TOWARDS FUTURE 4G MOBILE NETWORKS: A REAL-WORLD IMS TESTBED

Sara El Alaoui, Fatima Zohra Smaili, Omar Bougamza, Mohamed Riduan Abid

School of Science and Engineering,AlAkhawayn University in Ifrane, Morocco
Sa.elalaoui@aui.ma
F.Smaili@aui.ma
O.Bougamza@aui.ma
R.Abid@aui.ma

ABSTRACT

In the near future, current mobile communication networks will converge towards an All-IP network in order to provide richer applications, stronger customer satisfaction, andfurther return on investment for the industry. However, such a convergence induces a strong level of complexity when handling interoperability between different operators and different handset vendors. In this context, the 3GPP consortium is working on the standardization of the convergence, and IMS is emerging as the internationally agreed upon standard that is multi-operator and multi-vendor. In this paper, we shed further light on the subtleties of IMS, and we delineate a blueprint for the implementation of a real-world IMS testbed. An open source Presence Server is deployed as well. The operation of the IMS testbed and the Presence Server are checked to assess their conformance with 3GPP standards. A simple third party application is developed on top the IMS testbed to further assess its operation.

KEYWORDS

4G, IMS, SIP, Presence Server, Third Party Applications

1. Introduction

The design of next generation mobile networks is mainly driven by IP being the successful technology for present and future mass market communications. While these latter are becoming multimodal (e.g., voice, video, presence, messaging), the end-user is becoming more and more demanding in terms of Quality of Service (HD video, and high quality voice), especially when using mobile terminals.

Current radio access technologies (e.g., Wi-Fi and 3G) are limited in terms of QoS, and 3GPP is actively working to overcome this limitation. In this context the 4G LTE (Long Term Evolution) technology [1] is quite mature and ready to use. The only remaining problem towards an All-IP NGN (Next Generation Network) network is solving the interoperability problem between the various and different existing operators' infrastructure networks. So far, there is only one technology that is multi-vendor, multi-operator, and internationally standardized, that is IMS.

IMS is evolving as the signaling core for telecommunications industry to embrace the IP technology. It will provide global interoperability between all handsets (whatever), and all Telecommunications and Networking operators worldwide (wherever and whenever). Thus, IMS will handle universal service access, i.e., wherever an end user is roaming in the world, he/she

DOI : 10.5121/ijngn.2012.4303

should be provided with the same set of services. This should be seamless as the user does not need to do any configuration or whatever to assure a transparent roaming.

In this paper we shed further light on the IMS architecture, and provide a blueprint for developers to build innovative applications on top of IMS. The blueprint constitutes of: delineating the steps to deploy a real-world IMS testbed using open source software, delineating the steps to deploy a Presence Server as the crucial Application Server in IMS, and surveying APIs to provide a richer platform for developing Third Party Applications. These latter are very likely to constitute the future killer applications in IT, and this stems from the main fact that, in NGNs, we can reach everyone without worrying about the operator he/she is affiliated with or the handset he/she is using at the very moment.

The rest of the paper is organized as follows. Section 2 introduces 4G Mobile networks. IMS is highlighted in Section 3. The details of the deployment of the IMS system are delineated in section 4, followed by the testing of our system using a simple Sip Client (X-Lite) and an IMS Client (UCT IMS Client) respectively in section 5. Section 6 highlights the deployment of the presence server. In section 7, an example of a third party application, developed using SIP JAIN, is presented.Finally, we conclude in Section 8.

2. 4G Mobile Networks

Mobile communications are rapidly changing from being an expensive communications technology used by few people to a ubiquitous technology used by most people in the planet. Mobile communications started with the first generation (1G) in 1980s which were analog; This wasborrowed from the military. The second generation (2G) appeared in early 1990s (GSM, IS-95 CDMA), and it was the first digital mobile communication system. This introduced, for the first time, data services, e.g., text messaging. However, these data services were still carried over the circuit-switched network. Data over packet-switched networks became a reality in the second half of 1990s with the introduction of GPRS (General Packet Radio Systems) and EDGE (Enhanced Data for GSM Evolution). This was referred to by 2.5G. The migration towards 3G (e.g., IMT-2000, UMTS) happened in early 2000s, and it carried data over packet-switched networks while voice carrier remained over circuit-switched networks. 4G, which is the next generation of mobile networks, adoptedLTE as the new wireless broadband access technology; this will allow users to access networks with rate up to 1 Gbps (This will enable watching TV and movies from a cellular phone), a fact that will render mobile technology more and more attractive. 4G will have both data and voice carried over packet-switched, thus becoming an ALL-IP network.

Indeed, and thanks to the success of IP networks, next generation of mobile networks will be all connected to the packet-switched IP network, i.e., the Internet, and this will pave the path towards interoperability between the different operators networks, which are currently using different core networks, e.g., X25, Frame Relay, etc.

Once interoperable, the different operators' networks can develop richer applications rendering the end-used more and more attached to use his/her handset in connecting to the world. A typical goal, is to allow a seamless roaming between all operators while having a multi-device-access feature, e.g., a user can use his WiFi-enabled device to call his/her partner who is using a mobile phone or setting in front of his/her desktop PC.

Providing such inter-operability is not an easy task since it involves multiple operators who are using different technologies, and multiple end-user equipment vendors that are loosely coupled to

the underlying network technology. 3GPP is actively working to finalize the standardization process that will cope with the interoperability issue. The main 3GPP main standard axes are [2]:

- LTE (Long Term Evolution): The Wireless Access technology.
- EPC (Evolved Packet Core): The All-IP core network that will provide all the functionalities that was realized through the two separate sub-domains of circuit-switched for voice, and packet-switched for data in previous generations (2G, 3G).
- IMS (IP Multimedia Subsystem): The signaling core that is responsible for establishing sessions/connections between end-users. The next section introduces IMS.

3. IMS - IP Multimedia Subsystem

In the telecommunications realm, locating called parties and initiating/maintaining connections are the first major tasks to provide. H.323 is the most prevalent protocol for session initiation, and it was largely adopted in PSTNs (Public Switched Telephone Networks) and other networks as well, e.g., ISDN and ATM. Even if some operators are still using it for IP networks as well, it has been proved that H.323 is not very suitable and efficient simply because it was initially designed and shaped for non-IP networks; and since 4G networks are converging to an All-IP network, the need for an initiation protocol that fits the IP stack appears crucial. In this context, 3GPP designated SIP (Session Initiation Protocol) as the standard initiation protocol for 4G networks.

IMS is the 3GPP standard architecture for delivering multimedia services in next generation networks. IMS is a logical layer that runs over IP and uses SIP for sessions' initiation. Besides, sessions' initiation and multimedia delivery, IMS will provide very efficient accounting and charging services for operators. The major strength of IMS is that it is a multi-operator and multi-handset-vendor platform that will allow end users to connect regardless of their location (i.e., operator) and their handset type. This will exponentially increase the connections establishments among users in the world which will generate an enormous return on investment for industry and for developers.

In contrast to current IP networks which only care about the connection (i.e., the IP address), IMS further focuses on the end-user profiles and on allowing for finding, authenticating, getting the right end user profile (e.g., presence), then establishing an IP connection. Once the connection is over, IMS will produce the charging. After all, the operators and service developers do not have to care about the interoperability between the different networks. In other words, this should be totally transparent. For instance, in current VoIP solutions, besides the fact that no QoS is assured, end-users are constrained to use the same software (e.g., Skype, Windows Live), which constrains connections between users using different platforms.

In conclusion, IMS ensures that the level of complexity induced by handling different operators, accesses, and handset manufacturers, is not an issue that the developer has to be concerned about. All above should be transparent to the developer, thus allowing him to concentrate only on the logic of the service to provide. And the most promising venue for developers is to adapt and create new applications that fit into handsets as these latter are becoming the most preferable communication devices,to the extent that this would shape future access to the Internet. Besides, in contrast to nowadays fixed phones which are attached to a place, handsets are instead attached to persons; thus we can easily define persons' profiles rather than places' profiles.

Besides, nowadays handsets are far more than an ordinary phone; they are a camera, an audio recorder, a browser, a gaming station, or even a credit card. In other words, future handsets are to contain quite whatever electronic you can take with you (An ALL-in-one electronic device). This further highlights the gigantesque existing floor for building new applications.

4. IMS Deployment

To deploy IMS we started first by investigating existing platforms, and most platforms we found were realized by companies/operators for commercial purposes mainly. Some of these main operators that are deploying IMS, considering IMS or have made a decision to deploy an IMS network are: Alcatel Lucent, Cisco, Nokia Siemens Networks, Tekelec Networks. As we went deeper into our research we landed at last on the Open IMS Core platform [3] which is developed by FokusFraunhofer Institute in Germany [4].

4.1. Fokus Open IMS Core

FOKUS is an independent research organization that worked closely with other research institutes (especially the Technical Institute of Berlin) to develop software for Research and Development (R&D) purposes. During the 2nd International FOKUS IMS Workshop on November 16, 2006 [3], FOKUS officially launched its Open Source IMS Core project. FOKUS Open IMS Core project aims to fill the IMS void in the Open Source software landscape, as there were practically no Open Source platforms to develop an IMS Core. Following the standard IMS architecture set by 3GPP, Open IMS Core consists of the following components (See Figure 1):

- The Call Session Control Functions (P-CSCF, I-CSCF, S-CSCF): The central components of the Open IMS Core. These three CSCFs are extensions of the SIP Expression Router (SER) [5] which is basically a free SIP server developed to act as a SIP registrar, proxy or redirect server.
- FOKUS Home Subscriber Server (HSS): The database that takes care of managing user profiles and associated routing rules.

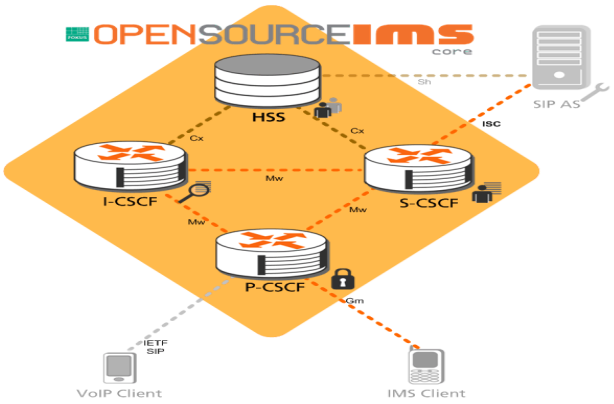

Figure 1: The Fokus Open IMS Core Architecture [3]

4.2. Deployment

The first step was deploying the four Fokus IMS Core components in one station in order to ensure minimum level of complexity and interdependency between the different IMS Core components. We went through of prerequisites steps [6, 7], which consisted on installing a Linux platform, JDK, GCC, MySQL-server, etc. We chose Ubuntu 11.04 to install our test bed for its availability and compatibility. After setting all the required packages, we began the installation process. To ease the installation process for readers, we are next highlighting some of the installation steps that induced problems:

4.2.1.Pre-requisites:

- Ubuntu 11.04 has by default and Open-JDK that you can use without downloading the JDK from Sun.
- You can use Ubuntu Software Center to install most of the packages.
- An important package that is mentioned in [7] but not in [6] is the libmysqlclient15-dev.
- In some packages, you may not find the exact version, so you should look for a more recent one, e.g., JDK 1.5

4.2.2. Source Code:

- Before proceeding, we need to install the Subversion package via this command:
  sudo apt-get install subversion
- We downloaded the source files of the IMS Core into two new directories that were made for this purpose:
  ~/opt/OpenIMSCore/ser_ims
  ~/opt/OpenIMSCore/FHoSS
- *ser_ims* folder is used for storing the configuration files of P-CSCF, I-CSCF, and S-CSCF, whereas FHoSS stores the HSS configuration files.

4.2.3. Environment Configuration

- At this stage, the default configuration of IMS Core is based on default Domain Name "open-ims.test" and the loop back address: 127.0.0.1
- Ideally, this configuration should be changed to the real IP address and real Domain Name.
- When the default configuration works well, you can change the configuration settings of all the files, either manually or via *ser_ims/cfg/configurator.sh*.
- Whenever we change the name server by altering *open-ims.dnszone*, *named.conf*, *named.conf.loca*l in the */etc/bind* directory, or *resolv.conf* and*hosts* in the */etc* directory, we should restart the bind server via the command:
  sudo /etc/init.d/bind9 restart.

4.2.4. Booting the Components

- All the 3 shell files (*pcscf.sh*, *icscf.sh*, and *scscf.sh*) should be run in at the same time in parallel.
- We should run *the FHoSS/deploy/startup.sh* database shell as well.
- The *JAVA_HOME* environment will create problems if it is not correctly set:
  Export JAVA_HOME=/usr/lib/jvm/your_jdk_version

After installing the four components (HSS, I-CSCF, S-CSCF, P-CSCF) in one station and assuring no bugs, we deployed each entity in a separate Linux station (Ubuntu 11.04, Kernel: 2.6.38-8-generic).We started by installing IMS core on the four Linux stations taking into consideration all the prerequisites needed to support Open IMSCore, and we defined the connections between the four entities: In the configuration files *(/etc/hosts*, */etc/bind/open-ims.dnszone*, and the configuration files CSCFs and HSS).We assigned to each of the IMS components the IP address of the local station, and we run the four components in parallel, one per station. We used the web console (on *http://hss IP address:8080*) to manage the users stored in the database, see Figure 2. Finally, to test the deployment we used both a SIP client and an IMS client, and tracked the exchanged messages using Wireshark [8]. The next section illustrates the testing phase.

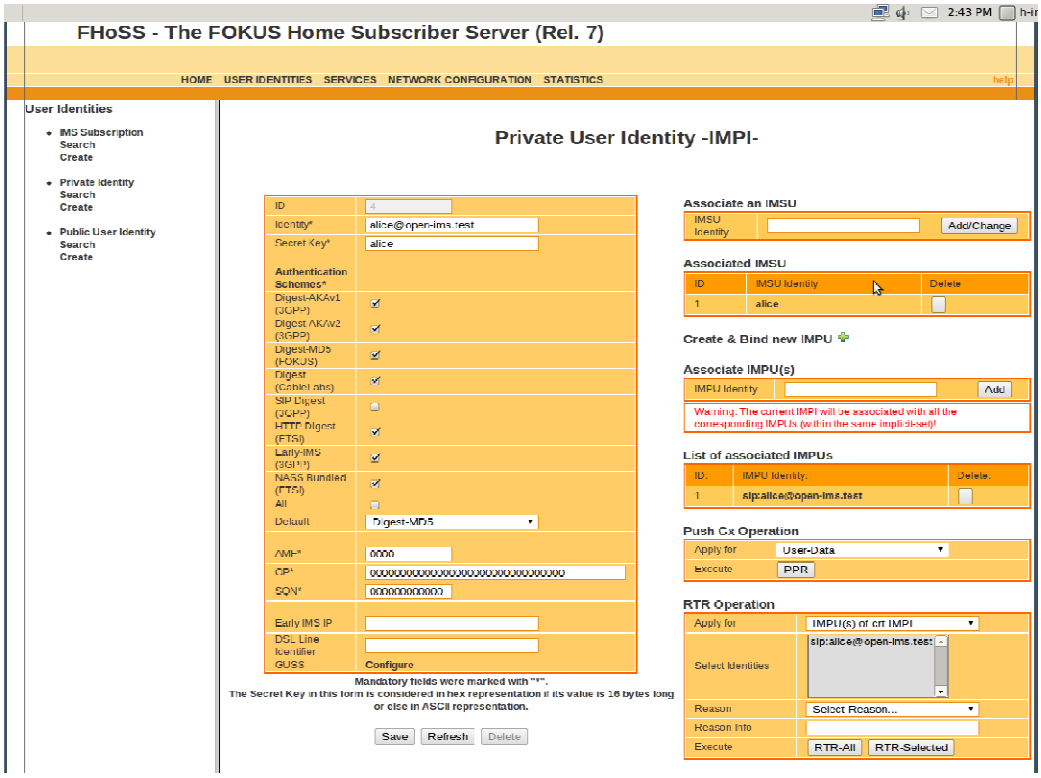

Figure 2FHoSS Web Interface: Users Management

5. Testing

After a successful installation of the four components of the Open IMS Core (CSCFs and HSS), we started the testing phase using both a SIP and an IMS client.

5.1. Testing using a SIP Client

We used the X-Lite [9] SIP client which is a free Open Source VoIP softphone that provides all the standard telephone services to its users and extends them to other services, mainly instant messaging, voice, three-way audio and video conferencing, presence using the SIMPLE protocol.We used X-Lite 4.0 and the first test consisted on using the two default accounts provided by Open IMS (*alice@open-ims.test* and *Bob@open-ims.test*). We updated the account settings accordingly (set proxy to: *pcscf.open-ims.test: 4060*) and established our first call, using instant messaging and then using voice. The second step was testing the system by establishing a call between two newly created accounts.

To assess that the installed IMS Core is compliant with the IMS standards, we used WireShark [8] to trace the messages exchanged during Registration and Call establishment between the different stations involved.The following figures(Figure 3 and Figure 4) illustrate the exchange of SIP messages between two X-Lite softphones. These captions prove the conformance between our OpenIMSCore and IMS standards.

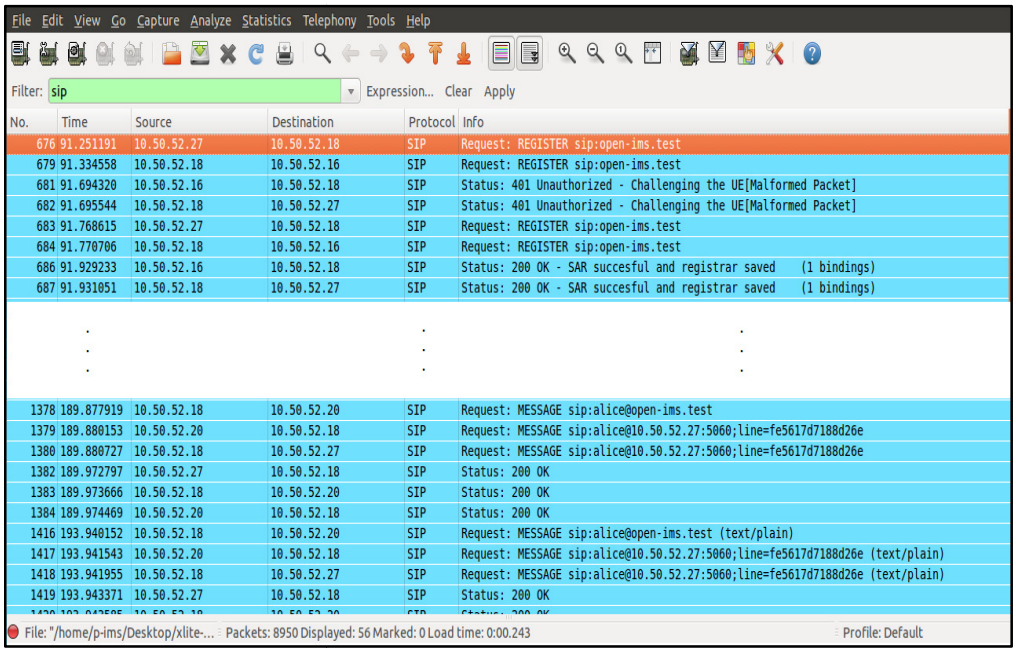

Figure 3 exchange of SIP messages between two users

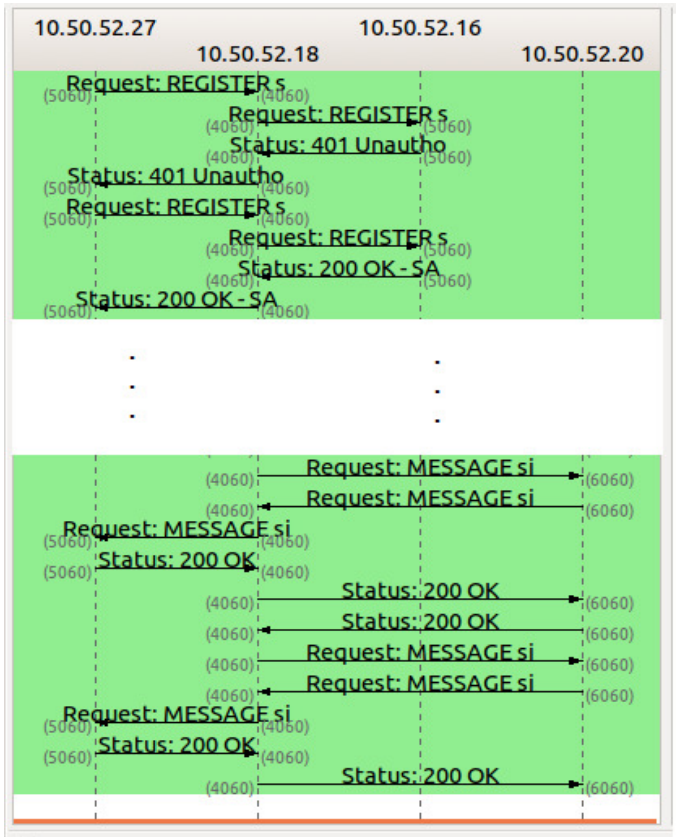

Figure 4 Flow-diagram of SIP messages between two users

5.2. Testing using an IMS Client

To further assess the IMS testbed compliance to the IMS standards, we used an IMS client for testing as well. We downloaded UCT IMS Client version 1.0.14 from [10]; an IMS Client developed by a team from University of Cape Town, South Africa (UCT), and recently released on May30Th, 2012.In order to identify the main differences between SIP client (X-lite) and IMS client (UCT IMS Client), we had recourse to WireShark, again. A brief analysis of the packets exchanged between IMS client and P-CSCF, on the one hand, and between SIP client and P-CSCF on the other hand showed that this difference consists of an additional authentication process that the IMS client goes through. Unlike SIP clients, IMS users need to go through authentication and authorization so as to access the applications [11]. Figure 5 depicts this difference:

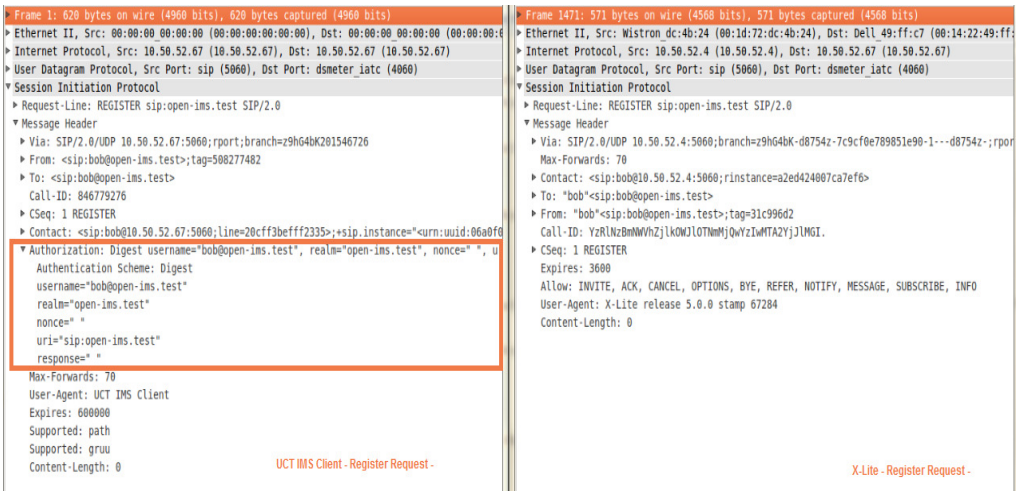

Figure 5 REGISTER Request - IMS Client vs. SIP Client

6. Presence Server

Becoming one the most powerful and important applications in telecommunication, presence is the dynamic profile of the user that is shared with others. One may think that presence is limited exclusively to sharing statuses and availability information (available, busy, absent …). However, its services go beyond this scope and provide information about the "application status" (i.e. whether it is provided by client or by server and information), about "device capability" (similar to telling which service it provides, i.e.Push-To-Talk over a data network, MMS, video sharing and so on), and about "dynamic network status" as well(i.e. the use of 2G or 3G and the terminal being on or off). As a consequence, presence will mold all the facets of mobile communication which will lead to a presence-based mobile communication. Presence represents, in addition to this, a promising field for businesses because it can be used in many lucrative ways such as advertising using means that consumer are comfortable with.

At the very beginning, presence appeared as a standalone application in 3GPP release 6; it was first introduced to IMS by OMA (Open Mobile Alliance) [12] creating what was called later: OMA Presence Architecture, which was composed of eight main building blocks. The most powerful and useful applications were, then, those that make use of the various pieces of information provided by the presence service. This latter enables the applications to do so by implementing some methods using SIP. In this context, PUBLISH was implemented to enable a

user to share his/her status and availability information with other people called watchers. This request is sent to the S-CSCF through the P-CSCF so as to be forwarded to the right presence server. Another important method is SUBSCRIBE; this latter allows a user to register to the watchers' list of a specific user. For example, if Alice wants to get presence information of Bob, she needs to send a SUBSCRIBE. Finally, another very used command if NOTIFY which delivers *presentity*'s presence state.

During our surveying of the common presence server platforms,we first investigated Mobicents SIP Presence Service [13]. This was the first alternative, but we did not go for it because of all the necessary components for a presence server are merged in one package, which is not practical with regards to debugging and to configuration. In addition to this, it limits flexibility. We finally decided to use OpenSips [14] combined with OpenXCAP [15] along with two other components which are *OpenSips-mi-proxy* and *soap-simple-proxy*. OpenSips was introduced under the name openSER [5]and was enhanced; its main role is handling the presence statuses of users and communicating with the S-CSCF. On the other hand, OpenXCAP is the component that stores the watchers' list and takes care of the privacy rules. Figure 6 depicts the way IMS servers are linked to the presence. The interface between the end-user and XCAP server was meant to avoid the overload on the presence server. Besides, XCAP and Presence Server are linked through sharing the same database.

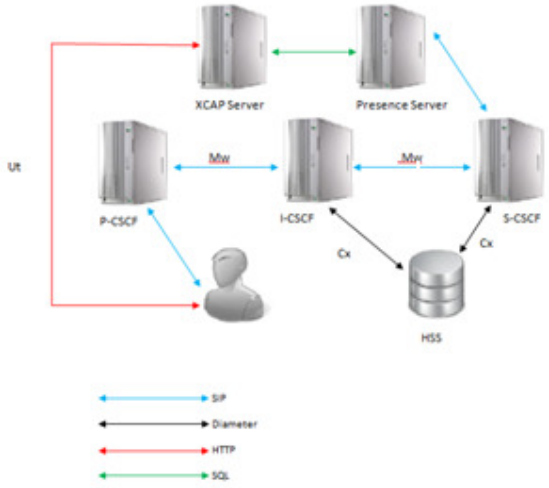

Figure 6 Presence Server Architecture [7]

After installing OpenSips first, we moved to installing openXCAP. However, since we were using Ubuntu 12.04 we discovered that there is no version of openXCAP compatible with this version of Ubuntu, so we had to switch to Ubuntu 11.04. After a successful installation, we configuredopenXCAP in addition to OpenSips, soap-simple-proxy, and OpenSips-mi-proxy. Still, when we tried first to start OpenSips, there were some errors in the configuration files. Those errors were stated in "*syslog*" file located in */var/log*. Basically, there were some missing load-modules as well as some inconsistencies in the "*route*" function. For instance, an extra curly bracket of the function route was added in line 329 of the configuration file of OpenSips (*opensips.cfg*) as soon as OpenSips was up and running.We used FHoSS web interface to set the new value of the IP address of the presence server as shown in the figure below.

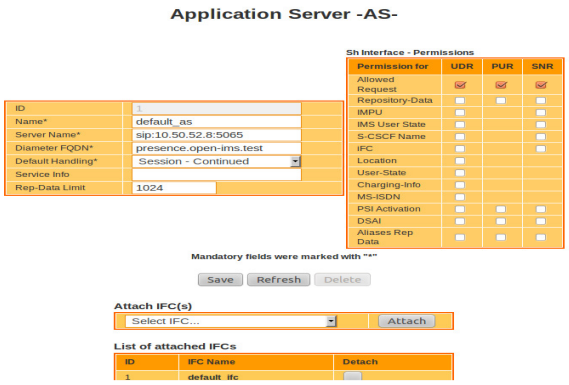

Figure 7FHoSS Web Interface: Setting Presence Server IP Address

At this stage, we used WireSharkto ascertain the connection between the presence server and IMS was established in the right way. We connected to the network using UCT IMS Client, and we enabled presence in this latter. The packets sniffed by WireShark and displayed were the messages we were expecting; that is to say, the requests exchanged were: SUBSCRIBE, PUBLISH and NOTIFY. Figure 8 illustrates the messages:

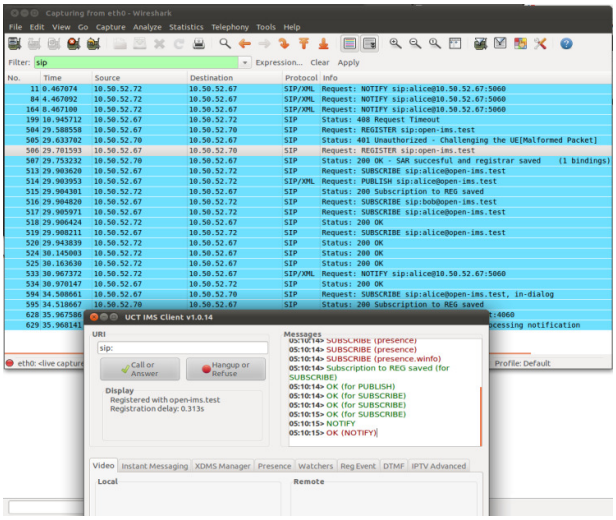

Figure 8 Exchanged messages between the Presence Server and the IMS Core

7. A Third Party Application using JAIN SIP

SIP is a standard communication protocol for controlling communication sessionssuch as voice and video calls over Internet Protocol (IP). The Java Community Process (JCP) [16] developed an API for telecommunications that was named the Java API for Integrated Networks (JAIN). Later, JAIN SIP was released as the Java interface to SIP for desktop and server applications. It is also considered to be the first attempt to enhance the development of SIP-based multimedia applications. As it is a high level API, JAIN SIP provides abstractions for SIP Protocol, dialog handling, etc.

The JAIN SIP API [17] is composed of the following main packages:

- *javax.sip:*This package contains the main interfaces that model the architecture from both an application developer and a stack vendor view. Main interfaces of this package are *ListeningPoint*, *ServerTransaction*, *SipListener*, *SipProvider*, and *SipStack.*
- *javax.sip.address***:** this package contains interfaces that represent the Addressing components of the SIP protocol. The main interfaces of this package are Address, and AddressFactory.
- *javax.sip.message***:** This package contains the interfaces representing SIP messages. Interfaces of this package are Message, MessageFactory, Request, and Response.
- *javax.sip.header***:** This package contains all the headers' interfaces supported by this specification.

To gain a better understanding of JAIN SIP API, we refer the reader to an open source Instant Messaging application [18]. The application is built over three main classes: *TextClient*, *SipLayer*, and *MessageProcesso.* The *TextClient* class declares a *SipLayer* instance given a username, IP address, and a port number (by default, the IP address is set to the *Localhost* address, but we changed itinthe source code in order to allow running Instant Messaging application between two different stations). *TextClient* prompts a graphical interface to output the messages exchanged between the two user agents (or the two *TextClient* instances).When *SipLayer* is instantiated at the *TextClient* level, it first creates a *SipFactory* instance that would allow for creating the backbone of the SIP implementation: instances of *SipStack*, *HeaderFactory*, *AddressFactory*, and *MessageFactory*. The *SipStack* instance serves for creating a *ListeningPoint* instance, which is used to create a *SipProvider* instance. All of these steps happen at the level of the *SipLayer* Constructor.

Thus, after creating two user agents from the *TextClient* class, we started testing. In order to send a message, the user should specify the URI of the other user device, and it has the following format: sip:username@ip_address:port, see Figure 9.

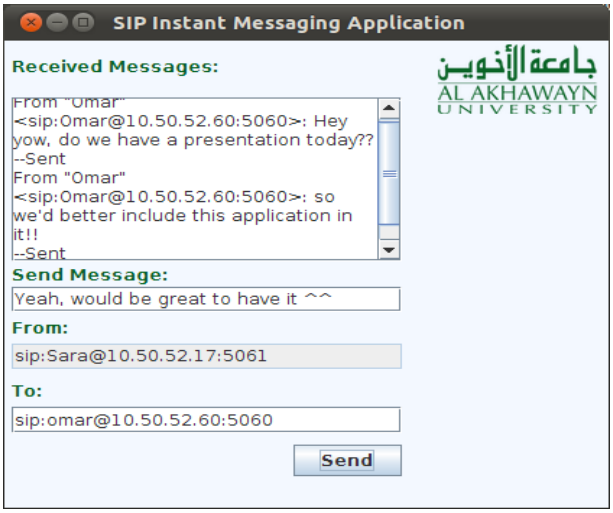

Figure 9 Customized JAIN SIP Application

8. Conclusion and Future Work

In this paper, we presented a detailed blueprint for the deployment of a real-world IMS testbed using open source software. Encountered problems as well as detailed steps are highlighted. The operation of the testbed is checked to conform to the 3GPP specifications.

The testbed will help the research community to establish an IMS core and thus set an important component in the establishment of future next generation networks. Once set, third party applications can be developed using common APIs. In this paper, we used JAIN SIP, and we showed how to build a simple messaging application on top of the deployed IMStestbed. More sophisticated applications can be developed, e.g., using the presence state.

As a future work, we are intending to extend the deployed IMS testbed to cover the whole university campus by using powerful stations, deploying multiple domains, and resolving NAT problems. Once installed, advanced applications for mobile handsets will be deployed, especially VoIP applications accounting for presence and user profiles.A future step is to acquire 4G femtocells, interface them to the IMS core, and thus paving a sold floor towards promoting richer 4G mobile applications.

References

[1] 3GPP Long Term Evolution, http://www.3gpp.org/LTE.

[2] P. Miikka, N. Aki, K. Hishamand M. Georg.*The IMS: IP Multimedia Concepts and Service*, 2nd ed. UK: John Wiley & Sons Ltd, 2006.

[3] Fraunhofer FOKUS NGNI. Open IMS Core Welcome to Open IMS Core's Homepage [Online]. Available: http://www. openimscore.org.

[4] The FokusFraunhofer Institute, http://www.fokus.fraunhofer.de/en.

[5] Opensips. (2012, August 15).*More about OpenSIPS* [Online].Available: http://www.opensips.org/.

[6] FOKUS. Open IMS Core Installation Guide[Online]. Available:http: //www.openimscore.org/installation _guide.

[7] W. Dave, O. Vitalis, and E. Asma. (2012, June 25)."UCT IMS Client." UCT IMS Client.University of Cape Town, South Africa, n.d.[Online].Available: http://uctimsclient.berlios.de/openimscore_on_ubuntu_howto.html.

[8] Wireshark, http://www.wireshark.org/.

[9] Counterpath Corporation, X-Lite Softphone. (2003).http://www.counterpath.com/x-lite.htm.

[10] BerliOS. (2012, May 13). UCT IMS Client [Online].Available:http://uctimsclient.berlios.de/.

[11] P. Dubravko and Miljenko.M. (2008).*Security risks of pre-IMS AKA access security solutions*[Online].Available: http://www.ericsson.com/hr/etk/dogadjanja/mipro_2008/1227.pdf.

[12] Open Mobile Alliance Organization, http://www.openmobilealliance.org/.

[13] Mobicents. (2008). *SIP Presence Service Introduction* [Online]. Available: http://www.mobicents.org/sip-presence/intro.html.

[14] OpenSips. (2012, February 22). *OpenSIPS 1.7.2* [Online].Available: http://opensips.org/pub/opensips /1.7.2/.

[15] L. J. Philippe. (2011, June 9). *OpenXCAP Installation* [Online]. Available: http://openxcap.org/projects/openxcap/wiki/Installation.

[16] JCP: The JAVA Community Process, http://jcp.org/.

[17] P. O. Doherty and R. Mudumbai.*NIST-SIP: The Reference Implementation for JAIN-SIP 1.2*[Online].Available:http://hudson.jboss.org/hudson/job/jainsip/lastSuccessfulBuild/artifact/javadoc/ overview-summary.html.

[18] P. Emmanuel. (2007, October 17). *Introduction to JAIN SIP* [Online].Available: http://www.oracle.com/technetwork/articles/entarch/introduction-jain-sip-090386.html.

Authors

Sara EL ALAOUI is a senior student at the Computer Science program [ABET Accredited] at Alakhawayn University in Ifrane, Morocco

Fatima Zohra SMAILI is a senior student at the Computer Science program [ABET Accredited] at Alakhawayn University in Ifrane, Morocco

Omar BOUGAMZA is a senior student at the Computer Science program [ABET Accredited] at Alakhawayn University in Ifrane, Morocco

Mohamed Riduan ABID received a Ph.D in Computer Science in 2010 from Auburn University,USA. He received in 2006 the Excellence Fulbright Scholarship. He is currently an Assistant Professor of Computer Science at Alakhawayn University,Morocco.