td>1,090415</td>
<td>1,840463</td>
<td>71</td>
<td>43,739625</td>
</tr>
<tr>
<td>3,272489</td>
<td>1,66982</td>
<td>120</td>
<td>38,187775</td>
</tr>
<tr>
<td>4,285157</td>
<td>1,748608</td>
<td>146</td>
<td>38,871075</td>
</tr>
<tr>
<td>1,438692</td>
<td>2,101656</td>
<td>86</td>
<td>50,0327</td>
</tr>
<tr>
<td>1,03347</td>
<td>1,288064</td>
<td>56</td>
<td>30,16325</td>
</tr>
<tr>
<td>1,320998</td>
<td>1,614088</td>
<td>71</td>
<td>37,77255</td>
</tr>
<tr>
<td>Time to get IP address</td>
<td>Arrival time of the 1st data packet</td>
<td>Number of lost packets</td>
<td>Estimated lost packets</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------------</td>
<td>------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>1.163328</td>
<td>1.915834</td>
<td>75</td>
<td>45.9168</td>
</tr>
<tr>
<td>1.071602</td>
<td>1.928209</td>
<td>73</td>
<td>46.20995</td>
</tr>
<tr>
<td>1.001843</td>
<td>1.579768</td>
<td>63</td>
<td>37.953925</td>
</tr>
<tr>
<td>7.245976</td>
<td>1.810724</td>
<td>220</td>
<td>38.8506</td>
</tr>
<tr>
<td>1.253443</td>
<td>1.955282</td>
<td>78</td>
<td>46.663925</td>
</tr>
<tr>
<td>1.188688</td>
<td>1.501687</td>
<td>66</td>
<td>36.2828</td>
</tr>
<tr>
<td>1.133547</td>
<td>1.473587</td>
<td>63</td>
<td>34.661325</td>
</tr>
<tr>
<td>1.202304</td>
<td>2.111656</td>
<td>80</td>
<td>49.9424</td>
</tr>
<tr>
<td>1.323836</td>
<td>1.991037</td>
<td>81</td>
<td>47.9041</td>
</tr>
<tr>
<td>1.459535</td>
<td>2.056948</td>
<td>85</td>
<td>48.511625</td>
</tr>
<tr>
<td>4.442411</td>
<td>1.672408</td>
<td>148</td>
<td>36.939725</td>
</tr>
<tr>
<td>1.088754</td>
<td>1.593664</td>
<td>65</td>
<td>37.78115</td>
</tr>
<tr>
<td>1.312763</td>
<td>1.838961</td>
<td>76</td>
<td>43.180925</td>
</tr>
<tr>
<td>1.328573</td>
<td>1.81684</td>
<td>76</td>
<td>42.785675</td>
</tr>
<tr>
<td>2.169392</td>
<td>1.828455</td>
<td>97</td>
<td>42.7652</td>
</tr>
<tr>
<td>7.631929</td>
<td>1.60601</td>
<td>223</td>
<td>32.201775</td>
</tr>
<tr>
<td>1.456594</td>
<td>2.053001</td>
<td>85</td>
<td>48.58515</td>
</tr>
<tr>
<td>1.290544</td>
<td>1.646456</td>
<td>71</td>
<td>38.7364</td>
</tr>
<tr>
<td>1.217954</td>
<td>2.142602</td>
<td>81</td>
<td>50.55115</td>
</tr>
<tr>
<td>5.223768</td>
<td>1.726796</td>
<td>168</td>
<td>37.4058</td>
</tr>
<tr>
<td>4.305015</td>
<td>2.054124</td>
<td>154</td>
<td>46.374625</td>
</tr>
<tr>
<td>2.327783</td>
<td>1.655592</td>
<td>97</td>
<td>38.805425</td>
</tr>
<tr>
<td>1.339481</td>
<td>1.856289</td>
<td>78</td>
<td>44.512975</td>
</tr>
<tr>
<td>1.511782</td>
<td>1.819946</td>
<td>81</td>
<td>43.20545</td>
</tr>
<tr>
<td>1.304659</td>
<td>2.099609</td>
<td>82</td>
<td>49.383525</td>
</tr>
<tr>
<td>1.094574</td>
<td>1.822374</td>
<td>71</td>
<td>43.63565</td>
</tr>
<tr>
<td>1.297626</td>
<td>2.046727</td>
<td>81</td>
<td>48.55935</td>
</tr>
<tr>
<td>1.266802</td>
<td>1.803497</td>
<td>75</td>
<td>43.32995</td>
</tr>
<tr>
<td>1.111064</td>
<td>1.477026</td>
<td>63</td>
<td>35.2234</td>
</tr>
<tr>
<td>1.217528</td>
<td>2.065883</td>
<td>80</td>
<td>49.5618</td>
</tr>
<tr>
<td>1.036904</td>
<td>2.16193</td>
<td>78</td>
<td>52.0774</td>
</tr>
<tr>
<td>1.432861</td>
<td>2.091446</td>
<td>85</td>
<td>49.178475</td>
</tr>
<tr>
<td>1.136864</td>
<td>1.674612</td>
<td>68</td>
<td>39.5784</td>
</tr>
<tr>
<td>1.451031</td>
<td>1.390946</td>
<td>69</td>
<td>32.724225</td>
</tr>
<tr>
<td>1.234857</td>
<td>2.107506</td>
<td>81</td>
<td>50.128575</td>
</tr>
<tr>
<td>1.083966</td>
<td>1.887039</td>
<td>72</td>
<td>44.90085</td>
</tr>
<tr>
<td>1.036159</td>
<td>1.938373</td>
<td>72</td>
<td>46.096025</td>
</tr>
<tr>
<td>1.35162</td>
<td>1.713805</td>
<td>75</td>
<td>41.2095</td>
</tr>
<tr>
<td>2.147212</td>
<td>1.382447</td>
<td>85</td>
<td>31.3197</td>
</tr>
<tr>
<td>1.16711</td>
<td>1.598828</td>
<td>67</td>
<td>37.82225</td>
</tr>
<tr>
<td>1.155913</td>
<td>2.075256</td>
<td>78</td>
<td>49.102175</td>
</tr>
</tbody>
</table>
F.1 MIP only

<table>
<thead>
<tr>
<th>time to get IP address</th>
<th>arrival time of the 1st data packet</th>
<th>number of lost packets</th>
<th>estimated lost packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.010232</td>
<td>1.926369</td>
<td>71</td>
<td>45,7442</td>
</tr>
<tr>
<td>3.535922</td>
<td>1.782491</td>
<td>129</td>
<td>40,60195</td>
</tr>
<tr>
<td>1.292959</td>
<td>1.481007</td>
<td>67</td>
<td>34,676025</td>
</tr>
<tr>
<td>8.900167</td>
<td>1.709969</td>
<td>257</td>
<td>34,495825</td>
</tr>
<tr>
<td>1.408863</td>
<td>2.038447</td>
<td>84</td>
<td>48,778425</td>
</tr>
<tr>
<td>1.489811</td>
<td>1.75326</td>
<td>79</td>
<td>41,754725</td>
</tr>
<tr>
<td>1.27506</td>
<td>2.052424</td>
<td>81</td>
<td>49,1235</td>
</tr>
<tr>
<td>1.409489</td>
<td>1.857432</td>
<td>79</td>
<td>43,762775</td>
</tr>
<tr>
<td>1.016258</td>
<td>2.10535</td>
<td>76</td>
<td>50,59355</td>
</tr>
<tr>
<td>5.607069</td>
<td>1.589715</td>
<td>174</td>
<td>33,823275</td>
</tr>
<tr>
<td>1.01855</td>
<td>2.005594</td>
<td>74</td>
<td>48,53625</td>
</tr>
<tr>
<td>1.0909</td>
<td>1.900611</td>
<td>72</td>
<td>44,7275</td>
</tr>
<tr>
<td>2.230698</td>
<td>2.028888</td>
<td>103</td>
<td>47,23255</td>
</tr>
<tr>
<td>8.767209</td>
<td>1.725917</td>
<td>254</td>
<td>34,81975</td>
</tr>
<tr>
<td>9.134161</td>
<td>1.52414</td>
<td>258</td>
<td>29,645975</td>
</tr>
<tr>
<td>4.261677</td>
<td>1.862528</td>
<td>148</td>
<td>41,458075</td>
</tr>
<tr>
<td>4.009175</td>
<td>1.85761</td>
<td>142</td>
<td>41,770625</td>
</tr>
<tr>
<td>1.41539</td>
<td>1.827982</td>
<td>79</td>
<td>43,61525</td>
</tr>
<tr>
<td>2.809307</td>
<td>2.052237</td>
<td>118</td>
<td>47,767325</td>
</tr>
<tr>
<td>2.006839</td>
<td>1.788754</td>
<td>92</td>
<td>41,829025</td>
</tr>
<tr>
<td>1.393315</td>
<td>1.653806</td>
<td>74</td>
<td>39,167125</td>
</tr>
<tr>
<td>1.307594</td>
<td>1.347219</td>
<td>65</td>
<td>32,31015</td>
</tr>
<tr>
<td>1.221361</td>
<td>2.150557</td>
<td>82</td>
<td>51,465975</td>
</tr>
<tr>
<td>1.494751</td>
<td>1.955609</td>
<td>84</td>
<td>46,631225</td>
</tr>
<tr>
<td>7.703803</td>
<td>1.743961</td>
<td>229</td>
<td>36,404925</td>
</tr>
<tr>
<td>1.293638</td>
<td>2.073161</td>
<td>82</td>
<td>49,65905</td>
</tr>
<tr>
<td>1.491872</td>
<td>1.944873</td>
<td>84</td>
<td>46,7032</td>
</tr>
<tr>
<td>1.42939</td>
<td>1.43641</td>
<td>70</td>
<td>34,26525</td>
</tr>
<tr>
<td>1.371894</td>
<td>1.921599</td>
<td>80</td>
<td>45,70265</td>
</tr>
<tr>
<td>1.162644</td>
<td>1.747507</td>
<td>70</td>
<td>40,9339</td>
</tr>
<tr>
<td>1.331562</td>
<td>1.860752</td>
<td>78</td>
<td>44,71095</td>
</tr>
<tr>
<td>1.296688</td>
<td>1.616302</td>
<td>70</td>
<td>37,5828</td>
</tr>
<tr>
<td>7.298673</td>
<td>1.500034</td>
<td>213</td>
<td>30,533175</td>
</tr>
<tr>
<td>8.012676</td>
<td>1.983378</td>
<td>243</td>
<td>42,8631</td>
</tr>
<tr>
<td>1.238658</td>
<td>1.277154</td>
<td>61</td>
<td>30,03355</td>
</tr>
<tr>
<td>1.241824</td>
<td>1.8631</td>
<td>76</td>
<td>44,9544</td>
</tr>
<tr>
<td>1.172735</td>
<td>1.853162</td>
<td>73</td>
<td>43,681625</td>
</tr>
<tr>
<td>1.260654</td>
<td>1.585715</td>
<td>69</td>
<td>37,48365</td>
</tr>
<tr>
<td>1.205712</td>
<td>2.111011</td>
<td>80</td>
<td>49,8572</td>
</tr>
<tr>
<td>1.037867</td>
<td>1.803606</td>
<td>69</td>
<td>43,053325</td>
</tr>
<tr>
<td>1.232676</td>
<td>1.730076</td>
<td>72</td>
<td>41,1831</td>
</tr>
<tr>
<td>time to get IP address</td>
<td>arrival time of the 1st data packet</td>
<td>number of lost packets</td>
<td>estimated lost packets</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------------</td>
<td>------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>7.433173</td>
<td>1.833837</td>
<td>225</td>
<td>39.170675</td>
</tr>
<tr>
<td>1.23216</td>
<td>1.456793</td>
<td>65</td>
<td>34.196</td>
</tr>
<tr>
<td>7.816515</td>
<td>1.483093</td>
<td>226</td>
<td>30.587125</td>
</tr>
<tr>
<td>7.676627</td>
<td>1.846393</td>
<td>230</td>
<td>38.084325</td>
</tr>
<tr>
<td>1.303643</td>
<td>2.054456</td>
<td>82</td>
<td>49.408925</td>
</tr>
<tr>
<td>7.691273</td>
<td>1.867408</td>
<td>232</td>
<td>39.718175</td>
</tr>
<tr>
<td>1.105065</td>
<td>1.710072</td>
<td>68</td>
<td>40.373375</td>
</tr>
<tr>
<td>1.02787</td>
<td>2.143471</td>
<td>77</td>
<td>51.30325</td>
</tr>
<tr>
<td>7.461889</td>
<td>1.668245</td>
<td>221</td>
<td>34.452775</td>
</tr>
<tr>
<td>8.261679</td>
<td>1.991402</td>
<td>248</td>
<td>41.458025</td>
</tr>
<tr>
<td>5.300662</td>
<td>1.825161</td>
<td>173</td>
<td>40.48345</td>
</tr>
<tr>
<td>4.749595</td>
<td>1.76761</td>
<td>158</td>
<td>39.260125</td>
</tr>
<tr>
<td>1.027821</td>
<td>2.087885</td>
<td>76</td>
<td>50.304475</td>
</tr>
<tr>
<td>1.24866</td>
<td>1.484911</td>
<td>66</td>
<td>34.7835</td>
</tr>
<tr>
<td>1.073117</td>
<td>1.837973</td>
<td>71</td>
<td>44.172075</td>
</tr>
<tr>
<td>5.482175</td>
<td>1.565423</td>
<td>170</td>
<td>32.945625</td>
</tr>
<tr>
<td>1.049164</td>
<td>1.813377</td>
<td>69</td>
<td>42.7709</td>
</tr>
<tr>
<td>1.362492</td>
<td>1.829118</td>
<td>78</td>
<td>43.9377</td>
</tr>
<tr>
<td>5.55577</td>
<td>1.956203</td>
<td>183</td>
<td>44.10575</td>
</tr>
<tr>
<td>7.507837</td>
<td>1.537248</td>
<td>219</td>
<td>31.304075</td>
</tr>
<tr>
<td>4.651461</td>
<td>1.293043</td>
<td>144</td>
<td>27.713475</td>
</tr>
<tr>
<td>1.373872</td>
<td>2.080598</td>
<td>84</td>
<td>49.6532</td>
</tr>
<tr>
<td>2.585574</td>
<td>1.798532</td>
<td>107</td>
<td>42.36065</td>
</tr>
<tr>
<td>3.498514</td>
<td>1.822281</td>
<td>130</td>
<td>42.53715</td>
</tr>
<tr>
<td>1.44625</td>
<td>1.394203</td>
<td>69</td>
<td>32.84375</td>
</tr>
<tr>
<td>1.198069</td>
<td>2.074154</td>
<td>80</td>
<td>50.048275</td>
</tr>
<tr>
<td>3.703346</td>
<td>2.134082</td>
<td>141</td>
<td>48.41635</td>
</tr>
<tr>
<td>1.433299</td>
<td>1.567462</td>
<td>73</td>
<td>37.167525</td>
</tr>
<tr>
<td>8.29015</td>
<td>1.956852</td>
<td>249</td>
<td>41.74025</td>
</tr>
<tr>
<td>2.42932</td>
<td>1.564795</td>
<td>97</td>
<td>36.267</td>
</tr>
<tr>
<td>10.281578</td>
<td>2.09426</td>
<td>300</td>
<td>42.96055</td>
</tr>
<tr>
<td>1.116129</td>
<td>1.844347</td>
<td>73</td>
<td>43.96775</td>
</tr>
<tr>
<td>9.429808</td>
<td>1.373778</td>
<td>261</td>
<td>25.2548</td>
</tr>
<tr>
<td>1.197504</td>
<td>1.659565</td>
<td>69</td>
<td>39.0624</td>
</tr>
<tr>
<td>5.315871</td>
<td>1.581354</td>
<td>168</td>
<td>35.103225</td>
</tr>
<tr>
<td>1.269619</td>
<td>2.000719</td>
<td>80</td>
<td>48.259525</td>
</tr>
<tr>
<td>6.194683</td>
<td>1.87593</td>
<td>196</td>
<td>41.132925</td>
</tr>
<tr>
<td>1.211229</td>
<td>1.935709</td>
<td>76</td>
<td>45.719275</td>
</tr>
<tr>
<td>2.163393</td>
<td>2.075072</td>
<td>103</td>
<td>48.915175</td>
</tr>
<tr>
<td>6.387365</td>
<td>2.084288</td>
<td>205</td>
<td>45.315875</td>
</tr>
<tr>
<td>1.16848</td>
<td>2.133466</td>
<td>80</td>
<td>50.788</td>
</tr>
<tr>
<td>time to get IP address</td>
<td>arrival time of the 1st data packet</td>
<td>number of lost packets</td>
<td>estimated lost packets</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------------</td>
<td>------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>1.331384</td>
<td>2.10495</td>
<td>83</td>
<td>49,7154</td>
</tr>
<tr>
<td>1.304553</td>
<td>1.973385</td>
<td>79</td>
<td>46,386175</td>
</tr>
<tr>
<td>1.233648</td>
<td>1.791247</td>
<td>74</td>
<td>43,1588</td>
</tr>
<tr>
<td>5.709985</td>
<td>1.935072</td>
<td>185</td>
<td>42,250375</td>
</tr>
<tr>
<td>1.230975</td>
<td>1.923167</td>
<td>77</td>
<td>46,225625</td>
</tr>
<tr>
<td>1.425564</td>
<td>1.950955</td>
<td>80</td>
<td>44,3609</td>
</tr>
<tr>
<td>1.137273</td>
<td>2.095486</td>
<td>77</td>
<td>48,568175</td>
</tr>
<tr>
<td>1.0391</td>
<td>2.00499</td>
<td>73</td>
<td>47,0225</td>
</tr>
<tr>
<td>1.119005</td>
<td>2.077427</td>
<td>76</td>
<td>48,024875</td>
</tr>
<tr>
<td>1.320904</td>
<td>1.947488</td>
<td>78</td>
<td>44,9774</td>
</tr>
<tr>
<td>1.433803</td>
<td>1.761854</td>
<td>76</td>
<td>40,154925</td>
</tr>
<tr>
<td>2.391036</td>
<td>1.556584</td>
<td>95</td>
<td>35,2241</td>
</tr>
<tr>
<td>0.994164</td>
<td>1.645421</td>
<td>63</td>
<td>38,1459</td>
</tr>
<tr>
<td>8.5021</td>
<td>1.679563</td>
<td>243</td>
<td>30,4475</td>
</tr>
<tr>
<td>6.270694</td>
<td>1.391201</td>
<td>184</td>
<td>27,23265</td>
</tr>
<tr>
<td>1.056863</td>
<td>2.087385</td>
<td>75</td>
<td>48,578425</td>
</tr>
<tr>
<td>1.192999</td>
<td>1.951316</td>
<td>75</td>
<td>45,175025</td>
</tr>
<tr>
<td>1.153023</td>
<td>2.016573</td>
<td>75</td>
<td>46,174425</td>
</tr>
<tr>
<td>1.078848</td>
<td>1.746202</td>
<td>67</td>
<td>40,0288</td>
</tr>
<tr>
<td>4.398683</td>
<td>1.841034</td>
<td>148</td>
<td>38,032925</td>
</tr>
<tr>
<td>1.510875</td>
<td>1.835852</td>
<td>80</td>
<td>42,228125</td>
</tr>
<tr>
<td>1.246799</td>
<td>1.484546</td>
<td>65</td>
<td>33,830025</td>
</tr>
<tr>
<td>1.249169</td>
<td>1.675299</td>
<td>70</td>
<td>38,770775</td>
</tr>
<tr>
<td>1.468078</td>
<td>1.405909</td>
<td>69</td>
<td>32,9805</td>
</tr>
<tr>
<td>1.167929</td>
<td>1.879989</td>
<td>72</td>
<td>42,801775</td>
</tr>
<tr>
<td>9.098607</td>
<td>2.144063</td>
<td>269</td>
<td>41,534825</td>
</tr>
<tr>
<td>1.066464</td>
<td>2.113733</td>
<td>75</td>
<td>48,3384</td>
</tr>
<tr>
<td>8.027866</td>
<td>1.910307</td>
<td>235</td>
<td>34,30335</td>
</tr>
<tr>
<td>1.132071</td>
<td>2.02486</td>
<td>76</td>
<td>47,698225</td>
</tr>
<tr>
<td>1.388017</td>
<td>1.969856</td>
<td>80</td>
<td>45,299575</td>
</tr>
<tr>
<td>1.208687</td>
<td>1.79126</td>
<td>71</td>
<td>40,782825</td>
</tr>
<tr>
<td>1.130581</td>
<td>1.230611</td>
<td>56</td>
<td>27,735475</td>
</tr>
<tr>
<td>1.392752</td>
<td>1.956661</td>
<td>80</td>
<td>45,1812</td>
</tr>
<tr>
<td>1.309038</td>
<td>2.081047</td>
<td>81</td>
<td>48,27405</td>
</tr>
<tr>
<td>1.425878</td>
<td>1.891531</td>
<td>78</td>
<td>42,35305</td>
</tr>
<tr>
<td>1.459546</td>
<td>1.745464</td>
<td>78</td>
<td>41,51135</td>
</tr>
<tr>
<td>1.04521</td>
<td>2.10764</td>
<td>75</td>
<td>48,86975</td>
</tr>
<tr>
<td>1.402169</td>
<td>1.637368</td>
<td>73</td>
<td>37,945775</td>
</tr>
<tr>
<td>1.514409</td>
<td>1.394155</td>
<td>70</td>
<td>32,139775</td>
</tr>
<tr>
<td>1.452713</td>
<td>1.810065</td>
<td>78</td>
<td>41,682175</td>
</tr>
<tr>
<td>1.063599</td>
<td>2.097998</td>
<td>76</td>
<td>49,410025</td>
</tr>
</tbody>
</table>
F.2 SIP only

After that, we have tested the SIP performance. It has already been done in IPv4 only, using wired network and changing the IP address using the command `ifconfig`. This test corresponds to a soft handover, and we have redone it using the wireless connection.

After the handover, the UAC registers again and reinvokes all its opened sessions. The data packets are stopped in the translator, and allowed again after the end of the renegotiation. The correspondent node stops sending data packets after receiving the INVITE request for the renegotiation.

<table>
<thead>
<tr>
<th>time to get IP address</th>
<th>arrival time of the 1st data packet</th>
<th>number of lost packets</th>
<th>estimated lost packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>7,405098</td>
<td>1,80196</td>
<td>218</td>
<td>32,87255</td>
</tr>
<tr>
<td>7,208837</td>
<td>2,022378</td>
<td>222</td>
<td>41,779075</td>
</tr>
<tr>
<td>1,29516</td>
<td>1,895822</td>
<td>77</td>
<td>44,621</td>
</tr>
<tr>
<td>1,016729</td>
<td>1,611915</td>
<td>63</td>
<td>37,581775</td>
</tr>
<tr>
<td>1,402174</td>
<td>2,04718</td>
<td>83</td>
<td>47,94565</td>
</tr>
</tbody>
</table>

| Average                |                                     |                        |                        |
| 2,835292029            | 1,820826694                         | 112,3779904            | 41,49568971            |

| Ecart Type             |                                     |                        |                        |
| 2,587875089            | 0,230184339                         | 62,28945841            | 6,188296112            |

| Variance               |                                     |                        |                        |
| 6,697097475            | 0,05298483                          | 3879,976629            | 38,29500877            |

| Confidence Interval 95%|                                     |                        |                        |
| min                    | 1,794714313                         | 40,79368215            |                        |
| max                    | 1,846939075                         | 42,19769728            |                        |

<table>
<thead>
<tr>
<th>time to get IP address</th>
<th>arrival time of the 1st data packet</th>
<th>number of lost packets</th>
<th>estimated lost packets</th>
<th>REGISTER duration</th>
<th>INVITE duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>6,497647</td>
<td>2,412345</td>
<td>210</td>
<td>47,558825</td>
<td>6,497647</td>
<td>0,105157</td>
</tr>
<tr>
<td>1,34509</td>
<td>1,702655</td>
<td>69</td>
<td>35,37275</td>
<td>0,105375</td>
<td>0,387809</td>
</tr>
<tr>
<td>0,642171</td>
<td>2,407004</td>
<td>70</td>
<td>53,945725</td>
<td>0,101397</td>
<td>0,28277932</td>
</tr>
<tr>
<td>1,333684</td>
<td>2,272246</td>
<td>82</td>
<td>48,6579</td>
<td>0,188213</td>
<td>0,31670995</td>
</tr>
<tr>
<td>1,059606</td>
<td>2,158687</td>
<td>74</td>
<td>47,50985</td>
<td>0,098843</td>
<td>0,368287</td>
</tr>
<tr>
<td>1,120779</td>
<td>1,853652</td>
<td>69</td>
<td>40,980525</td>
<td>0,105242</td>
<td>0,291635</td>
</tr>
<tr>
<td>1,177654</td>
<td>2,34639</td>
<td>81</td>
<td>51,55865</td>
<td>0,108399</td>
<td>0,31862998</td>
</tr>
<tr>
<td>4,156875</td>
<td>1,619099</td>
<td>136</td>
<td>32,078125</td>
<td>0,107724</td>
<td>0,356459856</td>
</tr>
<tr>
<td>0,874132</td>
<td>2,442586</td>
<td>76</td>
<td>54,1467</td>
<td>0,106202</td>
<td>0,326776</td>
</tr>
<tr>
<td>7,663457</td>
<td>1,721419</td>
<td>222</td>
<td>30,413575</td>
<td>0,104209</td>
<td>0,299753</td>
</tr>
<tr>
<td>1,480139</td>
<td>2,514239</td>
<td>92</td>
<td>54,996525</td>
<td>0,109054</td>
<td>0,364551</td>
</tr>
<tr>
<td>time to get IP address</td>
<td>arrival time of the 1st data packet</td>
<td>number of lost packets</td>
<td>estimated lost packets</td>
<td>REGISTER duration</td>
<td>INVITE duration</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------------------------------</td>
<td>-----------------------</td>
<td>-----------------------</td>
<td>------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>8.884264</td>
<td>1.828912</td>
<td>252</td>
<td>29.8934</td>
<td>0.103393</td>
<td>0.313040018</td>
</tr>
<tr>
<td>1.150474</td>
<td>1.620302</td>
<td>63</td>
<td>34.23815</td>
<td>0.109417</td>
<td>0.270164</td>
</tr>
<tr>
<td>7.961001</td>
<td>1.835289</td>
<td>231</td>
<td>31.974975</td>
<td>0.103431</td>
<td>0.346347</td>
</tr>
<tr>
<td>1.430499</td>
<td>2.014941</td>
<td>78</td>
<td>42.237525</td>
<td>0.105811</td>
<td>0.360441</td>
</tr>
<tr>
<td>1.061409</td>
<td>1.628646</td>
<td>62</td>
<td>35.464775</td>
<td>0.099402</td>
<td>0.352476</td>
</tr>
<tr>
<td>1.368376</td>
<td>2.287972</td>
<td>85</td>
<td>50.7906</td>
<td>0.110039</td>
<td>0.347714</td>
</tr>
<tr>
<td>1.022812</td>
<td>1.638732</td>
<td>61</td>
<td>35.4297</td>
<td>0.109217</td>
<td>0.328342</td>
</tr>
<tr>
<td>2.804147</td>
<td>1.595768</td>
<td>101</td>
<td>30.896325</td>
<td>0.109049</td>
<td>0.289181</td>
</tr>
<tr>
<td>0.773262</td>
<td>1.632844</td>
<td>54</td>
<td>34.68645</td>
<td>0.104205</td>
<td>0.285738</td>
</tr>
<tr>
<td>1.120415</td>
<td>1.791713</td>
<td>67</td>
<td>38.989625</td>
<td>0.101246</td>
<td>0.286284</td>
</tr>
<tr>
<td>1.254482</td>
<td>2.287183</td>
<td>82</td>
<td>50.63795</td>
<td>0.119239</td>
<td>0.296295</td>
</tr>
<tr>
<td>1.305637</td>
<td>1.594851</td>
<td>66</td>
<td>33.359075</td>
<td>0.110315</td>
<td>0.33017993</td>
</tr>
<tr>
<td>1.017871</td>
<td>1.56301</td>
<td>59</td>
<td>33.553225</td>
<td>0.112020</td>
<td>0.27815</td>
</tr>
<tr>
<td>1.121269</td>
<td>2.071535</td>
<td>73</td>
<td>44.968275</td>
<td>0.102394</td>
<td>0.345165</td>
</tr>
<tr>
<td>1.03294</td>
<td>1.857162</td>
<td>67</td>
<td>41.1765</td>
<td>0.104167</td>
<td>0.360012</td>
</tr>
<tr>
<td>1.198289</td>
<td>2.403684</td>
<td>83</td>
<td>53.042775</td>
<td>0.109245</td>
<td>0.316413</td>
</tr>
<tr>
<td>4.585903</td>
<td>2.398586</td>
<td>163</td>
<td>48.352425</td>
<td>0.109385</td>
<td>0.368860006</td>
</tr>
<tr>
<td>1.058577</td>
<td>1.934998</td>
<td>68</td>
<td>41.535575</td>
<td>0.102205</td>
<td>0.418484</td>
</tr>
<tr>
<td>1.451376</td>
<td>1.935652</td>
<td>78</td>
<td>41.7156</td>
<td>0.112386</td>
<td>0.29658</td>
</tr>
<tr>
<td>4.914794</td>
<td>2.246719</td>
<td>168</td>
<td>45.13015</td>
<td>0.097404</td>
<td>0.383937</td>
</tr>
<tr>
<td>1.119814</td>
<td>2.022114</td>
<td>71</td>
<td>43.00465</td>
<td>0.112425</td>
<td>0.334346</td>
</tr>
<tr>
<td>1.211581</td>
<td>1.933949</td>
<td>72</td>
<td>41.710475</td>
<td>0.10942</td>
<td>0.40925</td>
</tr>
<tr>
<td>1.121605</td>
<td>1.569328</td>
<td>61</td>
<td>32.959875</td>
<td>0.099148</td>
<td>0.374254</td>
</tr>
<tr>
<td>1.060559</td>
<td>1.879061</td>
<td>68</td>
<td>41.486025</td>
<td>0.109401</td>
<td>0.34656</td>
</tr>
<tr>
<td>1.069101</td>
<td>1.593285</td>
<td>61</td>
<td>34.272475</td>
<td>0.102394</td>
<td>0.338913</td>
</tr>
<tr>
<td>6.90517</td>
<td>1.930611</td>
<td>207</td>
<td>34.37075</td>
<td>0.107384</td>
<td>0.412082</td>
</tr>
<tr>
<td>0.748466</td>
<td>1.48154</td>
<td>51</td>
<td>32.28835</td>
<td>0.112225</td>
<td>0.30434</td>
</tr>
<tr>
<td>1.000319</td>
<td>1.903769</td>
<td>65</td>
<td>39.992025</td>
<td>0.101213</td>
<td>0.387782</td>
</tr>
<tr>
<td>1.247436</td>
<td>2.061653</td>
<td>77</td>
<td>45.8141</td>
<td>0.107047</td>
<td>0.327623</td>
</tr>
<tr>
<td>4.168289</td>
<td>1.804053</td>
<td>138</td>
<td>33.792775</td>
<td>0.10046</td>
<td>0.332164</td>
</tr>
<tr>
<td>1.451342</td>
<td>2.094401</td>
<td>80</td>
<td>43.71645</td>
<td>0.111213</td>
<td>0.372231</td>
</tr>
<tr>
<td>1.138093</td>
<td>2.247045</td>
<td>77</td>
<td>48.547675</td>
<td>0.107218</td>
<td>0.34159</td>
</tr>
<tr>
<td>1.442349</td>
<td>2.118703</td>
<td>81</td>
<td>44.941275</td>
<td>0.212902</td>
<td>0.296823</td>
</tr>
<tr>
<td>1.086428</td>
<td>2.374103</td>
<td>76</td>
<td>48.8393</td>
<td>0.144365</td>
<td>0.425115</td>
</tr>
<tr>
<td>1.195668</td>
<td>2.319635</td>
<td>79</td>
<td>49.1083</td>
<td>0.105888</td>
<td>0.310028</td>
</tr>
<tr>
<td>8.724267</td>
<td>2.148448</td>
<td>256</td>
<td>37.893325</td>
<td>0.108233</td>
<td>0.31652</td>
</tr>
<tr>
<td>8.790374</td>
<td>2.215569</td>
<td>261</td>
<td>41.24065</td>
<td>0.105127</td>
<td>0.318091</td>
</tr>
<tr>
<td>1.194733</td>
<td>2.445095</td>
<td>84</td>
<td>54.131675</td>
<td>0.107604</td>
<td>0.377127</td>
</tr>
<tr>
<td>6.645778</td>
<td>1.983278</td>
<td>204</td>
<td>37.85555</td>
<td>0.1034</td>
<td>0.330579</td>
</tr>
<tr>
<td>1.221259</td>
<td>1.751204</td>
<td>67</td>
<td>36.468525</td>
<td>0.100222</td>
<td>0.287359</td>
</tr>
<tr>
<td>time to get IP address</td>
<td>arrival time of the 1st data packet</td>
<td>number of lost packets</td>
<td>estimated lost packets</td>
<td>REGISTER duration</td>
<td>INVITE duration</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------------</td>
<td>------------------------</td>
<td>------------------------</td>
<td>------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>1.197054</td>
<td>1.982179</td>
<td>74</td>
<td>44.07365</td>
<td>0.104231</td>
<td>0.314221</td>
</tr>
<tr>
<td>3.112229</td>
<td>2.125289</td>
<td>122</td>
<td>44.194275</td>
<td>0.099222</td>
<td>0.392514</td>
</tr>
<tr>
<td>0.998475</td>
<td>1.97795</td>
<td>68</td>
<td>43.038125</td>
<td>0.113422</td>
<td>0.345173</td>
</tr>
<tr>
<td>1.031063</td>
<td>1.934852</td>
<td>68</td>
<td>42.224242</td>
<td>0.108387</td>
<td>0.293203</td>
</tr>
<tr>
<td>0.562656</td>
<td>1.67212</td>
<td>51</td>
<td>36.9336</td>
<td>0.103246</td>
<td>0.332261</td>
</tr>
<tr>
<td>1.066694</td>
<td>1.495869</td>
<td>58</td>
<td>31.3326</td>
<td>0.105313</td>
<td>0.361989975</td>
</tr>
<tr>
<td>1.020795</td>
<td>1.649427</td>
<td>61</td>
<td>35.480125</td>
<td>0.107368</td>
<td>0.340313</td>
</tr>
<tr>
<td>1.435462</td>
<td>1.594558</td>
<td>69</td>
<td>33.1134</td>
<td>0.108407</td>
<td>0.340333</td>
</tr>
<tr>
<td>1.121587</td>
<td>1.79225</td>
<td>66</td>
<td>37.960325</td>
<td>0.103866</td>
<td>0.317342</td>
</tr>
<tr>
<td>1.081641</td>
<td>1.729104</td>
<td>64</td>
<td>36.958975</td>
<td>0.110455</td>
<td>0.363505</td>
</tr>
<tr>
<td>3.795157</td>
<td>1.551958</td>
<td>122</td>
<td>27.121075</td>
<td>0.103208</td>
<td>0.350379944</td>
</tr>
<tr>
<td>6.968801</td>
<td>1.805171</td>
<td>206</td>
<td>31.779975</td>
<td>0.109215</td>
<td>0.319415</td>
</tr>
<tr>
<td>1.192956</td>
<td>1.816013</td>
<td>69</td>
<td>39.1761</td>
<td>0.106227</td>
<td>0.362944</td>
</tr>
<tr>
<td>7.505475</td>
<td>1.636582</td>
<td>215</td>
<td>27.363125</td>
<td>0.220946</td>
<td>0.270413</td>
</tr>
<tr>
<td>1.368546</td>
<td>2.459867</td>
<td>88</td>
<td>53.78635</td>
<td>0.108094</td>
<td>0.379956</td>
</tr>
<tr>
<td>1.324797</td>
<td>2.084777</td>
<td>79</td>
<td>45.880075</td>
<td>0.099218</td>
<td>0.341292</td>
</tr>
<tr>
<td>1.28088</td>
<td>1.585039</td>
<td>66</td>
<td>33.978</td>
<td>0.114134</td>
<td>0.336414</td>
</tr>
<tr>
<td>1.059615</td>
<td>2.282245</td>
<td>76</td>
<td>49.509625</td>
<td>0.101208</td>
<td>0.410414</td>
</tr>
<tr>
<td>1.142466</td>
<td>2.137752</td>
<td>74</td>
<td>45.43835</td>
<td>0.110213</td>
<td>0.337754</td>
</tr>
<tr>
<td>1.123398</td>
<td>2.098117</td>
<td>74</td>
<td>45.91505</td>
<td>0.099401</td>
<td>0.326864</td>
</tr>
<tr>
<td>1.183792</td>
<td>1.691454</td>
<td>65</td>
<td>35.4052</td>
<td>0.102216</td>
<td>0.292758</td>
</tr>
<tr>
<td>1.115914</td>
<td>1.77041</td>
<td>65</td>
<td>37.10215</td>
<td>0.12018</td>
<td>0.326135</td>
</tr>
<tr>
<td>1.137881</td>
<td>1.65117</td>
<td>63</td>
<td>34.552975</td>
<td>0.108029</td>
<td>0.371086</td>
</tr>
<tr>
<td>3.733229</td>
<td>2.135415</td>
<td>137</td>
<td>43.669275</td>
<td>0.10421</td>
<td>0.305592</td>
</tr>
<tr>
<td>1.257367</td>
<td>2.503804</td>
<td>88</td>
<td>56.565825</td>
<td>0.113202</td>
<td>0.336838</td>
</tr>
<tr>
<td>1.441263</td>
<td>2.022518</td>
<td>80</td>
<td>43.968425</td>
<td>0.0992</td>
<td>0.330284</td>
</tr>
<tr>
<td>9.19184</td>
<td>1.494403</td>
<td>252</td>
<td>22.204</td>
<td>0.111403</td>
<td>0.374019</td>
</tr>
<tr>
<td>1.307519</td>
<td>1.59323</td>
<td>65</td>
<td>32.312025</td>
<td>0.175976</td>
<td>0.334168</td>
</tr>
<tr>
<td>1.348539</td>
<td>2.155873</td>
<td>81</td>
<td>47.286525</td>
<td>0.100421</td>
<td>0.286590099</td>
</tr>
<tr>
<td>1.504851</td>
<td>2.161888</td>
<td>84</td>
<td>46.378725</td>
<td>0.103204</td>
<td>0.278344</td>
</tr>
<tr>
<td>6.933796</td>
<td>1.913415</td>
<td>208</td>
<td>34.6551</td>
<td>0.118132</td>
<td>0.345866</td>
</tr>
<tr>
<td>1.432053</td>
<td>2.3308</td>
<td>87</td>
<td>51.198675</td>
<td>0.104665</td>
<td>0.366947</td>
</tr>
<tr>
<td>4.48337</td>
<td>2.513403</td>
<td>163</td>
<td>50.91575</td>
<td>0.113104</td>
<td>0.392917</td>
</tr>
<tr>
<td>0.821531</td>
<td>2.084582</td>
<td>67</td>
<td>46.461725</td>
<td>0.103398</td>
<td>0.37869</td>
</tr>
<tr>
<td>1.009447</td>
<td>1.96154</td>
<td>68</td>
<td>42.763825</td>
<td>0.110424</td>
<td>0.323713</td>
</tr>
<tr>
<td>3.911348</td>
<td>2.353705</td>
<td>146</td>
<td>48.2163</td>
<td>0.099212</td>
<td>0.350397</td>
</tr>
<tr>
<td>7.703435</td>
<td>2.398491</td>
<td>239</td>
<td>46.414125</td>
<td>0.103217</td>
<td>0.280809879</td>
</tr>
<tr>
<td>1.316708</td>
<td>2.020122</td>
<td>77</td>
<td>44.0823</td>
<td>0.103777</td>
<td>0.344028</td>
</tr>
<tr>
<td>1.377447</td>
<td>1.770665</td>
<td>71</td>
<td>36.563825</td>
<td>0.107021</td>
<td>0.330515</td>
</tr>
<tr>
<td>1.194509</td>
<td>1.8431</td>
<td>70</td>
<td>40.137275</td>
<td>0.145511</td>
<td>0.383589983</td>
</tr>
</tbody>
</table>
### F.3 MIP+SIP without renegociation

<table>
<thead>
<tr>
<th></th>
<th>time to get IP address</th>
<th>arrival time of the 1st data packet</th>
<th>number of lost packets</th>
<th>estimated lost packets</th>
<th>REGISTER duration</th>
<th>INVITE duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>7,511633</td>
<td>1,9411</td>
<td>222</td>
<td>34,209175</td>
<td>0,111233</td>
<td>0,282189846</td>
<td></td>
</tr>
<tr>
<td>0,377656</td>
<td>1,740978</td>
<td>48</td>
<td>38,5586</td>
<td>0,106206</td>
<td>0,31705</td>
<td></td>
</tr>
<tr>
<td>5,114037</td>
<td>2,213065</td>
<td>173</td>
<td>45,149075</td>
<td>0,110233</td>
<td>0,338639</td>
<td></td>
</tr>
<tr>
<td>2,444254</td>
<td>2,41107</td>
<td>112</td>
<td>50,89365</td>
<td>0,174354</td>
<td>0,30018</td>
<td></td>
</tr>
<tr>
<td>1,086514</td>
<td>1,742671</td>
<td>65</td>
<td>37,83715</td>
<td>0,103419</td>
<td>0,337075</td>
<td></td>
</tr>
<tr>
<td>1,092611</td>
<td>1,936947</td>
<td>69</td>
<td>41,684725</td>
<td>0,10321</td>
<td>0,376202</td>
<td></td>
</tr>
<tr>
<td>5,793798</td>
<td>2,318295</td>
<td>192</td>
<td>47,15505</td>
<td>0,107216</td>
<td>0,306147</td>
<td></td>
</tr>
<tr>
<td>1,133402</td>
<td>1,869698</td>
<td>67</td>
<td>38,66495</td>
<td>0,103188</td>
<td>0,35288</td>
<td></td>
</tr>
<tr>
<td>5,038233</td>
<td>1,673264</td>
<td>158</td>
<td>32,044175</td>
<td>0,112195</td>
<td>0,342584</td>
<td></td>
</tr>
<tr>
<td>1,464638</td>
<td>2,018065</td>
<td>79</td>
<td>42,38405</td>
<td>0,10021</td>
<td>0,32638</td>
<td></td>
</tr>
<tr>
<td>0,316026</td>
<td>2,145395</td>
<td>55</td>
<td>47,09935</td>
<td>0,11919</td>
<td>0,370874</td>
<td></td>
</tr>
<tr>
<td>1,145417</td>
<td>2,053638</td>
<td>74</td>
<td>45,364575</td>
<td>0,115104</td>
<td>0,376984</td>
<td></td>
</tr>
<tr>
<td>0,640662</td>
<td>1,796204</td>
<td>55</td>
<td>38,98345</td>
<td>0,111055</td>
<td>0,378181</td>
<td></td>
</tr>
<tr>
<td>1,136257</td>
<td>2,104769</td>
<td>73</td>
<td>44,593575</td>
<td>0,104393</td>
<td>0,324651</td>
<td></td>
</tr>
<tr>
<td>1,12174</td>
<td>1,65162</td>
<td>63</td>
<td>34,9565</td>
<td>0,104239</td>
<td>0,334134</td>
<td></td>
</tr>
<tr>
<td>1,060607</td>
<td>1,657228</td>
<td>61</td>
<td>34,484825</td>
<td>0,100423</td>
<td>0,364517</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Average</th>
<th>Ecart type</th>
<th>Variance</th>
<th>Confidence Interval 95%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2,389576944</td>
<td>2,357470338</td>
<td>5,557666394</td>
<td>5,1245461</td>
</tr>
<tr>
<td></td>
<td>1,967848402</td>
<td>0,286236301</td>
<td>0,08193122</td>
<td>0,001245461</td>
</tr>
<tr>
<td></td>
<td>100,8411215</td>
<td>57,09197306</td>
<td>3259,493387</td>
<td>0,000409738</td>
</tr>
<tr>
<td></td>
<td>41,1016979</td>
<td>7,202354697</td>
<td>51,87391318</td>
<td>0,0001245461</td>
</tr>
<tr>
<td></td>
<td>0,111292944</td>
<td>0,020241988</td>
<td>0,00409738</td>
<td>0,0001245461</td>
</tr>
<tr>
<td></td>
<td>0,338020752</td>
<td>0,035291086</td>
<td>0,001245461</td>
<td></td>
</tr>
</tbody>
</table>

### F.3 MIP+SIP without renegociation

After an handover, the UAC changes its GGSN. In this case, we consider that the GGSN gets the filtering parameters from the previous GGSN, and accept them. We are not able to simulate the way for the GGSN to get these parameters. We just wait for a random delay between 50 and 90 ms: the average for a REGISTER request is 70 ms.

After the handover, the UAC registers also to its new GGSN and P-CSCF, by sending a REGISTER requester. In our prototype the routers don’t stop the data packets. This functionnality is done in the translator, which stops all the data packets after the handover: by changing the GGSN, the data packets can’t go through it, before it gets all the filtering parameters.
### F.3 MIP+SIP without renegociation

<table>
<thead>
<tr>
<th>time to get IP address</th>
<th>arrival time of the 1st data packet</th>
<th>number of lost packets</th>
<th>estimated lost packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.242236</td>
<td>2.05555</td>
<td>80</td>
<td>48.9441</td>
</tr>
<tr>
<td>1.494916</td>
<td>1.952963</td>
<td>80</td>
<td>42.6271</td>
</tr>
<tr>
<td>7.149835</td>
<td>2.281837</td>
<td>225</td>
<td>46.254125</td>
</tr>
<tr>
<td>9.438764</td>
<td>2.043885</td>
<td>271</td>
<td>35.0309</td>
</tr>
<tr>
<td>1.413072</td>
<td>1.930516</td>
<td>79</td>
<td>43.6732</td>
</tr>
<tr>
<td>1.386021</td>
<td>2.23704</td>
<td>87</td>
<td>52.349475</td>
</tr>
<tr>
<td>1.528638</td>
<td>2.291875</td>
<td>91</td>
<td>52.78405</td>
</tr>
<tr>
<td>1.330712</td>
<td>2.527873</td>
<td>92</td>
<td>58.7322</td>
</tr>
<tr>
<td>1.337709</td>
<td>2.155827</td>
<td>83</td>
<td>49.55725</td>
</tr>
<tr>
<td>1.353014</td>
<td>1.937398</td>
<td>78</td>
<td>44.17465</td>
</tr>
<tr>
<td>4.083152</td>
<td>2.234759</td>
<td>150</td>
<td>47.9212</td>
</tr>
<tr>
<td>1.10854</td>
<td>1.882297</td>
<td>71</td>
<td>43.2865</td>
</tr>
<tr>
<td>1.438771</td>
<td>2.211049</td>
<td>87</td>
<td>51.030725</td>
</tr>
<tr>
<td>1.407976</td>
<td>2.039425</td>
<td>83</td>
<td>47.8006</td>
</tr>
<tr>
<td>1.0875</td>
<td>1.796088</td>
<td>69</td>
<td>41.8125</td>
</tr>
<tr>
<td>1.086724</td>
<td>1.793692</td>
<td>69</td>
<td>41.8319</td>
</tr>
<tr>
<td>1.297548</td>
<td>1.663628</td>
<td>71</td>
<td>38.5613</td>
</tr>
<tr>
<td>7.714174</td>
<td>1.816095</td>
<td>228</td>
<td>35.1565</td>
</tr>
<tr>
<td>1.3261</td>
<td>2.043214</td>
<td>81</td>
<td>47.8475</td>
</tr>
<tr>
<td>1.437813</td>
<td>2.217689</td>
<td>87</td>
<td>51.054675</td>
</tr>
<tr>
<td>1.525127</td>
<td>1.902547</td>
<td>82</td>
<td>43.871825</td>
</tr>
<tr>
<td>0.998599</td>
<td>2.17947</td>
<td>76</td>
<td>51.035025</td>
</tr>
<tr>
<td>1.155651</td>
<td>1.588171</td>
<td>66</td>
<td>37.108725</td>
</tr>
<tr>
<td>6.141119</td>
<td>2.640824</td>
<td>210</td>
<td>56.472025</td>
</tr>
<tr>
<td>8.260481</td>
<td>2.533036</td>
<td>257</td>
<td>50.487975</td>
</tr>
<tr>
<td>1.232147</td>
<td>2.301381</td>
<td>85</td>
<td>54.196325</td>
</tr>
<tr>
<td>8.380598</td>
<td>2.299716</td>
<td>255</td>
<td>45.48505</td>
</tr>
<tr>
<td>5.273052</td>
<td>2.288352</td>
<td>181</td>
<td>49.1737</td>
</tr>
<tr>
<td>1.269569</td>
<td>2.504962</td>
<td>89</td>
<td>57.260775</td>
</tr>
<tr>
<td>1.509638</td>
<td>2.219736</td>
<td>90</td>
<td>52.25905</td>
</tr>
<tr>
<td>1.16207</td>
<td>2.322006</td>
<td>84</td>
<td>54.94825</td>
</tr>
<tr>
<td>3.793369</td>
<td>2.254347</td>
<td>144</td>
<td>49.165775</td>
</tr>
<tr>
<td>1.03626</td>
<td>2.437043</td>
<td>83</td>
<td>57.0935</td>
</tr>
<tr>
<td>1.315879</td>
<td>2.212752</td>
<td>85</td>
<td>52.103025</td>
</tr>
<tr>
<td>1.441205</td>
<td>2.153682</td>
<td>86</td>
<td>49.969875</td>
</tr>
<tr>
<td>1.069077</td>
<td>2.296356</td>
<td>82</td>
<td>55.273075</td>
</tr>
<tr>
<td>1.331954</td>
<td>2.606943</td>
<td>95</td>
<td>61.70115</td>
</tr>
<tr>
<td>1.486043</td>
<td>2.222836</td>
<td>89</td>
<td>51.848925</td>
</tr>
<tr>
<td>1.427</td>
<td>2.281324</td>
<td>89</td>
<td>53.325</td>
</tr>
<tr>
<td>1.497114</td>
<td>2.195123</td>
<td>88</td>
<td>50.57215</td>
</tr>
<tr>
<td>1.504143</td>
<td>1.920859</td>
<td>83</td>
<td>45.396425</td>
</tr>
</tbody>
</table>
### F.3 MIP+SIP without renegotiation

<table>
<thead>
<tr>
<th>time to get IP address</th>
<th>arrival time of the 1st data packet</th>
<th>number of lost packets</th>
<th>estimated lost packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,382114</td>
<td>2.177439</td>
<td>85</td>
<td>50,44715</td>
</tr>
<tr>
<td>1,188279</td>
<td>2.585975</td>
<td>91</td>
<td>61,293025</td>
</tr>
<tr>
<td>1,193597</td>
<td>2.490688</td>
<td>89</td>
<td>59,160075</td>
</tr>
<tr>
<td>1,114918</td>
<td>2.317489</td>
<td>83</td>
<td>55,12705</td>
</tr>
<tr>
<td>1,053135</td>
<td>2.296828</td>
<td>79</td>
<td>52,671625</td>
</tr>
<tr>
<td>1,291301</td>
<td>2.44433</td>
<td>89</td>
<td>56,717475</td>
</tr>
<tr>
<td>1,271407</td>
<td>2.220357</td>
<td>84</td>
<td>52,214825</td>
</tr>
<tr>
<td>1,134766</td>
<td>2.181346</td>
<td>80</td>
<td>51,63085</td>
</tr>
<tr>
<td>1,223653</td>
<td>1.984869</td>
<td>76</td>
<td>45,408675</td>
</tr>
<tr>
<td>1,181833</td>
<td>2.306535</td>
<td>84</td>
<td>54,454175</td>
</tr>
<tr>
<td>1,214029</td>
<td>2.707434</td>
<td>94</td>
<td>63,649275</td>
</tr>
<tr>
<td>3,824591</td>
<td>2.270614</td>
<td>147</td>
<td>51,385225</td>
</tr>
<tr>
<td>1,445011</td>
<td>2.330666</td>
<td>90</td>
<td>53,874725</td>
</tr>
<tr>
<td>1,249884</td>
<td>1.947908</td>
<td>77</td>
<td>45,7529</td>
</tr>
<tr>
<td>1,075938</td>
<td>2.415579</td>
<td>84</td>
<td>57,10155</td>
</tr>
<tr>
<td>1,200984</td>
<td>2.675698</td>
<td>93</td>
<td>62,9754</td>
</tr>
<tr>
<td>4,797303</td>
<td>1.771262</td>
<td>159</td>
<td>39,067425</td>
</tr>
<tr>
<td>4,113599</td>
<td>2.260066</td>
<td>154</td>
<td>51,160025</td>
</tr>
<tr>
<td>1,382601</td>
<td>2.225913</td>
<td>86</td>
<td>51,434975</td>
</tr>
<tr>
<td>1,255718</td>
<td>2.134825</td>
<td>82</td>
<td>50,60705</td>
</tr>
<tr>
<td>1,265434</td>
<td>2.24857</td>
<td>84</td>
<td>52,36415</td>
</tr>
<tr>
<td>1,08397</td>
<td>1.903693</td>
<td>71</td>
<td>43,90075</td>
</tr>
<tr>
<td>6,228785</td>
<td>2.625112</td>
<td>212</td>
<td>56,280375</td>
</tr>
<tr>
<td>1,325031</td>
<td>2.211862</td>
<td>84</td>
<td>50,874225</td>
</tr>
<tr>
<td>1,51664</td>
<td>2.260675</td>
<td>90</td>
<td>52,084</td>
</tr>
<tr>
<td>1,10906</td>
<td>2.02373</td>
<td>76</td>
<td>48,2735</td>
</tr>
<tr>
<td>1,355658</td>
<td>2.094549</td>
<td>82</td>
<td>48,10855</td>
</tr>
<tr>
<td>1,470784</td>
<td>2.323663</td>
<td>91</td>
<td>54,2304</td>
</tr>
<tr>
<td>8,941783</td>
<td>2.138653</td>
<td>268</td>
<td>44,455425</td>
</tr>
<tr>
<td>1,13232</td>
<td>1.736822</td>
<td>69</td>
<td>40,692</td>
</tr>
<tr>
<td>2,713252</td>
<td>2.24255</td>
<td>119</td>
<td>51,1687</td>
</tr>
<tr>
<td>1,284781</td>
<td>2.435588</td>
<td>90</td>
<td>57,880475</td>
</tr>
<tr>
<td>1,185135</td>
<td>1.801856</td>
<td>72</td>
<td>42,371625</td>
</tr>
<tr>
<td>6,53002</td>
<td>2.275247</td>
<td>210</td>
<td>46,7495</td>
</tr>
<tr>
<td>5,020233</td>
<td>2.351343</td>
<td>176</td>
<td>50,49175</td>
</tr>
<tr>
<td>1,114424</td>
<td>2.343139</td>
<td>83</td>
<td>55,1394</td>
</tr>
<tr>
<td>1,071983</td>
<td>1.90224</td>
<td>72</td>
<td>45,200425</td>
</tr>
<tr>
<td>4,542689</td>
<td>2.534648</td>
<td>169</td>
<td>55,432775</td>
</tr>
<tr>
<td>1,509989</td>
<td>2.637259</td>
<td>100</td>
<td>62,250275</td>
</tr>
<tr>
<td>2,479421</td>
<td>2.271025</td>
<td>114</td>
<td>52,014475</td>
</tr>
<tr>
<td>1,09997</td>
<td>1.970893</td>
<td>74</td>
<td>46,50075</td>
</tr>
</tbody>
</table>
### F.3 MIP+SIP without renegotiation

<table>
<thead>
<tr>
<th>time to get IP address</th>
<th>arrival time of the 1st data packet</th>
<th>number of lost packets</th>
<th>estimated lost packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.079477</td>
<td>2.065233</td>
<td>76</td>
<td>49.013075</td>
</tr>
<tr>
<td>1.061569</td>
<td>2.222963</td>
<td>79</td>
<td>52.460775</td>
</tr>
<tr>
<td>1.45789</td>
<td>2.30562</td>
<td>90</td>
<td>53.55275</td>
</tr>
<tr>
<td>5.571186</td>
<td>1.90648</td>
<td>179</td>
<td>39.72035</td>
</tr>
<tr>
<td>4.423207</td>
<td>2.59019</td>
<td>169</td>
<td>58.419825</td>
</tr>
<tr>
<td>1.262209</td>
<td>2.190275</td>
<td>83</td>
<td>51.444775</td>
</tr>
<tr>
<td>6.099459</td>
<td>2.429278</td>
<td>204</td>
<td>51.738525</td>
</tr>
<tr>
<td>3.235803</td>
<td>2.015358</td>
<td>126</td>
<td>45.104925</td>
</tr>
<tr>
<td>1.021317</td>
<td>2.615691</td>
<td>88</td>
<td>62.467075</td>
</tr>
<tr>
<td>1.207817</td>
<td>2.477452</td>
<td>88</td>
<td>57.804575</td>
</tr>
<tr>
<td>1.49968</td>
<td>2.318405</td>
<td>91</td>
<td>53.508</td>
</tr>
<tr>
<td>1.244359</td>
<td>1.730667</td>
<td>71</td>
<td>39.891025</td>
</tr>
<tr>
<td>8.788928</td>
<td>2.310031</td>
<td>265</td>
<td>45.2768</td>
</tr>
<tr>
<td>1.111875</td>
<td>2.315001</td>
<td>83</td>
<td>55.203125</td>
</tr>
<tr>
<td>1.24668</td>
<td>2.340247</td>
<td>86</td>
<td>54.833</td>
</tr>
<tr>
<td>1.260959</td>
<td>2.352587</td>
<td>87</td>
<td>55.476025</td>
</tr>
<tr>
<td>1.243238</td>
<td>2.710827</td>
<td>94</td>
<td>62.91905</td>
</tr>
<tr>
<td>7.46874</td>
<td>2.123262</td>
<td>231</td>
<td>44.2815</td>
</tr>
<tr>
<td>4.972672</td>
<td>1.819697</td>
<td>163</td>
<td>38.6832</td>
</tr>
<tr>
<td>1.127535</td>
<td>2.220657</td>
<td>81</td>
<td>52.811625</td>
</tr>
<tr>
<td>1.195362</td>
<td>2.194446</td>
<td>81</td>
<td>51.11595</td>
</tr>
<tr>
<td>2.942472</td>
<td>2.16865</td>
<td>122</td>
<td>48.4382</td>
</tr>
<tr>
<td>1.203672</td>
<td>2.21041</td>
<td>81</td>
<td>50.9082</td>
</tr>
<tr>
<td>1.284212</td>
<td>2.185117</td>
<td>83</td>
<td>50.8947</td>
</tr>
<tr>
<td>1.485568</td>
<td>2.173887</td>
<td>88</td>
<td>50.8608</td>
</tr>
<tr>
<td>1.038919</td>
<td>1.73852</td>
<td>67</td>
<td>41.027025</td>
</tr>
<tr>
<td>1.294891</td>
<td>2.487579</td>
<td>91</td>
<td>58.627725</td>
</tr>
<tr>
<td>4.259597</td>
<td>1.687046</td>
<td>143</td>
<td>36.510075</td>
</tr>
<tr>
<td>1.219</td>
<td>1.920719</td>
<td>76</td>
<td>45.525</td>
</tr>
<tr>
<td>1.084595</td>
<td>2.526907</td>
<td>86</td>
<td>58.885125</td>
</tr>
<tr>
<td>1.437927</td>
<td>2.249147</td>
<td>88</td>
<td>52.051825</td>
</tr>
<tr>
<td>8.388721</td>
<td>2.427345</td>
<td>258</td>
<td>48.281975</td>
</tr>
<tr>
<td>1.502975</td>
<td>2.46945</td>
<td>96</td>
<td>58.425625</td>
</tr>
<tr>
<td>1.454501</td>
<td>2.365218</td>
<td>91</td>
<td>54.637475</td>
</tr>
<tr>
<td>6.674248</td>
<td>2.100501</td>
<td>209</td>
<td>42.1438</td>
</tr>
<tr>
<td>1.18098</td>
<td>2.4344</td>
<td>87</td>
<td>57.4755</td>
</tr>
<tr>
<td>4.879715</td>
<td>1.662435</td>
<td>157</td>
<td>35.007125</td>
</tr>
<tr>
<td>1.141218</td>
<td>1.747463</td>
<td>69</td>
<td>40.46955</td>
</tr>
<tr>
<td>9.266299</td>
<td>2.224718</td>
<td>274</td>
<td>42.342525</td>
</tr>
<tr>
<td>4.781889</td>
<td>2.213664</td>
<td>168</td>
<td>48.452775</td>
</tr>
<tr>
<td>5.889524</td>
<td>2.152695</td>
<td>192</td>
<td>44.7619</td>
</tr>
</tbody>
</table>
### F.3 MIP+SIP without renegotiation

<table>
<thead>
<tr>
<th>time to get IP address</th>
<th>arrival time of the 1st data packet</th>
<th>number of lost packets</th>
<th>estimated lost packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.52533</td>
<td>2.081003</td>
<td>84</td>
<td>45.86675</td>
</tr>
<tr>
<td>1.378554</td>
<td>1.980351</td>
<td>81</td>
<td>46.53615</td>
</tr>
<tr>
<td>1.214084</td>
<td>2.603718</td>
<td>91</td>
<td>60.6479</td>
</tr>
<tr>
<td>3.281548</td>
<td>2.355247</td>
<td>136</td>
<td>53.9613</td>
</tr>
<tr>
<td>1.213873</td>
<td>2.11919</td>
<td>80</td>
<td>49.653175</td>
</tr>
<tr>
<td>5.214347</td>
<td>2.553283</td>
<td>185</td>
<td>54.641325</td>
</tr>
<tr>
<td>1.336567</td>
<td>2.703088</td>
<td>96</td>
<td>62.585825</td>
</tr>
<tr>
<td>1.176805</td>
<td>2.706099</td>
<td>92</td>
<td>62.579875</td>
</tr>
<tr>
<td>1.050862</td>
<td>2.267438</td>
<td>79</td>
<td>52.72845</td>
</tr>
<tr>
<td>1.420689</td>
<td>2.435058</td>
<td>92</td>
<td>56.482775</td>
</tr>
<tr>
<td>1.229836</td>
<td>2.394229</td>
<td>87</td>
<td>56.2541</td>
</tr>
<tr>
<td>1.196038</td>
<td>2.414693</td>
<td>86</td>
<td>56.09905</td>
</tr>
<tr>
<td>1.287214</td>
<td>2.165109</td>
<td>81</td>
<td>48.81965</td>
</tr>
<tr>
<td>1.052363</td>
<td>2.467746</td>
<td>84</td>
<td>57.690925</td>
</tr>
<tr>
<td>1.424075</td>
<td>1.616169</td>
<td>72</td>
<td>36.398125</td>
</tr>
<tr>
<td>5.387578</td>
<td>1.915174</td>
<td>174</td>
<td>39.31055</td>
</tr>
<tr>
<td>1.249992</td>
<td>2.280766</td>
<td>84</td>
<td>52.7502</td>
</tr>
<tr>
<td>1.190973</td>
<td>2.36449</td>
<td>85</td>
<td>55.22675</td>
</tr>
<tr>
<td>1.214508</td>
<td>1.920923</td>
<td>75</td>
<td>44.6373</td>
</tr>
<tr>
<td>1.261767</td>
<td>2.656817</td>
<td>95</td>
<td>63.45825</td>
</tr>
<tr>
<td>1.163949</td>
<td>2.317233</td>
<td>83</td>
<td>53.901275</td>
</tr>
<tr>
<td>1.154367</td>
<td>2.601408</td>
<td>90</td>
<td>61.140825</td>
</tr>
<tr>
<td>1.431887</td>
<td>2.069912</td>
<td>84</td>
<td>48.202825</td>
</tr>
<tr>
<td>1.452869</td>
<td>2.0252</td>
<td>82</td>
<td>45.678275</td>
</tr>
<tr>
<td>1.16616</td>
<td>2.092833</td>
<td>80</td>
<td>50.846</td>
</tr>
<tr>
<td>1.153142</td>
<td>2.376489</td>
<td>86</td>
<td>57.17145</td>
</tr>
<tr>
<td>1.219777</td>
<td>2.189993</td>
<td>83</td>
<td>52.505575</td>
</tr>
<tr>
<td>1.251684</td>
<td>2.503233</td>
<td>90</td>
<td>58.7079</td>
</tr>
<tr>
<td>1.074878</td>
<td>1.486415</td>
<td>61</td>
<td>34.12805</td>
</tr>
<tr>
<td>1.442862</td>
<td>2.604404</td>
<td>97</td>
<td>60.92845</td>
</tr>
<tr>
<td>6.830497</td>
<td>1.937589</td>
<td>209</td>
<td>38.237575</td>
</tr>
<tr>
<td>3.237031</td>
<td>2.383548</td>
<td>135</td>
<td>54.074225</td>
</tr>
<tr>
<td>1.334606</td>
<td>1.888275</td>
<td>77</td>
<td>43.63485</td>
</tr>
<tr>
<td>1.17277</td>
<td>2.464207</td>
<td>87</td>
<td>57.68075</td>
</tr>
<tr>
<td>1.485798</td>
<td>2.64304</td>
<td>98</td>
<td>60.85505</td>
</tr>
<tr>
<td>1.26619</td>
<td>1.952799</td>
<td>75</td>
<td>43.34525</td>
</tr>
<tr>
<td>7.544213</td>
<td>1.692738</td>
<td>221</td>
<td>32.394675</td>
</tr>
<tr>
<td>1.172309</td>
<td>2.362708</td>
<td>85</td>
<td>55.692275</td>
</tr>
<tr>
<td>1.083661</td>
<td>1.950788</td>
<td>72</td>
<td>44.908475</td>
</tr>
<tr>
<td>1.313794</td>
<td>1.626726</td>
<td>70</td>
<td>37.15515</td>
</tr>
<tr>
<td>1.403954</td>
<td>1.978803</td>
<td>81</td>
<td>45.90115</td>
</tr>
</tbody>
</table>
### F.3 MIP+SIP without renegotiation

<table>
<thead>
<tr>
<th>time to get IP address</th>
<th>arrival time of the 1st data packet</th>
<th>number of lost packets</th>
<th>estimated lost packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.397889</td>
<td>2.295576</td>
<td>135</td>
<td>50.052775</td>
</tr>
<tr>
<td>1.500439</td>
<td>2.328047</td>
<td>91</td>
<td>53.489025</td>
</tr>
<tr>
<td>1.534771</td>
<td>2.219227</td>
<td>90</td>
<td>51.630725</td>
</tr>
<tr>
<td>1.130968</td>
<td>1.891847</td>
<td>72</td>
<td>43.7258</td>
</tr>
<tr>
<td>1.030434</td>
<td>2.241455</td>
<td>77</td>
<td>51.23915</td>
</tr>
<tr>
<td>1.156926</td>
<td>2.324054</td>
<td>81</td>
<td>52.07685</td>
</tr>
<tr>
<td>1.45303</td>
<td>1.928366</td>
<td>82</td>
<td>45.67425</td>
</tr>
<tr>
<td>1.522783</td>
<td>2.514123</td>
<td>97</td>
<td>58.930425</td>
</tr>
<tr>
<td>2.971031</td>
<td>2.450547</td>
<td>129</td>
<td>54.724225</td>
</tr>
<tr>
<td>6.112289</td>
<td>1.968265</td>
<td>192</td>
<td>39.192775</td>
</tr>
<tr>
<td>1.404284</td>
<td>2.418477</td>
<td>90</td>
<td>54.8929</td>
</tr>
<tr>
<td>8.350632</td>
<td>1.901492</td>
<td>244</td>
<td>35.2342</td>
</tr>
<tr>
<td>1.237672</td>
<td>2.458245</td>
<td>89</td>
<td>58.0582</td>
</tr>
<tr>
<td>8.887074</td>
<td>2.541989</td>
<td>269</td>
<td>51.82315</td>
</tr>
<tr>
<td>1.155023</td>
<td>1.845359</td>
<td>72</td>
<td>43.124425</td>
</tr>
<tr>
<td>1.305142</td>
<td>2.444184</td>
<td>89</td>
<td>56.37145</td>
</tr>
<tr>
<td>1.192855</td>
<td>2.336971</td>
<td>84</td>
<td>54.178625</td>
</tr>
<tr>
<td>0.998497</td>
<td>2.077152</td>
<td>74</td>
<td>49.037575</td>
</tr>
<tr>
<td>1.377935</td>
<td>2.601941</td>
<td>95</td>
<td>60.551625</td>
</tr>
<tr>
<td>1.064403</td>
<td>2.556121</td>
<td>85</td>
<td>58.389925</td>
</tr>
<tr>
<td>1.142692</td>
<td>1.60206</td>
<td>66</td>
<td>37.4327</td>
</tr>
<tr>
<td>6.530089</td>
<td>2.567123</td>
<td>218</td>
<td>54.747775</td>
</tr>
<tr>
<td>1.381914</td>
<td>2.52316</td>
<td>93</td>
<td>58.45215</td>
</tr>
<tr>
<td>1.432837</td>
<td>1.808662</td>
<td>77</td>
<td>41.179075</td>
</tr>
<tr>
<td>1.019905</td>
<td>2.517042</td>
<td>83</td>
<td>57.502375</td>
</tr>
<tr>
<td>1.078041</td>
<td>2.213974</td>
<td>79</td>
<td>52.048975</td>
</tr>
<tr>
<td>6.065316</td>
<td>1.99252</td>
<td>192</td>
<td>40.3671</td>
</tr>
<tr>
<td>1.497987</td>
<td>2.340274</td>
<td>93</td>
<td>55.550325</td>
</tr>
<tr>
<td>1.25145</td>
<td>2.145243</td>
<td>82</td>
<td>50.71375</td>
</tr>
<tr>
<td>1.111539</td>
<td>2.32672</td>
<td>82</td>
<td>54.211525</td>
</tr>
<tr>
<td>6.744465</td>
<td>1.625803</td>
<td>198</td>
<td>29.388375</td>
</tr>
<tr>
<td>9.12958</td>
<td>1.937348</td>
<td>265</td>
<td>36.7605</td>
</tr>
<tr>
<td>1.045981</td>
<td>2.078234</td>
<td>73</td>
<td>46.850475</td>
</tr>
<tr>
<td>1.342819</td>
<td>2.35886</td>
<td>87</td>
<td>53.429525</td>
</tr>
<tr>
<td>4.229209</td>
<td>2.238343</td>
<td>155</td>
<td>49.269775</td>
</tr>
<tr>
<td>1.014378</td>
<td>2.015144</td>
<td>72</td>
<td>46.64055</td>
</tr>
<tr>
<td>4.738554</td>
<td>2.22265</td>
<td>165</td>
<td>46.53615</td>
</tr>
<tr>
<td>1.125503</td>
<td>2.127085</td>
<td>77</td>
<td>48.862425</td>
</tr>
<tr>
<td>1.211739</td>
<td>2.477819</td>
<td>88</td>
<td>57.706525</td>
</tr>
<tr>
<td>3.578687</td>
<td>2.147804</td>
<td>134</td>
<td>44.532825</td>
</tr>
<tr>
<td>4.391434</td>
<td>2.436909</td>
<td>162</td>
<td>52.21415</td>
</tr>
</tbody>
</table>
F.4 MIP+SIP with renegociation

The last test done corresponds to the previous one. But this time, the GGSN doesn’t accept the filtering parameters of the previous GGSN. For example, when the handover is done from A WLAN network to a UMTS network, the data rate can’t be the same.

In order to achieve the renegociation, the UAC sends a UPDATE and PRACK requests. In this case, we suppose that the correspondent node accept the new SDP parameters.
### F.4 MIP+SIP with renegotiation

<table>
<thead>
<tr>
<th>time to get IP address</th>
<th>arrival time of the 1st data packet</th>
<th>number of lost packets</th>
<th>estimated lost packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.39798</td>
<td>2.406508</td>
<td>88</td>
<td>53.0505</td>
</tr>
<tr>
<td>4.875376</td>
<td>1.925458</td>
<td>163</td>
<td>41.1156</td>
</tr>
<tr>
<td>7.793241</td>
<td>2.429556</td>
<td>244</td>
<td>49.168975</td>
</tr>
<tr>
<td>4.670925</td>
<td>2.003601</td>
<td>158</td>
<td>41.226875</td>
</tr>
<tr>
<td>1.218354</td>
<td>1.979549</td>
<td>74</td>
<td>43.54115</td>
</tr>
<tr>
<td>1.169724</td>
<td>2.51937</td>
<td>86</td>
<td>56.7569</td>
</tr>
<tr>
<td>1.12091</td>
<td>2.652102</td>
<td>90</td>
<td>61.97725</td>
</tr>
<tr>
<td>1.029136</td>
<td>2.176079</td>
<td>77</td>
<td>51.2716</td>
</tr>
<tr>
<td>6.659529</td>
<td>2.340419</td>
<td>215</td>
<td>48.511775</td>
</tr>
<tr>
<td>1.441836</td>
<td>2.171026</td>
<td>85</td>
<td>48.9541</td>
</tr>
<tr>
<td>1.137998</td>
<td>2.498268</td>
<td>86</td>
<td>57.55005</td>
</tr>
<tr>
<td>1.151205</td>
<td>2.25829</td>
<td>78</td>
<td>49.219875</td>
</tr>
<tr>
<td>1.034532</td>
<td>2.437906</td>
<td>83</td>
<td>57.1367</td>
</tr>
<tr>
<td>7.84691</td>
<td>2.369921</td>
<td>244</td>
<td>47.82725</td>
</tr>
<tr>
<td>1.433869</td>
<td>2.530851</td>
<td>94</td>
<td>58.153275</td>
</tr>
<tr>
<td>1.068295</td>
<td>2.480023</td>
<td>85</td>
<td>58.292625</td>
</tr>
<tr>
<td>1.333881</td>
<td>2.942611</td>
<td>102</td>
<td>68.652975</td>
</tr>
<tr>
<td>6.647023</td>
<td>2.658858</td>
<td>218</td>
<td>51.824425</td>
</tr>
<tr>
<td>1.299309</td>
<td>2.064001</td>
<td>79</td>
<td>46.517275</td>
</tr>
<tr>
<td>4.759196</td>
<td>2.095906</td>
<td>163</td>
<td>44.0201</td>
</tr>
<tr>
<td>7.261811</td>
<td>2.061638</td>
<td>223</td>
<td>41.454725</td>
</tr>
<tr>
<td>6.417242</td>
<td>2.733273</td>
<td>218</td>
<td>57.56895</td>
</tr>
<tr>
<td>1.519492</td>
<td>2.44086</td>
<td>92</td>
<td>54.0127</td>
</tr>
<tr>
<td>1.266688</td>
<td>2.291433</td>
<td>82</td>
<td>50.3328</td>
</tr>
<tr>
<td>1.466176</td>
<td>2.56011</td>
<td>94</td>
<td>57.3456</td>
</tr>
<tr>
<td>5.712976</td>
<td>2.326075</td>
<td>186</td>
<td>43.1756</td>
</tr>
<tr>
<td>1.112858</td>
<td>2.752388</td>
<td>92</td>
<td>64.17855</td>
</tr>
<tr>
<td>1.006438</td>
<td>2.116369</td>
<td>76</td>
<td>50.83905</td>
</tr>
<tr>
<td>8.190199</td>
<td>2.791517</td>
<td>262</td>
<td>57.245025</td>
</tr>
<tr>
<td>1.085122</td>
<td>2.567982</td>
<td>86</td>
<td>58.87195</td>
</tr>
<tr>
<td>7.417187</td>
<td>2.24446</td>
<td>227</td>
<td>41.570325</td>
</tr>
<tr>
<td>6.437412</td>
<td>2.206393</td>
<td>203</td>
<td>42.0647</td>
</tr>
<tr>
<td>1.279578</td>
<td>2.220798</td>
<td>81</td>
<td>49.01055</td>
</tr>
<tr>
<td>5.707171</td>
<td>2.505939</td>
<td>197</td>
<td>54.320725</td>
</tr>
<tr>
<td>1.340715</td>
<td>2.516241</td>
<td>93</td>
<td>59.482125</td>
</tr>
<tr>
<td>1.476891</td>
<td>1.871226</td>
<td>78</td>
<td>41.077725</td>
</tr>
<tr>
<td>1.358111</td>
<td>2.743103</td>
<td>96</td>
<td>62.047225</td>
</tr>
<tr>
<td>1.162764</td>
<td>2.326872</td>
<td>81</td>
<td>51.9309</td>
</tr>
<tr>
<td>1.173128</td>
<td>2.713959</td>
<td>92</td>
<td>62.6718</td>
</tr>
<tr>
<td>1.007045</td>
<td>2.098656</td>
<td>75</td>
<td>49.823875</td>
</tr>
<tr>
<td>5.33998</td>
<td>1.774122</td>
<td>170</td>
<td>36.5005</td>
</tr>
<tr>
<td>time to get IP address</td>
<td>arrival time of the 1st data packet</td>
<td>number of lost packets</td>
<td>estimated lost packets</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------</td>
<td>------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>1.447976</td>
<td>2.195786</td>
<td>87</td>
<td>50,8006</td>
</tr>
<tr>
<td>4.080428</td>
<td>2.189099</td>
<td>145</td>
<td>42,9893</td>
</tr>
<tr>
<td>1.295016</td>
<td>2.207035</td>
<td>82</td>
<td>49,6246</td>
</tr>
<tr>
<td>6.499126</td>
<td>2.643371</td>
<td>218</td>
<td>55,52185</td>
</tr>
<tr>
<td>1.218452</td>
<td>2.610665</td>
<td>91</td>
<td>60,5387</td>
</tr>
<tr>
<td>1.364852</td>
<td>2.085102</td>
<td>80</td>
<td>45,8787</td>
</tr>
<tr>
<td>5.444234</td>
<td>2.390545</td>
<td>184</td>
<td>47,89415</td>
</tr>
<tr>
<td>8,732236</td>
<td>2.335641</td>
<td>258</td>
<td>39,6941</td>
</tr>
<tr>
<td>7.112852</td>
<td>2.631666</td>
<td>227</td>
<td>49,1787</td>
</tr>
<tr>
<td>1.358062</td>
<td>2.645776</td>
<td>95</td>
<td>61,04845</td>
</tr>
<tr>
<td>1,372256</td>
<td>2.062005</td>
<td>82</td>
<td>47,6936</td>
</tr>
<tr>
<td>1,246268</td>
<td>2.48166</td>
<td>89</td>
<td>57,8433</td>
</tr>
<tr>
<td>1.071878</td>
<td>2.300168</td>
<td>81</td>
<td>54,20305</td>
</tr>
<tr>
<td>5,222294</td>
<td>2.581355</td>
<td>182</td>
<td>51,44265</td>
</tr>
<tr>
<td>1.273037</td>
<td>2.314253</td>
<td>86</td>
<td>54,174075</td>
</tr>
<tr>
<td>1,083884</td>
<td>1.912759</td>
<td>70</td>
<td>42,9029</td>
</tr>
<tr>
<td>6.10992</td>
<td>2.871523</td>
<td>215</td>
<td>62,252</td>
</tr>
<tr>
<td>1,120437</td>
<td>2.529318</td>
<td>88</td>
<td>59,989075</td>
</tr>
<tr>
<td>1,334393</td>
<td>2.517791</td>
<td>92</td>
<td>58,640175</td>
</tr>
<tr>
<td>1,087132</td>
<td>2.755204</td>
<td>90</td>
<td>62,8217</td>
</tr>
<tr>
<td>1,10348</td>
<td>2.747958</td>
<td>88</td>
<td>60,413</td>
</tr>
<tr>
<td>5,88208</td>
<td>2.194431</td>
<td>194</td>
<td>46,948</td>
</tr>
<tr>
<td>1.464688</td>
<td>2.293449</td>
<td>90</td>
<td>53,3828</td>
</tr>
<tr>
<td>1.402568</td>
<td>2.038249</td>
<td>82</td>
<td>46,9358</td>
</tr>
<tr>
<td>5,959063</td>
<td>2.357203</td>
<td>197</td>
<td>48,023425</td>
</tr>
<tr>
<td>1,197229</td>
<td>1.718377</td>
<td>68</td>
<td>38,069275</td>
</tr>
<tr>
<td>1,498961</td>
<td>2.186498</td>
<td>85</td>
<td>47,525975</td>
</tr>
<tr>
<td>7,983126</td>
<td>2.379023</td>
<td>241</td>
<td>41,42185</td>
</tr>
<tr>
<td>7,480943</td>
<td>2.045466</td>
<td>228</td>
<td>40,976425</td>
</tr>
<tr>
<td>5,807552</td>
<td>2.326441</td>
<td>194</td>
<td>48,8112</td>
</tr>
<tr>
<td>1,166018</td>
<td>2.261761</td>
<td>79</td>
<td>49,84955</td>
</tr>
<tr>
<td>1.474926</td>
<td>2.391088</td>
<td>90</td>
<td>53,12685</td>
</tr>
<tr>
<td>4,663251</td>
<td>2.72611</td>
<td>171</td>
<td>54,418725</td>
</tr>
<tr>
<td>9,814442</td>
<td>2.359937</td>
<td>292</td>
<td>46,63985</td>
</tr>
<tr>
<td>1,369911</td>
<td>2.448396</td>
<td>91</td>
<td>56,752225</td>
</tr>
<tr>
<td>1,378458</td>
<td>2.504581</td>
<td>93</td>
<td>58,53855</td>
</tr>
<tr>
<td>1,056744</td>
<td>2.301602</td>
<td>80</td>
<td>53,5814</td>
</tr>
<tr>
<td>6,650046</td>
<td>2.396044</td>
<td>212</td>
<td>45,74885</td>
</tr>
<tr>
<td>1,215008</td>
<td>2.242799</td>
<td>80</td>
<td>49,6248</td>
</tr>
<tr>
<td>1,034481</td>
<td>2.397105</td>
<td>80</td>
<td>54,137975</td>
</tr>
<tr>
<td>6,653963</td>
<td>2.19946</td>
<td>211</td>
<td>44,650925</td>
</tr>
</tbody>
</table>
### F.4 MIP+SIP with renegotiation

<table>
<thead>
<tr>
<th>time to get IP address</th>
<th>arrival time of the 1st data packet</th>
<th>number of lost packets</th>
<th>estimated lost packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.408637</td>
<td>2.528271</td>
<td>142</td>
<td>56,784075</td>
</tr>
<tr>
<td>1.046247</td>
<td>2.110973</td>
<td>73</td>
<td>46,843825</td>
</tr>
<tr>
<td>1.349577</td>
<td>1.790924</td>
<td>75</td>
<td>41,260575</td>
</tr>
<tr>
<td>4.099458</td>
<td>2.038666</td>
<td>143</td>
<td>40,51355</td>
</tr>
<tr>
<td>1.128968</td>
<td>2.464873</td>
<td>86</td>
<td>57,7758</td>
</tr>
<tr>
<td>3.544875</td>
<td>2.55637</td>
<td>145</td>
<td>56,378125</td>
</tr>
<tr>
<td>1.074813</td>
<td>2.252169</td>
<td>78</td>
<td>51,129675</td>
</tr>
<tr>
<td>1.188377</td>
<td>2.48615</td>
<td>86</td>
<td>56,290575</td>
</tr>
<tr>
<td>1.117363</td>
<td>2.225475</td>
<td>79</td>
<td>51,065925</td>
</tr>
<tr>
<td>1.446621</td>
<td>2.214954</td>
<td>88</td>
<td>51,834475</td>
</tr>
<tr>
<td>1.211652</td>
<td>2.68875</td>
<td>91</td>
<td>60,7087</td>
</tr>
<tr>
<td>1.466827</td>
<td>2.643213</td>
<td>98</td>
<td>61,329325</td>
</tr>
<tr>
<td>1.25369</td>
<td>2.194695</td>
<td>82</td>
<td>50,65775</td>
</tr>
<tr>
<td>7.834399</td>
<td>2.486308</td>
<td>247</td>
<td>51,140025</td>
</tr>
<tr>
<td>1.037827</td>
<td>2.375139</td>
<td>81</td>
<td>55,054325</td>
</tr>
<tr>
<td>1.195169</td>
<td>2.052748</td>
<td>78</td>
<td>48,120775</td>
</tr>
<tr>
<td>1.07309</td>
<td>2.537928</td>
<td>85</td>
<td>58,17275</td>
</tr>
<tr>
<td>1.271391</td>
<td>2.832797</td>
<td>97</td>
<td>65,215225</td>
</tr>
<tr>
<td>1.100539</td>
<td>2.799659</td>
<td>91</td>
<td>63,486525</td>
</tr>
<tr>
<td>1.012391</td>
<td>2.629148</td>
<td>87</td>
<td>61,690225</td>
</tr>
<tr>
<td>6.622707</td>
<td>2.395966</td>
<td>215</td>
<td>49,432325</td>
</tr>
<tr>
<td>3.097775</td>
<td>2.272249</td>
<td>128</td>
<td>50,555625</td>
</tr>
<tr>
<td>1.209117</td>
<td>2.291324</td>
<td>82</td>
<td>51,772075</td>
</tr>
<tr>
<td>1.50193</td>
<td>1.987861</td>
<td>82</td>
<td>44,45175</td>
</tr>
<tr>
<td>1.059048</td>
<td>2.644303</td>
<td>89</td>
<td>62,5238</td>
</tr>
<tr>
<td>1.055476</td>
<td>2.841097</td>
<td>92</td>
<td>65,6131</td>
</tr>
<tr>
<td>1.31691</td>
<td>2.743807</td>
<td>96</td>
<td>63,07725</td>
</tr>
<tr>
<td>1.354745</td>
<td>1.930196</td>
<td>76</td>
<td>42,131375</td>
</tr>
<tr>
<td>1.050031</td>
<td>2.210165</td>
<td>78</td>
<td>51,749225</td>
</tr>
<tr>
<td>5.354254</td>
<td>2.366336</td>
<td>185</td>
<td>51,14365</td>
</tr>
<tr>
<td>1.080337</td>
<td>2.676141</td>
<td>90</td>
<td>62,991575</td>
</tr>
<tr>
<td>5.503608</td>
<td>2.716623</td>
<td>194</td>
<td>56,4098</td>
</tr>
<tr>
<td>5.58447</td>
<td>2.094754</td>
<td>180</td>
<td>40,38825</td>
</tr>
<tr>
<td>1.040576</td>
<td>2.578112</td>
<td>85</td>
<td>58,9856</td>
</tr>
<tr>
<td>1.148531</td>
<td>2.444537</td>
<td>83</td>
<td>54,286725</td>
</tr>
<tr>
<td>3.226627</td>
<td>2.083613</td>
<td>127</td>
<td>46,334325</td>
</tr>
<tr>
<td>1.430954</td>
<td>2.434848</td>
<td>92</td>
<td>56,22615</td>
</tr>
<tr>
<td>1.462136</td>
<td>2.684248</td>
<td>99</td>
<td>62,4466</td>
</tr>
<tr>
<td>1.083835</td>
<td>2.304869</td>
<td>80</td>
<td>52,904125</td>
</tr>
<tr>
<td>2.64166</td>
<td>2.650799</td>
<td>124</td>
<td>57,9585</td>
</tr>
<tr>
<td>1.098403</td>
<td>2.734349</td>
<td>90</td>
<td>62,539925</td>
</tr>
</tbody>
</table>
## F.4 MIP+SIP with renegotiation

<table>
<thead>
<tr>
<th>time to get IP address</th>
<th>arrival time of the 1st data packet</th>
<th>number of lost packets</th>
<th>estimated lost packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.273182</td>
<td>2.507151</td>
<td>90</td>
<td>58,17045</td>
</tr>
<tr>
<td>1.355653</td>
<td>2.580844</td>
<td>92</td>
<td>58,108675</td>
</tr>
<tr>
<td>5.40343</td>
<td>2.476638</td>
<td>182</td>
<td>46,91425</td>
</tr>
<tr>
<td>1.335362</td>
<td>2.714717</td>
<td>95</td>
<td>61,61959</td>
</tr>
<tr>
<td>5.208595</td>
<td>2.466881</td>
<td>181</td>
<td>50,785125</td>
</tr>
<tr>
<td>1.432392</td>
<td>2.26177</td>
<td>88</td>
<td>52,1902</td>
</tr>
<tr>
<td>1.051863</td>
<td>2.708613</td>
<td>90</td>
<td>63,703425</td>
</tr>
<tr>
<td>3.521818</td>
<td>2.700164</td>
<td>145</td>
<td>56,95455</td>
</tr>
<tr>
<td>1.20265</td>
<td>2.42082</td>
<td>83</td>
<td>52,93375</td>
</tr>
<tr>
<td>2.496702</td>
<td>2.397397</td>
<td>115</td>
<td>52,58245</td>
</tr>
<tr>
<td>3.796686</td>
<td>2.612394</td>
<td>149</td>
<td>54,08285</td>
</tr>
<tr>
<td>1.171048</td>
<td>2.316294</td>
<td>83</td>
<td>53,7238</td>
</tr>
<tr>
<td>4.489719</td>
<td>2.244426</td>
<td>161</td>
<td>48,757025</td>
</tr>
<tr>
<td>1.252917</td>
<td>1.857879</td>
<td>73</td>
<td>41,677075</td>
</tr>
<tr>
<td>6.297043</td>
<td>2.316823</td>
<td>202</td>
<td>44,573925</td>
</tr>
<tr>
<td>9.471612</td>
<td>2.532402</td>
<td>281</td>
<td>44,2097</td>
</tr>
<tr>
<td>1.194131</td>
<td>2.751208</td>
<td>93</td>
<td>63,146725</td>
</tr>
<tr>
<td>1.020416</td>
<td>2.553112</td>
<td>82</td>
<td>56,4896</td>
</tr>
<tr>
<td>4.584184</td>
<td>1.908099</td>
<td>155</td>
<td>40,3954</td>
</tr>
<tr>
<td>5.403745</td>
<td>2.381981</td>
<td>185</td>
<td>49,906375</td>
</tr>
<tr>
<td>1.378316</td>
<td>2.023843</td>
<td>82</td>
<td>47,5421</td>
</tr>
<tr>
<td>1.25978</td>
<td>2.519264</td>
<td>88</td>
<td>56,5055</td>
</tr>
<tr>
<td>1.447942</td>
<td>2.4358</td>
<td>91</td>
<td>54,80145</td>
</tr>
<tr>
<td>4.934159</td>
<td>2.308833</td>
<td>171</td>
<td>47,646025</td>
</tr>
<tr>
<td>1.276638</td>
<td>2.306092</td>
<td>83</td>
<td>51,08405</td>
</tr>
<tr>
<td>1.125024</td>
<td>2.535923</td>
<td>87</td>
<td>58,8744</td>
</tr>
<tr>
<td>1.317209</td>
<td>2.026384</td>
<td>79</td>
<td>46,069775</td>
</tr>
<tr>
<td>1.006201</td>
<td>2.116989</td>
<td>74</td>
<td>48,84975</td>
</tr>
<tr>
<td>1.302189</td>
<td>2.356067</td>
<td>84</td>
<td>51,445275</td>
</tr>
<tr>
<td>4.482375</td>
<td>2.693183</td>
<td>166</td>
<td>53,940625</td>
</tr>
<tr>
<td>1.109234</td>
<td>2.70243</td>
<td>91</td>
<td>63,26915</td>
</tr>
<tr>
<td>1.168728</td>
<td>2.470219</td>
<td>85</td>
<td>55,7818</td>
</tr>
<tr>
<td>6.3921</td>
<td>2.67639</td>
<td>212</td>
<td>52,1975</td>
</tr>
<tr>
<td>1.4121</td>
<td>2.540908</td>
<td>93</td>
<td>57,6975</td>
</tr>
<tr>
<td>1.479712</td>
<td>2.434674</td>
<td>93</td>
<td>56,0072</td>
</tr>
<tr>
<td>1.232123</td>
<td>2.510443</td>
<td>90</td>
<td>59,196925</td>
</tr>
<tr>
<td>1.410327</td>
<td>2.349213</td>
<td>88</td>
<td>52,741825</td>
</tr>
<tr>
<td>1.074608</td>
<td>2.182002</td>
<td>78</td>
<td>51,1348</td>
</tr>
<tr>
<td>1.19901</td>
<td>2.597786</td>
<td>87</td>
<td>57,02475</td>
</tr>
<tr>
<td>9.931367</td>
<td>2.409193</td>
<td>287</td>
<td>38,715825</td>
</tr>
<tr>
<td>1.098656</td>
<td>2.663726</td>
<td>88</td>
<td>60,5336</td>
</tr>
</tbody>
</table>
## F.4 MIP+SIP with renegotiation

<table>
<thead>
<tr>
<th>time to get IP address</th>
<th>arrival time of the 1st data packet</th>
<th>number of lost packets</th>
<th>estimated lost packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.411065</td>
<td>2.32346</td>
<td>232</td>
<td>46,723375</td>
</tr>
<tr>
<td>1.079315</td>
<td>2.635922</td>
<td>90</td>
<td>63,017125</td>
</tr>
<tr>
<td>1.146083</td>
<td>2.141332</td>
<td>77</td>
<td>48,347925</td>
</tr>
<tr>
<td>1.022089</td>
<td>2.257304</td>
<td>79</td>
<td>53,447775</td>
</tr>
<tr>
<td>1.105072</td>
<td>2.059745</td>
<td>74</td>
<td>46,3732</td>
</tr>
<tr>
<td>8.544308</td>
<td>2.478803</td>
<td>257</td>
<td>43,3908</td>
</tr>
<tr>
<td>2.836597</td>
<td>2.487185</td>
<td>127</td>
<td>56,085075</td>
</tr>
<tr>
<td>5.103818</td>
<td>2.466216</td>
<td>180</td>
<td>52,40455</td>
</tr>
<tr>
<td>1.070138</td>
<td>2.360243</td>
<td>80</td>
<td>53,24655</td>
</tr>
<tr>
<td>1.207326</td>
<td>2.567871</td>
<td>87</td>
<td>56,81685</td>
</tr>
<tr>
<td>5.870761</td>
<td>1.847993</td>
<td>180</td>
<td>33,230975</td>
</tr>
<tr>
<td>5.702683</td>
<td>2.499517</td>
<td>194</td>
<td>51,432925</td>
</tr>
<tr>
<td>1.242914</td>
<td>2.747624</td>
<td>96</td>
<td>64,92715</td>
</tr>
<tr>
<td>1.231269</td>
<td>1.951604</td>
<td>75</td>
<td>44,218275</td>
</tr>
<tr>
<td>1.25489</td>
<td>2.540296</td>
<td>90</td>
<td>58,62775</td>
</tr>
<tr>
<td>1.330208</td>
<td>2.113709</td>
<td>81</td>
<td>47,7448</td>
</tr>
<tr>
<td>1.15221</td>
<td>1.881834</td>
<td>70</td>
<td>41,19475</td>
</tr>
<tr>
<td>7.086608</td>
<td>2.321867</td>
<td>218</td>
<td>40,8348</td>
</tr>
<tr>
<td>1.423635</td>
<td>2.867381</td>
<td>102</td>
<td>66,409125</td>
</tr>
<tr>
<td>1.213257</td>
<td>2.527601</td>
<td>89</td>
<td>58,66875</td>
</tr>
<tr>
<td>9.50674</td>
<td>1.921808</td>
<td>273</td>
<td>35,3315</td>
</tr>
<tr>
<td>1.353298</td>
<td>2.365769</td>
<td>86</td>
<td>52,11755</td>
</tr>
<tr>
<td>9.417211</td>
<td>2.485937</td>
<td>280</td>
<td>44,569725</td>
</tr>
<tr>
<td>1.126783</td>
<td>2.81133</td>
<td>94</td>
<td>65,830425</td>
</tr>
<tr>
<td>1.257824</td>
<td>2.508117</td>
<td>88</td>
<td>56,5544</td>
</tr>
<tr>
<td>4.121687</td>
<td>2.782169</td>
<td>160</td>
<td>56,957825</td>
</tr>
<tr>
<td>1.020158</td>
<td>2.507799</td>
<td>84</td>
<td>58,49605</td>
</tr>
<tr>
<td>3.249311</td>
<td>2.600053</td>
<td>137</td>
<td>55,767225</td>
</tr>
</tbody>
</table>

**Average**

- 2.764713543
- 2.390794476
- 121,7019231
- 52,5840845

**Ecart type**

- 2.447864933
- 0.257697263
- 57,97723688
- 7,133510035

**Variance**

- 5,992042731
- 0.066407879
- 3361,359996
- 50,88696541

**Confidence Interval 95%**

- min 2.361490808
- max 2.420098144
- 51,77290781
- 53,39526118
Bibliography


