OpenIMS and Interoperability with Asterisk/Sip Express VOIP Enterprise Solutions

by

Fei Yao

Li Zhang

Thesis in partial fulfilment of the degree of
Master in Technology in
Information and Communication Technology

Agder University College
Faculty of Engineering and Science

Grimstad
Norway

May 2007

Abstract

Nowadays, with the development of the next-generation network, IMS becomes very important, and it will play a key role in the future All-IP infrastructure. But it is still in the developing stage, and it will take time for all 3G mobile networks upgrade to 3GPP networks. Therefore, the research project will study the interoperability solutions between Open Source IMS and Open Source VoIP/SIP implementations. This project is supported by auSystems and HiA.

OpenIMS (Open Source IMS) was developed by FOKUS (a Germany institution) in December 2006. It is a free IMS enterprise solution. As it is so new, there are still many doubts about using it. Therefore, it is important to evaluate it. auSystems and HiA have already established a research testbed for Enterprise VoIP solutions. For us, as a starting point, we extended the existing testbed with a new IMS sub-domain based on the "Open Source IMS Core". And then, Wireshark is used to trace the signaling to validate if the functionality of each component of the OpenIMS is conformant with the 3GPP R6 specification in TS 24.229 and TS 29.229.

After that we evaluated the performance of the OpenIMS as well. In this part, SIPp is used as the testing tool to do experiments. According to the results, we evaluated the performance of OpenIMS in the following aspects: the numbers of subscribers the OpenIMS can handle; the reliability of OpenIMS; does it work simultaneously or stably; how often does it response? We also discuss what client or devices Open IMS can support.

In this project, the other important task is to find a possible interoperability between OpenIMS and the existing SIP/VoIP enterprise solution. In order to implement the challenge, we divided the problem into two parts:
Firstly, we solved the interpretability from the SIP clients to the IMS clients in a single domain. In this step, we proposed a possible solution, designed the structure of OpenIMS and implemented it.
The last challenge of the project is to researche the interconnect way of OpenIMS with existing SIP/VoIP solutions from OpenIMS domain to two domains which contain OpenIMS domain and non-OpenIMS enterprise domain. In addition, state-of-the-art of SIP/VoIP solutions had been studied. We tried to implement the "client-based" solution of interconnecting OpenIMS with existing SIP/VoIP. This solution was proposed in [8].

Keywords: IMS, OpenIMS, CSCFs, FHoSS, SIP signaling

Preface

This report is the master thesis in Master of Science program in Information and Communication Technology (ICT) at Agder University College (AUC), Faculty of Engineering and Science in Grimstad, Norway. The project has been carried out in collaboration with auSystems in Grimstad from January to May 2007.

First we would like to thank Professor Dr. Frank Reichert, our supervisor at Agder University College, for his guidance and constructive advice throughout the project period. Then we would like to thank auSystems in Grimstad providing us with a comprehensive and exciting work. In this context, we also want to thank our supervisor at anSystems, Anders Hagelskjær Aasgaard, for his good help and support on our research. And we would like to give our special thanks to our co-supervisor in Agder University College, Andreas Häber, for his valuable suggestions and constant help during the work.
Finally, we give our thanks to Head of Studies, Stein Bergsmark, for his contributions and good arrangement.

Grimstad, May 2007

Fei Yao Li Zhang

Table of Contents

Figure Lists

Table Lists

1 Introduction

1.1 Background

Nowadays, with the development of the next-generation network (NGN), some new standards are coming up for voice and video services. VoIP (Voice over IP) is becoming very popular, especially amongst global enterprise businesses. It is developing quickly, gradually replacing the traditional telephony services gradually because of its flexibility and cheaper costs. Besides, VoIP has the potential to pave the way for converged data and voice networks that can deliver advanced communications applications to users, anytime, anywhere. Different VoIP protocols and solutions have created compatibility and interoperability obstacles. Security and QoS are very important for enterprises in choosing the right solution.

As the key element in NGN, IMS (IP Multimedia Subsystem) plays an important role in offering key features such as QoS, security, group management, and instant voice messaging. IMS makes it easier for operators to provide new services, in comparison to GSM where this is very limited The IMS is an open, standardized architecture that aims to merge multimedia services across the cellular world and IP networks, using the same standard protocols for both mobile and fixed IP services. It is defined by the 3rd Generation Partnership Project (3GPP). For service providers, IMS (IP Multimedia Subsystem) will enable faster and easier deployment of new services based on standards while cutting cost. For end users, IMS could afford a new, flexible and personalized real-time communication service across any network and any device, whether it is a PDA, PC, mobile phone or TV. With these features, IMS would be the core network of NGN.

This thesis is written in association with auSystems. auSystems in Norway is one of the world's leading independent 2.5G/3G competence centers, working with major Network Equipment Providers and global Mobile Operators. They have a broad competence within real-time systems, mobile networks, 2.5G/3G mobile data communication and services/products within machine-to-machine (M2M) communication. This company works with GPRS/UMTS/IMS Software development for Ericsson and offers professional services. Since its major customers are migrating to an All-IP infrastructure, it is important for auSystems to keep up with this area and to have the knowledge to explore new and promising business opportunities, e.g., in enterprise solutions in the VoIP/SIP/IMS area. According to some existing solutions, they plan to further extend their testbed with a new sub-domain based on the “Open Source IMS core” furthermore.

Since IMS has already been used gradually and there are already many Open Source projects established in the plain VoIP area for SIP clients, proxies, stacks and tools around the IETF sip standards, there are currently virtually no Open Source projects with specific focus on the IMS. Therefore, the Fraunhofer Institute FOKUS came up with the Open Source IMS core that aims to fill the currently existing IMS void in the Open Source software landscape with a flexible and extendable solution that has already proven its conformance and performance in several national and international R&D projects. [FOKUS 01]

Open Source IMS Core System [FOKUS 02] is an IP Multimedia System for test. It has been developed by the Fraunhofer Institute FOKUS. They point out that this Open Source IMS Core System is not intended to become or act as a product in a commercial context. Its sole purpose is to provide an IMS core reference implementation for IMS technology testing and IMS application prototyping for research purposes, typically performed in IMS test-beds. This target has also motivated the decision to use open source software (i.e. SER (SIP Express Router) based on GPL).

1.2 Problem description

The research project will study the interoperability solutions between Open Source IMS and Open Source VoIP/SIP implementations. The project is provided by auSystems and HiA who have established a research testbed for Enterprise VoIP solutions based on Open Source systems such as “Asterisk” and “SipExpress Router”. Our task is to extend the testbed with a new IMS domain based on the “Open Source IMS Core”, and research the interconnect SIP and IMS solutions as well as try to implement some of the solutions.

1.3 Problem statement

The project aims to research interoperability solutions between IMS and Open Source VoIP/SIP implementations. In order to achieve this purpose, the project will discuss the following:

- functional evaluation of an OpenIMS
- performance evaluation of an OpenIMS
- study of integration issues with existing SIP/VoIP solutions

1.3.1 Functional evaluation of an Open IMS

auSystmes and HiA have already established a research testbed for Enterprise VOIP solutions based on Open Source systems which contains all the necessary function of a typical enterprise installation. Therefore, as a starting point, we plan to extend the existing testbed with a new IMS sub-domain based on the “Open Source IMS Core”.

Since the tested needs to be extended based on the "Open Source IMS Core "[FOKUS 02], research and understand of many state-of-the-art aspects in Open Source IMS is essential. The Open Source IMS Core System is an IP Multimedia System for test. It has been developed by the Fraunhofer Institute FOKUS. The Open Source IMS Core consists of Call Session Control Functions (CSCFs), including Proxy-CSCF, Interrogating-CSCF, and Serving-CSCF, a SIP2IMS Gateway and a Home Subscriber Server (HSS) [FOKUS 02].

Our purpose in functional evaluation of the OpenIMS is to validate whether the system developed is conformant to the requirement specified. Since IMS was first released in 3GPP Release 5, then further developed in 3GPP Release 6, Release 7 is too new for us to use as the specification. Therefore, we choose 3GPP Release 6 as our requirement specified. That means our task is to evaluate if the functionality of the OpenIMS is conformant with IMS specifications in 3GPP Release 6. We will check if any features of OpenIMS are missing, or whether there are features not mentioned in 3GPP Release 6 but mentioned in 3GPP Release 7.

We need to evaluate the central components of the Open Source IMS Core. They are:

- HSS
- P-CSCF
- I-CSCF
- S-CSCF
- Sip2ims
- interfaces between each component: Cx, Mw and Gm

If time permits, the application service may be included as well, with its associated interfaces.

Therefore, we need to look at each point for each function and send it input and check the output from the system to see if it is behaving as specified. Sessions record is very important for us when we do such test, because we will use it as our test data and our analyze will based on it. Through the analysis, we could tell whether it is conformant to the specification or not. SIP clients or IMS clients will be used as the end users, and we will choose the basic SIP signaling procedures for testing: registration, subscribing and call sessions set up. If time permits, presence and instant message will be also implemented.

1.3.2 Performance evaluation of OpenIMS

The second task is to evaluate the performance of OpenIMS. We plan to do some tests and discuss the test results with the following question in mind:

- How many users can it handle?
- Will the system always be able to handle user's requests? Or it will only be able to work sporadically?
- If the system can provide services under normal operations, will it also be able to

react under abnormal situations, for example, when the system is overloaded?

- What is the latency for the system in response to users' requests? Moreover, what are the variations in this time?
- We could define a particular time interval and check how many user requests the system can handle during that period. Will there be any data loss during the process?

1.3.3 Study of integration issues with existing SIP/VoIP solutions

The third task is to study how OpenIMS interconnects to the rest of the world since OpenIMS works well.

In this part, we will consider these aspects:
What could we get if we use the interoperability solution between IMS and VoIP/SIP? Why should we combine them? Several ways to combine them have already been discovered. Which way is better? We need to study and compare them and find the best way to implement it.

With the development of VoIP, enterprises are not satisfied with only using traditional telephone services. Therefore, more and more SIP/VoIP solutions have already been developed. Interconnecting applications such as voicemail and video conference are interesting and innovative. There are some basic requirements for a valid SIP/VoIP enterprise solution. First, it should support [8] for a wide array of operation systems. Secondly, it should include SIP, video and NAT/firewall traversal supporting. And it should also be compatible with a large array of soft/hard phones.

Our task in this part will be divided into four parts:

- State-of-the-art SIP/VoIP solutions will be evaluated at first. For example, we will evaluate software SIP PBX that is the key element (Asterisk vs. SIP Express Router).
- We will then try to implement the acceptable solutions of interconnecting OpenIMS with existing SIP/VoIP.
- After that, we will find the way of interconnecting OpenIMS with security solutions. Firewalls will be involved.
- Last, we will discuss what client or devices Open IMS can support.

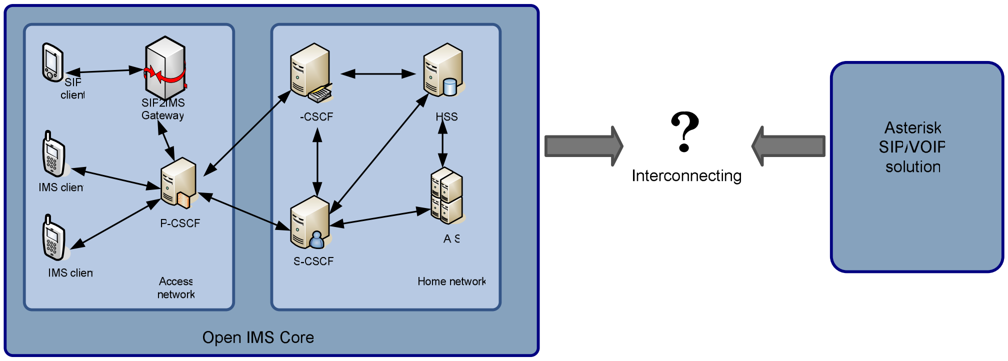


Figure 1-1 Interconnection of SIP and IMS solutions

Additionally, SIP2IMS, one of the key elements of the OpenIMS Core should be discussed in this part. This is because in order to accelerate testing and to integrate with SIP UEs and test tools, a gateway that helps SIP traffic to work in an IMS environment was required [FOKUS 05]. For example, we could learn the logical architecture of the enterprise IMS gateway at this stage.

1.4 Problem premises and Temporary assumptions

Our research is based on prior research done by auSystems and HiA which has already established a research testbed for Enterprise VOIP solutions based on Open Source systems that contains all the necessary functions of a typical enterprise installation. Under the assumption that the Open IMS is available during our study, we will extend the testbed by setting up a new IMS sub-domain based on the Open Source IMS core, and verify whether the new domain we will establish is fully functional.

We will study and describe the solutions to interconnect SIP and IMS. In this migration stage, we assume that the enterprise user may have both IMS enabled terminals and non-IMS SIP clients. We will demonstrate, if possible, selected key features, but not full and very detailed IMS integration solutions. Considering the devices we will use, no IMS phones will be available in time; therefore we will use portable PC, if available.

At the end, if time permits, we will try to implement the SIP to IMS interconnecting solution, which is being forwarded in prior research, and try to put forward a new IMS migration solution. However, if the time is limited, we may not complete this task.

1.5 Motivation

The technologies involved in this project are very new. All of them are broadly used now or will be in the future. It is these new technologies in mobile communication that interest us most.

IMS provides a range of key enablers; it allows easier integration and supports legacy mobile phones. As standardization enables competition, it will lead to better prices and solutions and will be less costly to deploy and operate. With these feature, IMS will be the key element for the next-generation network. There hws been great development in IMS recently. It will be involved in broad area of new technology.

Voice-over-IP services based on SIP are widely spread and accepted by end-users, so it is natural that we also share interest in it. Technologies related to VoIP are also widely deployed.

SIP is a new key technology in the internet for handling end-to-end sessions. It is quite clear to us is that SIP will play a much more important role in the future, especial for the next generation of communication devices.

As a result of this project, we can gain insights into IMS, VoIP and SIP technology. We consider this subject as a very important factor for the further research with the field, one that will make use of upcoming technology. This new knowledge and the experience of this project will be much appreciated by us, as it will help us with our future work in the field.

Additionally, the project gives us good theoretical and practical experience with VoIP/SIP/IMS, and a chance to demonstrate our capabilities to solve complex system engineering problems, which will be very rewarding for us.

We also find that auSystems is a really a good company with advanced IT technology in mobile communication area; the work experience with such a company will provide us with priceless insight that can help us in our future careers.

1.6 Importance of the study

The importance of doing a research is showed in two aspects:

- With ourselves in mind, this research’s importance lies in the information and experiences we will accumulate through the study. The competence gained in internet telephone, SIP and IMS, the experience accumulated by working with a real company and the understanding we gain through usable engineering is extremely benefited when applying for a job.

- With the project itself in mind, knowing whether the research contributes for the field is important.

  This project is support by auSystems, a company that is one of the world's leading independent 2.5G/3G competence centers. In this project, our main task is to find an enterprise solution with interoperability between open source IMS and VoIP/SIP. That is to say, if the solution successfully shows good results through tests, auSystems will have a more effective and an easier way to combine the solution between OpenIMS and VoIP/SIP and help them to decide future investment strategies.

  On the hand, if the result is not as good as expected, out work result could show auSystems that our solution is a less advantageous way to integrate open source IMS and VoIP/SIP, saving them time and money since they know to go with another method. Thus our work will still be important in guiding future research in the right direction.

  Lastly, it is possible that our thesis could serve as a helpful reference for future projects on using interoperability between open IMS and VoIP/SIP.

1.7 Report Outline

This report is structured as follows.

Chapter 1: introduces the master thesis that is the current chapter.

Chapter 2: provides the basic theory about IMS/OpenIMS. This will establish a foundation for understanding the later work described in the report.

Chapter 3: introduces the installation and configuration for set up OpenIMS in the testbed.

Chapter 4: gives a detail evaluation for the functionality of OpenIMS.

Chapter 5: proposes a solution of interoperability between IMS and SIP clients within a single domain and describes how we implemented it.

Chapter 6: describe how we try to implement a solution of the interoperability between OpenIMS and existing SIP/VoIP solutions, and discuss the problems we met.

Chapter 7: evaluates the performance of OpenIMS.

Chapter 8: discusses all the results we have achieved in this project.

Chapter 9: gives the conclusion of our project work and point out possible further work.

2 Theory and state-of-the-art

As the next step of 3G, IMS is discussed hot and broadly nowadays. Quite a few papers on the topic have been written in recent years. Papers discussing this topic based on SIP solution are not hard to find on a scientific database or mobile communication web site. Most of the literatures that we used in this chapter could be searched for at IEEE or ACM database. Some application solutions we got from the Ericsson website. As for Open Source IMS, which is not so commonly used, we can find some information on the website of Fraunhofer Institute FOUKUS. The definitions are from Third Generation Partnership Project (3GPP).

The 3GPP was created in 1998 as a collaboration agreement to develop a third generation telecommunication system based on the GSM specifications. The 3GPP is organized into a number of working groups, known as the Project Co-ordination Group (PCG), which is responsible for the overall management of 3GPP, and Technical Specification Groups (TSGs) whose work is to produce technical work. The 3GPP working groups' output is in the form of Technical Specifications (TS) a Technical Reports (TR), and '3GPP [1] its specifications in what is called a Release'. The first version of IMS was introduced in 3GPP Release 5.

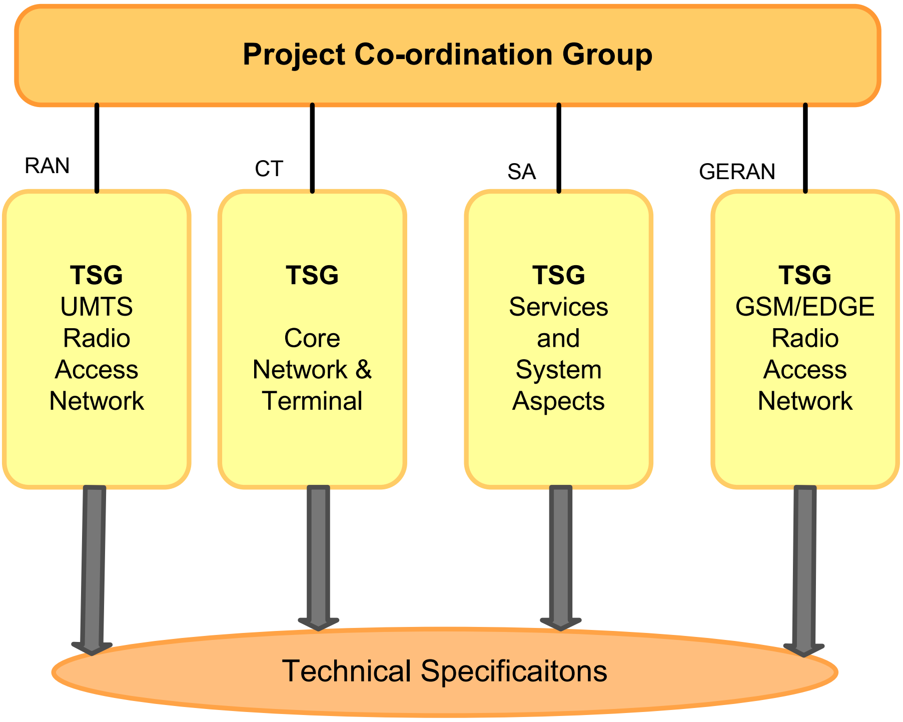

Project Co-ordination Group
RAN
CT
SA
GERAN
TSG UMTS Radio Access Network
TSG Core Network & Terminal
TSG Services and System Aspects
TSG GSM/EDGE Radio Access Network
Technical Specificaitons

Figure 2-1 3GPP new internal structure

2.1 Theory

2.1.1 IMS

At present, cellular networks have already provided a wide range of service, and cellular users can access the Internet using a data connection and access any services the Internet may provide. So why do we need the IMS (IP Multimedia Subsystems)?

As we known, there are different domains in the 3G network, which are named the circuit-switched domain and the packet-switched domain respectively.

In the circuit-switched domains the circuits are used to transport voice and video, or are used to transport instant messages. There are two different planes in the circuit-switched networks: the signaling plane and the media plane. 'The signaling plane includes the protocols used to establish a circuit-switched path between terminals and the media plane includes the data transmitted over the circuit-switched path between the terminals.' [1] And the media plane also includes the encoded voice exchanged between users.

The 3GPP (Third Generation Partnership Project) defines the MSC (Mobile Switching Center) with two parts: the MSC server, and the media gateway. The MSC server will work for the signaling plane while the media gateway will handle the media plane. The IMS also has a signaling plane and a media plane just as the circuit-switched networks and the 3GPP.

Because of the packet-switched domain, users can connect to the Internet. They are capable of making a VoIP call over the packet-switched domain.

Now back to our question: why do we need the IMS?

The first reason for creating the IMS was to provide the service with QoS (Quality of Service). [1] The packet-switched domain provides a best-effort service without QoS. So the quality of a VoIP conversation can have dramatically changes. For example, at a certain point, the person's voice may sound perfectly clear, but it can become impossible to understand afterwards. So we created the IMS to provide the QoS so users can enjoy their conversations. The IMS allows operators to control the QoS a user gets, so that operators can differentiate certain groups of customers from others.

Another reason for creating the IMS was to enable charges of multimedia sessions appropriately. 'The IMS provides information about the service being invoked by the user, and with this information the operator decides whether to use a flat rate for the service, apply traditional time-based charging, apply QoS-based, or perform any new type of charging.' [1]

The third main reason for the existence of the IMS was to provide integrated services to users. Service developers use the standard interface defined by the IMS, so that operators can integrate services and create new services.

The operators today want to provide more packet-switched services to users, that is, the mobile Internet needs to become more attractive to its users. In this condition, the IMS has been established.

So the IMS aims to [1]:

1 *combine the latest trends to technology*
2 *make the mobile Internet paradigm come true*
3 *create a common platform to develop diverse multimedia services*
4 *create a mechanism to boost margins due to extra usage of mobile packet-switched networks*

There are some requirements that led to design of the 3GPP IMS. [1]

1 *'Support for establishing IP Multimedia Sessions.*
2 *Support for a mechanism to negotiate Quality of Service (QoS).*
3 *Support for interworking with the Internet and circuit-switched networks.*
4 *Support for roaming.*
5 *Support for strong control imposed by the operator with respect to the services delivered to the end-user.*
6 *Support for rapid service creation without requiring standardization.*

2.1.2 IMS Architecture

The 3GPP standardize functions instead of nodes, so the 'IMS architecture is a collection of functions linked by standardized interfaces.' [1] The Figure 2-2 shows an overview of the IMS architecture. From the figure we can see that the IMS terminal uses a radio link to attach to the network, and the IMS supports other types of devices and accesses. For example, Personal Digital Assistants and computers are devices that can connect to the IMS, and WLAN or ADSL are examples of access methods. Besides, SIP is used as the main signaling protocol in IMS.

The nodes including in Figure 2-2 are [1]:

1 Home Subscriber Servers (HSS) and Subscriber Location Functions(SLFs)
2 Call/Session Control Functions(CSCFs)
3 Application Servers(ASs)
4 Media Resource Functions(MRFs), each one further divided into Media Resource Function Controllers(MRFCs) and Media Resource Function Processes(MRFPs)
5 Breakout Gateway Control Functions(BGCFs)
6 PSTN gateways, each one divided into an Signaling Gateway(SGW) and Media Gateway Controller Function(MGCF) and an Media Gateway(MGW)

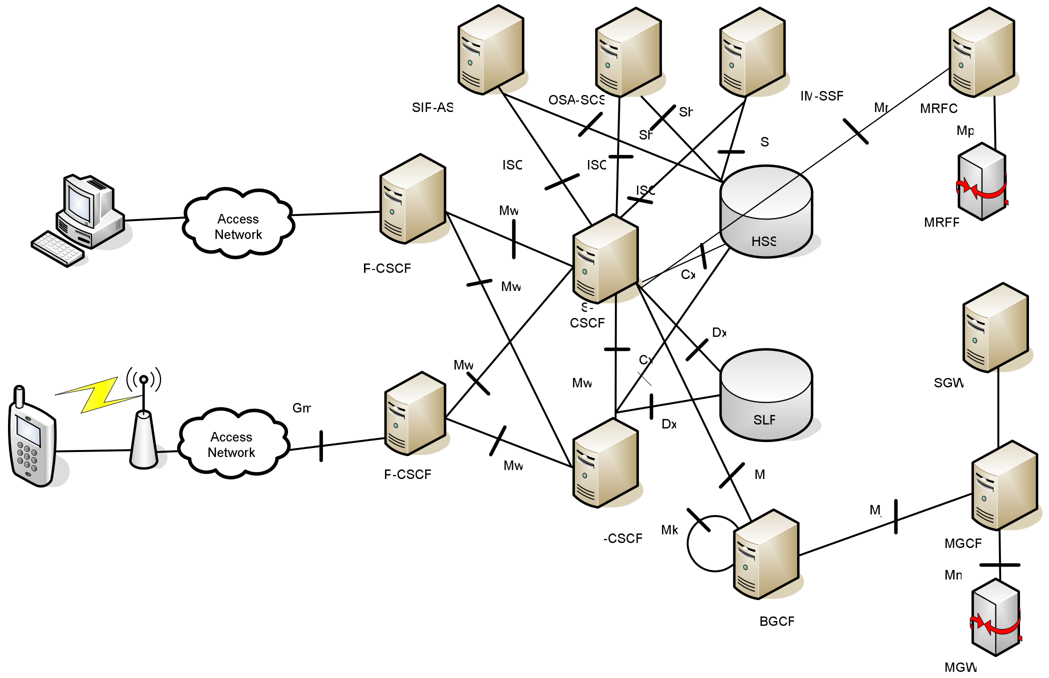


Figure 2-2 3GPP IMS architecture [1]

2.1.2.1 Identification in the IMS

Identification is one of the most important abilities of a network. Users have to be identified in any kind of network, such as when calls can be directed to the appropriate user. Additionally, services need to be identified as well when they are provided, typically using some special number. IMS also has its own way to identify users and provides mechanisms to identify services.

Private User Identities

Unlike Public User Identity, each IMS subscriber has one Private User Identity. The Private User Identity takes the format of NAI (Network Access Identifier) as: RFC 2486 [RFC01]

username@operator.com.

The Private User Identity is used for subscription identification and authentication. It is stored in a smart card and there is no need for the user to know it.

Public User Identities

Each IMS user can be allocated one or more Public User Identities, which is used to route SIP signaling in IMS. The Public User Identity can be a SIP URI or a TEL URI. If Public User Identity is a SIP URI, the form is as follow: RFC 3261[RFC02]

sip: first.last@operator.com

In SIP URI, it's possible to contain a telephone number in the SIP URI, using the following format:

sip: +1-212-555-0293@operator.com; user=phone

When a PSTN (Public Switched Telephone Network) subscriber makes a call to IMS terminal or receives a call from IMS user, the TEL URI is always needed, because the identification in PSTN can only be presented as numbers. So, TEL URI uses an international way to present a phone number: RFC 3966[RFC03]

tel: +1-212-555-0293

In 3GPP Release 5, the relationship between the Public User Identity and Private Use Identity is shown in figure 2-3:

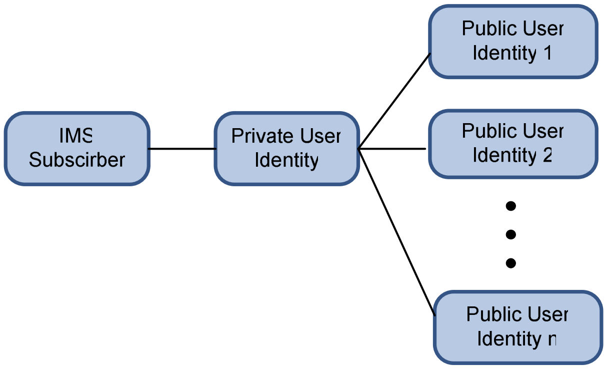


Figure 2-3 Relationship of Private and Public User Identities in 3GPP R5 [1]

From this figure, each IMS Subscriber is allocated only one Private User Identity and a set of Public User Identities. And the all the Private User Identity and Public User Identities are stored in HSS which is used as a database.

In 3GPP Release 6, the situation has been extended. Each IMS Subscriber could have more than one Private User Identity, which means that users may have different smart cards (each smart card stores only one Private User Identity) that insert in different IMS terminals. And one Public User Identity can be used in one or more Private User Identities, as the following figure 2-4 shows, Public User Identity 2 is used in both Private User Identity 1 and Private User Identity 2.

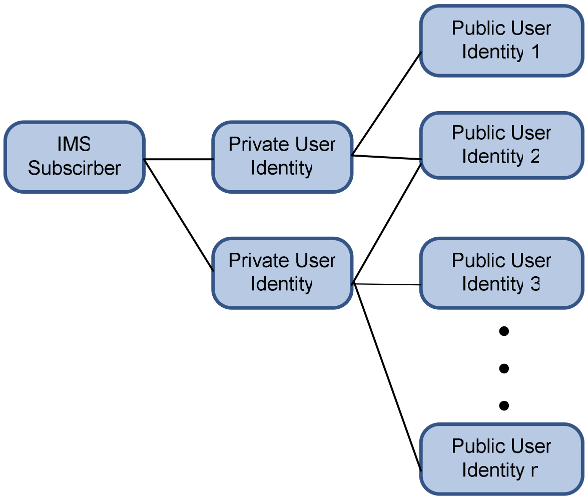


Figure 2-4 Relationship of Public User Identity and Private User Identities in 3GPP R6 [1]

Public Server Identities

The Public Server Identities are allocated to the server hosted in Application Server. And as the same as the Public User Server, the Public Server Identities can also take the format of SIP URI or TEL URI. [1]

2.1.2.2 Home Networks and Visited Networks

When we talk about the home networks, we mean that we use the infrastructure that is given by our network operator. However, if we roam out of the range of our home networks, we will no longer be using the infrastructure that is provided by our own network operator, instead, we will get it from another operator. In this condition, we call the infrastructure a visited network, and we are the visiting users to this network. Whenever we visit to a visited network, a roaming agreement is always needed between visited network operator and home network operator.

Almost all the IMS nodes are located in the home network, but P-CSCF is a special case that can be located in both the home network and the visited network. [1]

2.1.2.3 HSS

The HSS (Home Subscriber Server) is the main user database of the IMS. It is used to store the authentication information of users and subscription-related information (user profiles) and users' Initial Filter Criteria. [Wikipedia 01] Technically, the HSS is an evolution of the HLR (Home Location Register). A network may contain more than one HSS, but in any case, all of the data related to a particular user is stored in a single HSS.

2.1.2.4 SLF

An SLF (Subscriber Location Function) is needed when multiple HSSs are used. It is a database used to map the users' address to HSSs. Both the HSS and the SLF can implement the Diameter protocol. [1] The HSS can connect to both Cx interface and Dx interface, and the SLF can connect to Dx interface.

2.1.2.5 CSCF

The Call/Session Control Function is a SIP service and is very important to IMS. There are three types of CSCF: P-CSCF; I-CSCF; S-CSCF

P-CSCF

A P-CSCF (Proxy-CSCF) is a SIP proxy server that is the first point of contact for the

IMS terminal. The P-CSCF can act as a proxy. For example, it accepts requests and services. The P-CSCF may also behave like a User Agent, where for example, it could generate SIP transactions in abnormal conditions. The P-CSCF has several important functions.

1 Related to security.
It 'establishes a number of IPsec security associations which offer integrity protection toward the IMS terminal '. [1] The P-CSCF informs the identity of the user to other nodes in the network after it identifies the user. This identity of the network user will have some purposes, for example, to provide personalized services and generating account records. After the rest of the nodes in the network receive the identity, they will listen to P-CSCF and will not authenticate the user anymore. Besides, the P-CSCF authenticates the SIP requests which are sent by the IMS terminal.

2 The P-CSCF forwards SIP requests and responses in the suitable paths.
It forwards the SIP register request received from the UE (User Equipment) to an entry point determined using the home domain name, as provided by the UE; It forwards the SIP requests or response to the UE; It forwards SIP messages to the SIP server whose name has been received by P-CSCF. [8]

3 It can also compress and decompress SIP messages.
Sometimes, SIP messages are large and to transmit them over a narrowband channel may take a few seconds. In order to reduce the time of transmitting a large SIP message, the P-CSCF compresses the message and sends it over the air interfaces, and then decompresses it at the end.

As we mentioned before, the P-CSCF can be located in visited network or in the home network just as the GGSN (Gateway GPRS Support Node).

I-CSCF

An I-CSCF (interrogating-CSCF) 'is a SIP proxy located at the edge of an administrative domain'. [1] It is the entry point from the visit network to the home network, and is also the main connection of IMS and other PLMN. The DNS (Domain Name System) records of the domain stores the address of the I-CSCF. 'There may be multiple I-CSCFs within an operator's network.' [8] There are functions performed by the I-CSCF.

1 The I-CSCF has an interface to the HSS and SLF, which is based on the Diameter protocol. It can obtain the address of S-CSCF from HSS. And it could also retrieve user location information and route a SIP request received from another network towards the appropriate destination, i.e., the S-CSCF.

2 The I-CSCF 'assigning a S-CSCF to a user performing SIP registration'. [8] If the SIP messages contain some information about the domain, such as their DNS name, the number of servers in the domain or their capacity, the I-CSCF could encrypt parts of the SIP messages.

3 The I-CSCF may perform transit routing functions. When the I-CSCF determines

the destination of the session is not in the IMS, it may forward the request or return with a failure response.

The I-CSCF is usually located in the home network. However, for some special cases, for example, an I-CSCF (THIG: Topology Hiding Internetwork Gateway), it may also be located in visited network.
Talk to THIG, it has roles including the firewall function for signaling, topology hiding and conversion between IPv4 and IPv6, and the provision of NAPT.

S-CSCF

A S-CSCF (Serving-CSCF) is the central node of the signaling plane in IMS. The S-CSCF has a session state and can take charge of session control. It can also perform as a SIP server or a SIP registrar. A network usually contains a number of S-CSCFs and each S-CSCF serves a number of IMS terminals.

1 It can behave as proxy server, i.e. it services the receiving requests or forwards them on. And the S-CSCF can behave as a User Agent as well, i.e. it may terminate and generate SIP transactions.
2 The S-CSCF shall notify subscribers about registration changes. And it implements a Diameter interface to the HSS in order to download the user profile from the HSS. If a user tries to access the IMS through the HSS, the S-CSCF will download the authentication vector.
3 One of the main functions of the S-CSCF is to provide SIP routing. If the user dials a telephone number instead of a SIP URI (Uniform Resource Identifier), the S-CSCF provides translation services.

The S-CSCF usually located in the home network.

2.1.2.6 Application Server

Application server (AS) is the service creation and execution platform that interacts with the S-CSCF using ISC interface.

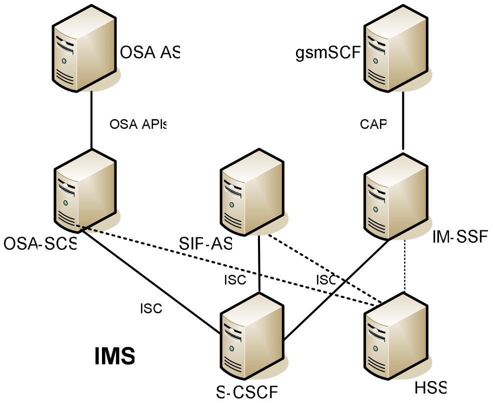


Figure 2-5 Three types of Application Servers [1]

1 **SIP-AS** [1]**:** The SIP-AS is the service part in the IMS, supporting well defined signaling and administration interfaces. It is triggered by the S-CSCF and it comprises filter rules for choosing applications for the handling of the session.
2 **OSA-SCS:** An Open Service Access - Service Capability Server interfaces with OSA Application Servers using Parlay
3 **IM-SSF:** An IP Multimedia Service Switching Function interfaces with CAMEL Application Servers using CAP

The SIP interface the AS may provide a Sh interface to the HSS. [1] The SIP-AS and OSA-SCS interfaces to the HSS are based on the Diameter protocol. They are used to upload or download the data to a user maintained in the HSS. And the IM-SSF interface towards the HSS is based on MAP (Mobile Application Part).

2.1.2.7 BGCF

'A BGCF *(*Breakout Gateway Control Function) [1] is a SIP server that includes routing functionality based on telephone numbers'. It's only used when calling from the IMS to a phone in a circuit switched network, to select an appropriate network or a suitable PSTN/CS gateway. If the S-CSCF has determined that the session cannot be routed using DNS, the BGCF will process requests for routing. It' determines the next hop for routing the SIP message.' [8]

If the PSTN/CS (Public Switched Telephone Network/Circuit Switched) Domain breakout is to occur in the network as the BGCF is located, then the BGCF shall select a MGCF (Media Gateway Control Function) to interwork with the PSTN/CS Domain. If the routing determination of the breakout is to occur in another network, the BGCF will forward this session signaling to another BGCF. In addition, if the routing determination results in the session moving towards another IMS network, the BGCF shall forward the message to an I-CSCF in the network.

There may be more than one BGCF in an operator's network.

2.1.2.8 PSTN Gateways

A PSTN/CS gateway interfaces with PSTN circuit switched (CS) networks. It allows IMS terminal to make calls to the PSTN or receive calls from the PSTN.

1 **MGCF:** An MGCF (Media Gateway Controller Function) is the main role of the PSTN/CS gateway. It does call control protocol conversation and maps SIP to ISUP. It also controls the resources in an MGW with a H.248 interface.
2 **SGW:** An SGW (Signaling Gateway) provides an interface towards the signaling plane of the CS network. 'It [Wikipedia 01] transforms lower layer protocols as SCTP (which is an IP protocol) into MTP (which is a SS7 protocol), to pass ISUP

(ISDN User Part) From the MGCF to the network.'

3 **MGW:** 'An MGW (Media Gateway) [1] interfaces with the media plane of the PSTN or CS network'. The MGW can send or receive IMS media and can transcode when the codes cannot match.

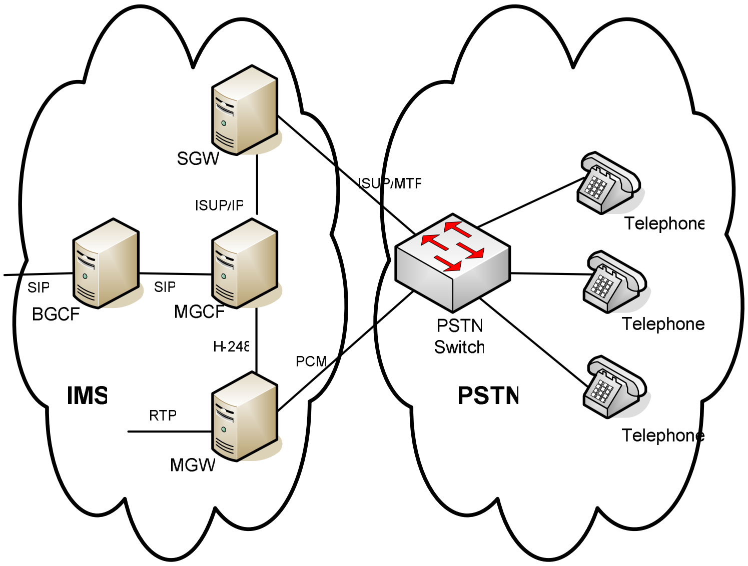


Figure 2-6 The PSTN/CS gateways interfacing a CS network [1]

2.1.2.9 Security Gateway

The SEG (Security Gateway) is used for network security which deals with securing traffic between different security domains. All the traffic will go through a SEG when they come or leave a security domain. Figure 2-7 shows that when traffic is sent from one domain to another it will traverse two SEGs.

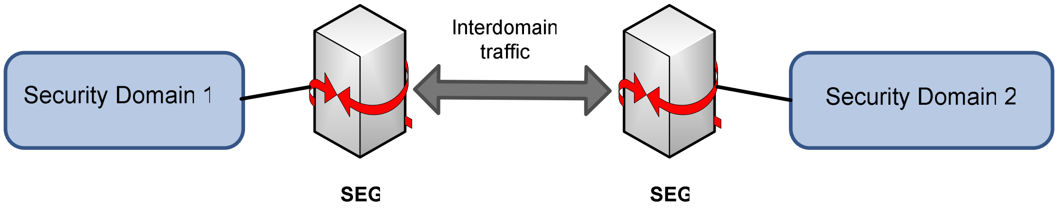


Figure 2-7 Inter-domain traffic through two security gateways [1]

In a security domain,' network entities exchange traffic with the SEGs of the domain using IPsec'. [1] Figure 2-8 shows that security view SEGs are treated as another network entity within the domain. The interface between SEGs from different domains is Za and the interface between network entities and SEGs is the Zb interface. The IMS use the Zb interface to protect the IMS signaling plane.

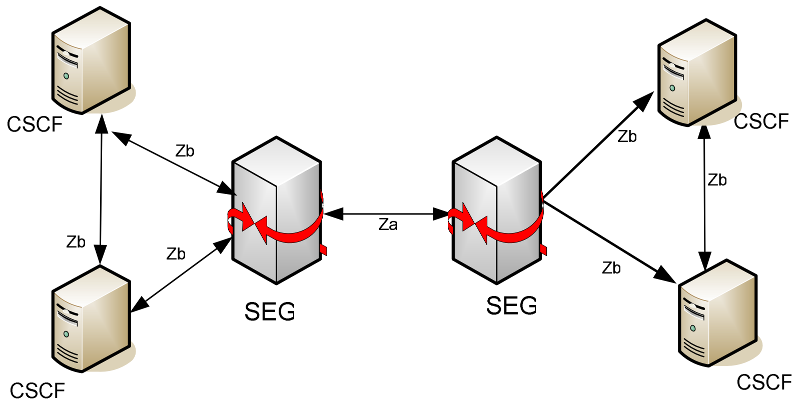


Figure 2-8 Za and Zb interfaces [1]

2.1.2.10 Protocols and interfaces

The Diameter protocol was defined by IETF that performs Authentication, Authorization, and Accounting (AAA) in the IMS and NGN.
The Diameter protocol provides the following facilities:

- *Connection and session management*
- *User authentication and capabilities negotiation*
- *Reliable delivery of attribute value pairs (AVPs)*
- *Extensibility, through addition of new commands and AVPs*
- *Basic accounting services*

Session Initiation Protocol (SIP) is a IP phones/Multimedia Session protocol based on text coding which is defined by IEIF.[9] It is used to control multimedia sessions between users, it can set up, modify and stop the multimedia sessions.
SIP protocol is used in User Agent and Server systems. It supports user's mobility, uses Client-Server solutions in HTTP protocols and it can combine many other IETF protocols. Besides, forking in SIP protocol makes SIP easily carry out the service. [10]

Cx, Dx, and Sh interfaces are based on the Diameter protocol. The Cx interface is between I-CSCF and HSS or between S-CSCF and HSS. It's used to exchange location information and authentication information, or to authorize a user to access the IMS, or to download changes in the user data stored in the server. And the Dx interface is used by CSCF to locate the HSS serving the subscriber. The Dx interface is between I-CSCF or S-CSCF and SLF.
The Sh interface is used by the AS (Application Servers) or OSA/Parlay Gateway to interface to the HSS. The Sh interface can operate in two modes: Data Handling and Subscriptions/Notification. [7] It is used to download user data from HSS or store user data into HSS. And, it allows server and client 'to request and send notifications on changes on user data'. [3GPP 01]

Table 2-1 Interfaces description [Wikipedia 01]

Interface Name	**IMS entities**	**Protocol**	**Description**
Gm	UE, P-CSCF	SIP	Used to exchange messages between UE and CSCFs
Mw	P-CSCF, I-CSCF, S-CSCF	SIP	Used to exchange messages between CSCFs
ISC	S-CSCF, I-CSCF, AS	SIP	Used to exchange messages between CSCF and AS
Mg	MGCF-> I-CSCF	SIP	MGCF converts ISUP signaling to SIP signaling and forwards SIP signaling to I-CSCF
Mi	S-CSCF->BGCF	SIP	Used to exchange messages between S-CSCF and BGCF
Mj	BGCF->MGCF	SIP	Used to exchange messages between BGCF and MGCF in the same IMS network
Mk	BGCF->BGCF	SIP	Used to exchange messages between BGCFs in different networks
Mr	S-CSCF，MRFC	SIP	Used to exchange messages between S-CSCF and MRFC
Cx	I-CSCF, S-CSCF, HSS	Diameter	Used to communicate between I-CSCF/S-CSCF and HSS
Dx	I-CSCF, S-CSCF, SLF	Diameter	Used by I-CSCF/S-CSCF to find a correct HSS in a multi-HSS environment
Sh	SIP AS, OSA,SCS, HSS	Diameter	Used to exchange information between SIP AS/OSA SCS and HSS
Dh	SIP AS, OSA, SCF, IM-SSF, HSS	Diameter	Used by AS to find a correct HSS in a multi- HSS environment
Go	PDF, GGSN	Diameter (Rel6+)	Allows operators to control QoS in a user plane and exchange charging correlation information between IMS and GPRS network
Gq	P-CSCF, PDF	Diameter	Used to exchange policy decisions-related information between P-CSCF and PDF

2.1.3 Open IMS

The IMS playground@FOKUS is an open technology test field. This playground works to implement existing and new IMS standards; all major FOKUS implemented IMS components, i.e. CSCFs, HSS, MG (Media Gateway), MRF, Application Servers and so on, and integrate them into a single environment. It also provides different service platforms, such as “Open Service Access (OSA)/Parlay, JAIN Service Logic Execution Environment (SLEE), Web services/Parlay X, SIP Servlets, Call Processing Language (CPL)”. [2]

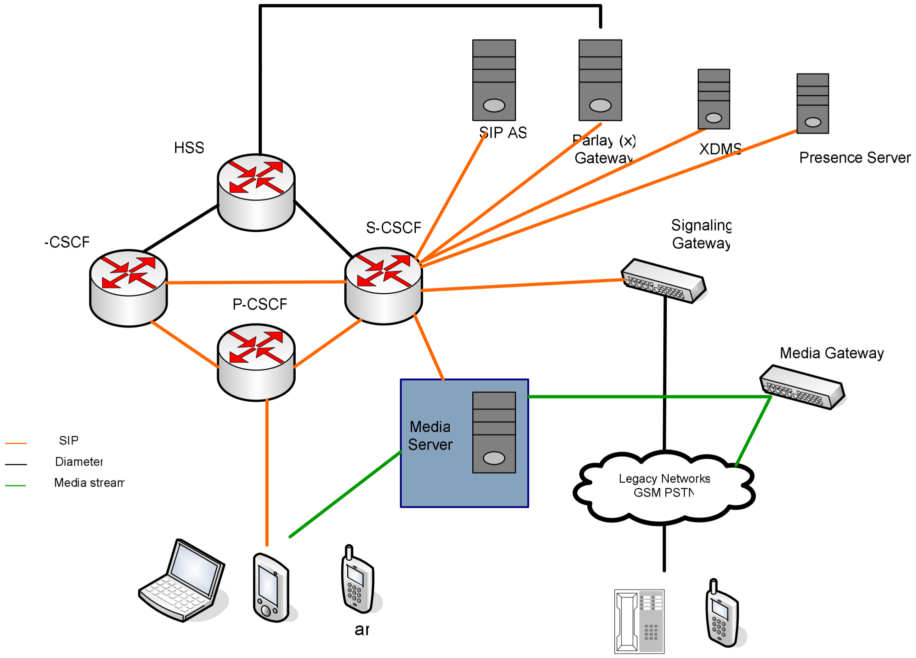


Figure 2-9 Overview of IMS playground@FOKUS [FOKUS 06]

Open Source IMS Core System [FOKUS 02] is an IP Multimedia System for test. It was created by the Fraunhofer Institute FOKUS. They point out that this Open Source IMS Core System is not intended to become or act as a product in a commercial context. Its sole purpose is to provide an IMS core reference implementation for IMS technology testing and IMS application prototyping for research purposes, typically performed in IMS test-beds. This target has also motivated the decision to use open source software (i.e. SER based on GPL).

Since IMS has already been used gradually and much efforts has been put forwards it, the main efforts now is for developing services. While there are already many Open Source projects established in the plain VoIP area for SIP clients, proxies, stacks and tools around the IETF sip standards, there are currently practically no Open Source projects with specific focus on the IMS.

The Fraunhofer Institute FOKUS focuses on Open Source software which is a flexible and extendable solution.

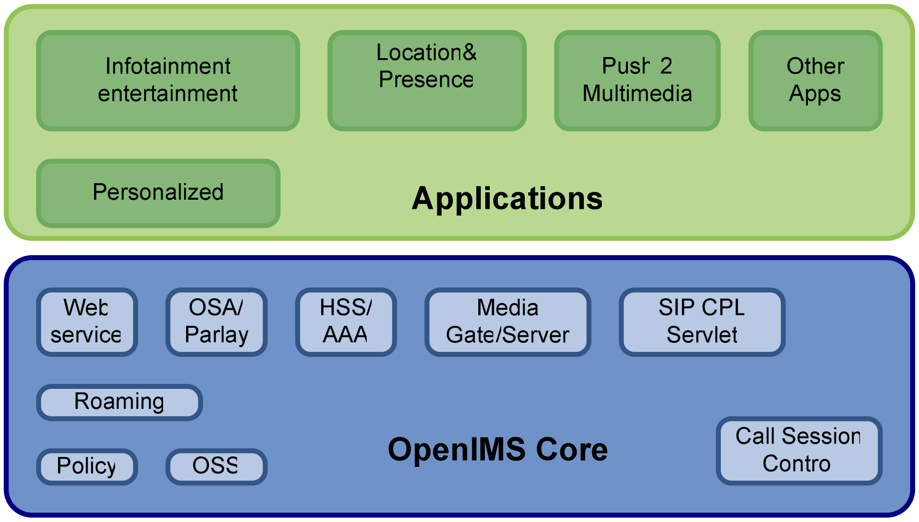


Figure 2-10 OpenIMS testbed at FOKUS [2]

We could see clearly from the following figure 2-11 that some of the key components are the same in both IMS and OpenIMS, but some are missing in OpenIMS, and the SIP2IMS gateway only exist in OpenIMS.

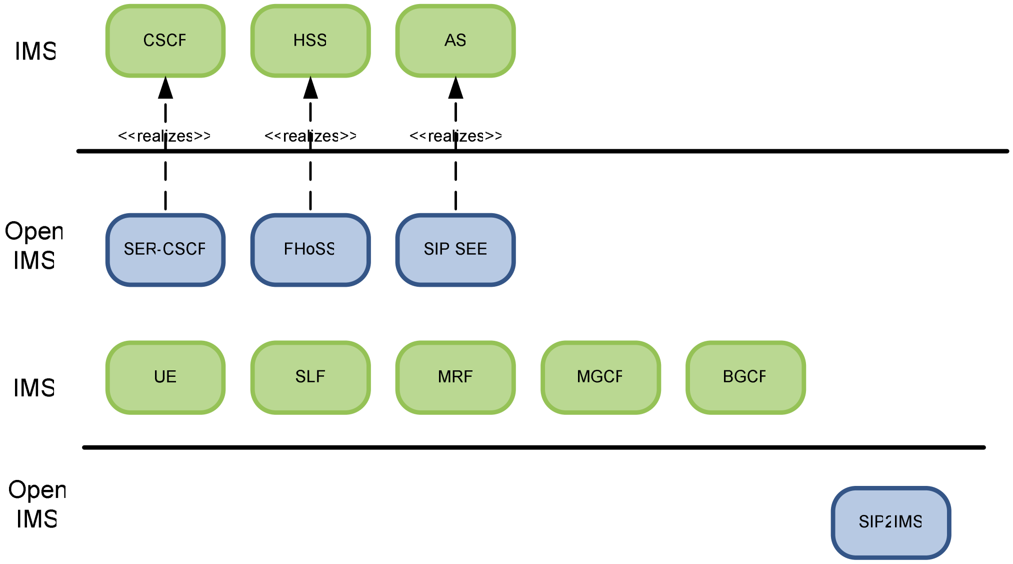


Figure 2-11 Comparisons of IMS and OpenIMS

2.1.4 Asterisk

Asterisk is an open source software implementation of a private branch exchange (PBX) which was created by Mark Spencer of Digium. [Wikipedia 03]

Asterisk [http://www.asterisk.org/] runs on OpenBSD, FreeBSD, Mac OS X and Sun Solaris. It supports Voice over IP protocols, such as SIP, H.323, IAX (inter-Asterisk eXchange) and MGCP (Media Gateway Control Protocol). Thus, Asterisk can interoperate with many SIP telephones and Asterisk PBXes. Asterisk contains many features including voice mail, conference calling, interactive voice response and automatic call distribution. All these made Asterisk a very popular software implementation of PBX.

The current release version is Asterisk 1.4.1 as of March 6.2007.

2.2 IMS Application

How does IMS provide services? We discuss some of the most significant services that will be provided by IMS.

2.2.1 Presence Service in the IMS

For presence service, the clients can send information that shows their status to the server and the server will inform availabilities or willingness of those clients to other users in the group.

Presentities choose what information they want to publish, and when watchers get the information, they decide how and when to communicate with the presentities. What's more, not only end-users but also other services can get the presence information. 'For example, an answering machine server is interested in knowing when the user is online to send them an instant message announcing that they have pending voicemails stored in the server'. [1] Therefore, the presence service is to be considered as the foundation of all the services.

As we see, Figure 2-12 'depicts the presence IMS architecture that maps the already defined roles in presence to existing functional entities in the IMS.'[1] In IMS, terminal has two roles: watcher and Presence User Agent (PUA). We refer the Presence Agent (PA) as a Presence Service (PS), and we also implemented Resource List Server (RLS) as Application Server (AS). From the figure we see that most of the interfaces are still exist in IMS SIP or Diameter interfaces, thought they change their name to start with 'P'. Compared to 3GPP the IMS architecture, there are 2 new interfaces in presence IMS architecture, which are Pen interface and Ut interface. Pen interface allows an AS work as a PUA, and Ut interface can be used between any AS and IMS terminal. [1]

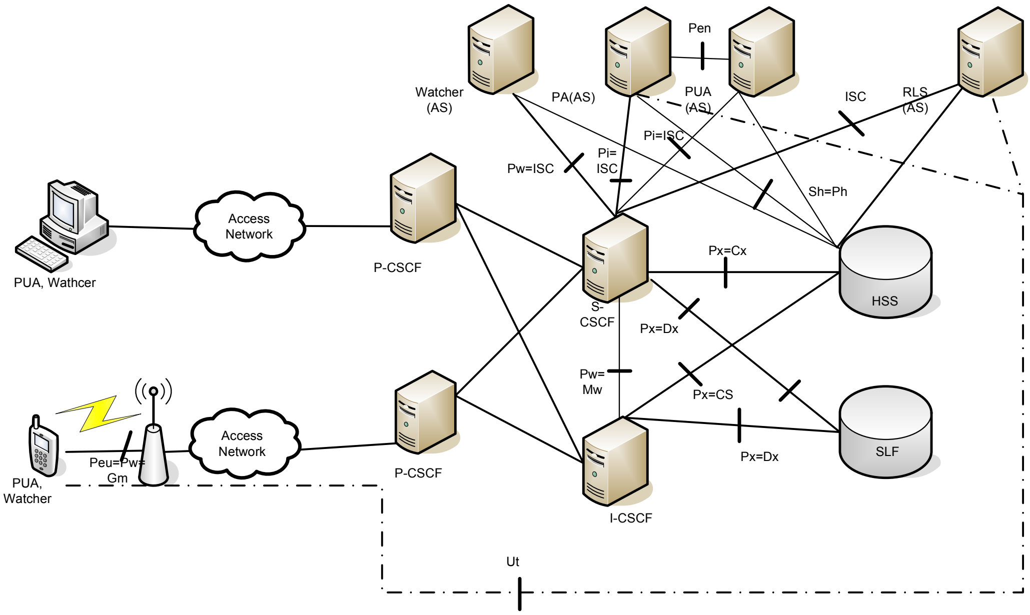


Figure 2-12 SIP-based presence architecture in the IMS [1]

2.2.2 Instant Messaging

A group of people can communicate with each other over a network, and as we know, instant messaging is one of the ways to achieve. A user can choose targets to send messages to, depending on the presence status of other users. [Wikipedia 02]

If the instant messages are stand-alone messages then they will be sent in pager-mode, and if the messages belong to some existing session, they will be sent in Session-mode. [1]

2.2.3 SmartMessenger

The SmartMessenger 'is a multi-channel message delivery service.' [2] Users can communicate with each other by sending SMS and instant messages through the Parlay interface.

The SmartMessenger is also a Click2Dial application. Messages can be read by a Text-to-Speech (TTS) on the phone, and the SmartMessenger uses the Call Control interface to make calls between VoIP and/or PSTN. [2]

2.2.4 IMS Push-to-Talk

Push to talk (PTT) provides one-to-one and one-to-many high-margin voice service for both business users and private consumers. IMS Push to talk is a good example of PTT, and includes key features of PTT, such as presence, one-to-many communication. It also has some particular services, for example, filtering of incoming calls or do-not-disturb status. Those services make IMS PTT a more intelligent and convenient service, and at the same time, it is easy to use, so today IMS PTT has a large market with a variety of users.

Ericsson IMS Push to talk solution is based on the Open Mobile Alliance, Push to Talk over Cellular (OMA PoC) standard, which defines a rich set of features such as presence and over-the-air provisioning. [6]

2.3 Related works

[8] is the most important reference for us. Our project will be done basic on this paper. In this paper, State-Of-the-Art Enterprise SIP Solutions were identified first. And then, they design a solution for SIP/VoIP enterprise. The final problem involves research on migrating enterprise SIP/VOIP solutions towards IMS.

In order to approve their concept and research based on the SIP technology, they established the “HiA-Teleca SIP/VOIP test-bed”. It is now a common research platform for HiA and auSystems (Teleca) to build and develop competence in VOIP/SIP/IMS area. It will contribute development in future “All-IP” products and services. The test-bed supports SIP session handling, IPvoice, voicemail, conference calls, PC software and SIP hardware clients, PSTN connectivity via a SIP ISP, presence, TAPI for Microsoft Office integration. This part is very useful for us, because we also need to implement our solution basic on the test-bed afforded by auSystems with Open Source IMS core.

Except design a SIP/VoIP enterprise solution, this paper also involves finding the way to migrate the existing SIP/VoIP solution towards IMS. Based on their solution, they find four possible ways or enterprise SIP/VoIP solutions to integrate legacy and future cellular phones using IMS.

Enterprise SIP/VoIP to IMS migration alternatives are:

1 Forking is the simplest solution and has no requirement on clients or servers.
2 The client based solution will give access to more features but requires an advanced SIP client and more configuration knowledge.
3 The presence solution is a network based solution. It handles all necessary location updates without involving the client. It supports legacy phones.
4 Link registration solution is network based, too. It directly links the registration

procedure by using “subscriber registration update service” provided by IMS-enabled operator. It may appear in the later phase of IMS migration.

3 Installation and Configuration of an OpenIMS

Since auSystems and HiA have already established a research testbed for Enterprise VOIP solutions based on Open Source systems which contains all the necessary functions of a typical enterprise installation. As a starting point, we plan to extend the existing testbed with a new IMS sub-domain based on the “Open Source IMS Core”, so that we can have the environment of OpenIMS to test and evaluate how it works.

Considering what we learned in the State-of-the-Art section, we have some understanding about how to set up this testbed, we need to do some installations and configurations of the key components of OpenIMS which are FHoSS, CSCFs, SIP2IMS, etc. And with the leading of Installation-Guide, we can implement it step by step. The figure 3-1 shows the main steps of establishing OpenIMS Testbed:

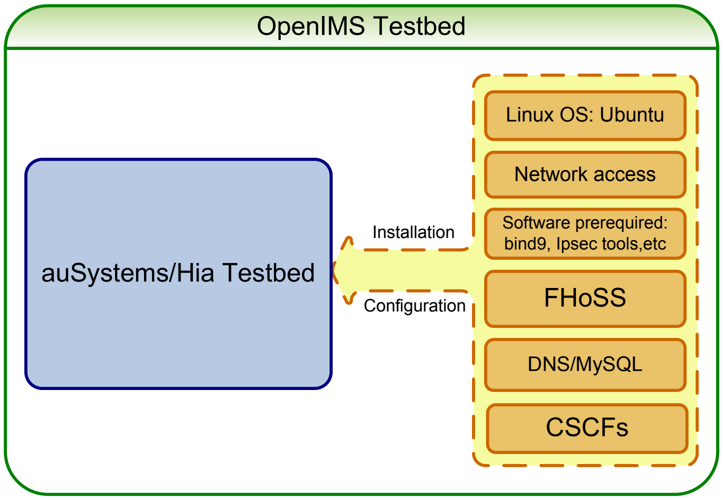


Figure 3-1 Setting up of OpenIMS testbed

3.1 Installation of Prerequisites

As the figure 3-1 shows, before installing the key components of OpenIMS, the testbed requires prerequisites in hardware, network and software.

For the environment of hardware, we need a current Linux desktop class machine. Here, we use Ubuntu 6.10 [http://www.ubuntu.com/] as our Linux Operating System. Besides, in order to get ultimate performance, we need to add several gigabytes of RAM and as many CPUs/Cores as needed.

As for the network access aspect, Inter-domain NAT is not something we are considering. Therefore, a public IP address is enough. The software should include the following requirements:

In order to ensure the system will work well, 100 MBytes of disk space are needed. Subversion is required to be installed so that we could find fresh code. GCC3/4, JDK1.5, ant which uses for Java development is needed to be installed firstly. In our case, we use MySQL as a database management system (DBMS) which is supported by FHoSS, I-CSCF and other functions which require a DBMS. Developed libxml2 and libmysql are required. Linux kernel 2.6 and ipsec-tools which is used to setkey are needed to use IPSec SAs. Besides, we use bind 9 as our name server. And we choose Firefox which can connect to the box as our browser.

After we make sure that all the requirements have already fulfilled, we can begin to install and configure the FHoSS, which is the core component of OpenIMS.

3.2 Installation and Configuration of FHoSS

The Open Source IMS Core would be incomplete without a Home Subscriber Server. FOKUS developed its own prototype HSS, the FOKUS HSS (FHoSS) which is entirely written in Java and based upon Open Source software. As its purpose in the Open Source IMS Core is that of a database, the FHoSS is targeted mainly towards conformance rather than performance. It is mostly the glue between a Database Management System and the Diameter interfaces with the CSCFs and IMS application layer.

FHoSS stores the IMS user profiles and provides the location information of the user. Additionally, it is also a web-based management console. The architecture of FHoSS below illustrates the main components and the interfaces that are used in.

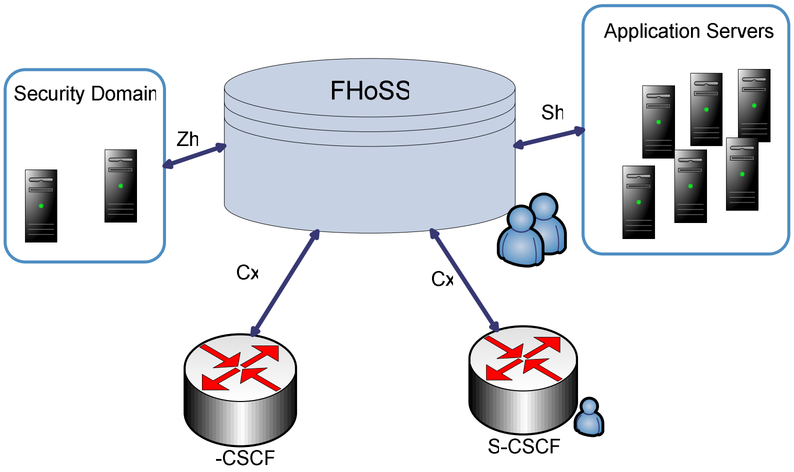


Figure 3-2 Architecture of FHoSS

From the figure 3-2, we can see that the entities that communicate with the FHoSS are the application server (AS) that hosts and execute services in the IMS environment, security domain and the Call State Control Function servers (CSCF). And the interface layer describes the external behavior of the HSS. FHoSS stores the user files via Sh interface point to Application Servers, and it communicates with CSCFs via Cx interface. Besides, it connects with Security Domain via Zh interface.

It extends the Open Source project ‘Open Diameter’ and implements the complex functionalities of the Cx and the Sh reference points based on Java. [2] The FHoSS stores the IMS user profiles along with authentication and authorization information and provides user status information along with a notification service via the Sh reference point to Application Servers. Additionally, it supplies S-CSCFs with the current filtering information on a user base over the Cx reference point.

3.2.1 Interface layer

The core of the FHoSS is the HssDiameterStack. It uses the DiameterPeer to send requests to other entities and retrieves the requests and responses via CommandListener. There are three interfaces used in FHoSS, who are Sh,Cx,and Zh.

They can be found in the de.fhg.fokus.cx, de.fhg.fokus.sh and the de.fhg.fokus.zh package. For each interface there is a direct implementation. This implementation can be found in the de.fhg.fokus.hss.server and subsequent packages. For every method of the interface there is a related operation class in the op packages.

These operations will be called by the interface implementations and will call in turn the diameter commands. As you can read in the 3gpp specification, every interface method is mapped to a number of Diameter requests. These requests are realized by implementing the CommandAction and CommandListener classes. You will find a action for every command which is sent through the diameter peer, in the de.fhg.fokus.hss.diam.cx, de.fhg.fokus.hss.diam.sh and de.fhg.fokus.hss.diam.zh packages. For every command which will be received by the Hss one listener exists. The command listener will dispatch the requests to the interface methods.

3.2.2 Data Access Layer (DAL)

The operational data of the FHoSS is stored in a database. The Hibernate persistence framework was used to build a data access layer that is used to change the database system. The related data classes can be found in the de.fhg.fokus.hss.model package.

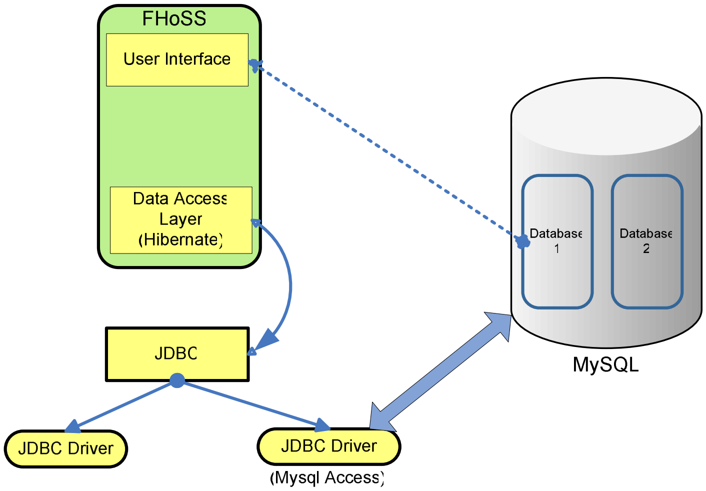


Figure 3-3 structure of databases

The user data is kept inside a MySQL database. However, there is a Data Access Layer (DAL) which is based on JDBC, and then any database can be used as long as we have a JDBC-driver for it. In our configuration, we use MySQL as the back-end database engine. Since MySQL is far from optimal for this database, we will consider the other database LDAP. We did some checking and there actually exists a JDBC-driver for it, and Hibernate supports for LDAP. Therefore, in theory it should be possible to also use LDAP here [http://www.openldap.org/jdbcldap/]. If time permits, we will test with both MySQL and OpenLDAP as the back-end database.

3.2.3 GUI

To manage and maintain the FHoSS, a web based management interface is provided. This provides a clear structure and separation of logic and GUI related tasks. The implementation of the GUI logic can be found at de.fhg.fokus.hss.form and package de.fhg.fokus.hss.action. The rendering is done by several Java Server Pages which can be found in the src-web folder.

3.2.4 Installation steps

3.2.4.1 Get source code

As a start, we find fresh code at http://svn.berlios.de/svnroot/repos/openimscore where we can get source code - FHoSS/trunk. The source code is pre-configured to work from a standard file path. Firstly, we create the directory ‘OpenIMSCore’ in ‘opt’ and go there. After that we create a new directory ‘FHoSS’ and check if the HSS is there:

```
mkdir /opt/OpenIMSCore

cd /opt/OpenIMSCore

mkdir FHoSS

svn checkout http://svn.berlios.de/svnroot/repos/openimscore/FHoSS/trunk FHoSS
```

3.2.4.2 Compile

Before compilation, we must make sure we have a JDK >=1.5. And then, we use Ant to build and install the FHoSS.

To build the FHoSS we must first execute the gen target, to generate the data classes.

```
ant gen
```

After that use compile to build the binaries

```
ant compile
```

Installation is also done using ant. With the deploy target we can install the FHoSS

```
ant deploy
```

3.2.4.3 Configure the environment

We need a database in order to use the FHoSS. We can find two sql scripts for the MySQL database in the root directory of our installation. Use these scripts to create a new MySQL database and to populate it with default values.

- to create the database and the tables:

```
mysql -u admin -p <hssdb.sql
```

- to create two demo users and initial values for service profiles:

```
mysql -u admin -p <userdata.sql
```

After creating the database, we check that the database isin there and accessible.

3.2.4.4 Configure the IMS core

By now MySQL is working and we need to take a look at the configuration files in FHoSS/deploy/

We read these files and modify some parameters as appropriate for our installation. Configure the required configuration files located in the root of the deployment directory. [Open IMS Core 02]

- "DiameterPeerHSS.xml": we need modify the peer configuration here: like the FQDN, Realm, Acceptor Port or Authorized identifiers.
- "hibernate.properties": what we should configure are the main properties for hibernate; implicitly is configured to connect to the mysql on the localhost (127.0.0.1:3306). The most relevant properties are: hibernate.connection.url=jdbc:mysql://127.0.0.1:3306/hssdb hibernate.connection.username=hss hibernate.connection.password=hss
- "hss.properties": Specify configuration like: on which address the tomcat is listening (e.g. host=localhost) and the relative path of the web interface of the FHoSS. (e.g. appPath=/hss.web.console). Other parameters like: operatorId, amfId or defaultPsiImpi can be specified here.
- "log4j.properties": Contains configuration for the logger. The most relevant things here are the output file path of the logger and the level of logging.

3.3 Installation and Configuration of CSCFs

As the extension of IMS, OpenIMS CSCFs use SER (SIP Express Router) to perform all CSCFs. SER is a worldwide recognized SIP reference implementation. SER can be used as SIP registrar, proxy or redirect server and is capable of handling many thousands of calls per second [4]. OpenIMS CSCFs are required to maintain as much of SER’s performance as possible. So that OpenIMS could have a flexible exploitation. Open IMS CSCFs it consists of four main components: P-CSCF, I-CSCF, S-CSCF, SIP2IMS Gateway. The figure 3-4 illustrates how the main components are located and the interfaces that are used in them.

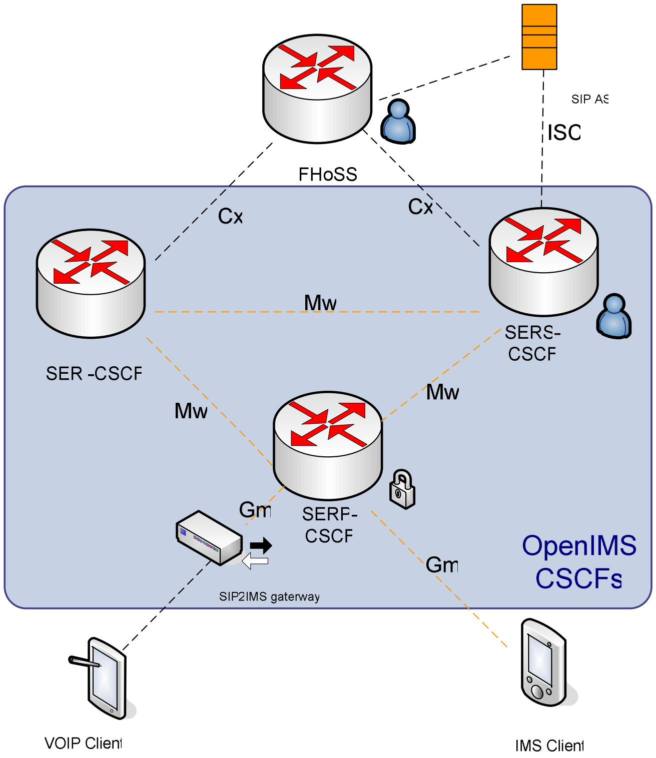


Figure 3-4 Architecture of OpenIMS CSCFs [FOKUS 02]

From the figure 3-4, it is obvious that the P-CSCF is the initial entity from the mobile terminals to OpenIMS through a SIP2IMS gateway. P-CSCF uses Gm interface to exchange messages between user endpoint (UE) and CSCFs. SIP affords this process. And only the registered endpoints can insert the message to IMS. I-CSCF and S-CSCF are located in Home Domain. Both of them connect with HSS using Diameter over Cx interface. As to the communication of P-CSCF, I-CSCF and S-CSCF, SIP is used to exchange messages over Gm interface. The general flow is as follows: the User Agent, who wants to access OpenIMS, should be registered through P-CSCF; and then, P-CSCF finds the related Home Domain and also sends the message to I-CSCF; I-CSCF forwards to HSS to select the right S-CSCF.

3.3.1 Modules of OpenIMS CSCFs

The modules in CSCFs can parallel process and supplementary state information can be kept. Besides, there is a special focus towards scalability for both load distribution and data quantity. The main modules of OpenIMS CSCFs are: [Open IMS 03]

- The CDiameterPeer Module (cdp)

This module is used for realm routing, we have to specify each time the FQDN (Fully Qualified Domain Name) of the destination host when it is used. It is supposed to allow efficient Diameter communication to and from SER.

- The IMS Service Control Module (isc)

isc module is supposed to provide support for the ISC interface between the Serving-CSCF and the Aplication Servers. To use it, we need the the Serving-CSCF Module (scscf) loaded because it uses the registrar in there for Initial Filter Criteria storage.

- The Proxy-CSCF Module (pcscf)

pcscf module is supposed to provide the functionality required for an Proxy-CSCF.

- The Interrogating-CSCF Module (icscf)

icscf module is supposed to provide the functionality required for an Interrogating-CSCF. We need the cpd module loaded to use it. This is because this module communicates using Diameter with the Home Subscriber Server over the IMS_Cx interface.

- The Serving-CSCF Module (scscf)

This module is supposed to provide the functionality required for a Serving-CSCF. It is the same as icscf module. To use it, we need the cpd module loaded. The reason is it communicates using Diameter with the Home Subscriber Server over the IMS_Cx interface.

- The SIP-to-IMS Gateway Module (sip2ims)

It provides NAT Helper for SIP clients. This module is supposed to provide the translation capabilities to enable normal SIP User Endpoints to use the Open IMS Core through the Proxy-CSCF.

To accelerate testing and to integrate with SIP UEs and test tools, a gateway that helps SIP traffic to work in an IMS environment was required. The SIP-to-IMS gateway performs this adaptation tasks. At the moment it translates between MD5 and AKAv1-MD5 authentications and helps with special headers. The SIP2IMS Gateway can be been regarded as the helper module of the Open source IMS Core project to allow the verification of services with current state-of-the art VoIP clients. Yet, due to its gateway nature, it filters much of the IMS advantages and should not be regarded as a full blown IMS solution.

The Open Source IMS SIP2IMS Gateway allows transformation of IETF SIP messages to IMS conformant messages. It translates MD5 authentication to IMS AKA authentication.SIP2IMS can also enable developers to access core elements and to trial multimedia services by using a non-IMS VoIP client.

3.3.2 Installation steps

3.3.2.1 Get the Source Code

On the same page http://svn.berlios.de/svnroot/repos/openimscore where we get the code of HSS, we need to get the code of CSCFs - ser_ims/trunk. The source code is pre-configured to work from a standard file path. We have already had the directory /opt/OpenIMSCore, and now we need to create new directory ser_ims under it, using ‘mkdir’ command. After that, we need to check if the CSCFs are there.

```
mkdir ser_ims

svn checkout http://svn.berlios.de/svnroot/repos/openimscore/ser_ims/trunk ser_ims
```

3.3.2.2 Compile

In this step, we are asked to make libs install all in ser_ims. This step takes some time to finish and we have to promise that all the prerequisites had been installed.

```
cd ser_ims

make install-libs all

cd ..
```

3.3.2.3 Configure the Environment

All the installation examples are configured to work only on the local loop back and the default domain configured as “open-ims.test”. Therefore, we replace 127.0.0.1 with our IP address and replace the home domain with our own one. Additionally, DNS needs to be configured in this step as well. We could find a sample DNS zone file and copy it to our bind configuration directory. And then, we edit named.conf and insert the file there. After that, we restart the name server to test if the names are resolvable. We need to also install icscf.sql and sip2ims.sql into MySQL in this step. Checking out if the databases are in there and accessible cannot be forgotten.

```
mysql -u root -p -h localhost < ser_ims/cfg/icscf.sql
```

3.3.2.4 Configure the IMS Core

By now, MySQL and DNS can work well. To configure the IMS Core, we just need to copy several files to /opt/OpenIMSCore. The files that need to be copied are: pcscf.cfg, pcscf.sh, icscf.cfg, icscf.xml, icscf.sh, scscf.cfg, scscf.xml, scscf.sh, sip2ims.cfg, sip2ims.sh. What should be noticed is that we need to edit these files to our own preferences.

```
cp ser_ims/cfg/*.cfg .

cp ser_ims/cfg/*.xml .

cp ser_ims/cfg/*.sh .
```

3.4 Start the components

The next step after installation is to start each module of CSCFs: pcscf.sh, icscf.sh, scscf.sh and sip2ims.sh at the same time. Besides, startup.sh is also used to start the FHoSS on a Linux/UNIX system. Note whether if the JAVA_HOME variable is set correctly.

```
./pcscf..sh

./icscf.sh

./scscf.sh

./sip2ims.sh

./startup.sh
```

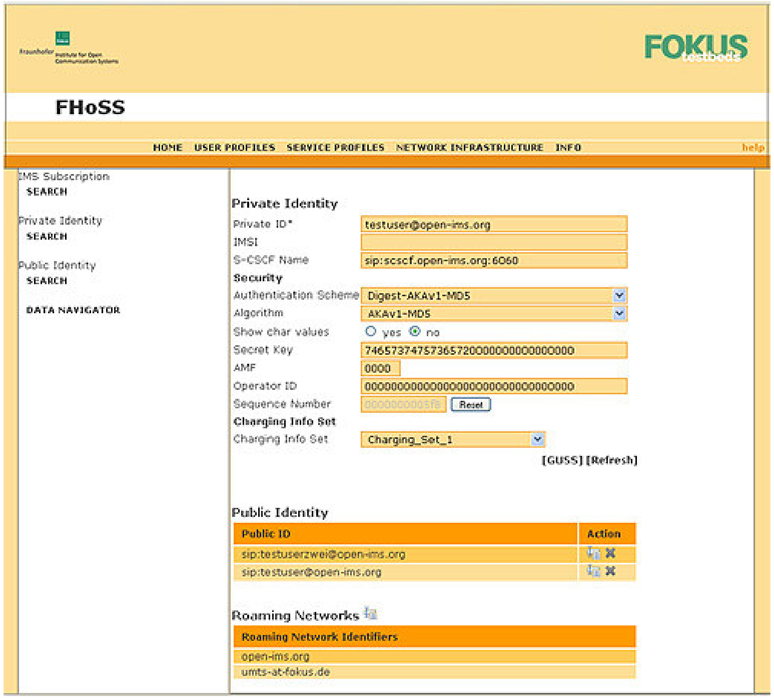
Make sure that each component can connect to HSS. And in this process, we could see periodical log messages with the content of the registrar and with the opened diameter links by default.

After that, we can check the web interface on http://localhost:8080/

3.5 Configure Subscribers

We can access the management console using a web browser on http://localhost:8080/hss.web.console. The operator can enter in this website as ‘hssAdmin’ to read and manage the console. But the ‘hss’ can be used just as a reader. Both of the two users have the same code which has already been given. [Open IMS Core 02]

After we login in as ‘hssAdmin’, more information can be found. FHoSS comes provisioned with a couple of sample users by default. One user is ‘alice’, whose Public User Identity is: alice@open-ims.test. And the other is ’bob’, whose Public User Identity is: bob@open-ims.test. In addition to this obvious information, some concealed information can be read as well, such as each users’ Private User Identities. With this information, we could create a new subscriber and its Private User Identity and Public User Identity. There are some buttons to click on. The new subscribers that we created are ‘fei’ and ‘li’. It is easy to create new users by using FHoSS User Profiles. The figure 3-5 shows how the web-interface looks.



Private Identity

Private ID*: testuser@open-ims.org
IMSI:
S-CSCF Name: sip:scscf.open-ims.org:6060

Security

Authentication Scheme: Digest-AKAv1-MD5
Algorithm: AKAv1-MD5
Show char values: ○ yes ● no
Secret Key: 7465737475736572000000000000000
AMF: 0000
Operator ID: 00000000000000000000000000000000
Sequence Number: 0000000005f8 Reset

Charging Info Set

Charging Info Set: Charging_Set_1

[GUSS] [Refresh]

Public Identity

Public ID	Action
sip:testuserzwei@open-ims.org	✖
sip:testuser@open-ims.org	✖

Roaming Networks

Roaming Network Identifiers
open-ims.org
umts-at-fokus.de

Figure 3-5 FHoSS User Profile

As it is mentioned in State-of-the-art, each IMS User is assigned at least one Private and one or more Public User Identities. IMS Private User Identity uses the format of NAI which is:

username@operator.com.

And the Public User Identity uses the format of SIP URI or TEL URI that can be presented as:

sip:first.last@operator.com or tel: +1-212-555-0293.

Based on this, when we create the Private User Identity for the new subscriber, we followed NAI format which is: fei@open-ims.test and the Public User Identity is: sip: fei@open-ims.test.

In order to test, we may add more Public User Identities for each subscriber after we finish the whole installation. And we may try the situation extended in 3GPP R6, that is a subscriber is assigned more than one Private User Identities and one Public User Identities can be used in for Private User Identities.

This step is continued with configuration Subscribers of FHoSS. We have already created a new Subscriber and its Private User Identity and its Public User Identities. And now, we can find that the SIP-to-IMS Gateway comes provisioned with the same couple of sample users. One is alice@open-ims.test, the other is bob@open-ims.test. We could use these in the sip2ims.credentials mysql table at first. And we can also insert new ones.

The details of the new subscribers' configuration:

```
Fei:
Private Identity: fei@open-ims.test
Secret Key: fei
OP: 0x00...0
AMF: 0x00...0
Use of Anonimity Key: enable
Public Identity 1: sip:fei@open-ims.test
Public Identity 2: tel:+4797430726
```

```
Li:
Private Identity: li@open-ims.test
Secret Key: li
OP: 0x00...0
AMF: 0x00...0
Use of Anonimity Key: enable
Public Identity 1: sip:li@open-ims.test
```

4 Functionality evaluation of OpenIMS

Basic on the OpenIMS Testbed that we established in the previous section, what we focus on in this section is testing to evaluate if each component of OpenIMS conforms to the 3GPP specifications. That is to say, we have to do some experiments to make sure all the necessary functionality will be available to us. We will select the SIP Client or the IMS Client firstly, and then use these to make call session connected with OpenIMS to analyze the log message showing in Wireshark. We will use a table to show if the functionalities conform to 3GPP Release 6. By the way, we choose 3GPP Release 6 as the OpenIMS specification, and compare all functionalities of OpenIMS with it.

4.1 User Equipment

As the prerequisites, we start our testing with configuration of several SIP/IMS Clients. We choose to install several SIP/IMS clients to find one that can support all functions and also are easy to operate.

4.1.1 SIP Hardphone

A SIP Hardphone is an IP hardphone using SIP Protocol to register as a user. We use Grandstream GXP-2000, who was the winner of internet telephone in 2005, as one of the SIP Clients.

4.1.1.1 Grandstream GXP-2000

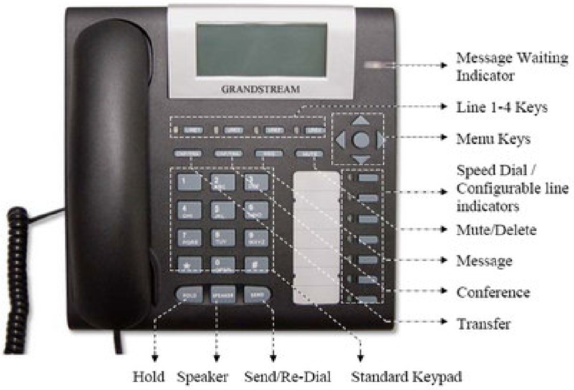


Figure 4-1 Grandstream GXP-2000

Grandstream GXP-2000 is a next-generation SIP Enterprise Phone which is designed for small to large business enterprises.

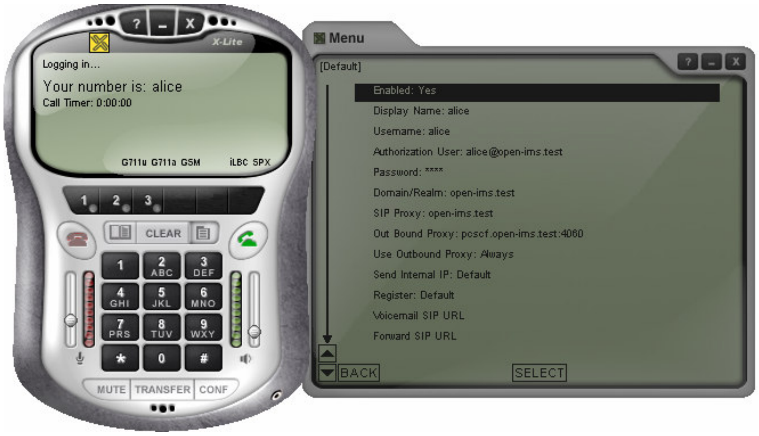
Grandstream GXp-2000 has open standards compatible including SIP 2.0, TCP/IP/UDP, PPPoE, RTP/RTCP, and SRTP by SDES, HTTP/HTTPS, ARP/RARP, ICMP, DNS, DHCP, NTP/SNTP, and TFTP. It has four individual lines and up to 11 lines indicators and dual 10/100 Mbps Ethernet ports with auto detection. And its graphical LCD can display up to 8 lines and 22 characters per line. Grandstream GXP-2000 also supports Caller ID display, and Power-over-Ethernet. It has very good audio quality with advanced digital signal processing and silence suppression

And there are two ways available for operator to configure it: one is using the phone interface, the other it to use the phone's web interface.

4.1.2 SIP Softphone

4.1.2.1 X-Lite

X-Lite [http://www.xten.com] is CounterPath's next-generation SIP based softphone. It works over IP-based system. Users can use it within an enterprise LAN or VoIP service provider network.



Logging in...
Your number is: alice
Call Timer: 0:00:00

[Default]

Enabled: Yes
Display Name: alice
Usemame: alice
Authorization User: alice@open-ims.test
Password: ****
Domain/Realm: open-ims.test
SIP Proxy: open-ims.test
Out Bound Proxy: pcscf.open-ims.test:4060
Use Outbound Proxy: Always
Send Internal IP: Default
Register: Default
Voicemail SIP URL
Forward SIP URL

Figure 4-2 X-Lite softphone

As other softphones, X-Lite has all the standard telephone features, including Two lines, Mute, Do not disturb, Three-way audio and video conferencing and so on.
However, it also has some improved features and functions make X-Lite a popular softphone on the market.

- Instant messaging and presence using the SIMPLE protocol
- "Zero touch" configuration
- "Zero touch" detection to detect the bandwidth that a user's computer can access
- Support the RFC 3261 SIP standard, support for H.263, H.263 1998

4.1.3 UCT IMS Client

The UCT IMS Client [http://uctimsclient.berlios.de/] is a soft-phone designed especially for to use together with the Fraunhofer FOKUS Open IMS Core. It is run on a Linux-based operating system, and is very easy to configure and use. It support the AKA algorithm, support instant messages, the current version also support voice calls. Because it is still under improvement, there are some known bugs and the user interface is also very simple.

4.1.4 Configuration of SIP/IMS Client

As there are two defaults IMS UE by FHoSS, we use the existing UEs' information to configure SIP UE so that they can register by OpenIMS server successfully.

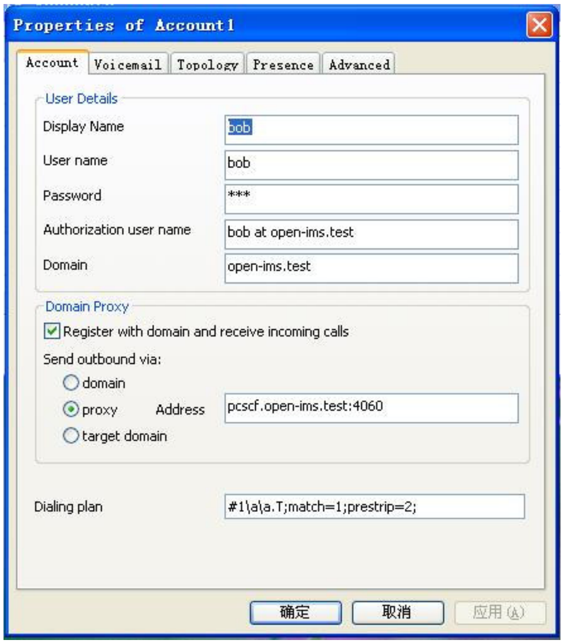
Configuration files of SIP UE

```
Alice:
User part of the SIP URI: alice
Host part of the SIP URI/Domain/realm: open-ims.test
Password: alice
Strict Outbound Proxy: sip:pcscf.open-ims.test:4060
```

The main components that we need to evaluate are at least: P-CSCF, I-CSCF, S-CSCF, FHoSS, SIP2IMS and the interfaces between them (Cx, Gm, Wm). Considering that the testing case could be quite complicated because of involving so many components, we will start our experiment with 'Registration at the IMS-level'.

4.1.4.1 Configuration of X-Lite Softphone

We installed X-Lite 3.0 in one computer, and use "fei" which is added by ourselves into the FHoSS to register. First we add a new account before using X-Lite to place or receive calls. Click "SIP Account Settings" in the menu and select the Enable checkbox for the desired account.



User Details

Display Name: bob

User name: bob

Password: ***

Authorization user name: bob at open-ims.test

Domain: open-ims.test

Domain Proxy

☑ Register with domain and receive incoming calls

Send outbound via:

○ domain

● proxy Address pcscf.open-ims.test:4060

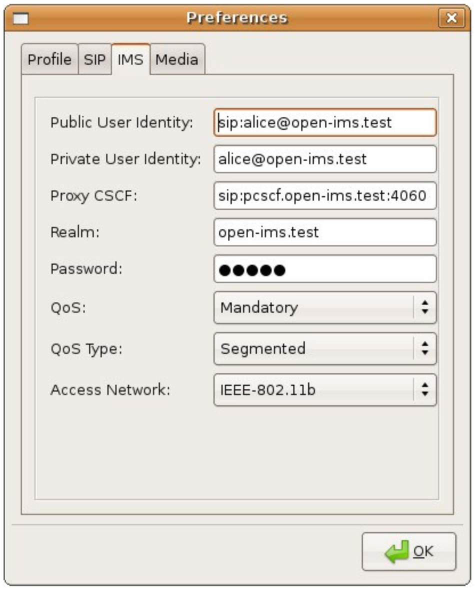
○ target domain

Dialing plan: #1\a\a.T;match=1;prestrip=2;

Figure 4-3 Setting up Account

4.1.4.2 Configuration of UCT IMS client

We installed the UCT IMS Client 1.0.3 in the Linux OS/Ubuntu as the IMS client. After installing, we used the “make” command from the src directory to compile, and then “./uctimsclient” is used to run the software. It’s much easier for us to use it because two defaulted users “alice” and “bob” have already been configured in it.



Public User Identity: sip:alice@open-ims.test

Private User Identity: alice@open-ims.test

Proxy CSCF: sip:pcscf.open-ims.test:4060

Realm: open-ims.test

Password: ●●●●●

QoS: Mandatory

QoS Type: Segmented

Access Network: IEEE-802.11b

.

Figure 4-4 Preferences of UCT IMS client

Considering that we don’t have QoS requirements, we set “QoS” to “None” for both alice and bob register.

4.2 Test tools

Wireshark[http://www.wireshark.org/] is the open source software which works as a network packet analyzer. It can capture network packets and try to display the packet data in detail. It can be used to examine what is going on inside a network cable, so that we can use it to examine kinds of problems, to debug protocol implementations, or to learn network protocol internals. In addition to these, Wireshark is also available for both Linux and Windows OS, and it is easy to use for searching and filtering packets on many criteria packets display based on filters. It is used to analyze each packet, the SIP protocol and the Diameter protocol in our project.

SIPp [http://sipp.sourceforge.net/] is the Open Source test tools used for performance testing of SIP protocol. SIPp is one of the best tools to do the performance testing because it can establish and release multiple calls with the INVITE and BYE methods. Besides, it can also read XML scenario files describing any performance testing configuration. It features the dynamic display of statistics about running tests, periodic CSV statistics dumps, TCP and UDP over multiple sockets or multiplexed with retransmission management, regular expressions and variables in scenario files, and dynamically adjustable call rates. It is also very useful to emulate thousands of user agents calling the SIP system.

Although SIPp is much powerful, it is not so easy to use. As the reason that it is not the software using window interface, the operator has to write commands to control the signaling procedure. The register-alice.xml and register.csv should also be written by the operator.

SIPp can be used to test many real SIP equipments like SIP proxies, B2BUAs, SIP media servers, SIP/x gateways, SIP PBX, etc. It is also very useful in emulating thousands of user agents calling the SIP system.

4.3 SIP signaling in OpenIMS

Session Initiation Protocol (SIP) is IP phone/Multimedia Session Protocol based on text coding which is defined by IETF. [11]

In SIP protocol, if A send request to B, then A is Client and B is server. Contrariwise, if B send request to A, then B is Client and A is server.

SIP used to control multimedia sessions between users. What it can do is set up, modify and stop the multimedia sessions. There are 6 ways to carry out those functions: [12]

- INVITE-- Sending a call via invite user.
- ACK--Using together with INVITE information to confirm the UAC has received

the finally response of INVITE request.

- BYE--User Agent uses this method to release call.
- CANCEL--Used to cancel an unfinished request, has nothing to do with completed request.
- REGISTER--User using this method to register the addresses in to server. User Agent sending REGISTER request to address when it starts working, in order to accomplish the registration to local server.
- OPTIONS--Used to inquire after available User Agent or Servers.

The figure 4-5 shows the different cases we will use in our test.

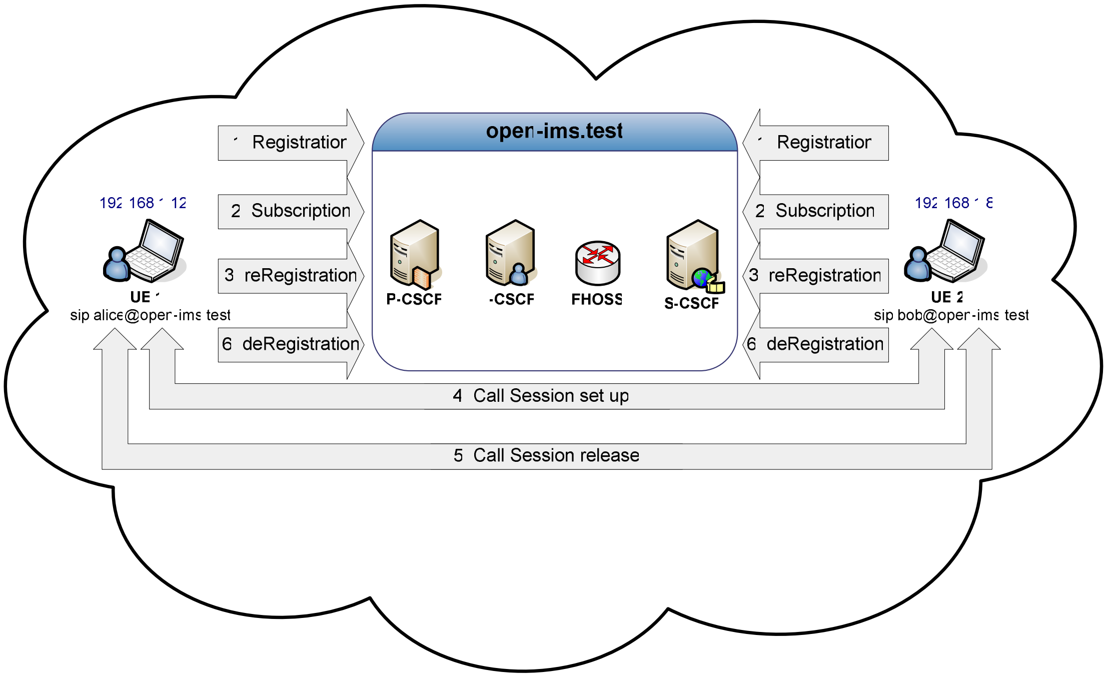


Figure 4-5 Test cases in evaluating OpenIMS

4.3.1 Registration Procedures

Before an IMS terminal begins any operation, there are several required prerequisites that have to follow. First thing is to establish an IMS service contract or subscription between the end-user and IMS service provider so that the end-user can be authorized when registering, making calls or doing other services.

After this establishment, IMS terminal needs to access to an IP-CAN (IP Connectivity Access Network) such as GPRS, ADSL or WLAN, because IP-CAN can provide the IMS terminal connection to the IMS home network or IMS visited network. But in our situation, we just concentrate on the Open IMS core network. In order to make the whole structure much easier, we put the IMS terminal in the same network as the IMS server. Therefore, the IP-CAN is not considered in the project.

In addition to the above aspects, IMS terminal also needs to discover the IP address of the P-CSCF which acts as an outbound/inbound SIP proxy server. This step is so important because IMS terminal can send and receive SIP signaling to or from the P-CSCF when P-CSCF discovery procedure is finished.

When the prerequisites are fulfilled firstly, the IMS terminal can start to register to the IMS network. It's a regular SIP registration.

The figure 4-6 shows the general procedure of a IMS terminal registering to the OpenIMS Core network.

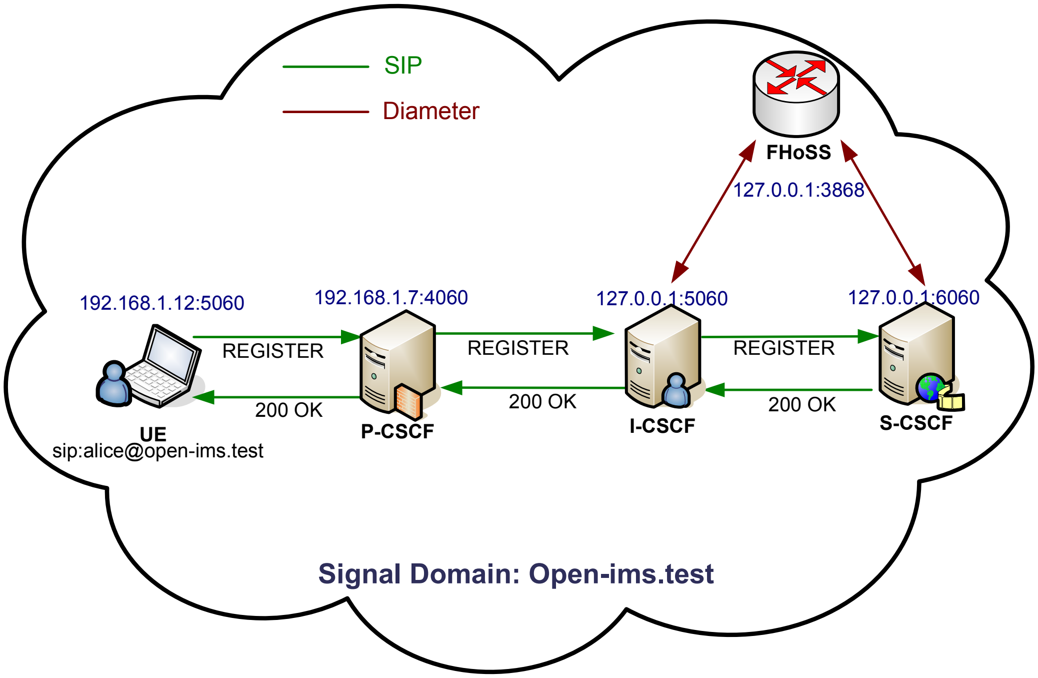


Figure 4-6 Registration for IMS client in OpenIMS

As shown in the figure 4-6, UE sends the SIP register request to P-CSCF, P-CSCF will send the register messages that contain P-CSCF name and user information to I-CSCF. I-CSCF obtains the HSS address and exchanges UMTS information with HSS through Cx interface, while at the same time HSS chooses the suitable S-CSCF for UE according to the request of I-CSCF and sends the address back to I-CSCF.

After that, I-CSCF will deliver the register information to the selected S-CSCF. HSS stores register information with user identity and the corresponding S-CSCF name. At the end, S-CSCF answers the 200 ok to UE, and then sends the connect information back to I-CSCF and releases all register information. At that time, UE is prepared to use needed multimedia services.

The purpose of registration for end-users is that they can be authorized to access the IMS network and use the IMS services. IMS Registration is accomplished by a SIP

REGISTER request. First IMS terminal retrieves from ISIM the Private User Identities, Public User Identities and home network domain URI, then it creates a SIP REGISTER request including the following four parameters.

The registration URI: this is used to identify the home network domain, and it included in the *Request-URI* of the REGISTER request. In our situation, it is: *sip:open-ims.test.*

The Public User Identity: it is a SIP URI that represents the user ID under registration. And it is included in the *To* header field value of the REGISTER request.
In out case, it is: *sip:alice@open-ims.test.*

The Private User Identity: this is only used for authentication, in our case, it is showed as: *alice@open-ims.test*. We can find it in the in the *username* parameter of the *Authorization* header field value of SIP REGIDTER request.

The Contact address: this is a SIP URI which includes the IP address of the IMS terminal or a host name where the user is reachable. It is contained in the SIP *contact* header. And it is showed as: *sip:alice@192.168.1.12:5060* in our case.

```
Internet Protocol, Src: 192.168.1.12 (192.168.1.12), Dst: 192.168.1.7 (192.168.1.7)
User Datagram Protocol, Src Port: 5060 (5060), Dst Port: 4060 (4060)
Session Initiation Protocol
    Request-Line: REGISTER sip:open-ims.test SIP/2.0
    Message Header
        Via: SIP/2.0/UDP 192.168.1.12:5060;rport;branch=z9hG4bK1480421391
        Route: <sip:pcscf.open-ims.test:4060;lr>
        From: <sip:alice@open-ims.test>;tag=1286929239
        To: <sip:alice@open-ims.test>
        Call-ID: 1298980893@192.168.1.12
        CSeq: 1 REGISTER
        Contact: <sip:alice@192.168.1.12:5060>
        Authorization: Digest username="alice@open-ims.test", realm="open-ims.test", nonce="
", uri="sip:open-ims.test", response=" "
        Max-Forwards: 70
        User-Agent: eXosip/2.2.2
        Expires: 600000
        Supported: path
        Content-Length: 0
```

4.3.2 Subscription procedure

By this stage, the registrar accepts the registration and creates a registration state.

And the IMS terminal will subscribe to its registration-state information. The IMS terminal sends the SUBSCRIBER request for the event: reg to the P-CSCF and P-CSCF should proxy the request to the S-CSCF. After the S-CSCF receives the request, it acts as a SIP notifier and sends a 200 (OK) response, and the P-CSCF forwards the response to the terminal. Besides, the S-CSCF should also send a NOTIFY request and the P-CSCF shall forward it to the terminal. Contained in this NOTIFY request are all the Public User Identities allocated to the user during the user's registration state. At the end, the terminal answers with a 200 (OK) response and the P-CSCF forwards the response to the S-CSCF.

4.3.3 Call Session procedure

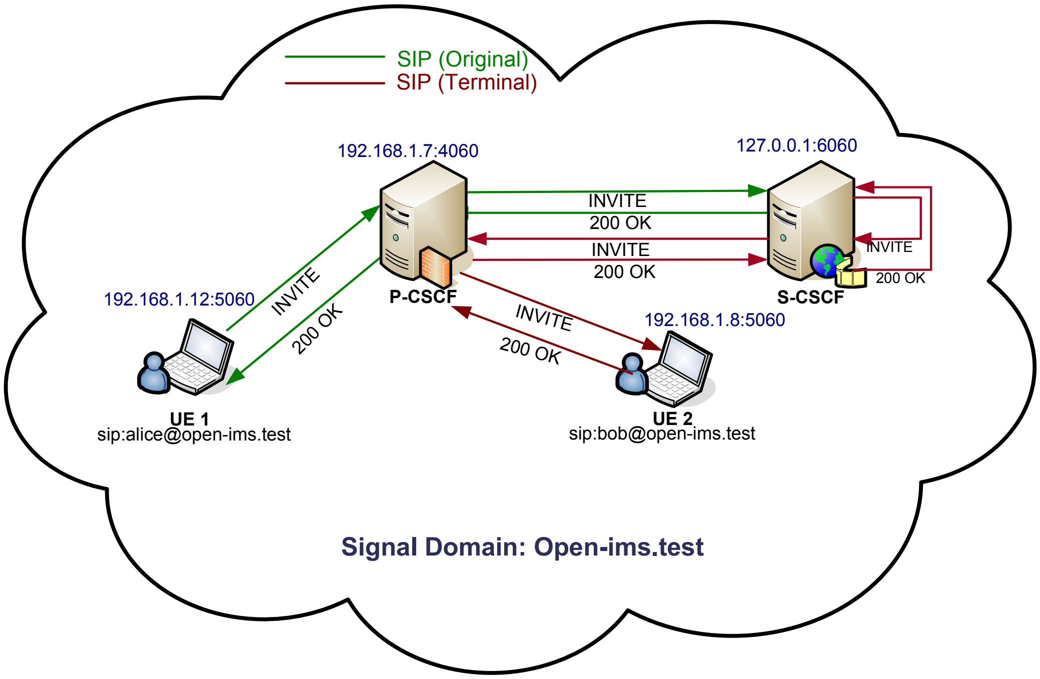


Figure 4-7 IMS client basic call setup

The IMS Terminal Sends an INVITE Request

The log message below shows our situation where the IMS client sends an INVITE request to the network.

Taking a look at the message header field; we see that this field is consisting of following header fields.

- The Request-URI header field: It contains the Public User Identity that shows Alice's identity which is belongs to the operator "open-ims.test".
- The Via header field: In this header field, there are the IP address and the port number showing where the IMS terminal responses to the INVITE request. This header also shows what transport protocol will be used to transport SIP messages, in our case the protocol is "UDP".

- The Contact header field: The value of this header is set to the SIP URI that the IMS terminal is supported to receive subsequent request.
- The Router header field: The value of this header points to the P-CSCF in the visited network and the S-CSCF in the home network. As shown below, because our call is made within one network, the visit network and the home network are the same which is known as "open-ims.test".

  Route: <sip:pcscf.open-ims.test:4060;lr>
  Route: <sip:orig@scscf.open-ims.test:6060;lr>
- The P-Preferred-Identity header field: The value of this header set to "Alice" and a SIP URI. When the P-CSCF forwarding the INVITE request to the home network, it will change this header filed into a P-Asserted-Identity header field containing the same value.
- The P-Asserted-Identity header field: Whenever the P-CSCF forwards the SIP request, the request always contains a P-Asserted-Identity header field which includes a Public User Identity value.
- The P-Assess-Network-Info header field
- The From and To header fields
- The Privacy header field: This header field used to show that the user is willing to indicate some private information to called party.
- The Require, Proxy-Require, and Supported header fields

  The Supported header field declares the SIP extensions that will be used in the response. For instance, in our example, the IMS terminal shows that it supports the provisioned response extension "100rel".
- The Allow header fields: Though this header field is optional, it is important. It shows the SIP methods that the IMS terminal supports.
- The Content-Type and Content-Length header fields: The values of those two headers rely on the body included in the INVITE request.

```
Request-Line: INVITE sip:bob@open-ims.test SIP/2.0
    Message Header
        Via:SIP/2.0/UDP 192.168.1.12:5060;rport;branch=z9hG4bK2099595392
        Route: <sip:pcscf.open-ims.test:4060;lr>
        Route: <sip:orig@scscf.open-ims.test:6060;lr>
        From: "Alice" <sip:alice@open-ims.test>;tag=1990381184
        To: <sip:bob@open-ims.test>
        Call-ID: 1621887217@192.168.1.12
        CSeq: 20 INVITE
        Contact: <sip:alice@192.168.1.12:5060>
        Max-Forwards: 70
        User-Agent: UCT IMS Client
        Subject: IMS Call
        Expires: 120
        P-Preferred-Identity: "Alice" <sip:alice@open-ims.test>
        Privacy: none
        P-Access-Network-Info: IEEE-802.11a
        Require: sec-agree
        Proxy-Require: sec-agree
        Supported: 100rel
        Allow: INVITE, ACK, UPDATE, INFO, CANCEL, BYE, OPTIONS,
REFER, SUBSCRIBE, NOTIFY, MESSAGE
        Content-Type: application/sdp
        Content-Length: 322
```

The Originating P-CSCF Processes the INVITE response

The P-CSCF first checks if the Router header contains the value that the Service-Router header field the IMS terminal received. Since the Router header contains the value, P-CSCF knows that the Router header is correctly populated. Then, P-CSCF checks whether a P-Preferred-Identity header is in the INVITE request. In our example, the P-Preferred-Identity header exists, then P-CSCF deletes it in the INVITE and inserts a P-Asserted-Identity header field and set its value to a SIP registered Public User Identity of the user.

The P-CSCF records the router and inserts a Record-Router header field with its own SIP URI when a SIP proxy wants to remain in the path for subsequent operation.

```
Request-Line: INVITE sip:bob@open-ims.test SIP/2.0
        Message Header
                Route: <sip:orig@scscf.open-ims.test:6060;lr>
                Record-Route: <sip:mo@pcscf.open-ims.test:4060;lr>
                Via: SIP/2.0/UDP 192.168.1.7:4060;branch=z9hG4bK17b.09719f96.0
                Via:                                                                   SIP/2.0/UDP
192.168.1.12:5060;rport=5060;branch=z9hG4bK2099595392
                From: "Alice" <sip:alice@open-ims.test>;tag=1990381184
                To: <sip:bob@open-ims.test>
                Call-ID: 1621887217@192.168.1.12
                CSeq: 20 INVITE
                Contact: <sip:alice@192.168.1.12:5060>
                Max-Forwards: 16
                User-Agent: UCT IMS Client
                Subject: IMS Call
                Expires: 120
                Privacy: none
                P-Access-Network-Info: IEEE-802.11a
                Require: sec-agree
                Proxy-Require: sec-agree
                Supported: 100rel
                Allow: INVITE, ACK, UPDATE, INFO, CANCEL, BYE, OPTIONS, REFER,
SUBSCRIBE, NOTIFY, MESSAGE
                Content-Type: application/sdp
                Content-Length: 322
                P-Asserted-Identity: "Alice" <sip:alice@open-ims.test>
                P-Charging-Vector:     icid-value="P-CSCFabcd00000000460ead1600000075";
icid-generated-at="127.0.0.1"; orig-ioi="open-ims.test"
```

The Originating S-CSCF Processes the INVITE Request

The S-CSCF tries to route the SIP request based on the destination in the Request-URI. In our case the Request-URI is set to <sip:bob@open-ims.test>, so the S-CSCF tries to find a SIP server in the network named “open-ims.test”.

Since the request is forwarded within the home work, S-CSCF still keeps the P-Access-Network-Info header field in the request.

The Terminating S-CSCF Processes the INVITE Request

The S-CSCF in the terminating network used to take care of the callee receives the INVITE request. First, it verifies the callee by the Request-URI in the request, and it adds its own SIP URI to the Record-Router header field value to remains in the path. Then it continues with the processing of the INVITE request. It creates a new Request-URI with the contents of the Content header field value that were registered by the callee (<sip:bob@192.168.1.8:5060 SIP/2.0>). And it sets the Router header to that of the Path header which contains the P-CSCF in as showed below.

Router: <sip:term@pcscf.open-ims.test:4060;lr>

The S-CSCF is retargeted, so it inserts a P-Called-Party-ID header field which is set to the original Request-URI.

P-Called-Party-ID: <sip:bob@open-ims.test>

Then S-CSCF forwards the INVITE request including the P-Called-Party-ID header field and at the end the request will reach to the P-CSCF.

The Terminal P-CSCF Processes the INVITE request

In our case the caller set the Privacy header field to “none”, therefore no privacy is required, so the P-CSCF keeps the P-Asserted-Identity header field in the INVITE request. Also, the P-CSCF extracts the Public User Identity from the P-Called-Party-ID header of the SIP INVITE request and identifies the Public User Identity of the callee.

The Callee’s Terminal Processes the INVITE Request

The IMS terminal checks the P-Asserted-Identity header field and gets the result that it presents, so the IMS terminal extracts the identity of the caller. Then it inspects the value of the P-Called-Party-ID to determine where the INVITE request is addressed to. At the same time the IMS terminal inserts a Contact header whose value is a SIP URI that including the IP address and the port number.

The Callee’s IMS Terminal Processes the PRACK request

When the PRACK is received at the callee the IMS terminal generates a 200OK response which is different from the 200OK to the INVITE request.

The situation of SIP client is very similar to the IMS client case. However, there are some differences we described below.

As showed below, when the SIP terminal sends an INVITE request, there is no P-Preferred-Identity header, no P-Access-Network-Info header, no Privacy header and no Require, Proxy-Require, Supported header fields in the message header field.

```
Request-Line: INVITE sip:li@open-ims.test SIP/2.0
    Message Header
        Via:                                                SIP/2.0/UDP
192.168.1.15:34156;branch=z9hG4bK-d87543-d852681fd15532
72-1--d87543-;rport
        Max-Forwards: 70
        Route: <sip:orig@scscf.open-ims.test:6060;lr>
        Contact: <sip:fei@192.168.1.15:34156>
        To: "li (Softphone)"<sip:li@open-ims.test>
        From: "fei"<sip:fei@open-ims.test>;tag=ef22a914
        Call-ID: M2IwM2FlN2QxZTJiZTcxNGFmNjEzZTFhNjNkZDE4MzI.
        CSeq: 1 INVITE
        Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY,
MESSAGE, SUBSCRIBE, INFO
        Content-Type: application/sdp
        User-Agent: X-Lite release 1006e stamp 34025
        Content-Length: 325
```

From the basic session setup figure for SIP terminal, we can see, between the S-CSCF in the originating home network and the P-CSCF in terminating visited network, there is a terminating home network containing I-CSCF, HSS and S-CSCF. Therefore, we discuss another situation below.

The Terminating I-CSCF Processes the INVITE request

The S-CSCF has found the I-CSCF at the SIP server in the home network. The I-CSCF is used to identify the callee in the Request-URI of the INVITE request and has to forward the SIP request to the S-CSCF allocated to the callee.

To discover the address of the S-CSCF which is allocated to the callee, the I-CSCF queries the HSS where the address of the S-CSCF is stored with a Diameter Location-Information-Request (LIR) message. After the HSS received the message, it gets the stored S-CSCF address and inserts it in a Diameter Location-Information-Answer (LIA) message, sending it to the I-CSCF. Till then, I-CSCF can route the INVITE request to the S-CSCF that is allocated to the callee.

Another difference from the IMS case is that there is no PRACK response in the session for SIP case.

4.4 Evaluation of OpenIMS components

4.4.1 Evaluation at the UE

Table 4-1 Evaluation at the UE

TS 24.229 v6			R6 Conform	R6 Non-conforma	comments
5.1 Procedures at the UE					
	5.1.1.2 Initial registration		*		
		On sending a REGISTER request		*	no security-client; no P-Access-Network-Info header
		On reseiving the 200 (OK)response to the REGISTER	*		without considering security
	5.1.1.3 Initial subscription to the registration-state event package			*	no P-Access-Network-Info header; no Event & Expires header
	5.1.1.5 Authentication				
		On receiving a 401 (Unauthorized)reponse to the REGISTER request		*	
		the 401 (Unauthorized) reponse to the REGISTER request is deemed to be valid		*	
		On receiving the 200 (OK) reponse for the security association protected REGISTER request		*	
	5.1.1.6 User-initiated deregistration				
		On sending a REGISTER request		*	no Security-Verity; no security-client; no P-Access-Network-Info header
		On receiving the 200 (OK) reponseto the REGISTER request	*		
	5.1.2.1 Notification about multiple registered pubblic user identities		*(IMS)	*(SIP)	
	5.1.3.1 Initial INVITE request		*(IMS)		

As showed in the table, since our testing is within one single domain and no security is required, so for both IMS and SIP clients, in most cases such as registration, deregistration and authentication, there are no security-client or security-server headers, no security-association, no P-Access-Network-Info and P-Associated-URI headers. Besides, when UE initials subscription to the registration-state event package, the event header should set to “reg”. But for SIP clients, the event header set to “message-summary”. Otherwise, our testing situations are mostly consistent with the 3GPP TS 24.229 specification.

In Appendix 2-6 we described the testing situations in detail according to the sub-clauses in the specification 3GPP TS 24.229.

4.4.2 Evaluation at the P-CSCF

Table 4-2 Evaluation at the P-CSCF

TS 24.229 v6	R6 Conformant	R6 Non-conformant	comments
5.2 Procedure at the P-CSCF			
5.2.1 General	✻		
5.2.2 Registration			
when the P-CSCF receives a REGISTER request from the UE		✻	not protected; no Security-Client header
when the P-CSCF receives a 401(Unauthorized) response to a REGISTER request	✻		without considering security
when the P-CSCF receives a 200(OK) response to a REGISTER request		✻	no contact header; no P-Asserted-Identity header; no P-Charging-Function-Address; term-ioi is not received header
5.2.3 Subscription to the user's registration-state event package			
Upon receipt of a 200(OK) response to the initial REGISTER request		✻	From header is not set to P-CSCF's SIP URI; the Expire header is still the same
5.2.5 Deregistration		✻(SIP)	no functionality of deregistration for SIP
5.2.5.1 User-initiated deregistration		✻	does not remove the Public User Identity found in the To header field.
5.2.6 General treatment for all dialogs and standalone transactions excluding the REGISTER method			
5.2.6.3 Requests initiated by the UE		✻	no P-Charging-Function-Address header
5.2.6.4 Requests terminated by the UE	✻		
5.2.7 Initial INVITE			
5.2.7.2 Mobile-originating case		✻	doesn't insert the P-Media-Authorization header
5.2.8.1 P-CSCF-initiated Call release			
5.2.8.1.2 Release of an existing session	✻		for IMS: P-CSCF serves the calling user. for SIP: P-CSCF serves the called user .

Generally speaking, the functionality of the P-CSCF is mostly conformant to the specification of 3GPP R6 TS 24.229. The P-CSCF of OpenIMS support the Path and Service-Route headers, and the Path header is only used in the REGISTER request and its 200 (OK) response, while the Service-Route header is only applicable to the 200 (OK) response of REGISTER request.

The difference in OpenIMS is: for both IMS Client and SIP Client, there is not P-Charging-Function-Addresses header. Therefore, the functionality of P-CSCF with P-Charging-Function-Addresses header is not considered. The other difference is that there is no P-Media-Authorization header in OpenIMS, because what our project concentrates on is just OpenIMS Core in which the AS is not included.

As for the reason that the security is not considered, the REGISTER request is not protected. Therefore, the REGISTER request is not protected in our case, and the Security-Client header does not exist. The related information regarding security is different from the specification. For example, there are no security associations, Security-Server, or reg-await-auth timer. And the parameter “integrity-protected” is inserted with the value “no”. And the architecture of the OpenIMS Core in our case is too simple to include the security, because all the components are fixed in a single domain.

The next difference is that P-CSCF cannot store the values received in the P-Charging-Function-Address header. The last difference is that a term-ioi parameter is not received in the P-Charging-Vector header.

For the situation where it receives a 200 (OK) response to the initial REGISTER request, The P-CSCF will generate a SUBSCRIBE request but the From header is not set to the P-CSCF’s SIP URI. It set as: sip:alice@open-ims.test which is a Public User Identity’s SIP URI. And the Expires header is still set to 600000 which are the same as the Expires header indicated in the 200 (OK) responses to the REGISTER request.

In deregistration case, for the SIP Client, it doesn’t support the functionality of deregistration. For the IMS Client, there are some functionalities of deregistration are different from the specification.

When the P-CSCF receives an initial request for a dialog or a request for a standalone transaction, the request of IMS client contains a P-Preferred-Identity header, so the P-CSCF shall identify the initiator of the request by that public user identity. As to the SIP client, the situation is different. The request of the SIP client doesn’t contain a P-Preferred-Identity header, so the P-CSCF shall identify the initiator of the request by a default public user identity.

There is no Service-Route header in our situation, and therefore we don’t consider the related cases.

Both the IMS and SIP client add their own address to the Via header, a situation which is conformant to the specifications.

When the P-CSCF receives a 1xx or 2xx response to the before request, the P-CSCF shall not store the values received in the P-Charging-Function-Address header, because we don’t have this header in our cases.

When the P-CSCF receives an INVITE request from the UE, the P-CSCF shall respond to all INVITE requests with a 100 (Trying) provisional response which is conformant to the specification. But the P-CSCF doesn’t insert the P-Media-Authorization header containing that media authorization token.

For the situation of call release, for the IMS clients, the P-CSCF serves the *calling* user of the session it shall generate a BYE request based on the information saved for the related dialog. And for SIP client, the P-CSCF serves the *called* user of the session it shall generate a BYE request based on the information saved for the related dialog.

The situation of security association and access network are not considered.

In Appendix 2-6 we described the testing situations in detail according to the sub-clauses in the specification 3GPP TS 24.229.

4.4.3 Evaluation at the I-CSCF

Table 4-3 Evaluation at the I-CSCF

TS 24.229 v6	**R6 Conformant**	**R6 Non-conformant**	**comments**
5.3 Procedure at the I-CSCF			
5.3.1 Registration procedure			
5.3.1.2 Normal procedures		✻	the SLF is not included
5.3.2 Initial requests	✻		don’t consider the IP connective access network
5.3.2.1 Normal procedures		✻	for IMS: remove SIP URI, route the request; for SIP: contains more than one Route header.
5.3.3 THIG functionality in the I-CSCF (THIG)		✻	in single domain

Generally speaking, the I-CSCF behaves as a stable proxy during the registration procedure.

The I-CSCF decides which HSS to query, and possibly as a result of a query to the Subscription Locator Functional (SLF) entity. But in the OpenIMS Core, the SLF is not included.

All components in this situation are in a signal domain. Therefore, we don't consider the IP connective access network. That's the reason why we don't have *P-Access-Network-Info* headers. As the same reason, we can not see the procedures about I-CSCF shown in the Wireshark log messages.

There is a situation which is different on IMS Client and SIP Client:
When the I-CSCF receives an initial request for a dialog or standalone transaction, we traced the log messages about IMS Client, and found that the I-CSCF removes its own SIP URI from the topmost *Route* header, and routes the request based on the *Request-URI* header field. With the trace on SIP Client, the situation is different. I-CSCF contains more than one *Route* header, and I-CSCF at first removes its own SIP URI from the topmost *Route* header, and then forwards the request based on the topmost *Route* header.

The situation of THIG is not considered, because the visited network and the home network are the same in our case.

In Appendix 2-6 we described the testing situations in detail according to the sub-clauses in the specification 3GPP TS 24.229.

4.4.4 Evaluation at the S-CSCF

Table 4-4 Evaluation at the S-CSCF

TS 24.229 v6			R6 Conformant	R6 Non-conformant	comments
5.4 Procedure at the S-CSCF					
	5.4.1.1 Introduction		✻		
	5.4.1.2 Initial registration and user-initiated reregistration				
		5.4.1.2.1 Unprotected REGISTER	✻(IMS)	✻(SIP)	for SIP: no ik.ck field and no reg-await-auth header
	5.4.1.4 User-initiated deregistration			✻	for IMS: integrity-protected =0; for SIP doesn't support function of deregistration
	5.4.2.1 Subscriptions to S-CSCF events				
		5.4.2.1.1 Subscription to the event providing registration state	✻		without considering association security
	5.4.3.1 Determination of mobile-originated or mobile-terminated case		✻		
	5.4.3.2 Requests initiated by the served user			✻	no P-charging-Verity header; no P-charging-Function-Address header; no need of network-hiding; no P-Access-Network-Info header
	5.4.4.1 initial INVITE		✻		
	5.4.4.2 Subsequent requests				
		5.4.4.2.1 Mobile-originating case		✻	
		5.4.4.2.2 Mobile-terminating case		✻	
	5.4.5.1 S-CSCF-initiated session release				
		5.4.5.1.2 Release of an existing session	✻		

As we showed in the table 4-4, our testing for registration procedures, both IMS clients and SIP clients support the Path header and Service-Route header. However, for IMS cases, those two headers also appear when the S-CSCF receives the “401 Unauthorized-Challenging the UE” which is not accordance to the specification 3GPP TS 24.229. According to the specification, the initiated register request in our testing cases is unprotected. Under this condition, when receiving a REGISTER request with the “integrity-protected” parameter set to “no” or without the “integrity-protected” parameter, the S-CSCF behaves almost as the specification says, but for both IMS and SIP clients, it doesn’t start the timer reg-await-auth.

When an incoming SUBSRIBER request addressed to S-CSCF arrives containing the Event header, for both SIP client and IMS client, the S-CSCF can find the identity for authentication of the subscription in the P-Asserted-Identity header received in the SUBSRIBER request and also stores the value of the orig-ioi parameter received in the P-Charging-Vector header.
However, for SIP client, the SUBSCRIBER request is subscribed to the event "message-summary" instead of event "reg" described in the specification, so it cause all notify messages to give the "487 Package not Supported" information.

Looking at our testing cases for call session procedures, we can see from the table 4-4 that when a request is initiated by served users, for both IMS and SIP clients, the S-CSCF doesn't insert P-Charging-Function-Addresses and there is no access-network-charging-info parameter in the P-Charging-Vector header field, and the S-CSCF doesn't remove the P-Access-Network-Info header based on the destination URI. Besides, for both the SIP clients and the IMS clients, when the S-CSCF receives an INVITE request, the S-CSCF processes the initial INVITE request without examining the SDP.

In Appendix 2-6 we described the testing situations in detail according to the sub-clauses in the specification 3GPP TS 24.229.

4.4.5 Evaluation at the SIP2IMS

To accelerate testing and to integrate with SIP UEs and test tools, a gateway that helps SIP traffic to work in an IMS environment was required. The SIP-to-IMS gateway performs this adaptation tasks. At the moment it translates between MD5 and AKAv1-MD5 authentications and helps with special headers. The SIP2IMS Gateway can been regarded as the helper module of the Open source IMS Core project to allow the verification of services with current state-of-the art VoIP clients. Yet, due to its gateway nature, it filters much of the IMS advantages and should not be regarded as a full blown IMS solution.

The Open Source IMS SIP2IMS Gateway should allow transformation of IETF SIP messages to IMS conformant messages. It translates MD5 authentication to IMS AKA authentication.SIP2IMS can also enable developers to access core elements and to trial multimedia services by using a non-IMS VoIP client.

But in the real experiment, the gateway SIP2IMS is useless, we can just use SIP client by changing the authentication-algorithm into MD5, while use IMS client by changing the authentication-algorithm into AKAv1-MD5. The SIP2IMS gateway doesn't afford the functionality to translate MD5 authentication to AKAv1-MD5 when we use SIP client change into IMS client.

For its useless in OpenIMS core, FOKUS would delete it later.

4.4.6 Evaluation at the Cx

In our situation, there are several commands that appear which are User-Authorization-Request (UAR), User-Authorization-Answer (UAA), Server-Assignment-Request (SAR), Server-Assignment-Answer (SAA), Location-Info-Request (LIR), Location-Info-Answer (LIA),Multimedia-Auth-Request (MAR), Multimedia-Auth-Answer (MAA). For both IMS clients and SIP clients, our examples are mostly in accord with the 3GPP TS 29.229. We have all the mandatory AVPs and most optional AVPS in those commands. However, there are no "Registration-Termination-Request (RTR)", "Registration-Termination-Answer (RTA)", "Push-Profile-Request (PPR)" and "Push-Profile-Answer" commands in our examples.

For both IMS client and SIP client in our example, there are two values standing for success that are "DIAMETER_RIRST_REGISTRATION" (2001) and "DIAMETER_SUBSEQUENT_REGISTRATION" (2002).

The "DIAMETER_RIRST_REGISTRATION" (2001) appears in MAA, SAA and LIA commands while the "DIAMETER_SUBSEQUENT_REGISTRATION" (2002) appears in UAA command during the registration process.

There are also several AVPs that are showed in the table6.3.1 of 3GPP TS 29.229 that appeared in our examples, namely, Visited-Network-Identifier AVP (600), Public-Identity AVP (601), Server-Name AVP (602), User-Data AVP (606), SIP-Number-Auth-Items AVP (607), SIP-Auth-Data-Item AVP (612), Server-Assignment-Type AVP (614), Charging-Information AVP (618), User-Authorization-Type AVP (623), User-Data-Already-Available AVP (624)

In Appendix 2-6 we described the testing situations in detail according to the sub-clauses in the specification 3GPP TS 24.229.

4.4.7 Evaluation at the abnormal cases

At first, we choose SJphone as the SIP Client with which we are always got the following error:

- **403 forbidden-not registered! You must register first with a s-cscf**

We checked it and found that SJ Phone doesn't put Expires header to the REGISTER request, so although we could get 200 OK, it comes with 0 binding, which means it is not registered. In addition, there is another problem with SJ Phone, which is that doesn't follow Service Route headers. Therefore, we use X-Lite which can at least work instead of SJphone.

During the register process, we met several errors.

- **503 : Server Unavailable**

To solve this problem, we just need to change the DNS address to: 192.168.1.7. This is because the DNS service on that host knows about the open-ims.test DNS zone.

- **600: Busy everywhere**

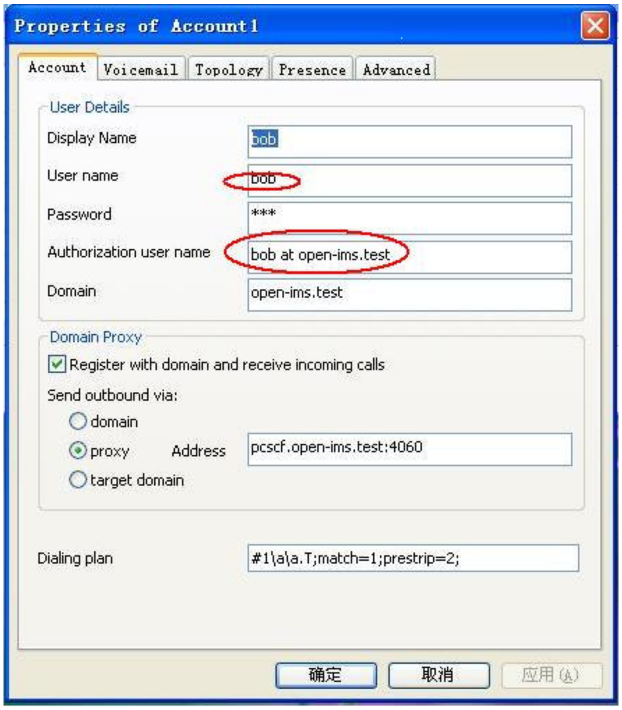
  We get the following trace when we got this error:

```
Status-Line: SIP/2.0 600 Busy everywhere-Forwarding to S-CSCF failed
         status-code: 600
         [Resent Packet: False]
Message Header
         via: SIP/2.0/UDP 192.168.1.12:65290;
         To: "bob"<sip:bob at open-ims.test;tag=a6alc5f60faecf035alae5b6e96e979a-c29e
        From:"bob" <sip:bob at open-ima.test>; tag=f66a8220
        Call-ID: YjY4M7FjMGExM2FmZTNhZTI2ZGNKNDI2NmUxNzU5N2Y
        Cseq: 1 REGISTER
        Server: sip Express router(2.1.0-dev1-OpenIMSCore (x86_64/Linux))
        Content-Length: 0
        Warning: 392 127.0.0.1:5060 "Noisy feebback tells: pid=13230
req_src_ip=192.168.1.7 req_src_port=4060 in_uri=sip:open-ims.test
out_uri=sip:scscf.open-ims.test:4060 via_cnt==0"
```

We just restarted the system and this problem was resolved, however, we got another error as shown below;

- **403 forbidden -- HSS user unknown**

Since it shows ‘user unknown’, we focus on check logs that are connected to user and see what the response is. After checking, we find that we made some mistakes when we set up the account.



User Details

Display Name: bob

User name: bob

Password: ***

Authorization user name: bob at open-ims.test

Domain: open-ims.test

Domain Proxy

☑ Register with domain and receive incoming calls

Send outbound via:

○ domain

● proxy Address: pcscf.open-ims.test:4060

○ target domain

Dialing plan: #1\a\a.T;match=1;prestrip=2;

Figure 4-8 Mistakes in properties

As shown in figure 4-8, we had some mistakes at ‘user name’ and ‘Authorization user name’. We have to delete the space after ‘bob’ in ‘user name’, and have to change ‘Authorization user name’ into ‘bob@open-ims.test’.
After we made these two changes, we tried to register again, and this time registration was successful!

During the register process, we only met one problem said:

- **403 Forbidden—HSS returned no authentication vectors**.

We know that the S-CSCF sends the request for the HSS regarding the authentication vectors and also specifies the preferred scheme, and the HSS is looking on request, for an authentication scheme specified by the S-CSCF, which has to be used. Then, the HSS verifies if the user supports that scheme. If the user does not support the requested scheme, the HSS will send an error message to the S-CSCF with Authentication Scheme not supported. So to solve this problem we have to configure the S-CSCF and the HSS for the same scheme.
What we did was to first change the “Authentication algorithm” from “Digit-MD5” to”Digit-AkAV1-MD5” from the HSS web interface. And then in the “scscf.cfg” file, change

```
 modparam(“scscf”,”registratiotn-default-algorithm”,”MD5”)
#modparam(“scscf”,”registration-default-algorithm”,”AKAV2-MD5”)
#modparam(“scscf”,”registrationi-default-algorithm”,”AKAV1-MD5”)
```

To：

```
 modparam(“scscf”,”registratiotn-default-algorithm”,”AKAV1-MD5”)
#modparam(“scscf”,”registration-default-algorithm”,”AKAV2-MD5”)
#modparam(“scscf”,”registrationi-default-algorithm”,”MD5”)
```

After changing, we restarted S-CSCF, and then tried to register again, and the problem was solved.

5 Solutions of interoperability between IMS and SIP

Through various kinds of experiments, we found that most kinds of cases can be obtained, but they are just limited in using the same user equipment. For example, we can set up the call session by using SIP Client - SIP Client, or IMS Client - IMS Client. In fact, that is too limited when used in enterprise solutions because some of the clients can only support SIP, and some can support IMS. No doubt companies want to find a way for the interoperability between SIP and IMS so that the establishment can be set up.

5.1 Use two S-CSCFs

The way to find out how to solve this problem is to know the reason why IMS service can only be used in the same user equipment. We make the experiment by using a SIP Client and an IMS Client at the same time. And then we use the log message shown on Wireshark to analyze the reason.

During the register process, we met the following problem:
403 Forbidden—HSS returned no authentication vectors
After analyzing, we found that the reason is something about authentication and the related components are may be S-CSCF and HSS.

Like what we got in the evaluation of OpenIMS, we know that one main feature of S-CSCF in OpenIMS is to retrieve authentication vectors. It supports several ways of authentication: AKAv1-MD5, AKAv2-MD5 and MD5. The authentication algorithm AKAv1-MD5 is supported for IMS Client, while the authentication algorithm MD5 is supported for SIP Client. The S-CSCF will choose one authentication algorithm when the clients register or call through OpenIMS. Therefore, if the clients belong to SIP clients or IMS clients, the S-CSCF could work correctly, but if the clients who want to connect to the OpenIMS belong to different kinds, the S-CSCF will be confused about which authentication algorithm to choose, and then the error above will be shown.

Therefore, we consider that if it is possible to use two S-CSCFs which support different authentication algorithms, one supporting ‘AKAv1-MD5’ for IMS Client, and the other supporting ‘MD5’ for SIP Client, so that the SIP Client and IMS Client can register at the same time, and implement some IMS services between them. This is one solution for the interoperability between IMS and SIP.

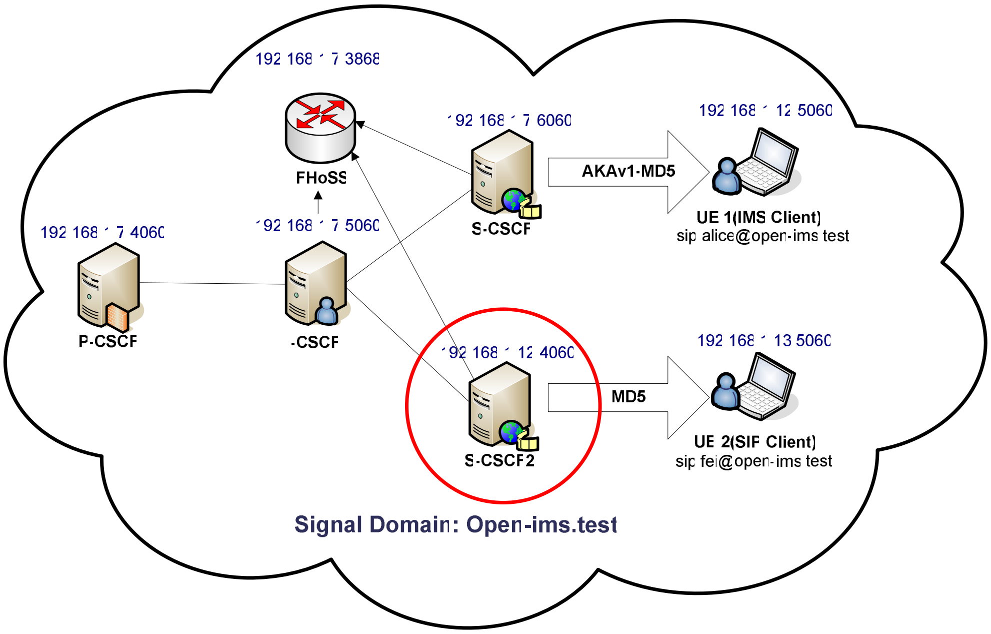


Figure 5-1 interoperability between SIP and IMS

5.2 Installation and configuration of S-CSCF2

In order to reach our purpose, we install the S-CSCF2 on another computer with the IP address and the port 168.192.1.12:4060, while the S-CSCF with the different IP address and the port number is: 168.192.1.7:5060. Some changes in the configuration files are required.

5.2.1 Add S-CSCF2 in DNS

```
cd var/named/etc/sites/open-ims.test

sudo vim open-ims.zone
```

This command is used to modify the zone.files. After we open it, the scscf2 is added in the zone.file which is shown as follow:

```
$ORIGIN open-ims.test.
$TTL 1W
@                   1D IN SOA        localhost. root.localhost. (
                                     2006101001       ; serial
                                     3H               ; refresh
                                     15M              ; retry
                                     1W               ; expiry
                                     1D )             ; minimum
                    1D IN NS         ns
ns                  1D IN A          192.168.1.7
pcscf               1D IN A          192.168.1.7
open-ims.test.      1D IN A          192.168.1.7
icscf               1D IN A          192.168.1.7
_sip                1D SRV 0 0 5060  icscf
_sip._udp           1D SRV 0 0 5060  icscf
_sip._tcp           1D SRV 0 0 5060  icscf
open-ims.test.               1D   IN   NAPTR   10   50   "s"   "SIP+D2U"   ""
_sip._udp.open-ims.test.
open-ims.test.               1D   IN   NAPTR   20   50   "s"   "SIP+D2T"   ""
_sip._tcp.open-ims.test.
scscf               1D IN A          192.168.1.7
scscf2              1D IN A          192.168.1.12
sip2ims             1D IN A          192.168.1.7
hss                 1D IN A          192.168.1.7
ue                  1D IN A          192.168.1.7
presence            1D IN A          192.168.1.7
```

5.2.2 Installation of s-cscf2

The installation steps are shown as follows
The first step is to create a new directory on the new computer. Then, to create the directory for ser_ims under the directory /opt/OpenIMSCore, we need get the Code of CSCFs - ser_ims/trunk from the page http://svn.berlios.de/svnroot/repos/openimscore . After that, we need to check if the CSCFs are there. The following step is to compile the CSCFs, which is to make libs install all in ser_ims. This step takes some time to finish and we have to make sure all the prerequisites have been installed. We just need the new S-CSCF, so it is not necessary to install icscf.sql or other components of CSCFs into MySQL. The last step is to configure the IMS core; we just need to copy the files of icscf: icscf.cfg, icscf.xml, icscf.sh to /opt/OpenIMSCore. Note that the necessary changes are needed in our own situation.

```
mkdir /opt/OpenIMSCore

cd /opt/OpenIMSCore

mkdir ser_ims

svn checkout http://svn.berlios.de/svnroot/repos/openimscore/ser_ims/trunk ser_ims

cd ser_ims

make install-libs all

cd ..

cp ser_ims/cfg/icscf.* .
```

5.2.3 Add s-cscf2 in FHoSS

After the installation of s-cscf2, it should be edited in the icscf database and also inserted into the new one. The tables of icscf in database are shown as follows:

```
+---------------------------+
| Tables_in_icscf           |
+---------------------------+
| nds_trusted_domains       |
| s_cscf                    |
| s_cscf_capabilities       |
+---------------------------+
```

Then we select the data from s_cscf in this table, and edit the default S-CSCF as IMS S-CSCF with the uri as: sip:scscf.open-ims.test:6060 while inserting the new s-cscf2 as SIP S-CSCF with the uri as: sip:scscf2.open-ims.test:4060.

```
+----+-----------------+--------------------------------------+
| id | name            | s_cscf_uri                           |
+----+-----------------+--------------------------------------+
| 1  | IMS S-CSCF | sip:scscf.open-ims.test:6060              |
| 2  | SIP S-CSCF | sip:scscf2.open-ims.test:4060             |
+----+-----------------+--------------------------------------+
```

Next, we select the data from s_cscf_capabilities and edit it. The 0, 1 of capability represents the authentication algorithms of AKAv1-MD5 and MD5. Therefore, 0 is assigned to IMS S-CSCF and 1 is assigned to the SIP S-CSCF.

```
+----+--------------+-----------------+
| id | id_s_cscf     | capability    |
+----+--------------+-----------------+
|   1 |              1 |               0 |
|   2 |              1 |               0 |
|   3 |              2 |               1 |
|   4 |              2 |               1 |
+----+--------------+-----------------+
```

Then, we select the data from the diam_servers table. There is a default server and host for S-CSCF, and we insert the new server and host for the S-CSCF2.

```
+--------------------------------------+----------------------------+
| server                                | host                       |
+--------------------------------------+-------------------------- +
| sip:scscf.open-ims.test:6060    | scscf.open-ims.test    |
| sip:scscf2.open-ims.test:4060 | scscf2.open-ims.test |
+--------------------------------------+----------------------------+
```

5.2.4 Modification in s-cscf/s-cscf2.cfg

As in the evaluation of abnormal cases in chapter 4, if the authentication algorithm in S-CSCF is not defined in right way, we will receive the 403 error which is describe as HSS returning no authentication vectors. In case the error occurs, we need to define the authentication algorithm in each s-cscf for each kind of client.

In the file of scscf.cfg, we defined the default authentication algorithm as ”Digit-AkAV1-MD5”.

```
#modparam("scscf","registration_default_algorithm","MD5")
#modparam("scscf","registration_default_algorithm","AKAv2-MD5")
 modparam("scscf","registration_default_algorithm","AKAv1-MD5")
```

And in the file of scscf2.cfg, we define the default authentication algorithm as ”Digit--MD5”.

```
 modparam("scscf","registration_default_algorithm","MD5")
#modparam("scscf","registration_default_algorithm","AKAv2-MD5")
#modparam("scscf","registration_default_algorithm","AKAv1-MD5")
```

After these steps, we could register SIP Client and also IMS Client at the same time. Additionally, the call session can be established as well.

5.3 Registration and call session with S-CSCFs

As we described earlier, we still put the terminal and the server in the same network that is: open-ims.test. Following, we will describe the situation where the SIP client registered and the call session to OpenIMS network use S-CSCF2.

5.3.1 Registration with S-CSCF 2

As showed in the figure 5-2, first the SIP terminal creates a SIP REGISTER request including four parameters which are the registration URI, the Public User Identity, the Private User Identity and the Contact address. In our situation:

- the registration URI is: *sip:open-ims.test*
- the Public User Identity is: *sip:fei@open-ims.test* or *sip:li@open-ims.test*
- the Private User Identity is: *fei@open-ims.test* or *li@open-ims.test*
- the Contact address is: *fei@192.168.1.13:51066* or *li@192.168.1.5:5062*

The SIP terminal sends the request to the P-CSCF. The P-CSCF inserts a P-Visited-Network-ID that shows where the P-CSCF is located. In our example this is shown: open-ims.test. And the P-CSCF also inserts a Path header with its own SIP URI to request the home network to forward all SIP requests. In our example, this SIP URI is: sip:term@pcscf.open-ims.test:4060. The P-CSCF discovers an I-CSCF in the home network and forwards the SIP REGISTER request to it.

The I-CSCF queries the HSS with a Diameter UAR message and the HSS answers with the Diameter UAA message. Eventually, the I-CSCF forwards the REGISTER request to the S-CSCF2.

The S-CSCF2 creates a Diameter MAR message and sends it to the HSS. The HSS then stores the S-CSCF2 URI in the user data for further query and answers with a Diameter MAA message. In this way, the S-CSCF2 has downloaded the authentication data from the HSS. After this, the S-CSCF2 creates a SIP 401 (Unauthorized) response and forwards it to the SIP terminal via I-CSCF and P-CSCF.

In response, the SIP terminal sends a new REGISTER request to the P-CSCF. Because the authentication was successful before, when the request is received by the S-CSCF2, it sends a Diameter SAR message to the HSS to inform it that the user is now registered and at the same time to download the user profile. Then, the S-CSCF2 sends a 200 (OK) response to the SIP terminal showing that the REGISTER request is successful.

By this stage, in theory the SIP terminal should send the SUBSCRIBER request for the event: reg to the P-CSCF and P-CSCF should proxy the request to the S-CSCF. The S-CSCF shall act as a SIP notifier and sends a 200 (OK) response, and the

P-CSCF forwards the response to the terminal. In addition, the S-CSCF should also send a NOTIFY request and the P-CSCF shall forward it to the terminal. At the end, the terminal answers with a 200 (OK) response and the P-CSCF forwards the response to the S-CSCF.

However, in our example, the situation is quite different. First, after the SIP terminal receives a 200 (OK) response for the registration process from the S-CSCF2, it generates a new SUBSCRIBE request to event package: message-summary instead of event: reg. This request is forwarded to the S-CSCF2 via P-CSCF and I-CSCF, and the S-CSCF2 return also with the SUBSCRIBE request to the terminal. When the terminal receives the request, it sends a “489: Event Package not Supported” message to the P-CSCF and then the P-CSCF forwards the message to the S-CSCF2. The S-CSCF2 also forwards the message back to the terminal via I-CSCF and P-CSCF. This process is unexpected according to the theory, so we think our SIP client is not supporting the event package: message-summary.

After this, instead of sending a SUBSCRIBE request to event: reg by the SIP terminal, the P-CSCF directly sends the SUBSCRIBE request towards the event package: reg to the I-CSCF, and the I-CSCF forwards the message to the S-CSCF2. Then the S-CSCF2 answers with a “200 Subscription to REG saved” message and the I-CSCF forwards the message to P-CSCF. Additionally, the S-CSCF sends a NOTIFY message to the P-CSCF and gets the answer 200 (OK).

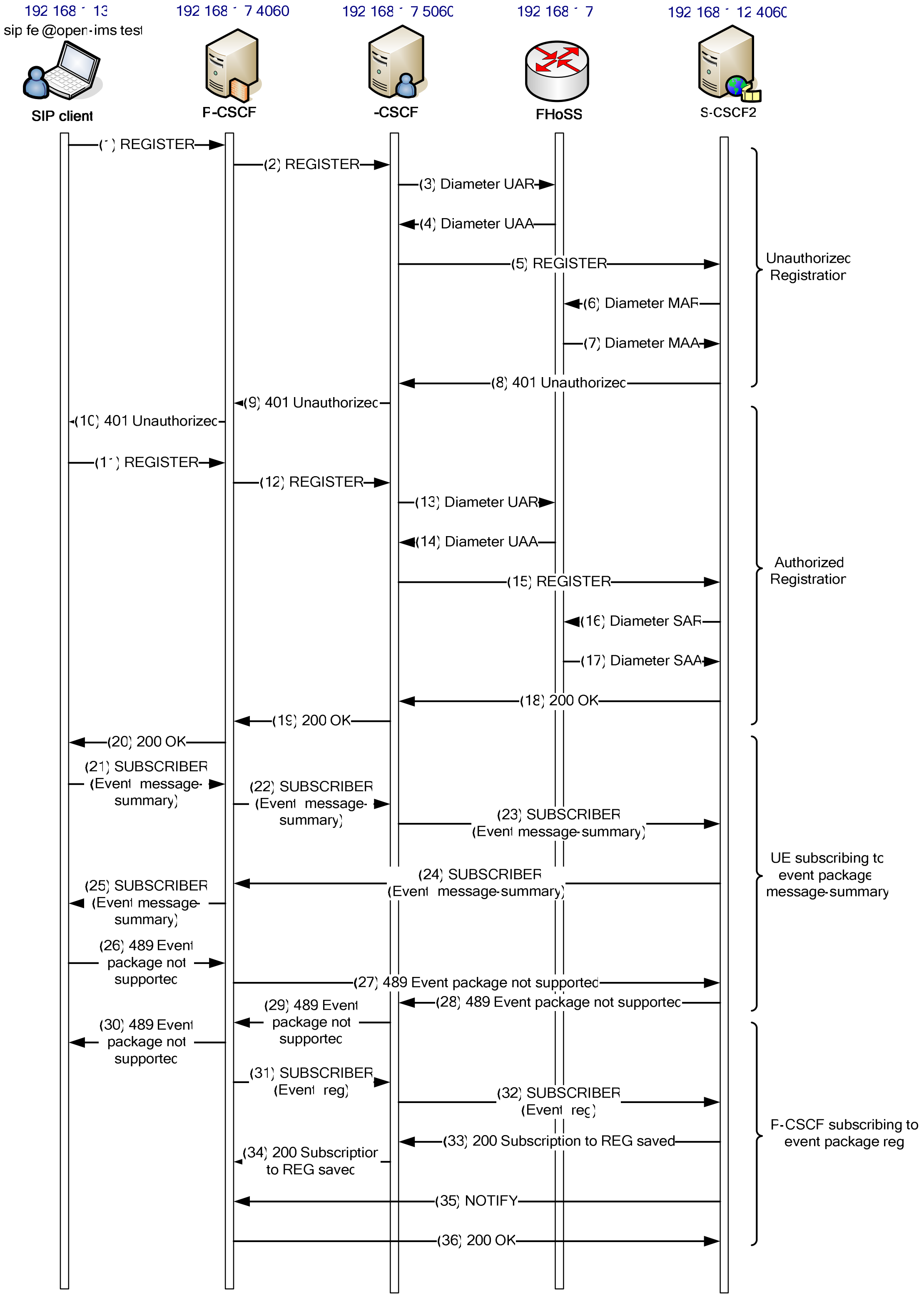

192 168 · 13
sip fe @open-ims test
SIP client
192 168 · 7 4060
P-CSCF
192 168 · 7 5060
-CSCF
192 168 · 7
FHoSS
192 168 · 12 4060
S-CSCF2
(1) REGISTER
(2) REGISTER
(3) Diameter UAR
(4) Diameter UAA
(5) REGISTER
(6) Diameter MAR
(7) Diameter MAA
(8) 401 Unauthorized
(9) 401 Unauthorized
(10) 401 Unauthorized
Unauthorized Registration
(11) REGISTER
(12) REGISTER
(13) Diameter UAR
(14) Diameter UAA
(15) REGISTER
(16) Diameter SAR
(17) Diameter SAA
(18) 200 OK
(19) 200 OK
(20) 200 OK
Authorized Registration
(21) SUBSCRIBER (Event message-summary)
(22) SUBSCRIBER (Event message-summary)
(23) SUBSCRIBER (Event message-summary)
(24) SUBSCRIBER (Event message-summary)
(25) SUBSCRIBER (Event message-summary)
(26) 489 Event package not supported
(27) 489 Event package not supported
(28) 489 Event package not supported
(29) 489 Event package not supported
(30) 489 Event package not supported
UE subscribing to event package message-summary
(31) SUBSCRIBER (Event reg)
(32) SUBSCRIBER (Event reg)
(33) 200 Subscription to REG saved
(34) 200 Subscription to REG saved
(35) NOTIFY
(36) 200 OK
P-CSCF subscribing to event package reg

Figure 5-2 Registration SIP client to OpenIMS (with two S-CSCF)

5.3.2 Call session with S-CSCF 2

The figure 5-3 below shows the flow of call session setup between the SIP client and the IMS client. From our Wireshark logs, we cannot get detailed information about the S-CSCF2. From the figure, it can be seen that the call process between SIP and the IMS client is almost the same as the call process between SIP clients as we described in chapter 4.5.3. We should to mention here that the “101 Dialog Establishment” message has not been defined in the 3GPP specification, so we can consider it as an unexpected situation.

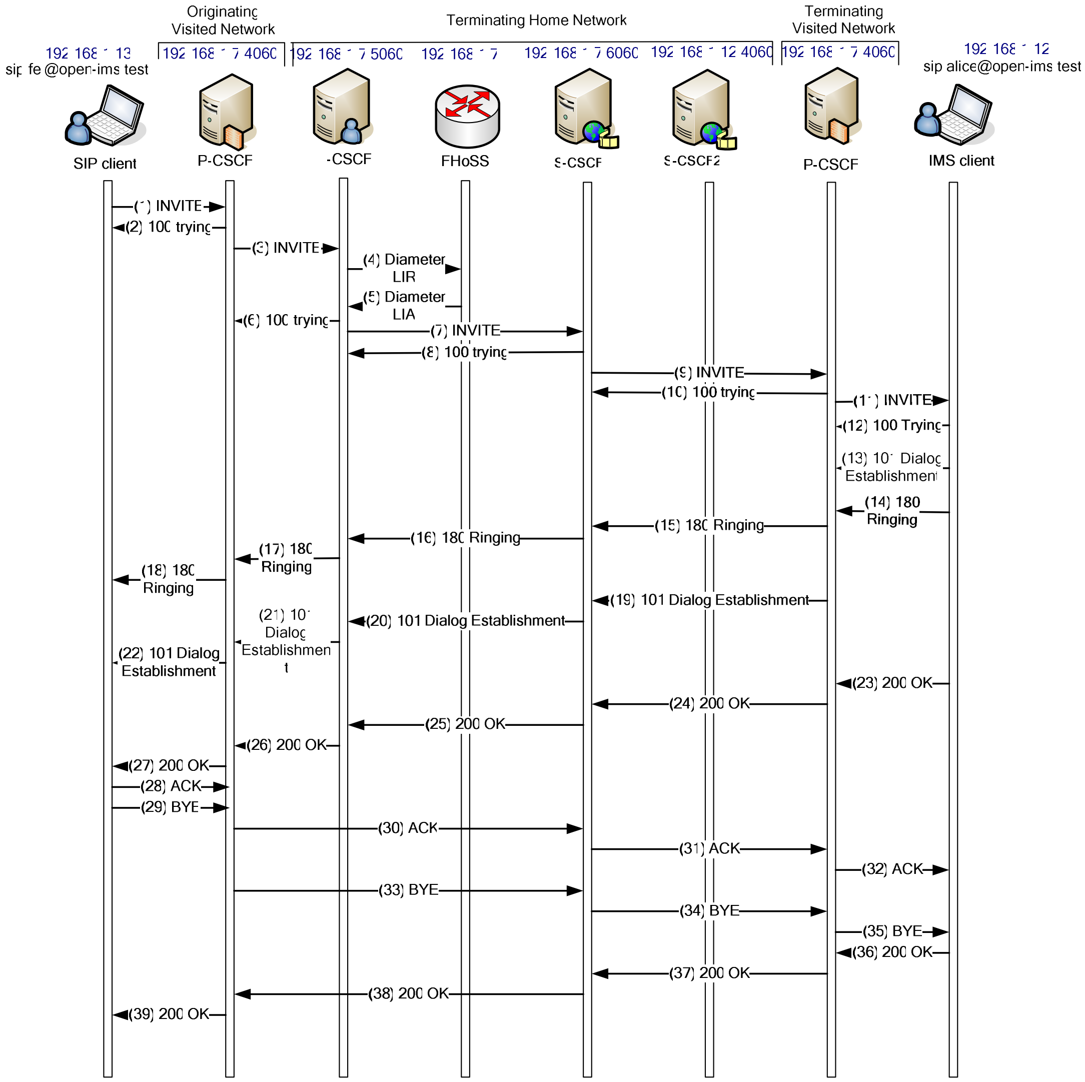


Figure 5-3 Call session setup between SIP client and IMS client

5.4 Evaluation about the solution with two S-CSCFs

The experiment shows it is possible to use two S-CSCFs supporting different authentication algorithms for SIP Client and IMS Client at the same time, so that the two clients can register to the OpenIMS at the same time and further establish the call session. Although the solution can be implemented, there are still some latent problems.

5.4.1 Instability

After each registration or call session, we cannot continue to repeat the testing again, but to get **403 errors - HSS returned no authentication vectors**.
As we have already described in the Chapter 4, this error occurs when there is no matched authentication algorithm in S-CSCF. Therefore, we check the scscf.cfg and the database in MySQL. We find that there is no change in scscf.cfg or scscf2.cfg, but the data in MySQL/hssdb/impi is changed automatically. IMPI is IP Multimedia Private Identity while IMPU is IP Multimedia Public Identity. They are contained in HSS user database.

The example below will illustrate how it changes. The data is too long to show, so it is shown with only the changeable parts.
The table is the data from MySQL/hssdb/impi, and is the state before testing. We can derive from it that fei@open-ims.test and li@open-ims.test are SIP Clients, and they both register to the S-CSCF 2 by using the “Digest-MD5” authentication algorithm, while bob@open-ims.test and alice@open-ime.test are IMS Clients who use “Digest-AKAv1-MD5” and register to the S-CSCF. Obviously, it is configured in the right way.

```
+--------------------+----+------------------------------+----+----------------------- +----+---------+
|   impi_string      |......|          scscf_name          |......|   auth_scheme       |......| algorithm |
+--------------------+----+--------------------------------+---+-----------------------+---+---------+
|bob@open-ims.test|......|sip:scscf.open-ims.test:6060|......| Digest-AKAv1-MD5 |......| AKAv1|
|fei@open-ims.test |......|sip:scscf2.open-ims.test:4060|......|      Digest-MD5      |......| NULL |
|li@open-ims.test    |......| sip:scscf2.open-ims.test:4060 |……|    Digest-MD5 |…… | NULL |
|alice@open-ims.test|......|sip:scscf.open-ims.test:6060|......|Digest-AKAv1-MD5|......|AKAv1|
+----------------------+---+---------------------------------+---+-----------------------+---+--------+
```

We can use the right configured data to test once, after that the data is changed by itself as following table.

```
+--------------------+----+------------------------------+----+---------------------- +----+---------+
|   impi_string      |......|         scscf_name          |......|   auth_scheme       |......| algorithm |
+--------------------+----+--------------------------------+---+------------------------+---+---------+
|bob@open-ims.test|......|sip:scscf.open-ims.test:6060|......| Digest-AKAv1-MD5 |......| AKAv1|
|fei@open-ims.test |......|sip:scscf.open-ims.test:6060|......|                          |......| NULL |
|li@open-ims.test|......|sip:scscf.open-ims.test:6060|……|Digest-AKAv1-MD5|……| NULL |
|alice@open-ims.test|......|sip:scscf.open-ims.test:6060|......|Digest-AKAv1-MD5|......|AKAv1|
+----------------------+---+--------------------------------+---+-----------------------+---+--------+
```

As it shown in the table, the scscf_name may be changed with the other one using in IMS Client, or the auth_scheme may be changed into the other authentication algorithm or even into nothing. Especially, the changes is not fixed, sometimes, it changes as this; sometimes, it changes in the other similar way and sometimes it may not change. The situation is randomly.

The reason for the instability is because:
We assigned the S-CSCF 2 in the impi table manually, but this is not the normal way to do. As the trunk version of OpenIMS does not offer capability supporting, we have no right to make and modifications in the impi table. Therefore, our changes in the databases do not change anything in the HSS functionality and that is also the reason that the assigned S-CSCF always remains the default S-CSCF after registration or other operations. Capabilities can be added to the branch version of FHoSS, but the capability functionality is not yet fully tested.

5.4.2 Call session released automatically

The other problem is the call session is always released by the remote client automatically as soon as the callee receives the call. From the RTP package, the message can be sent from caller, while the callee cannot reply.

When we change the X-Lite to SIPp to make a call initiated by fei (SIP client) and invite alice (IMS client), the call session can be established in the normal way.

When a call is initiated the request is transferred with SIP, while the actual audio is transferred over the Real-time Transport protocol (RTP) on another port. The end-to-end media transfer (RTP) contains only details of the private addresses and ports from the computer it was sent from. So when the client on the other side tries to return the media information it will fail, because the private address doesn't mean anything on the public network. [13]

6 Integration with SIP/VoIP solutions

In chapter 5, we discussed the solution of the interoperability between IMS and SIP. However, this solution is only valid in one domain which is our open-ims.test domain. Yet there are many users who still use non-IMS clients which directly connect to IMS, so a migration path to IMS is necessary.

In the migration stage, we assume a user A has two identities, one for entry to the IMS domain while the other is used to enter the non-IMS domain. In this case, when someone tries to call user A in non-IMS domain, the user's terminals which are registered with the IMS domain needs to be reached. In this case, we know that the IMS domain and the non-IMS domain need to interoperate, so our problem becomes how to achieve this goal?

6.1 Solution for migrate towards IMS

In [8], the authors came up with four possible solutions. In our project, we try to implement their "client based" solution. They described this solution as:

"This solution requires an advanced client to inform enterprise SIP PBX that his location has moved to another domain, all incoming call to sip:userA@enterprise.com should be redirected to sip:userA@operate.com.

When user turns on the IMS terminal, it will register on IMS domain as sip:userA@operate.com. Then it will register on the enterprise SIP PBX as sip:userA@enterprise.com and inform SIP PBX about the location change.

When an incoming call from sip:userA@enterprise.com reaches SIP PBX, the SIP PBX will acts as a redirect server and sends a "302 Moved Temporarily" SIP message to caller userB and instruct userB to try userA's new location sip:userA@operate.com. Then userB will initiate a new call to sip:userA@operator.com which is directly sent to IMS domain, and userA's terminal."

In our situation, the "operator.com" which is IMS domain is our "open-ims.test" domain in HiA, and the "enterprise.com" is non-IMS domain in auSystems "agder-ikt104.hia.no".

Figure 6-1 and 6-2 shows the registration and basic call setup cases in theory.

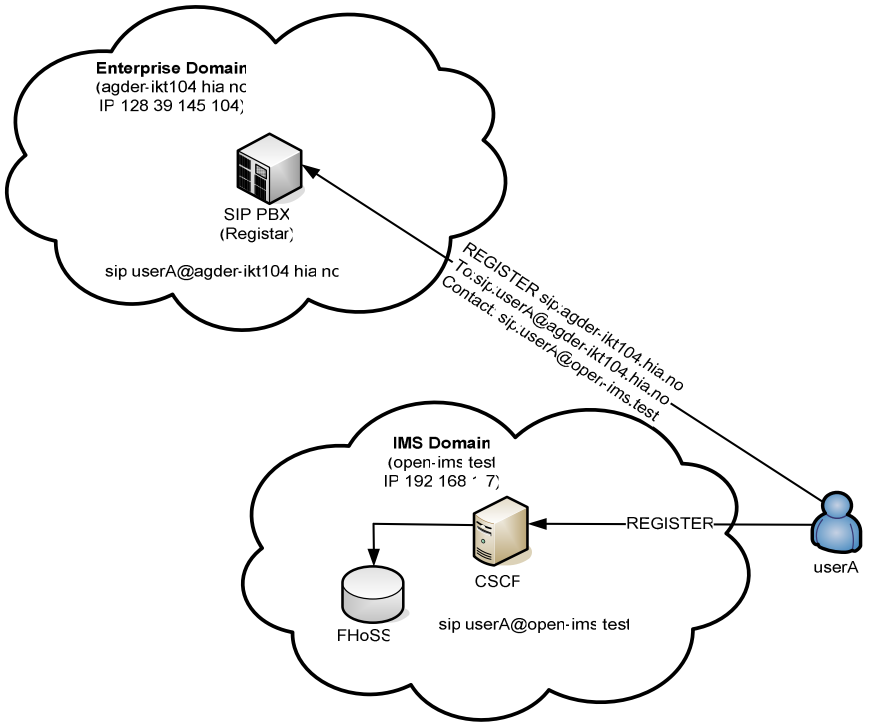


Figure 6-1 client based solution—Registration case

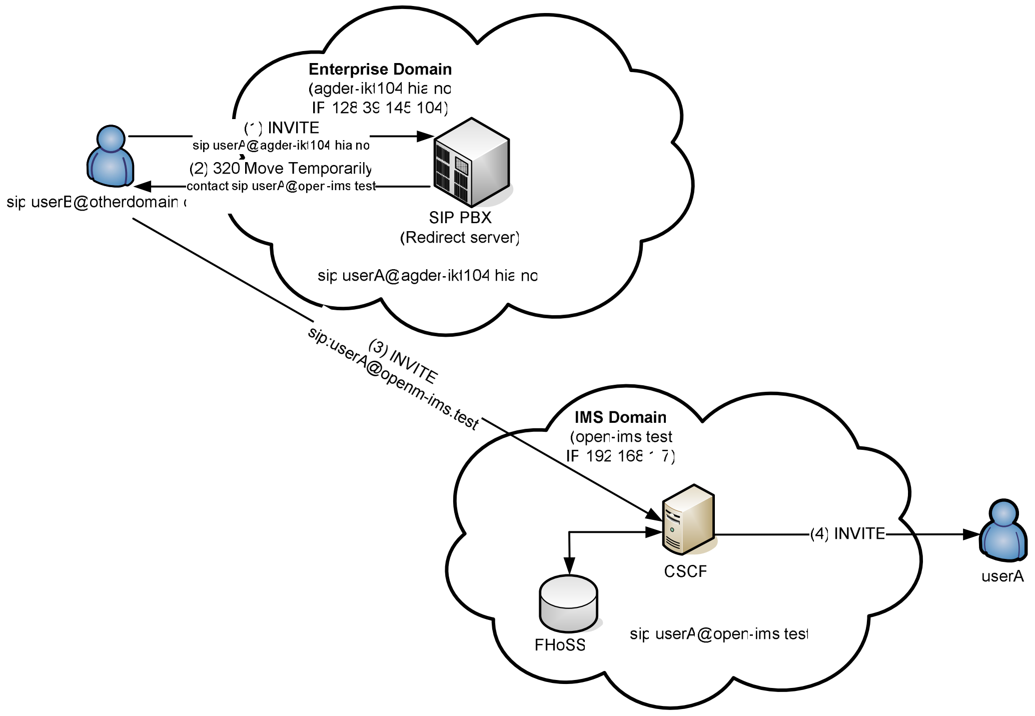


Figure 6-2 client based solution—Call session setup case

6.2 Implementation of the "client based" solution

As we showed in section 6.1, the "client based" solution needs a SIP PBX to inform about location changing.

The SIP PBX in the HiA/auSystems testbed is already setup with two different SIP servers, one is SIP PBX: Asterisk 1.2.4, while the other is an alternative SIP server: openSER v1.0.1. Currently the Asterisk SIP server is working, while the openSER SIP server was a test installation which is not working anymore. Hence, we tried approaches with both Asterisk and openSER servers.

6.2.1 Using Asterisk server

Follow the introduction in [8], if we use the Asterisk server, then this solution requires that the user terminal be SIP enabled and that domain settings are configurable, so that it can send SIP messages to the SIP PBX to inform the new location of the user.

Therefore, we cannot use normal SIP clients like X-Lite and GXP2000 which we used before. Instead, we use SIPp as our terminal where we can set the "contact" address by ourselves.

6.2.2 Using openSER server

To make the situation easy, we still want to use normal SIP clients. Therefore, we choose to add the openSER back to the testbed and let it work as a redirect server. After we change the old Asterisk server to openSER redirect server, we no longer need to use a "contact" function since the redirect server will automatically redirect all invites.

6.3 Registration and Call session setup with redirect server

As we see from the figure 6-3, User A registers to enterprise domain with the redirect server so that it gives the new location for further contact-information forwarding.

When User B tries to reach the User A in enterprise domain, the redirect server will tell User B the new location of User A, that is the redirect information. After User A receives this redirect information it will resend an INVITE request to the given address.

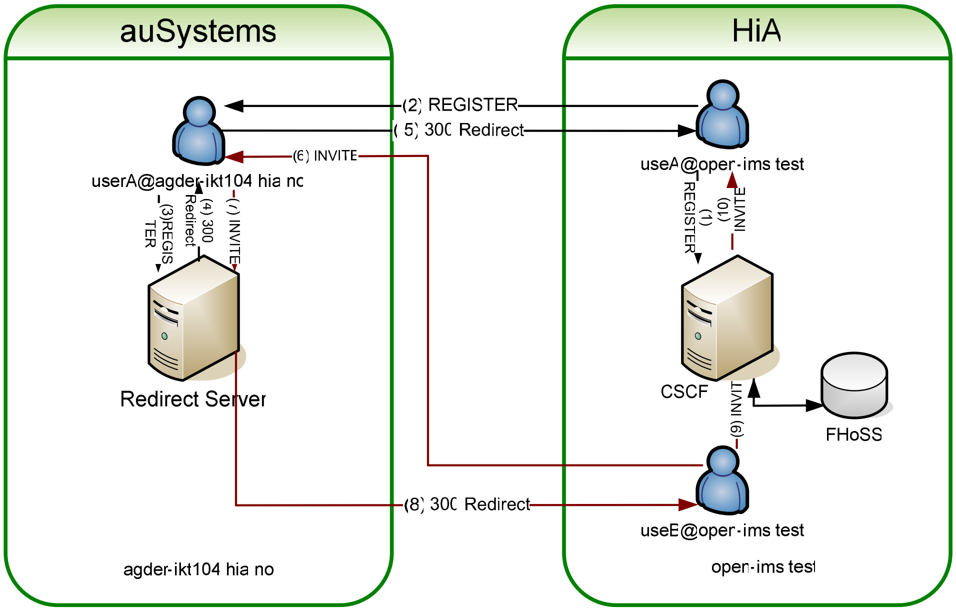


Figure 6-3 Registration and Call session setup with redirect server

In our case, we use “li” register to enterprise domain (agder-ikt104.hia.no) with X-Lite, and let the redirect server tell the new location as “li@open-ims.test “. Below we show the “300 redirect” message:

```
Session Initiation Protocol
Status-Line: SIP/2.0 300 Redirect
Message Header
Via:SIP/2.0/UDP
192.168.1.13:32824;branch=z9hG4bK-d87543-b523ee1a6248f76a-1--d87543-;rport=57621;recei
ved=128.39.145.250
To: "li"<sip:li@agder-ikt104.hia.no>;tag=b27e1a1d33761e85846fc98f5f3a7e58.cd9b
From: "li"<sip:li@agder-ikt104.hia.no>;tag=67468573
Call-ID: OWU3YzI0Njc2NTA2MTJkMmQ2ZDk0OWFmNGNiY2JkOTU.
CSeq: 1 SUBSCRIBE
Contact: sip:li@open-ims.test
Server: Sip EXpress router (0.9.6 (i386/linux))
Content-Length: 0
Warning: 392 128.39.145.104:5060 "Noisy feedback tells: pid=12416
req_src_ip=128.39.145.250 req_src_port=57621 in_uri=sip:li@agder-ikt104.hia.no
out_uri=sip:li@open-ims.test via_cnt==1"
```

Then we use registered IMS client “alice” and SIP client “fei” to call “li@agder-ikt104.hia.no” and let the call redirect to HiA domain “li@open-ims.test”.

In our experiment, the INVITE request didn’t reach the destination “li@open-ims.test”.

When we used IMS client to initiate the call, after the client had been told the new location of callee, there was no new INVITE request sent out to the new address. And when we used SIP client to call "li@agder-ikt104.hia.no", the new INVITE request only reached the P-CSCF in the OpenIMS and the P-CSCF sent another "300 Redirect" message instead of forwarding the request to the terminal client. In figure 6-4 below us show the process for our experiment when we used SIP client "fei" initiates the call.

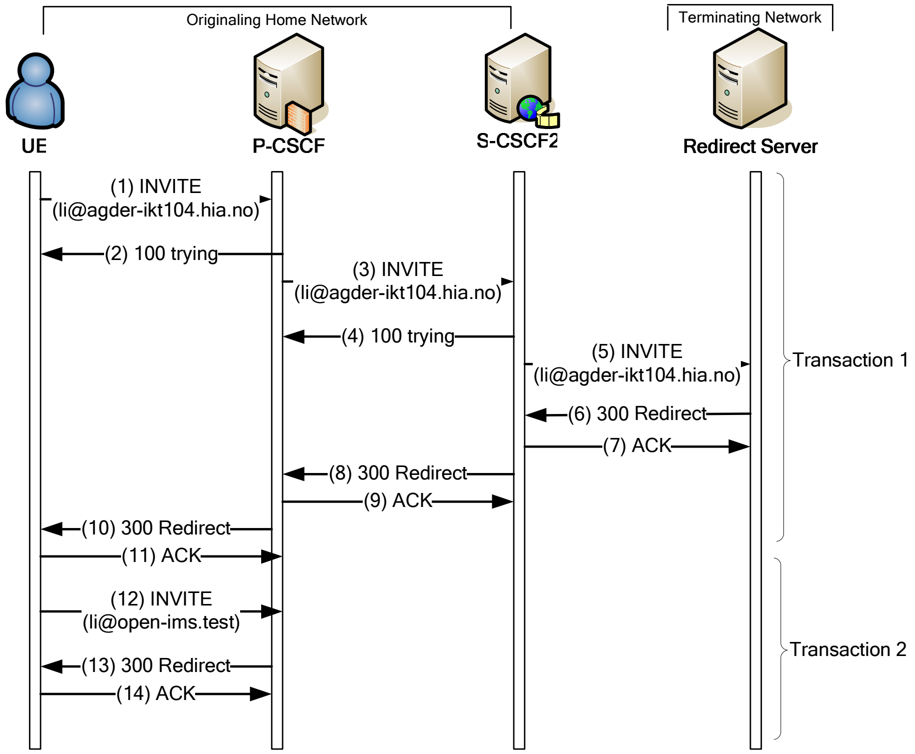


Figure 6-4 Call session setup between two domains

As we see from the figure 6-4, there are two transactions in this dialog. For transaction 1 the CSeq equals 1, and for transaction 2 the CSeq value should equals 2 because it happened after transaction 1, but the Call-IDs for those two transactions should be the same since they are in the same dialog. We show the (1) INVITE, (10) 300 Redirect, (11) ACK, (12) INVITE and (13) 300 Redirect messages in Appendix 2-7.

We checked our Wireshark trace for this call dialog and found out that the situation for INVITE requests in transaction 1 and in transaction 2 had different CSeq values and the same Call-ID values as they are suppose to, However, the "300 Redirect" response sent by P-CSCF in transaction 2 still had the same CSeq value as in the transaction 1. This shows that the P-CSCF still treats the new INVITE request to "li@open-ims.test" as the resend INVITE request to "li@agder-ikt104.hia.no", so it still responds with "300 Redirect" message instead of forwarding the request to the terminal user.

Due to the limited time for our project, we haven't solved this problem. It could be considered as the future work.

6.4 Evaluation of "client-based" solution

6.4.1 NAT issues

Since the "client-based" solution is related to two different domains, so it will refer to NAT (Network Address Translator) problem.

NAT is "a method by which IP addresses are mapped from one realm to another in an attempt to provide transparent routing to hosts". [14] If a SIP server A is behind a NAT gateway, SIP server B which is on another side of NAT will not able to contact server A.

For our situation, open-ims.test domain is located in the Agder Mobility Laboratory network which has only one external IP address. Therefore, when messages are transmitted within open-ims.test domain there are no NAT issues involved. However, when the messages are transmitted between the open-ims.test domain and agder-ikt104.hia.no domain, the NAT only helps for outgoing communication, that is, when the clients located in the open-ims.test domain want to send messages to clients in agder-ikt104.hia.no domain, the messages can go through the gateway and reach the clients in agder-ikt104.hia.no domain. But when the external clients in agder-ikt104.hia.no domain want to communicate with internal clients in open-ims.test domain, the messages will only reach the gateway but not the internal clients because they only know open-ims.test domain's external IP address.

So we have to do reconfiguration and let the gateway redirect the traffic. Therefore the messages which are sent from another domain can reach the intended recipient.

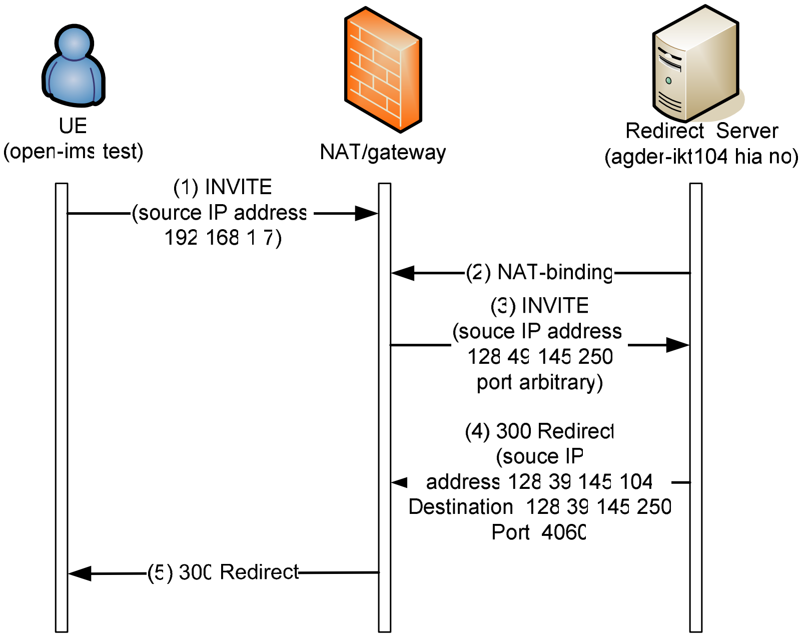


Figure 6-5 Call session setup related to firewall

6.4.2 Configuration of P-CSCF

400 Bad Request----Not following indicated Service-Routes

After we reconfigured the firewall, we tried to call from fei@open-ims.test to li@agder-ikt104.hia.no again, but we got a "400 Bad Request" response.

We checked the Wireshark trace and found out that the initiated INVITE request to li@agder-ikt104.hia.no and the new INVITE request to li@open-ims.test after redirection have the same Call-ID. According to the default configuration for P-CSCF, the P-CSCF will reply with "400 Bad Request----Not following indicated Service-Routes" message when the P-CSCF detects that the Call-IDs are the same for different INVITE request; therefore we got the 400 response.

To solve this problem, we changed the commands in the P-CSCF configuration file, letting it ignore the Call-ID problem and enforce the routes and let the dialog continue.

```
>               sl_send_reply("400","Bad Request - Not following indicated dialog routes");
>               break;
>               #Variant 2- enforce routes and let the dialog continue
> # P_enforce_dialog_routes("term");
……
```

Figure 6-6 original commands in pcscf.cfg

```
>               #sl_send_reply("400","Bad Request - Not following indicated dialog routes");
>               #Variant 2- enforce routes and let the dialog continue
>   P_enforce_dialog_routes("term");
>               break;

……
```

Figure 6-7 changed commands in pcscf.cfg

As shown in figure 6-6 and figure 6-7, we changed the commands, so the P-CSCF no longer sends “400 Bad Request” response. Instead, it will enforce routes and let the dialog continue.

7 Performance evaluation of OpenIMS

We have already established the OpenIMS testbed and have implemented the possible solution for interoperability between SIP and IMS. From the formal experiments, we know that the OpenIMS can work as well as the solution for interoperability between SIP and IMS. The task now is to evaluate if the OpenIMS works well enough. As the problem describes in 1.3.2, we will do the experiments on performance evaluation in such aspects.

- How many users can OpenIMS handle?
- Will the system always possible to handle user's requests? Or it will only be able to work sporadically?
- If the system can provide services under normal operations, will it also be able to react under un-normal situation, for example, when the system is overloaded?
- What is the latency for the system to respond users' requests? And what are the variations in this time?
- We could define a particular time interval and check how many users' request the system can handle during that period. And will there be any data loss during the process?

In this part, we choose SIPp as our test tool, for the reason that it is very powerful, flexible and free so that we can send all kinds of RFC3261 message with it.

7.1 Testing steps

As we refer to the command using in SIPp, there are several pre-requests before the test. The xml file and the csv file need to be written correctly first. Besides, in order to test how many users OpenIMS can handle, we need many subscribers existing in FHoSS firstly.

7.1.1 Add 100 subscribers in databases

As described in the first test case, the key problem is showing how many users OpenIMS can handle lies in how to get more subscribers in FHoSS as there are just two default ones in it. We could add several ones by manually to test if the OpenIMS works, but it is time consuming to add hundreds of thousands of subscribers in this way.

In our case, we decide to generate 100 subscribers automatically by using simple java programming. 'AddUser.java' is the file to generate 100 subscribers. It is attached in the Appendix 2-3. In this file, we generate 100 new subscribers with the names ranging from alice0001 to alice0100. Take alice0010 as an example. Its accordingly username is alice0010@open-ims.test and the keyword is alice0010. The other subscribers are configured in the same way. This should be confirmed because they will be used in xml and csv files for registration and call session.

After writing the java file, we need insert it in the sql file, which we can find under the directory */opt/OpenIMSCore/FHoSS/deploy/userdata.sql.* Then, the new 100 subscribers are established in the databases, and we can check them in the FHoSS user profile.

Note that the new 100 subscribers are created as SIP clients who register to the S-CSCF2 with authentication algorithm Digest-MD5. The reason is that we initiate the testing in the simplest way for SIP client – SIP client. And if time permits, we will extend it to SIP client – IMS client and IMS client – IMS client.

7.1.2 Write the xml files

As referred in the 7.1, the xml files are the most important parts is using SIPp. We write our own SIPp scenarios including *sip-reg.xml, sip-inv-uac.xml* and *sip-inv-uas.xml* basics on the templates gotten from [http://sipp.sourceforge.net/]. The xml files are attached in the Appendix 2-4 and 2-5. They are the cases for register and session set-up on the OpenIMS platform.

7.1.3 Use the commands to test

The following commands are using for registration and call sessions. The scenarios enable direct communication with the P-CSCF. And they can be called after saving the relevant files as xml-files.

```
sipp -sf reg-fei.xml 192.168.1.7:4060 -i 192.168.1.3 -p 5061 -m 1
sipp -sf reg-li.xml 192.168.1.7:4060 -i 192.168.1.3 -p 5060 -m 1
sipp -sf uas-fei-li.xml 192.168.1.7:4060 -i 192.168.1.3 -p 5060 -m 1
sipp -sf uac-fei-li.xml 192.168.1.7:4060 -i 192.168.1.7 -p 5061 -m 1
```

From these commands, we can see that the IP address of registrar is 192.168.1.7:4060 which towards to P-CSCF of OpenIMS. And the local IP address is 192.168.1.3, while fei uses 5061 to register and li uses 5060 to register.

-m means how many times the experiment will run. And in the above command, it just runs once.

```
sipp -sf reg.xml -inf reg.csv 192.168.1.7:4060 -i 192.168.1.3 -p 5061 -m 10
```

The above command is another situation. It uses 10 different subscribers to register. The *reg.xml* file is shown as follows:

```
REGISTER sip:[field0]@open-ims.test SIP/2.0
Via: SIP/2.0/[transport] [local_ip]:[local_port]
From: [field0]<sip:[field0]@open-ims.test>
To: [field0]<sip:[field0]@open-ims.test>
Call-ID:[call_id]
CSeq: 2 REGISTER
Contact: sip:[field0]@[local_ip]:[local_port]
[field1]
Content-Length: [len]
```

And the *reg.csv* is shown as follows:

```
fei; [authentication username=fei@open-ims.test password=fei]
li; [authentication username=li@open-ims.test password=li]
alice0001; [authentication username=alice0001@open-ims.test password=alice0001]
alice0002; [authentication username=alice0002@open-ims.test password=alice0002]
alice0003; [authentication username=alice0003@open-ims.test password=alice0003]
alice0004; [authentication username=alice0004@open-ims.test password=alice0004]
alice0005; [authentication username=alice0005@open-ims.test password=alice0005]
alice0006; [authentication username=alice0006@open-ims.test password=alice0006]
alice0007; [authentication username=alice0007@open-ims.test password=alice0007]
alice0008; [authentication username=alice0008@open-ims.test password=alice0008]
```

As shown from the above two figures, the blue color username is mapping to [field0], while the red color authentication information is mapping to [field1] in registration using for return 401 un-authentication message.

Note that the xml file and csv file are different from registration cases in regards to when to test call session.

```
INVITE sip:[field0]@open-ims.test SIP/2.0
Via: SIP/2.0/[transport] [local_ip]:[local_port]
From: [field0]<sip:[field0]@open-ims.test>
To: [field1]<sip:[field1]@open-ims.test>
Call-ID:[call_id]
Content-Length: [len]
```

```
fei; li
li; alice0001
alice0001; alice0002
alice0002; alice0003
alice0003; alice0004
alice0004; alice0005
alice0005; alice0006
alice0006; alice0007
alice0007; alice0008
alice0008; alice0009
```

These two figures show SIPp making 10 calls which are initiated by blue color users inviting red color users.

7.2 Performance evaluation of OpenIMS

When we use all 100 subscribers to register, the SIP signaling process is the same as it registers using only one subscriber, which means the OpenIMS can support 100 subscribers to registering in the same period.

When we do the experiments for making a call session, we design the cases as following:

First, we make 20 calls from SIPp to UCT IMS client, which means use 'fei' to call 'alice' (SIP to IMS client). In each call, the caller just sends one invite message. The results can be shown in the table 7-1 below.

Table 7-1 SIPp-UCT IMS client (SIP to IMS)

20 calls with 1 call each time	**number**	**reason**
Successful calls	19	
Failed calls	1	603 Decline error

And then we make the same situation from SIPp to X-Lite, which means use 'fei' to call 'li' (SIP to SIP client). The table 7-2 shows the numbers.

Table 7-2 SIPp-X-Lite (SIP to SIP)

20 calls with 1 call each time	**number**	**reason**
Successful calls	18	
Failed calls	2	600 Busy Everywhere error

The next case that we design is to make 20 calls, where the caller will send 30 invite message each time. We choose 30 calls each time because the SIPp is limited in sending 30 successful calls. The numbers are too small to see which way of using in OpenIMS network is better.

The table 7-3 shows the situation from SIPp to UCT IMS client (SIP to IMS)

Table 7-3 SIPp-UCT IMS client (SIP to IMS)

20 testing cases with 30 calls each time	**Original calls**	**Successful calls**
1	30	14
2	30	7
3	30	15
4	30	10
5	30	8
6	30	5

7	30	7
8	30	7
9	30	8
10	30	9
11	30	13
12	30	10
13	30	10
14	30	8
15	30	9
16	30	5
17	30	14
18	30	15
19	30	9
20	30	11

The table 7-4 shows the situation from SIPp to X-Lite (SIP to SIP)

Table 7-4 SIPp – X-Lite (SIP to SIP)

20 testing cases with 30 calls each time	**Original calls**	**Successful calls**
1	30	28
2	30	17
3	30	29
4	30	19
5	30	19
6	30	24
7	30	25
8	30	28
9	30	18
10	30	21
11	30	17
12	30	20
13	30	18
14	30	15
15	30	16
16	30	18
17	30	7
18	30	20
19	30	15
20	30	21

From these two tables and the numbers, the graphs can be compared as follow, in the figure the lengthways axes shows the number of the successful calls while the horizontal axes shows the sequence number of our testing cases.

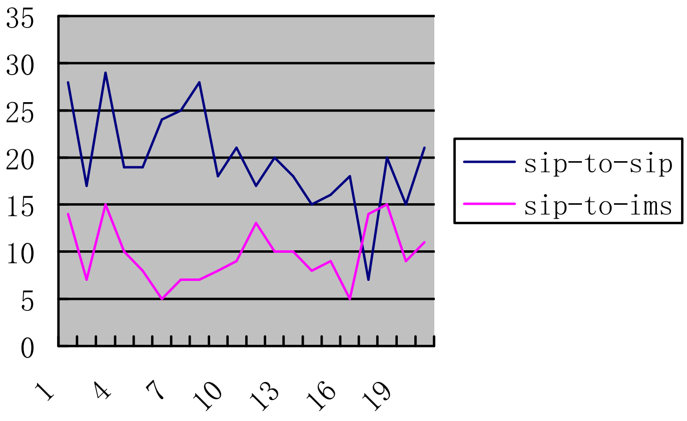


Figure 7-1 the difference cases access to OpenIMS

It is very clear from this graph that the performance of SIP-to-IP client is much better than the SIP-to-IMS client when accessing to the OpenIMS.

As for the reliability of OpenIMS, it is not so good for SIP client. The reason is that the SIP client registers to the S-CSCF2 which is added by us and the information of subscribers around it are always changing randomly. In that case, we always have to change the databases. But for the IMS client, the situation is much better because the IMS client register to the S-CSCF.

The OpenIMS can work well in the normal situation, as well as in the abnormal situation.

- When we make infinite calls through OpenIMS, it can work well for the first period, but it will become overloaded as the calls increase. The **600 busy** will be presented to warn.
- The other situation is that a **408 time out** will be shown.
- When the OpenIMS finds that the user is not in the databases, it will show **403 HSS forbidden** to inform operator to add it firstly.

And we also design the other situations. We define the fixed seconds to see how the message package changes in each procedure. The fixed time is 150 seconds, and the caller will send 30 calls each time. The table 7-5 will show the clear changes from SIP to UCT IMS client.

Table 7-5 SIP to UCT IMS

	10s	**15s**	**20s**	**30s**	**40s**	**50s**
REGISTER	30	30	30	50	47	43
401	4	30	30	14	47	43
REGISTER	4	30	30	14	47	43
200 OK	4	21	20	14	37	34
INVITE	4	21	20	14	37	34

100 trying	4	21	20	14	37	34
101 Dialog Establishment	4	17	14	14	27	28
180 Ringing	0	2	0	1	0	3
200 OK	0	0	0	0	0	0
ACK	0	0	0	0	0	0

The table 7-6 is almost is the same as the above one, just for SIP to SIP.

Table 7-6 SIP to SIP

	10s	**15s**	**20s**	**30s**	**40s**	**50s**
REGISTER	46	50	57	50	60	70
401	45	25	53	50	60	70
REGISTER	45	25	53	50	60	70
200 OK	35	25	41	42	49	65
INVITE	35	25	41	42	49	65
100 trying	35	25	41	42	49	65
101 Dialog Establishment	0	25	0	0	0	0
180 Ringing	0	0	0	0	0	0
200 OK	0	0	0	0	0	0
ACK	0	0	0	0	0	0

From the data of the above two tables, we can easily find that the performance of SIP-to-SIP is better than performance of SIP-to-IMS when they access the OpenIMS. But the SIP-to-IMS is much more stable than SIP-to-SIP.

8 Discussion

8.1 Functionality evaluation of OpenIMS

8.1.1 Evaluate SIP and IMS clients

8.1.1.1 UCT IMS

The UCT IMS client who was created directly for OpenIMS is much more stable than the other kinds of clients. It has very simple interface so it's easy to operate. However, the functionalities of UCT IMS client are limited. For example, it doesn't support multiple accounts simultaneously.

8.1.1.2 X-Lite

X-Lite was used as SIP client and it was impressive to use. It has a very nice and easy user interface with all the common controls, therefore, it's easy to operate. The functionalities of X-Lite are considerable, for example, it supports multiple simultaneous connected accounts. However, sometimes it's unstable.

8.1.1.3 Grandstream GXP-2000

GXP-2000 was also used as SIP client for our project. It has powerful functionalities and it's more stale compare to X-Lite. GXP-2000 has a web interface that can be used to configure general or advanced settings and up to four accounts. But we found out it's not so easy to operate. For example, by using the keys of the phone to dial username, we have to use phonebook and the processes are very complex. And every time after we change the information for accounts, we need to reboot it for updating and this take some time.

Table 8-1 compare of three SIP/IMS clients

	Reliability in OpenIMS	Functionality	Operation
UCT IMS client	Most steady	Less function	Middling
X-Lite 3.0 SIP client	Least steady	More function	Easy to operate
GXP-2000 SIP client	Middling	More function	Hard to operate

8.1.2 Functionality evaluation

To evaluate the functionalities of OpenIMS is very time-consuming. Because 3GPP TS 24.229 Release 6 has many sub-clauses described for different situation, so we need to check it very carefully and that took us some time.

The testing results showed that IMS clients' scenarios conform to 3GPP TS 24.229 better than SIP clients' scenarios. But all in all, for both IMS and SIP clients, the results are mostly conform to the specification. For detail information, we had described in chapter 4 and Appendix2-6.

8.2 Solutions of interoperability between IMS and SIP

We installed two S-CSCFs support different authentication algorithm for SIP clients and IMS clients at the same time. Although the solution is feasible, there are still some problems.

This part that gave us most headaches is instability of S-CSCF2 which was added by us to support SIP clients. After each registration of call, we couldn't continue to repeat the testing, but to get **403 errors - HSS returned no authentication vectors**. That is because the users' data in MySQL always change automatically and randomly. In section 5.4.1 of this thesis and in section 6 of [3GPP 04] Release 5, the reason for this problem have been discussed. After changing, the authentication algorithm and the selected S-CSCF for users cannot match. For this problem, we need to change the database back to set value manually each time we do new experiment.

In this stage, we also met another problem that is the call sessions were always released by the remote client automatically as soon as the UCT IMS client received calls from X-Lite. However, after we use SIPp as SIP clients instead X-Lite, this problem no more appeared. So we can say this is due to the drawback of X-Lite.

8.3 Integration with SIP/VoIP solutions

In this task we tried to implement "client based" solution for interoperability of non-IMS domain and IMS domain.

This solution was come up in [8] and was based on using Asterisk as SIP PBX. But to be supported by the idea, more functionality from the client are required. The client has to be SIP enabled and domain setting configurable, so it can send SIP message to SIP PBX to inform location changing. Many clients are not supported by this solution. Although this problem can be worked out in some situations, for example, "if the enterprise SIP PBX has a web GUI to maintain registrar, the current location can be updated through the internet" [8], it is still not very flexible.

Under this situation, we considered to use openSER as a redirect server instead Asterisk, so there are no extra requirement for clients.

Since this solution refer to two domains, so the NAT issues also considered in the project. However, this solution hasn't been implemented completely. Clients could register to non-IMS network with redirect server. But we failed to make calls. After redirect server told the caller about the redirect information, caller resend a new INVITE request to callee who located in OpenIMS domain in order to establish the call session. But the P-CSCF of OpenIMS didn't do its job to forward the request to callee so the call session setup failed. To solve this problem, we probably need to change the configuration file of P-CSCF. Nevertheless, the time was limited, so we didn't fix this problem before we finished project.

9 Conclusion & future work

9.1 Conclusion

In the period of doing this project, we finished three main tasks: evaluation of the functionality of OpenIMS components, evaluation the performance of OpenIMS and implementation of interoperability between OpenIMS and existing SIP/VoIP solutions.

For the functional evaluation, we validated if OpenIMS was conforming to 3GPP TS 24.229 Release 6 [3GPP 06]. The result is that most functionality of OpenIMS is conforming to 3GPP TS 24.229 R6, some of the functionalities are still based on 3GPP TS 24.229 R5. One of the components called SIP2IMS gateway is no longer used in OpenIMS core and can be deleted from it.

For the performance evaluation of OpenIMS, SIPp was used to test all kinds of scenarios. From the testing, we can see that OpenIMS can at least handle 100 subscribers. OpenIMS works well in “normal” situation as well as in the abnormal situation, and it always works well for IMS clients. However, for SIP clients, OpenIMS works not reliably. The reason was that for SIP clients who register to the S-CSCF2, which was added by us, but the information of subscribers around S-CSCF2 were always changed randomly.

The last task required implementation of a solution of interoperability between OpenIMS and existing SIP/VoIP solutions. Firstly, we solved the interpretability between SIP clients to IMS clients in a single domain. We used two S-CSCFs and let one support the authentication algorithm for SIP clients while the other one supported the algorithm for IMS clients. Therefore both SIP clients and IMS clients could register to OpenIMS at the same time, and the call could be established between SIP and IMS clients. In this step, we found out that the the S-CSCF2 is very unstable. In future project, this should be improved and probably involves some programming.

After this, we continued our work with study and implementation of the solutions for OpenIMS to interoperate with existing SIP/VoIP solutions in two domains. Here, we chose the ‘client-based’ solution which is proposed in [8]. This solution is using a redirect server so that the call can be established from a non-IMS network to IMS network. However, this solution hasn’t been implemented completely. Clients could register to non-IMS network with redirect server. But we failed to make call between two domain, and the reason is related to the P-CSCF in OpenIMS domain. Due to the limited time, we didn’t fix this problem before we finished the project.

As the OpenIMS was only published at the end of last year, it is still under development and improving, Some of our experiments and valuations are based on the version of OpenIMS of March, 2007.

9.2 Future work

Our research of OpenIMS is mainly on the core components which include FHoSS, P-CSCF, I-CSCF, S-CSCF, SIP2IMS and the interfaces between these components. In the future the OpenIMS core can be extended broadly with components such as MRF (Media Resource Function) and AS.

The next problem could involve application research. We have already studied OpenIMS and found it works sufficiently well. Future work can continue to study it as an application platform. This would be an important extension of the IMS reference implementation. The OpenIMS core provides different and necessary functionalities to Application Server to support various advanced applications and services, for example, presence and instant messaging, or conference and video support. It can be also outlined what 'enables' OpenIMS can offer to people, for example, QoS, security, etc.

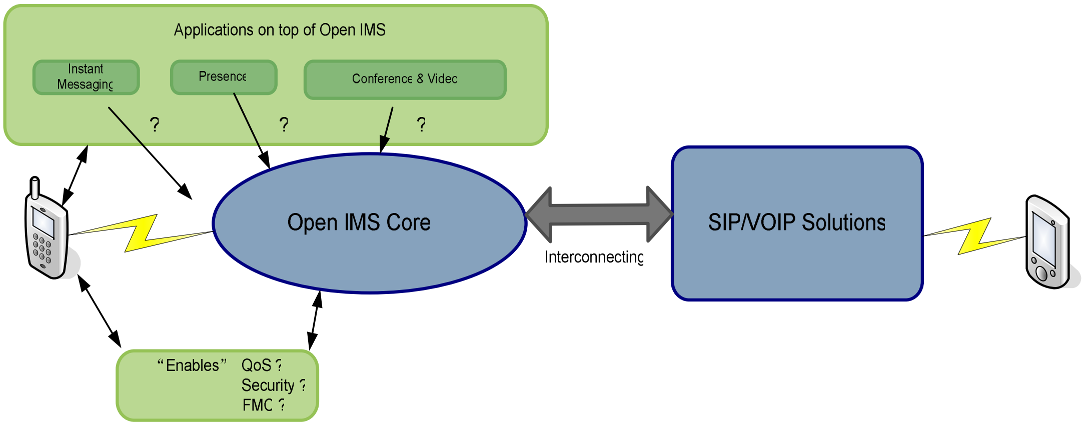


Figure 9-1 Application of OpenIMS

In our project all the components of OpenIMS were implemented in a single domain. So NAT and security were not considered, but we believe in the future is very necessary to consider these security issues.

10 References

[1] Gonzalo Camrillo, Miguel A. Garcia-Martin, “The 3G IP Multimedia Subsystem (IMS)”, ISBN 0-470-01818-6, May 2006

[2] T. Magedanz, D. Witaszek, K. Knuettel: “The IMS playground @ FOKUS-an open testbed for generation network multimedia services”, Testbeds and Research Infrastructures for the Development of Networks and Communities, 2005. Tridentcom 2005. First International Conference on 23-25 Feb. 2005

[3] Thomas Magedanz, Overview of the IMS – Principles, Architecture and Applications, FOKUS-IMS-Tutorial-public, 2006

[4] SER=SIP Express Router
URL: http://www.iptel.org/ser

[5] ERISSON white paper - IMS – IP Multimedia Subsystem, October, 2004
URL:http://www.ericsson.com/technology/whitepapers/ims_ip_multimedia_subsystem.pdf

[6] “Ericsson Push-to-Talk” from Ericsson Solutions
URL: http://www.ericsson.com/products/hp/IMS_Push_to_Talk_bs.shtml

[7] Dev2Dev – understanding the home subscriber server (HSS) Sh interface
URL: http://dev2dev.bea.com/pub/a/2006/10/home-subscriber-server.html

[8] Lian Wu; Anders Aasgaard,"Migration of VOIP/SIP Enterprise Solutions towards IMS", Master Thesis Report, Agder University College, June 2006
URL: http://ikt.hia.no/aml/papers/Report_2006_Lian_and_Anders-final.pdf

[9] Zhaopeng Chen,” Application of SIP Protocol in IMS system”, China Science and Technology Information, Jan 2006

[10] Ning Xue, “Next generation of VoIP protocol”, Telecommunication Technology, August 2005

[11] Zhaopeng Chen, “Application of SIP protocol in IMS system”, China Science and Technology Information, Jan 2006

[12] Jinke Wang, Yaliang Zhao, Shilei Shen, “ The Design and the implementation of SIP-Based Stack for IMS”, Journal of Henan University (Natural Science), Mar 2006, TP 393.08

[13] “NAT Traversal for Multimedia over IP Services” (visited Feb. 2006)
URL: http://www.newport-networks.com/whitepapers/nattraversal1.html

[14] Jae Cheon Han, Wook Hyun and Sun Ok Park, *“An Application Level Gateway for Traversal of SIP Transaction through NATs”*, Advanced Communication Technology, Vol 3. 2006

[FOKUS 01] The 2nd international FOKUS IMS Workshop “From FMC to Triple Play and NGN” on November 16/17, 2006

[FOKUS 02] “the Open Source IMS Core” from the 2nd International FOKUS IMS Workshop, the Fraunhofer Institute FOKUS, November 16, 2006
URL:http://www.fokus.fraunhofer.de/bereichsseiten/testbeds/ims_playgr

ound/OSIMS.php?lang=en

[FOKUS 03] "the Open Source IMS Core - CSCFs" from the 2nd International FOKUS IMS Workshop, the Fraunhofer Institute FOKUS, November 16, 2006

URL:http://www.fokus.fraunhofer.de/testbeds/ims_playground/CSCF.php?lang=en

[FOKUS 04] "the Open Source IMS Core - Presentation" from the 2nd International FOKUS IMS Workshop, the Fraunhofer Institute FOKUS, November 16, 2006

[FOKUS 05] "the Open Source IMS Core – SIP2IMS" from the 2nd International FOKUS IMS Workshop, the Fraunhofer Institute FOKUS, November 16, 2006
URL:http://www.fokus.fraunhofer.de/testbeds/ims_playground/SIP2IMS_GW.php?lang=en

[FOKUS 06] "the Open Source IMS Core - playground" from the 2nd International FOKUS IMS Workshop, the Fraunhofer Institute FOKUS, November 16, 2006
URL:http://www.fokus.fraunhofer.de/bereichsseiten/testbeds/ims_playground/playground/playground.php?lang=en

[FOKUS 07] "the Open Source IMS Core - FHoSS" from the 2nd International FOKUS IMS Workshop, the Fraunhofer Institute FOKUS, November 16, 2006
URL:http://www.fokus.fraunhofer.de/testbeds/ims_playground/FHoSS.php?lang=en

[FOKUS 08] FOKUS Home Subscriber Server (FHoSS)
URL:http://www.fokus.fraunhofer.de/bereichsseiten/testbeds/ims_playground/FHoSS.php?lang=en

[3GPP 01] 3GPP TS 23.002, Sh interface

[3GPP 02] 3GPP TS 29.328, Sh interface

[3GPP 03] RFC3588, DIAMETER Protocol, 3GPP TS 29.229 Diameter application,

[3GPP 04] 3GPP TS 29.228, Cx interface

[3GPP 05] 3GPP TS 33.220, Zh interface

[3GPP 06] 3GPP TS 24.229 Internet Protocol (IP) multimedia call control protocol based Session Initiation Protocol (SIP) and Session Description Protocol (SDP)

[Open IMS 01] Installation Guide
URL: http://www.openimscore.org/docs/install.html

[Open IMS 02] Open IMS Core FHoSS - Getting Start
URL: http://www.openimscore.org/docs/FHoSS/index.html

[Open IMS 03] Open IMS Core CSCFs – modules
URL: http://www.openimscore.org/docs/ser_ims/html/index.html

[Wikipedia 01] 'IP Multimedia Subsystem'

URL:http://en.wikipedia.org/wiki/IP_Multimedia_Subsystem#Architecture

[Wikipedia 02] 'Instant messaging'
URL: http://en.wikipedia.org/wiki/Instant_messaging

[Wikipedia 03] 'Asterisk (PBX)'
URL: http://en.wikipedia.org/wiki/Asterisk_%28software%29

[RFC01] B.Aboba and M.Beadles. The Network Access Identifier. RFC 2486, Internet Engineering Task Force, January 1999.

[RFC02] J.Rosenberg, J.Weinberger,C.Huitema, and R.Mahy. SIP:Session Initiation Protocol. RFC 3261, Internet Engineering Task Force, June 2002.

[RFC03] H.Schulzrinne.The tel URI for Telephone Number. RFC 3966, Internet Engineering Task Force, December 2004.

Appendices

Appendix 1 Glossary & Abbreviations

3GPP	3rd Generation Partnership Project
AKA	Authentication and Key Agreement
AS	Application Server
BGCF	Breakout Gateway Control Function
CSCE	Call State Control Function Server
CSCF	Call/Session Control Function
DAL	Data Access Layer
DNS	Domain Name System
FHoSS	FOKUS HSS
GGSN	Gateway GPRS Support Node
GPL	GNU General Public License
GSM	Global System for Mobile Communication
GRUU	Globally Routable User Agent
GUI	Graphical User Interface
HLR	Home Location Register
HSS	Home Location Register
I-CSCF	Interrogating-CSCF
IETF	Internet Engineering Task Force
IM SSF	IP Multimedia-Services Switching Function
IMS	IP Multimedia Subsystem
IP	Internet Protocol
IP CAN	IP Connectivity Access Network
ISUP	ISDN User Part
MAP	Mobile Application Part
MGCF	Media Gateway Controller Function
MGW	Media Gateway
MRF	Media Resource Function
MRFC	Media Resource Function Controllers
MRFP	Media Resource Function Processes
MSC	Mobile Switching Center
NGN	Next Generation Networking
OSA-SCS	Open Service Access-Service Capability Server
PA	Presence Agent
PCG	Project Co-ordination Group
P-CSCF	Proxy CSCF
PLMN	Public Land Mobile Network
PSTN/CS	Public Switched Telephone Network/Circuit Switched

PTT	Push to Talk
QoS	Quality of Service
S-CSCF	Serving CSCF
SEG	Security Gateway
SER	SIP Express Router
SGW	Signaling Gateway
SLF	Subscriber Location Function
SIP	Session Initiation Protocol SIP ISP
TAPI	Telephony Application Programming Interface
THIG	Topology Hiding Internetwork Gateway
TR	Technical Reports
TS	Technical Specifications
TSG	Technical Specification Group
TTS	Text-to-Speech
UE	User Equipment
URI	Uniform Resource Identifier
VoIP	Voice over Internet Protocol
WLAN	Wireless LAN

Appendix 2 Project Management

Appendix 2-1 DNS zone-file

```
$ORIGIN open-ims.test.
$TTL 1W
@                       1D IN SOA         localhost. root.localhost. (
                                          2006101001        ; serial
                                          3H                ; refresh
                                          15M               ; retry
                                          1W                ; expiry
                                          1D )              ; minimum
                        1D IN NS          ns
ns                      1D IN A           127.0.0.1
pcscf                   1D IN A           192.168.1.7
open-ims.test.          1D IN A           127.0.0.1
icscf                   1D IN A           127.0.0.1
_sip                    1D SRV 0 0 5060   icscf
_sip._udp               1D SRV 0 0 5060   icscf
_sip._tcp               1D SRV 0 0 5060   icscf
open-ims.test.                    1D   IN   NAPTR   10   50   "s"   "SIP+D2U"   ""
_sip._udp.open-ims.test.
open-ims.test.                    1D   IN   NAPTR   20   50   "s"   "SIP+D2T"   ""
_sip._tcp.open-ims.test.
scscf                   1D IN A           127.0.0.1
sip2ims                 1D IN A           127.0.0.1
hss                     1D IN A           127.0.0.1
ue                      1D IN A           127.0.0.1
presence                1D IN A           127.0.0.1
```

Appendix 2-2 UML figures of SIP signaling

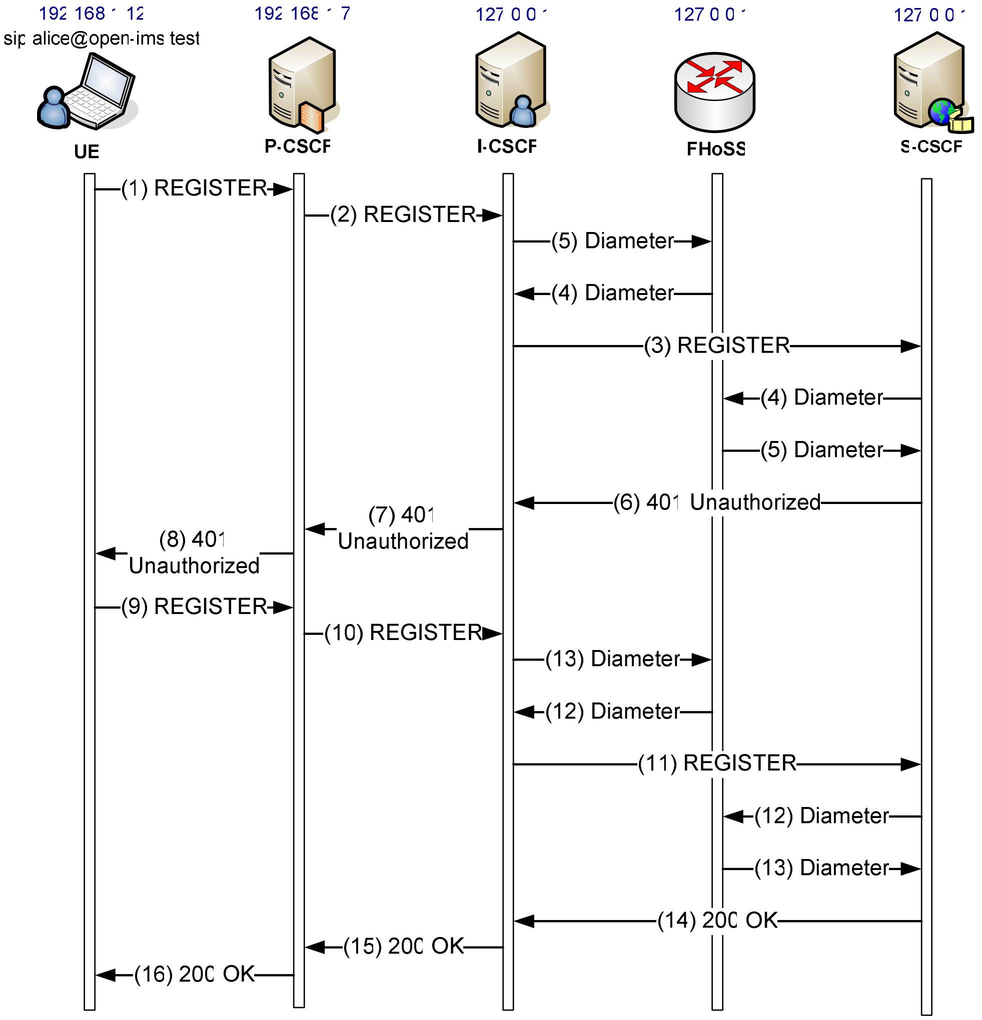


Figure A Registration for IMS client in the OpenIMS

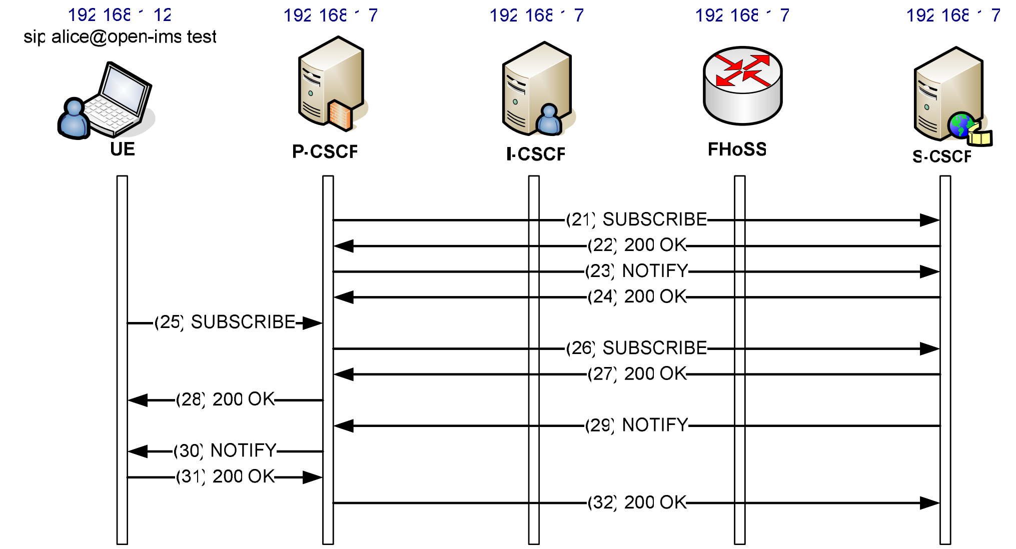


Figure B Subscribe to reg Event

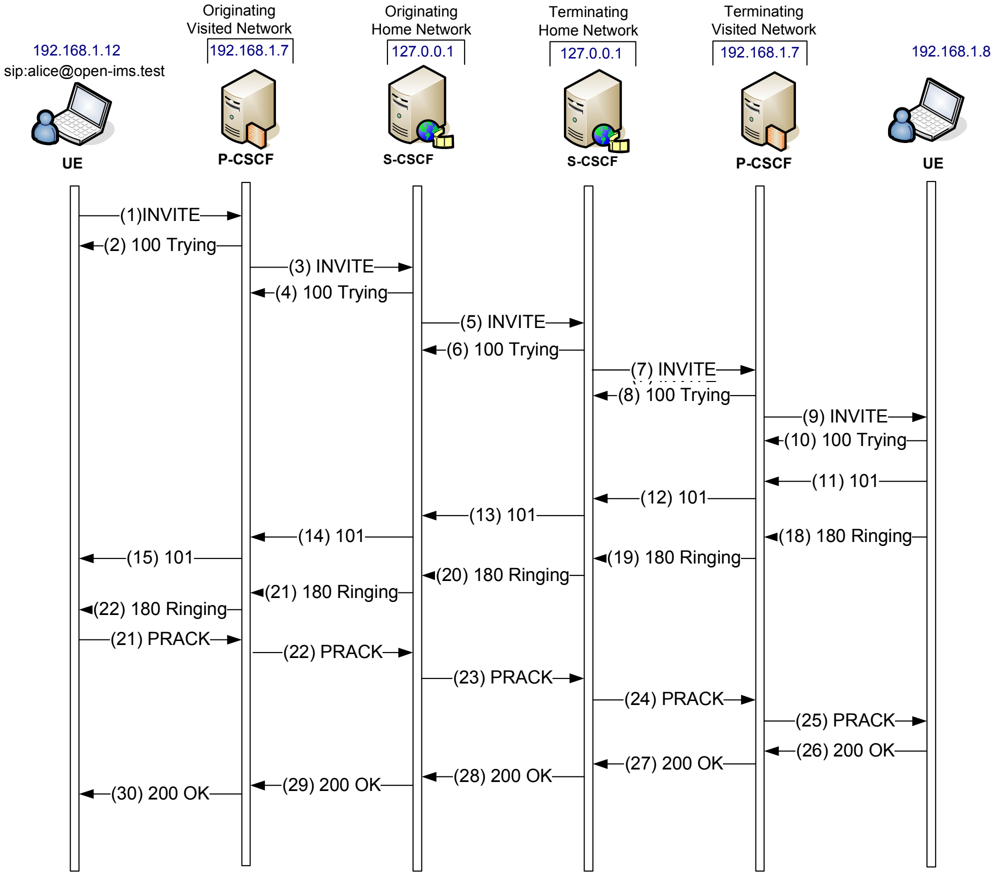


Figure C IMS client basic session setup, part 1

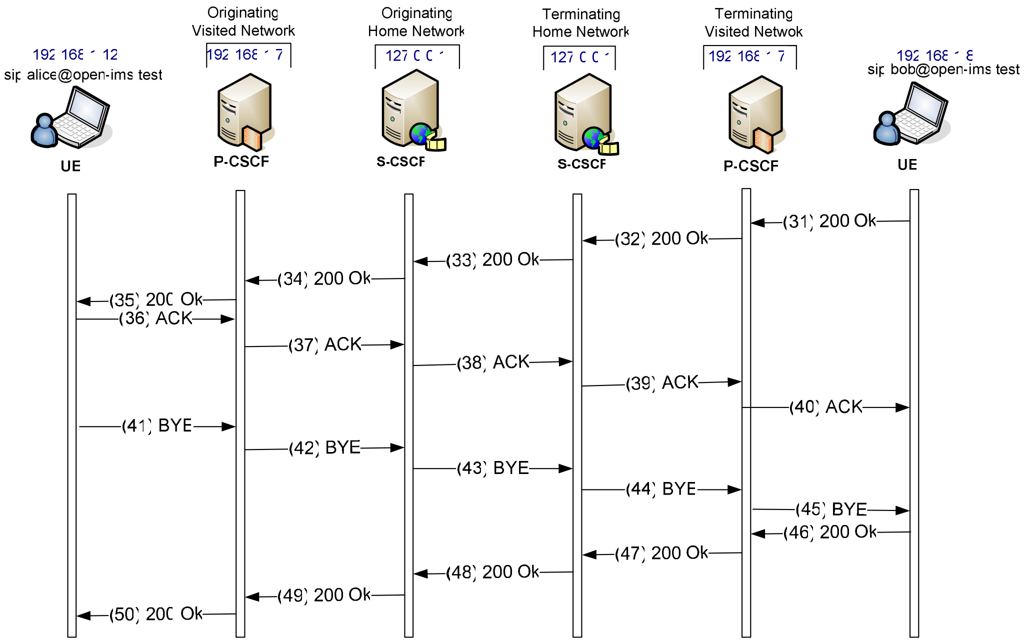


Figure D IMS client basic session up, part 2

Appendix 2-3 AddUser.java file

```
public final class AddUser {
        public static void main(String[] args) {

                System.out.println("use hssdb;");

                for (int i = 1; i <= 100; i++) {
                String num = "" + i;
                int zeroesToAdd = 4 - num.length();

                for (int j = 0; j < zeroesToAdd; j++) {
                     num = "0" + num;
                }

                System.out.println("insert into imsu(name) values ('alice" + num + "_imsu');");
                System.out.println("insert into impi(impi_string, imsu_id, imsi, scscf_name,
s_key, chrg_id, sqn) values('alice" + num + "@open-ims.test', (select imsu_id from imsu
where imsu.name='alice" + num + "_imsu'), 'alice" + num + "_ISDN_User_part_ID',
'sip:scscf2.open-ims.test:4060', '616c6963650000000000000000000000', (select chrg_id from
chrginfo where chrginfo.name='default_chrg'), '000000000000');");
                System.out.println("insert into impu(sip_url, tel_url, svp_id) values ('sip:alice"
+ num + "@open-ims.test','tel:00491234" + num + "', (select svp_id from svp where
svp.name='default_sp'));");
                System.out.println("insert into impu2impi(impi_id, impu_id) values ((select
impi_id from impi where impi.impi_string='alice" + num + "@open-ims.test'), (select
impu_id from impu where impu.sip_url='sip:alice" + num + "@open-ims.test'));");
                System.out.println("insert into roam(impi_id, nw_id) values((select impi_id
from impi where impi.impi_string='alice" + num + "@open-ims.test'), (select nw_id from
networks where networks.network_string='open-ims.test'));");
                }
        }
}
```

Appendix 2-4 XML file of REGISTER using in SIPp

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE scenario SYSTEM "sipp.dtd">

<scenario name="sip-to-sip call">

<send retrans="500">
<![CDATA[
REGISTER sip:open-ims.test SIP/2.0
Via: SIP/2.0/[transport] [local_ip]:[local_port]
Route: <sip:pcscf.open-ims.test:4060;lr>
Max-Forwards: 70
From: "fei" <sip:fei@open-ims.test:4060>
To: "fei" <sip:fei@open-ims.test:4060>
P-Access-Network-Info: 3GPP-UTRAN-TDD;utran-cell-id-3gpp=C359A3913B20E
Call-ID: [call_id]
Contact: <sip:fei@[local_ip]:[local_port]>;transport=[transport]
Content-Length: 0
Supported: path
Expires: 300
CSeq: 1 REGISTER
User-Agent: Sipp v1.1-TLS, version 20061124
]]>
</send>

<recv response="401" auth="true" rtd="true">
<action>
<ereg regexp=".*" search_in="hdr" header="Service-Route" assign_to="1" />
</action>
</recv>

<send retrans="500">
<![CDATA[
REGISTER sip:open-ims.test SIP/2.0
Via: SIP/2.0/[transport] [local_ip]:[local_port]
Route: <sip:pcscf.open-ims.test:4060;lr>
Max-Forwards: 70
From: "fei" <sip:fei@open-ims.test>
To: "fei" <sip:fei@open-ims.test>
P-Access-Network-Info: 3GPP-UTRAN-TDD;utran-cell-id-3gpp=C359A3913B20E
Call-ID:[call_id]
CSeq: 2 REGISTER
Contact: <sip:fei@[local_ip]:[local_port]>
Expires: 300
Content-Length: 0
[authentication username=fei@open-ims.test password=fei]
Supported: path
User-Agent: Sipp v1.1-TLS, version 20061124
]]>
</send>

<recv response="200">
</recv>

</scenario>
```

Appendix 2-5 XML file of INVITE using in SIPp

UAC

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE scenario SYSTEM "sipp.dtd">

<scenario name="sip-to-sip call">

<send retrans="500">
<![CDATA[
INVITE sip:li@open-ims.test SIP/2.0
Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
Route: <sip:orig@scscf2.open-ims.test:4060;lr>
From: "fei" <sip:fei@open-ims.test:4060>;tag=[call_number]
To: "li" <sip:li@open-ims.test:4060>
Call-ID: [call_id]
CSeq: [cseq] INVITE
Contact: <sip:fei@[local_ip]:[local_port]>
Max-Forwards: 70
Subject: Performance Test
Content-Type: application/sdp
Content-Length: [len]

v=0
o=- 0 2 IN IP4 192.168.1.3
s=-
c=IN IP4 192.168.1.3
t=0 0
m=audio [media_port] RTP/AVP 0
a=rtpmap:0 PCMU/8000

]]>
</send>
<recv response="100" optional="true"/>
<recv response="180" optional="true"/>
<recv response="200" rtd="true"/>

<send>
<![CDATA[
ACK sip:li@[local_ip]:[local_port] SIP/2.0
Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
Route: <sip:mo@pcscf.open-ims.test:4060;lr>
Route: <sip:mt@scscf.open-ims.test:6060;lr>
Route: <sip:mt@scscf.open-ims.test:6060;lr>
From: "fei"<sip:fei@open-ims.test>;tag=[call_number]
To: "li"<sip:li@open-ims.test>[peer_tag_param]
Call-ID: [call_id]
CSeq: [cseq] ACK
Contact: <sip:fei@[local_ip]:[local_port]>
Max-Forwards: 70
Subject: Performance Test
Content-Length: [len]
]]>

</send>
<![CDATA[
BYE sip:fei@[local_ip]:[local_port] SIP/2.0
Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
From: "fei"<sip:fei@open-ims.test>;tag=[call_number]
To: "li"<sip:li@open-ims.test>[peer_tag_param]
Call-ID: [call_id]
CSeq: [cseq] BYE
Contact: <sip:li@[local_ip]:[local_port]>
Max-Forwards: 70
Subject: Performance Test
Content-Length: [len]
]]>
</send>

<recv response="200"crlf="true"/>

</scenario>
```

UAS

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE scenario SYSTEM "sipp.dtd">

<scenario name="uac-uas(sip-sip call), server-side">

<recv request="INVITE">
</recv>

<send>
<![CDATA[
SIP/2.0 180 Ringing
[last_Via:]
[last_Record-Route:]
[last_From:]
[last_To:];tag=[call_number]
[last_Call-ID:]
[last_CSeq:]
Contact: <sip:li@[local_ip]:[local_port]>
Content-Length: [len]
]]>
</send>

<pause milliseconds="2000"/>

<send retrans="500">
<![CDATA[
SIP/2.0 200 OK
[last_Via:]
[last_Record-Route:]
[last_From:]
[last_To:];tag=[call_number]
[last_Call-ID:]
[last_CSeq:]
Contact: <sip:li@[local_ip]:[local_port]>
Allow:
INVITE,REGISTER,ACK,BYE,INFO,REFER,NOTIFY,SUBSCRIBE,MESSAGE,CAN
CEL
Content-Type: application/sdp
Content-Length: [len]

v=0
o=- 0 2 IN IP4 [local_ip]
s=-
c=IN IP4 [media_ip]
t=0 0
m=audio 40000 RTP/AVP 8 0 18
a=rtpmap:8 PCMA/8000
a=rtpmap:0 PCMU/8000
a=rtpmap:18 G729/8000
]]>
</send>

<recv request="ACK" crlf="true">
</recv>
```

```
<recv request="BYE">
</recv>

<send>
<![CDATA[
SIP/2.0 200 OK
[last_Via:]
[last_From:]
[last_To:]
[last_Call-ID:]
[last_CSeq:]
Content-Length: 0
]]>
</send>

</scenario>
```

Appendix 2-6 Testing results

In the square bracket "[ ]" show the sub-clause in 3GPP TS 24.229 Release 6 specification.

- Evaluation at the UE

[5.1.1] Registration and authentication
[5.1.1.2] initial registration
IMS client
On sending a REGISTER request, the UE populate the header fields with an Authorization header, a From header, a To header, a Contact header, a Via header, an Expires header, a Request-URI, a Supported header that accord with the 3GPP TS 24.229 completely. However, there are some parts are not consistent with the standards.
-The UE suppose to associate two parts, a protected client port and a protected server port, but in our situation, we only find the protected server port without association
-We have no Security-Client header
-There is no P-Access-Network-Info header.

On receiving the 200(OK) response to the REGISTER request, as showed in the table, our situation accord with most of the standards, but there are still some differences:
-There is no P-Associated-URI header.
-There is no security association lifetime shows.

When a 401(Unauthorized) response to a REGISTER is received the UE is barely behave as the standards says. Except that derive the keys CK and IK as described in 3GPP TS 32.203, there is no temporary set of security associations has been set up, no Security-Client header and there is no Authorization header.

SIP client
On sending a REGISTER request, the UE populate the header fields contain an Authorization header, a From header set to the SIP URI containing the public user identity, a To header set to the SIP URI containing the public user identity to be registered, and a Contact header, a Via header, and a Request-URI that are consistent to the standards. But there is no security-client header, no P-Access-Network-Info header, and no Supported header. Besides, the expire parameter in the Contact header set to the value 3600 but not 600 000 seconds.

On receiving the 200(OK) response to the REGISTER request, as showed in the table, our situation accord with most of the standards, but there are still some differences:
-The UE doesn't store the expiration time of the registration.
-There is no P-Associated-URI header.

-There is no security association lifetime shows.

[5.1.1.3] Initial subscription to the registration-state event package

IMS client

On sending a SUBSCRIBER request, the UE populate the header fields with a Request URI set to a SIP URI that contains the public user identity used for subscription, and a From header, a To header, an Event header, an Expires header, a Contact header that are consistent as the standards described. The only difference is that there is no P-Access-Network-Info header.

Upon receipt of a 200 response to the SUBSCRIBE request, the UE stores the information for the established dialog and the expiration time as indicated in the Expires header of the received response.

SIP client

On sending a SUBSCRIBER request, when UE populate the header fields, there are some parts are not consistent of the standards.
-There is no P-Access-Network-Info header
-The Event header set to “message-summary” instead “reg”.
-The Expires time set to”300”, but not”600000”.

2xx response never reached UE, it only forward to the P-CSCF.

[5.1.1.5] Authentication

[5.1.1.5.1] General

The 401 situation is already discussed in 5.1.1.2.

On receiving the 200(OK) response for the protected REGISTER request, for both SIP client and IMS client, there is no security association provided.

[5.1.1.5.2] Network-initiated re-authentication

Since there is no timer F expires at the UE, so we don’t consider this situation that described in this sub-clause.

[5.1.1.6] User-initiated deregistration

IMS client

On sending a REGISTER request, the UE populate the header fields with an Authentication header, a From header, a To header, a Contact header, a Via header, an Expires header, and a Request-URI that accord with the description in 3GPP TS 24.229 completely. The differences are:
-There is no Security-Client header.
-There is no Security-Verify header.
-There is no P-Access-Network-Info header.

On receiving the 200 (OK) responses to the REGISTER request, the UE removed all the registration details relating to the public user identity. And since there are no more public user identities registered, the UE deleted the related keys that may towards to the IM CN subsystem.

SIP client

The X-Lite does not support deregistration.

[5.1.1.7] Network-initiated deregistration

Upon receiving the NOTIFY request on the dialog which was generated during subscription to the reg event package, the UE contains a <registration> element with the state attribute set to “terminated”. But the event attribute is a little different from the standards: it is set to “unregistrated” but not to “rejected” or “deactived”.

[5.1.2] Subscription and notification

[5.1.2.1] Notification about multiple registered public user identities

[5.1.2.2] General SUBSRIBER requirements

The UE doesn’t receive a 503 response, so we don’t need to consider what described in this sub-clause.

[5.1.3] Call initiation-mobile originating case

[5.1.3.1] Initial INVITE request

For both SIP client and IMS client, our situation is the originating UE does not require local resource reservation.

Upon generating an initial INVITE request, the UE indicates the support for reliable provisional response and the support for the preconditions mechanism by using the Supported header. And it doesn’t indicate the requirement for the precondition mechanism by using the Require header mechanism.

● Evaluation at the P-CSCF

Generally speaking, the functionality of the P-CSCF is conformant to the specification of 3GPP R6.

[5.2.1] General

As the description of 3GPP TS 24.229, the P-CSCF of OpenIMS support the Path and Service-Route headers, and the Path header is only used in the REGISTER request and its 200 (OK) response, while the Service-Route header is only applicable to the 200 (OK) response of REGISTER request.

The difference in our case is: there is not P-Charging-Function-Addresses header. Therefore, the functionality of P-CSCF with P-Charging-Function-Addresses header is not considered.

The other difference is without P-Media-Authorization header in our case, because what we concentrate on is just OpenIMS Core which the AS is not included.

Both IMS Client and SIP Client get the same situation.

[5.2.2] Registration

In the registration, the P-CSCF is preparing to receive only the initial REGISTER requests on the SIP default port values or on the port advertised to the UE during the P-CSCF discovery procedure.

Most procedures in registration are conformant with TS 24.229. But, we don't consider the security, so, the REGISTER request is not protected. And the parameter "integrity-protected" is inserted with the value "no".

Although the REGISTER request is not protected in our cases, the Security-Client header is not existed. The reason is that, the architecture of the OpenIMS Core in our case is too simple to include the security, because all the components are fixed in a single domain.

For the state that P-CSCF receives a 401 (Unauthorized) response to a REGISTER request, the P-CSCF perform almost the same as the specification, but we could not evaluate the security around it, because there are not security associations, Security-Server, reg-await-auth timer in our case.

For the state that P-CSCF receives a 200 (OK) response to a REGISTER request, some of the functionality is different. At first, there is no Contact header can be checked. And then, there is no P-Asserted-Identity header. Next difference is P-CSCF cannot store the values received in the P-Charging-Function-Address header for the reason that in our case, there is no P-Charging-Function-Address header. The last difference is a term-ioi parameter is not received in the P-Charging-Vector header. the security association is not considered.

[5.2.3] Subscription to the user's registration-state event package

For the situation that upon receipt of a 200 (OK) response to the initial REGISTER request, the different cases for P-CSCF performs as following.

The P-CSCF will generate a SUBSCRIBE request but the From header is not set to the P-CSCF's SIP URI. It set as: sip:alice@open-ims.test which is a Public User Identity's SIP URI. And the Expires header is still set to 600000 which is the same as the Expires header indicated in the 200 (OK) response to the REGISTER request.

[5.2.5] Deregistration

For the SIP Client, it doesn't support the functionality of deregistration. For the IMS Client, there are some functionalities of deregistration are different from the specification.

[5.2.5.1] User-initiated deregistration

When the P-CSCF receives a 200 (OK) response to a REGISTER request sent by the

UE, the Expires header will be checked, in the situation that the expires parameter equal zero, the difference for the P-CSCF of OpenIMS does not remove the Public User Identity found in the To header field.

[5.2.6] General treatment for all dialogs and standalone transactions excluding the REGISTER method

[5.2.6.3] Requests initiated by the UE

When the P-CSCF receives an initial request for a dialog or a request for a standalone transaction, the request of IMS client contains a P-Preferred-Identity header, so the P-CSCF shall identify the initiator of the request by that public user identity. As to the SIP client, the situation is different. The request of SIP client doesn't contain a P-Preferred-Identity header, so, the P-CSCF shall identity the initiator of the request by a default public user identity.

There is no Service-Route header in our situation, therefore, we don't consider the related cases.

Both of the IMS and SIP client add its own address to the Via header which the situation is conformant to the specifications.

When the P-CSCF receives a 1xx or 2xx response to the before request, the P-CSCF shall not store the values received in the P-Charging-Function-Address header, cause we don't have this header in our cases.

5.2.6.4 Request terminated by the UE

When adding P-CSCF's own SIP URI to the top of the list of Record-Route headers and save the list, the P-CSCF build the P-CSCF SIP URI in a format that contains the rport parameter where is not conformant to the specification.

In the situation that P-CSCF receives a 1xx or 2xx response to the request, the P-CSCF performs mostly conformant to the specification. But the case is different for SIP client and IMS client when P-CSCF verifies the list of URIs received in the Record-Route header.

5.2.7 Initial INVITE

5.2.7.1 Mobile-originating case

When the P-CSCF receives from the UE an INVITE request, the P-CSCF shall respond to all INVITE requests with a 100 (Trying) provisional response which is conformant to the specification. But the P-CSCF doesn't insert the P-Media-Authorization header containing that media authorization token.

And it is the same in **5.2.7.3 Mobile-terminating cas**e, cause we don't have the P-Media-Authorization header.

And for **5.2.7.4 Access network charging information**, we don't consider it because we don't have the access network.

5.2.8 Call release

5.2.8.1.2 Release of an existing session

The situation is conformant to the specification, but it is different from IMS client to SIP client here. For IMS client, the P-CSCF serves the *calling* user of the session it shall generate a BYE request based on the information saved for the related dialog. And for SIP client, the P-CSCF serves the *called* user of the session it shall generate a BYE request based on the information saved for the related dialog.

And we don't consider the situation about security association.

● Evaluation at the I-CSCF

[5.3.1] Registration procedure

Generally speaking, the I-CSCF behaves as a stateful proxy during the registration procedure.

[5.3.1.2] Normal procedures

The I-CSCF decides which HSS to query, and possibly as a result of a query to the Subscription Locator Functional (SLF) entity. But in the OpenIMS Core, the SLF is not included.

[5.3.2] Initial requests

The I-CSCF behaves as a stateful proxy for initial requests.

[5.3.2.1] Normal procedures

All components in our situation are in a signal domain, therefore, we don't consider the IP connective access network. That's the reason why we don't have *P-Access-Network-Info* headers

Besides, as the same reason, we can not see the procedures about I-CSCF shown in the Wireshark log messages.

There is a situation is different on IMS Client and SIP Client:

When the I-CSCF receives an initial request for a dialog or standalone transaction, we trace the log messages about IMS Client, and found that, the I-CSCF remove its own SIP URI from the topmost *Route* header, and route the request based on the *Request-URI* header field. While the trace on SIP Client, the situation is different. I-CSCF contains more than one *Route* header, and I-CSCF at first remove its own SIP URI from the topmost *Route* header, and then forwarding the request based on the topmost *Route* header.

[5.3.3] THIG functionality in the I-CSCF

We don't consider the situation about THIG, as the reason that the visited network and the home network are the same in our case

● Evaluation at the S-CSCF

[5.4.1] Registration and authentication

[5.4.1.1] Introduction

The S-CSCF acts as the SIP registrar for UA belonging to the IM CN subsystem.

IMS client

For IMS client situation, the S-CSCF supports the Path header, the Service-Router header, the Require header, and also the Supported header. But it still cannot accord with the standards completely. Because according to the standard, the Path header should only applicable to the REGISTER request and its 200OK, and the Service-Router header should only applicable to the 200OK of REGISTER, but in our situation, both of the header also appears when S-CSCF receiving the "401 Unauthorized-Challenging the UE".

SIP client:

In accordance with the 3GPP TS 24.229, the S-CSCF supports the Path header (only applicable to the REGISTER request and its 200OK), the Service-Router header (only applicable to the 200OK response of REGISTER), and also support the Require header. However, it does not support the Supported header.

[5.4.1.2] Initial registration and user-initiated reregistration

[5.1.1.2.1] Unprotected REGISTER

As says in NOTE 2, if a REGISTER request with Expires header value equal to zero should always be received protected, but for both SIP client and IMS client, the Expires header value are not equal to zero, so our REGISTER request is unprotected.

IMS client

When receiving a REGISTER request with the "integrity-protected" parameter set to "no", the IMS client accord with the standards better than SIP client. Except the timer reg-await-auth haven't been started, others are consistent to 3GPP TS 24.229.

SIP client:

Upon receipt of a REGISTER request without the "integrity-protected" parameter, the S-CSCF behave almost as the standards says, but there is no IK, CK parameters in the WWW-Authenticate header, and because is SIP client, so the security mechanism is MD5 but no AKAV1-MD5. Besides, in normal case, the S-CSCF doesn't start the timer reg-await-auth.

[5.4.1.2.2] Protected REGISTER

Since our REGISTER request is unprotected, so we don't consider this sub-clause.

[5.4.1.3] Authentication and re-authentication

This situation we already discussed in 5.4.1.2.

[5.4.1.4] User-initiated deregistration

IMS client

Since the "integrity-protected" parameter in Authorization header set to "no", according to the standard, S-CSCF apply the procedures described in sub-clause 5.4.1.2.1

SIP client

X-Lite cannot been deregistered by user.

[5.4.2] Subscription and notification

[5.4.2.1] Subscriptions to S-CSCF events

[5.4.2.1.1] Subscription to the event providing registration state

When an incoming SUBSRIBER request addressed to S-CSCF arrives containing the Event header with the reg event package, the S-CSCF shall check if the request was generated by a subscriber who is authorized to subscribe to the registration state of this particular user. For both SIP client and IMS client, the S-CSCF can find the identity for authentication of the subscription in the P-Asserted-Identity header received in the SUBSRIBER request. And the S-CSCF stores the value of the orig-ioi parameter received in the P-Charging-Vector header.

IMS client

When generate a 200 response to the SUBSCRIBER request, the S-CSCF populate an Expires header set to the same value as the Expires header in SUBSCRIBE request which is accord with the standards.

SIP client

When generate a 200 response to the SUBSCRIBER request, the S-CSCF populates an Expires header set to a value that is higher than the Expires header in SUBSCRIBE request, this is the opposite as described in the 3GPP TS 24.229.

[5.4.2.1.2] Notification about registration state

IMS client

For each NOTIFY on all dialogs which have been established due to subscription to the reg event package of the user, the S-CSCF set the Request-URI and Router header to the saved route information during subscription, and set the Event header to the "reg". In the body of the NOTIFY request contains a <registration> elements and for each <registration> element, the S-CSCF set the aor attribute to one public user identity, and set the<uri> sub-element inside the <contact> sub-element of the <registration> element to the contact address. Under this situation, if the public user identity has been deregistered, then S-CSCF sets the state attribute in the <registration> element to "terminated", sets the state attribute in the <contact>

element to “terminated” and set the event attribute in the <contact> element to “unregistered”.

However, there is no P-Charging-Vector header for the NOTIFY request which is different as the standard says.

SIP client

For SIP client X-Lite, we got “487 Event Package Not Supported”.

[5.4.3] General treatment for all dialogs and standalone transactions excluding requests terminated by the S-CSCF

[5.4.3.1] Determination of mobile-originated or mobile-terminated cases

For both IMS client and SIP client, upon receipt of an initial request or a target refresh request or a stand-along transaction, the S-CSCF perform the procedures for the mobile-originating case as described in 3GPP TS 24.229 sub-clause 5.4.3.2, and the S-CSCF remove the “orig” parameter from the topmost Route header.

[5.4.3.2] Requests initiated by the served user

IMS client

When S-CSCF receives an initial request for a dialog or a request for a standalone transaction from the served user, the S-CSCF first determines whether the request contains a barred public user identity in the P-Accessed-Identity header field of the request or not. For our situation, there is non-barred public user identity.

Our example accord with most of the situations as described in standards, but there are still some differences:

-The S-CSCF stores the value of the orig-ioi parameter received in the P-Charging-Vector header, but it doesn’t remove it from the forwarded request.

-The S-CSCF doesn’t insert a P-Charging-Function-Addresses header and have no knowledge that the SIP URI contained in the received P-Asserted-Identity header is an alias SIP URI for a tel URI (We didn’t use tel URI).

- Since the networking is not needed, so the S-CSCF doesn’t put the address of the I-CSCF to the topmost route header.

-The S-CSCF doesn’t remove the P-Access-Network-Info header based on the destination user (Request-URI) or when it receives a target refresh request from the served user.

-There is no access-network-charging-info parameter in the P-Charging-Vector header field.

SIP client

Almost all the situations are have the same result as IMS client example except that there is no original dialog identifier that the S-CSCF previously placed in a Router header is present in the topmost Route header of the incoming request.

[5.4.3.4] Original dialog identifier

As described before, our SIP client example doesn't show the original dialog identifier.

[5.4.4] Call initiation

[5.4.4.1] Initial INVITE

For both SIP client and IMS client, when the S-CSCF receives an INVITE request, the S-CSCF processes the initial INVITE request without examining the SDP.

[5.4.4.2] Subsequent requests

[5.4.4.2.1] Mobile-originating cases

According to the 3GPP TS 24.229, when the S-CSCF receives 1xx or 2xx response, the S-CSCF shall insert a P-Charging-Function-Addresses header and store the access-network-charging-info parameter in it when receiving the request containing the access-network-charging-info parameter in the P-Charging-Vector. But in our situation, for both SIP client and IMS client,the S-CSCF doesn't insert the P-Charging-Vector header.

When the S-CSCF receives any request or response (excluding ACK requests and CANCEL requests and responses) related to a mobile-originated dialog or standalone transaction, the S-CSCF may insert save value into P-Charging-Vector and P-Charging-Function-Addresses headers before forwarding the message within the S-CSCF home network, however in our testing, the S-CSCF didn't insert it.

[5.4.4.2.2] Mobile-terminating case

For both SIP client and IMS client, our situation is not consistent to the standards. When S-CSCF receives the any 1xx or 2xx response, the S-CSCF doesn't insert te P-Charging-Function-Addresses header, and when the S-CSCF receives 180(Ringing) or 200OK(to INVITE) response, the response are not contain the access-network-charging-info parameter, and not contain the P-Charing-Vector.

- ## Evaluation at the Cx

In the square bracket "[ ]" show the sub-clause in 3GPP TS 24.229 Release 6 specification.

[6] Diameter application for Cx interface

[6.1] Command-Code values

In our situation, there are several commands appear which are User-Authorization-Request (UAR), User-Authorization-Answer (UAA), Server-Assignment-Request (SAR), Server-Assignment-Answer (SAA), Location-Info-Request (LIR), Location-Info-Answer (LIA),Multimedia-Auth-Request (MAR), Multimedia-Auth-Answer (MAA). For both IMS client and SIP client, our examples are mostly accord with the 3GPP TS 29.229. We have all the mandatory AVPs and most optional AVPS in those commands. However, there are no "Registration-Termination-Request (RTR)", "Registration-Termination-Answer (RTA)", "Push-Profile-Request (PPR)" and "Push-Profile-Answer" commands in our examples.

[6.2] Result-Code AVP values

[6.2.1] Success

For both IMS client and SIP client in our example, there are two values stand for success that are "DIAMETER_RIRST_REGISTRATION" (2001) and "DIAMETER_SUBSEQUENT_REGISTRATION" (2002).

The "DIAMETER_RIRST_REGISTRATION"(2001) is appeared in MAA, SAA and LIA commands while the "DIAMETER_SUBSEQUENT_REGISTRATION" (2002) is appeared in UAA command during the registration process.

[6.2.2] Permanent Failures

When we use GXP-2000 as SIP client to register, there are "DIAMETER_ERROR_USER_UNKNOWN" (5001) stand for permanent failures. It appears in the last UAA command in the process of registration.

[6.3] AVPS

There are several AVPs that are showed in the table6.3.1 of 3GPP TS 29.229 appeared in our examples. We describe them individually below.

[6.3.1] Visited-Network-Identifier AVP (600)

For both IMS client and SIP client, it appears in the UAR command, and the values is: open-ims.test.

[6.3.2] Public-Identity AVP (601)

IMS client

The Public-Identity appears in UAR, MAR, SAR and LIR commands when using IMS client to register. The value is "sip:alice@open-ims.test".

SIP client

When using X-Lite as SIP client, it appears in UAR, MAR, SAR and LIR commands and the value is "sip:fei@open-ims.test".

When using GXP-2000 as SIP client, it appears in UAR command and the value is "sip:li@open-ims.test".

[6.3.3] Server-Name AVP (602)

When using IMS client and using X-Lite as SIP client to register, the Server-Name AVP appears in UAA, MAR, SAR and LIA commands and the value is “sip:scscf.open-ims.test:6060”.

When using GXP-2000 as SIP client to register, it only appears in UAA command and the value is also “sip:scscf.open-ims.test:6060”.

[6.3.7] User-Data AVP (606)

For both IMS client and X-Lite as SIP client, the User-Data AVP appears in SAA commands. When using GXP-2000 as SIP client, there is no User-Data AVP appears.

[6.3.8] SIP-Number-Auth-Items AVP (607)

For both IMS client and X-Lite as SIP client, the SIP-Number-Auth-Items AVP appears in MAR, MAA commands. When using GXP-2000 as SIP client, there is no SIP-Number-Auth-Items AVP appears.

[6.3.13] SIP-Auth-Data-Item AVP (612)

For both IMS client and X-Lite as SIP client, the SIP- Auth-Data-Items AVP appears in MAR, MAA commands. The value for IMS client is” Digest-AKAv1-MD5” while the value for X-Lite is “Digest-MD5”.When using GXP-2000 as SIP client, there is no SIP-Auth-Data-Item AVP appears.

[6.3.15] Server-Assignment-Type AVP (614)

For both IMS client and X-Lite as SIP client, the Server-Assignment-Type AVP appears in SAR command. When using GXP-2000 as SIP client, there is no Server-Assignment-Type AVP appears.

[6.3.19] Charging-Information AVP (618)

For both IMS client and X-Lite as SIP client, the Charging-Information AVP appears in SAA command. When using GXP-2000 as SIP client, there is no SIP-Number-Auth-Items AVP appears.

[6.3.24] User-Authorization-Type AVP (623)

Only when using GXP-2000 as SIP client to register the User-Authorization-Type appears. And the value is “REGISTRATION (0)”.

[6.3.25] User-Data-Already-Available AVP (624)

For both IMS client and SIP client, it appears in SAR command.

Appendix 2-7 Messages from call session for “client-based” solution

- **(1) INVITE**

```
Session Initiation Protocol
Request-Line: INVITE sip:li@agder-ikt104.hia.no SIP/2.0
Message Header
Via:SIP/2.0/UDP192.168.1.13:2668;branch=z9hG4bK-d87543-8409b2001c6c9c59-1
    --d87543-;rport
Max-Forwards: 70
Route: <sip:orig@scscf2.open-ims.test:4060;lr>
Contact: <sip:fei@192.168.1.13:2668>
To: "li@agder-ikt104.hia.no"<sip:li@agder-ikt104.hia.no>
From: "fei"<sip:fei@open-ims.test>;tag=ce7c1c2f
Call-ID: ZWZiZjVlMTJkM2E3ZWJkMDI5ZmUxOTZiNTM1MzhhNDY.
CSeq: 1 INVITE
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY, MESSAGE,
       SUBSCRIBE, INFO
Content-Type: application/sdp
User-Agent: X-Lite release 1006e stamp 34025
Content-Length: 325
```

- **(10) 300 Redirect**

```
Session Initiation Protocol
Status-Line: SIP/2.0 300 Redirect
Message Header
Via:SIP/2.0/UDP192.168.1.13:2668;branch=z9hG4bK-d87543-8409b2001c6c9c59-1
    --d87543-;rport=2668
To:"li@agder-ikt104.hia.no"<sip:li@agder-ikt104.hia.no>;tag=b27e1a1d33761e85846
   fc98f5f3a7e58.fc09
From: "fei"<sip:fei@open-ims.test>;tag=ce7c1c2f
Call-ID: ZWZiZjVlMTJkM2E3ZWJkMDI5ZmUxOTZiNTM1MzhhNDY.
CSeq: 1 INVITE
Contact: sip:li@open-ims.test
Server: Sip EXpress router (0.9.6 (i386/linux))
Content-Length: 0
Warning:  392   128.39.145.104:5060   "Noisy   feedback   tells:    pid=12415
          req_src_ip=128.39.145.250                        req_src_port=51836
          in_uri=sip:li@agder-ikt104.hia.no          out_uri=sip:li@open-ims.test
          via_cnt==2"
```

- **(11) ACK**

```
Session Initiation Protocol
Request-Line: ACK sip:li@agder-ikt104.hia.no SIP/2.0
Message Header
Via:SIP/2.0/UDP192.168.1.13:2668;branch=z9hG4bK-d87543-8409b2001c6c9c59-1
    --d87543-;rport
Route: <sip:orig@scscf2.open-ims.test:4060;lr>
To:"li@agder-ikt104.hia.no"<sip:li@agder-ikt104.hia.no>;tag=b27e1a1d33761e85846
   fc98f5f3a7e58.fc09
From: "fei"<sip:fei@open-ims.test>;tag=ce7c1c2f
Call-ID: ZWZiZjVlMTJkM2E3ZWJkMDI5ZmUxOTZiNTM1MzhhNDY.
CSeq: 1 ACK
Content-Length: 0
```

- **(12) INVITE**

```
Session Initiation Protocol
Request-Line: INVITE sip:li@open-ims.test SIP/2.0
Message Header
Via:SIP/2.0/UDP192.168.1.13:2668;branch=z9hG4bK-d87543-5a2372669626033f-1-
    -d87543-;rport
Max-Forwards: 70
Route: <sip:orig@scscf2.open-ims.test:4060;lr>
Contact: <sip:fei@192.168.1.13:2668>
To: "li@agder-ikt104.hia.no"<sip:li@agder-ikt104.hia.no>
From: "fei"<sip:fei@open-ims.test>;tag=ce7c1c2f
Call-ID: ZWZiZjVlMTJkM2E3ZWJkMDI5ZmUxOTZiNTM1MzhhNDY.
CSeq: 2 INVITE
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY, MESSAGE,
       SUBSCRIBE, INFO
Content-Type: application/sdp
User-Agent: X-Lite release 1006e stamp 34025
Content-Length: 325
```

- **(13) 300 Redirect**

```
Session Initiation Protocol
Status-Line: SIP/2.0 300 Redirect
Message Header
Via:SIP/2.0/UDP192.168.1.13:2668;branch=z9hG4bK-d87543-8409b2001c6c9c59-1
   --d87543-;rport=2668
To:"li@agder-ikt104.hia.no"<sip:li@agder-ikt104.hia.no>;tag=b27e1a1d33761e85846
   fc98f5f3a7e58.fc09
From: "fei"<sip:fei@open-ims.test>;tag=ce7c1c2f
Call-ID: ZWZiZjVlMTJkM2E3ZWJkMDI5ZmUxOTZiNTM1MzhhNDY.
CSeq: 1 INVITE
```

```
Contact: sip:li@open-ims.test
Server: Sip EXpress router (0.9.6 (i386/linux))
Content-Length: 0
Warning: 392 128.39.145.104:5060 "Noisy feedback tells: pid=12415
         req_src_ip=128.39.145.250 req_src_port=51836
         in_uri=sip:li@agder-ikt104.hia.no out_uri=sip:li@open-ims.test
         via_cnt==2"
```