https://doi.org/10.26552/com.C.2012.2.55-62

Pavel Segec – Tatiana Kovacikova *

IMPLEMENTATION OF IMS TESTBEDS USING OPEN SOURCE PLATFORMS

The IP Multimedia Subsystem (IMS) is a control architecture framework for the Next Generation Network (NGN). It is expected that this new network architecture will provide a new kind of IP multimedia services delivered in network converged environments. The NGN and IMS, even after the years of a standardization effort, are still hot networking topics. One of the communities, which adopt emerging communications standards into their products fast, is the open source community. In present days, there are a few open source products, which allow building the IMS environment. However, there are at least two domains for which they are especially helpful. It is teaching/learning of the IMS architecture and testing service deployment in a nearly real environment. This article provides a survey of available open source products. It also describes the experience with the implementation, establishment and running an IMS testbed, which has been built up on an open source client and server entities. At the server side, the focus is on the Kamailio IMS platform.

Keywords: *IMS, open source, Kamailio, testbed*

1. Introduction

Next Generation Network (NGN) presents an interesting concept which merge different communications paradigms, wireline and wireless, mobile and Internet. The idea behind the NGN is to operate a universal communications platform covering existing network heterogeneity; which is multiservice, multiprotocol, multiaccess, IP-based and providing feature rich multimedia services. The fixed-mobile converged network enables to create a single communication environment allowing its users a unified access to their services from any fixed or mobile terminal via any access network.

NGN framework was introduced by the International Telecommunication Union–Telecommunication Standardization Sector (ITU-T), and ETSI (European Telecommunications Standards Institute). The key component of the NGN is IP Multimedia Subsystem (IMS) control layer, formerly defined by the 3rd Generation Partnership Project (3GPP) for mobile networks of 3rd generation [1, 2] and incorporated then into the NGN. IMS allows the move from circuit-switched services toward IP packet-based services and allows building up “All-IP” communications architecture with all of its benefits [3]. IMS allows providing of multimedia services in secure, chargeable environment with different levels of quality guarantees. IMS specifies a set of standardized entities and interfaces and allows building different kinds of integrated innovative services, which should combine features of legacy telecommunication and IP services. This concept brings a new Internet and IT way of the service creation approach to the development of innovative and attractive services.

Prototyping and delivering of such multimedia services or trial concepts is a challenging task which requires testing environments that allow understanding and evaluating of the behavior and impacts of proposed solutions. This should be one of many factors which help to speed up IMS adoptions. One of the communities, which adopt emerging communications standards into their products very fast, is the open source community. There are a few open source products which allow building up IMS testing/teaching environment to support learning models, service and components development. This article provides the survey of existing open source products which are suitable for IMS testbed establishment. It also describes the experience with implementation, establishment and running an IMS testbed which has been built up on open source client and server entities.

The paper is organized into five chapters. Chapter two provides the overview of NGN and IMS system architecture, chapter three analyzes key open source solutions suitable for establishing an IMS testbed, chapter four provides a description of our approach and chapter five concludes the article.

2. Next Generation Networks and IP Multimedia Subsystem

2.1. NGN architecture overview

The objective of NGN is to enable convergence of different fixed and mobile networks whose central component is the IMS.

* **Pavel Segec, Tatiana Kovacikova**
Department of InfoComm Networks, Faculty of Management Science and Informatics, University of Zilina, Slovakia,
E-mail: Pavel.Segec@fri.uniza.sk

The NGN functional reference architecture [4] is designed as a system of horizontally oriented layers consisting of a number of cooperating subsystems. The subsystem design enables future subsystems to be added extending the main functionality and services, (for example, an IPTV in NGN Release 3). Each subsystem is specified as a set of entities interconnected through related interfaces. The essential NGN specification defines the NGN transport layer and the NGN service layer. The NGN transport layer provides IP connectivity for NGN users (UE) and hides the transport technology of the access and core networks below the IP. The transport layer is divided into two sub-layers, the transport processing functions sub-layer and the transport control sub-layer. In general, transport processing functions provide packet forwarding and routing functions together with more specific functions such as media processing/transcoding, relaying, IP level interconnection, resource control, etc. The transport control sub-layer provides attachment control (addressing, access authentication, authorization, location management) provided by the Network Attachment Subsystem (NASS) and functions responsible for the implementation of procedures and mechanisms handling policy-based resource reservation (Quality of Service (QoS), Network Address Translation (NAT)) and admission control provided by the Resource and Admission Control Subsystem (RACS) [4]. NGN service layer comprises a number of sub-systems such as Core IP Multimedia Subsystem (IMS), PSTN/ISDN emulation subsystem (PES), IPTV subsystem, other subsystems and common components whose functionalities are shared by the related subsystems. Finally, there is a user equipment (UE) subsystem defined as: “One or more devices allowing user access to network services delivered by TISPAN NGN networks” [5]. As an example of the UE, a telephone set can be mentioned which has to be NGN-compatible to make the full use of NGN services. The UE is sending/receiving user data over the transport layer. At the signalling layer, the UE communicates with NASS for network authentication/configuration and with IMS for handling services.

2.2. IMS architecture overview

The IMS is a core component around which the NGN is defined (Fig. 1). The IMS is end-to-end IP control framework that allows applications and services to be supported seamlessly across all types of networks. The IMS architecture capabilities are specified in a set of 3GPP documents that define components, services and interfaces [1, 2]. ETSI TISPAN NGN [6, 7] adopted the 3GPP IMS [1] for the NGN network architecture [4]. From an architectural point of view, the NGN IMS subsystem, also known as “Core IMS”, is a subset of the 3GPP IMS, where some entities such as Application servers (AS) and transport/media related entities (Multimedia Resource Function Processor function (MRFP), IP Multimedia Gateway Functions (IM-MGW)) are considered to be outside of the “core IMS” [7] (Fig. 2). From a functional point of view, the 3GPP IMS has been extended to support additional access network types, such as xDSL and WLAN. The control functionality of the Core IMS handles session signalling and comprises a number of distinct functions to process signalling flows, such as Call Session Control Functions (CSCF), Breakout Control Gateway Functions (BGCF), Media Gateway Control Function (MGCF), Multimedia Resource Function Controller (MRFC) and Interconnection Border Control Function (IBCF). Using protocols such as SIP [8], Diameter and H.248 MEGACO, the different elements are able to establish requested subscriber services. The ETSI TISPAN is harmonizing with the 3GPP on any modifications or improvements that may be needed for the NGN/IMS.

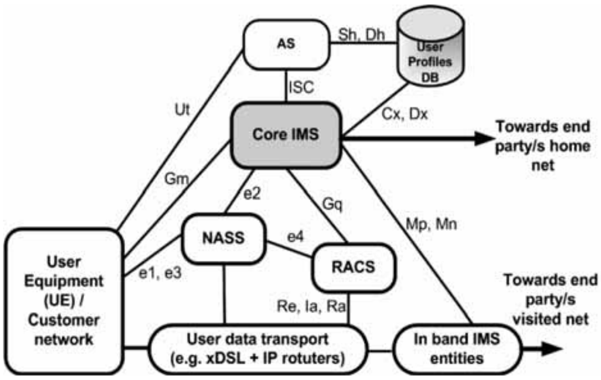
AS
Sh, Dh
User Profiles DB
ISC
Ut
Core IMS
Cx, Dx
Towards end party/s home net
Gm
e2
Gq
Mp, Mn
NASS
e4
RACS
User Equipment (UE) / Customer network
e1, e3
Re, Ia, Ra
Towards end party/s visited net
User data transport (e.g. xDSL + IP rotuters)
In band IMS entities

Fig. 1 IMS interfaces to the ETSI TISPAN NGN architecture

2.3. Core IMS components

The functionalities of a core IMS are included in the Call Session Control Function (CSCF). A CSCF is a SIP server which processes the IMS signalling traffic in order to control IMS sessions. There are three types of CSCF, Proxy CSCF (P-CSCF), Serving CSCF (S-CSCF) and Interrogating CSCF (I-CSCF) (Fig. 2).

The P-CSCF is the entry point for signalling traffic flowing into the IMS. A user is allocated to a P-CSCF as a part of the registra-

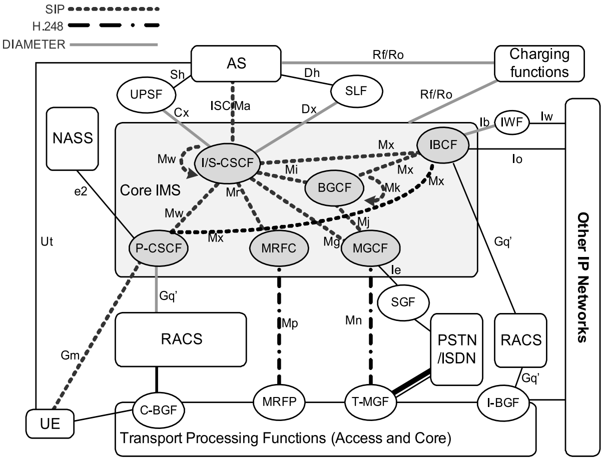

Fig. 2 Core IMS in the ETSI TISPAN NGN architecture

tion process. P-CSCF stays for a registered client on signalling traffic path for the duration of the session. P-CSCF should provide other functionalities as signalling compression, security association and resources management. The I-CSCF is usually located at the boundary of a domain that serves as a gateway to other domains, such as other service provider networks. I-CSCF may encrypt sensitive domain information (Topology Hiding Internetwork Gateway (THIG)) before forwarding the traffic. The S-CSCF is always located at a home provider network, provides the service logic which controls the delivery of requested service. The S-CSCF interacts with a profile database (HSS in a 3GPP environments, UPSF in a NGN environment) in order to determine user service eligibility by downloading the user profile; the S-CSCF is allocated for the duration of the registration.

3. Open Source IMS Testbed

Motivations behind the establishment of an IMS testbed are different; to expose complex communications concepts and components, to enable advanced NGN/IMS services development, to enable trial and performance testing, etc. For university education, there are all mentioned with an accent to popularize IMS technology on an advanced learning platform, to enable technology, protocol and services analyses and to provide safe and controlled environments to facilitate research and experimentations.

Similarly to SIP and IMS, an open source usually speeds up the implementation of emerging industry standards. Today's open source products allow the building of IMS testbed using standard hardware and operating systems (OSs) that provide cost effective solutions. This chapter provides a survey and feature analysis of suitable open source products available at the time of writing this paper.

3.1 Survey of IMS Core components

As defined in [1, 2, 6 and 7], the IMS consists of numerous entities mentioned before, including media and gateway controllers. It should provide services in secure, chargeable environments with different levels of QoS. This is a really challenging task which currently cannot be fulfilled by any of open source solutions. However, for establishing a simple IMS system with basic functionalities allowing call establishment and registrations, four components are necessary only, P-CSCF, S-CSCF, I-CSCF and a database (Home Subscriber Server – HSS) used to store user profiles. Therefore, main research and development effort is focusing on the control and application layer of the IMS, while the transport layer with QoS reservation concept, RACS and NASS subsystems, is currently omitted. IP connectivity is required as a fundamental prerequisite. Control layer components which perform signalling routing, authorization, registration and service provisioning are central parts of a testbed. To build up an IMS testbed, there are two open source solutions available, the Open IMS Core developed by the Fraunhofer Institute FOKUS [9] and Kamailio IMS extensions developed by the NG Voice [10].

3.1.1 Open IMS Core

Main motivation behind the development of the Open IMS Core, led by the Fraunhofer Institute FOKUS, was to provide an IMS core reference implementation suitable for IMS technology testing, service prototyping, technology research and many others. The Open IMS Core is neither designed nor suitable for commercial IMS architecture deployment. The Open IMS Core is an open source implementation (GPLv2 license) of IMS CSCFs signalling entities and the HSS [9] (Fig. 3) compliant with 3GPP specifications. The Open IMS Core is based upon open source solutions such as the SIP Express Router (SER) implementation of SIP Proxy server and the MySQL database. SER is characterized by powerful modular design and may act as a SIP registrar, a SIP redirect and proxy server. The Open IMS Core extends main IETF SIP functionalities of the SER express router with IMS modules (namely CDiameterPeer, IMS Service Control, Proxy-CSC, Interrogating-CSCF, Serving-CSCF modules [11, 12]) to provide core IMS signalling functionalities of IMS CSCF entities (called Open IMS CSCFs). As the IMS requires some information lookup (user and service profiles), a simple, lightweight version of an HSS entity (called Fokus HSS – FHoSS) has been developed and added to the Open IMS Core project. The FHoSS is written in Java and uses the MySQL database. The FHoSS contains its own Java based Diameter stack which supports Cx, Sh and Zh IMS Diameter interfaces. The FHoSS provides functions such as storage of user profiles and service subscriptions, charging information, generation of authentication/authorization data, and a support of subscriber locations. The Open IMS Core provides an ISC (IMS Service Control) interface to allow the development of IMS applications or modifications of routing rules or events.

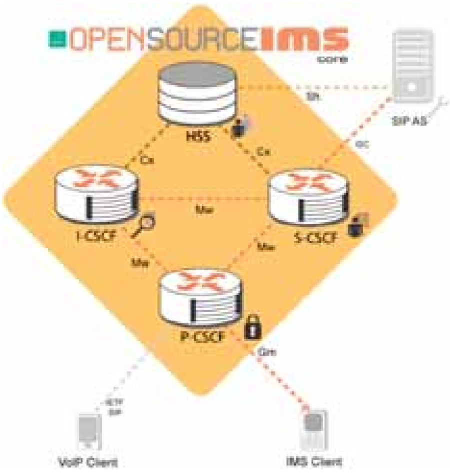

Fig. 3 The Fraunhofer Institute FOKUS Open IMS Core architecture

The Open IMS Core project provides quite comprehensive web documentation which describes the required steps for the testbed installation with basic IMS functionalities. The Open IMS

Core entities run over standard Linux based OSs. The installation requires compilation of installation packages, but the source codes contain the pre-prepared configuration files of CSCF entities. Main configuration requires small changes only related to the DNS names used in a testbed environment and port numbers on which CSCFs entities are listening. The user and service management is provided over the web-based FHoSS management graphical user interface (GUI).

The Open IMS Core project started on November 16, 2006. One of the goals of the project was to make the Open IMS Core an enabler for other testbeds and to create a community of IMS testers and developers that are using the Open IMS Core for their projects. Having analyzed the information available through the Internet, we found that this goal was met successfully [11, 13]. However, the last modification of the project was during 2009 and since that time the project has not been actively maintained.

3.1.2 Kamailio based IMS solution

As mentioned before, the Open IMS Core project is a perfect tool for the IMS playground establishment. However, there are several issues that could limit its future deployment. The Open IMS Core project is no more evolving neither supported now. Core CSCF entities of the Open IMS Core framework are based on an old version of the SER server (version 2.0 Beta).

The SER (SIP Express Router) server, nowadays known as the Sip-Router (since 2008), is a famous SIP proxy with a number of worldwide implementations and wide open-source community of developers, maintainers and supporters. The SER has been developed since 2001, and released in 2004 under the open source GPL license. The history behind the SER development is complicated, and leads to establishing several viable forked solutions, such as OpenSER in 2005 (renamed to Kamailio in 2008) and OpenSIPS in 2008. Nowadays, the developers of Kamailio and SIP-router projects are normalizing the situations and the merging of these two projects has been initiated. At present, Kamailio and SER should be considered as the same application (since release version 3.0). The OpenSIPS fork is still going its own development way.

Kamailio (former OpenSER) [14] is a very popular and viable SER based SIP proxy. The Kamailio server, released under the GPL license, is able to handle thousands of call setups per second. The Kamailio features include a support for asynchronous TCP, UDP and SCTP, secure communication via TLS for VoIP (voice, video), SIMPLE instant messaging and presence, ENUM, least cost routing, load balancing, routing fail-over, accounting, authentication and authorization against MySQL, Postgres, Oracle, Radius, LDAP, XMLRPC control interface, SNMP monitoring [14]. During 2011 the Kamailio IMS module extensions were introduced, which allow to build an IMS testbed with the latest Kamailio SIP server releases. The main developer of the Kamailio IMS extensions is Mr. Carsten Bock, a member of the Kamailio development team [10]. The Kamailio IMS module project aims to create an open-source based stable IMS infrastructure based on different GPL solutions [10]. As a basement for Kamailio IMS modules, former Open IMS Core modules were taken and ported for re-using with Kamailio. However, there are several issues identified. Since the Open IMS Core deployed all the IMS functionalities into a few modules, whose functionalities and programming code are partially duplicated by the Kamailio server, the development is progressing fast with many improvements (architecture re-designs, performance improvements, feature extensions, etc.). The future modules rebuild is being developed with the objective to define modular IMS modules design without duplicating functionalities of the Kamailio server and IMS modules, re-using functionalities of Kamailio modules wherever possible (for example, a dialog module), enhancing an IMS module wherever necessary, reusing Kamailio presence functionalities, and finally adding new features [15]. Actually, some improvements have been already provided, for example, support for Reg-Event, enhancements of a dialog module and RTP activity detection. Developers are currently working on other improvements (especially, on Diameter protocol). Kamailio IMS intended to provide high-performance, open-source and standards-compliant implementation of HSS for the use in IMS context. Due to lack of human resources the project was frozen. Instead, the Open IMS Core FHoSS is used. At present days, the Kamailio IMS project, although short-lived, allows to build up an operational IMS testbed. Kamailio based CSCF entities provide many features, many of them are fully open source based, standards compliant and providing stability and performance with full IPv6 support. The whole feature list is available at [10].

The Kamailio project provides wide, high-quality and up-to-date web based documentation flesh out with a lot of external internet documentation sources. The information about a Kamailio IMS testbed installation is documented in [10]. The knowledge base is also provided on the knowledge portal of the Department of information networks [16]. Kamailio IMS runs over standard Linux based OSs. Installation packages for 32bit debian operating systems are available through the package repository, while the other operating systems require their compilation. Configuration of Kamailio based IMS requires modifications of configuration files with correct DNS names of a testbed environment, modifications of port numbers on which CSCFs entities are listening and finally, modifications of Diameter configuration files with correct IP addresses and port numbers required for Diameter interconnection [16]. Comparing to the Open IMS Core installation process, Kamailio IMS configuration process is more difficult, which is caused by missing lines inside the Diameter configuration files which are required for establishing correct Diameter interconnection interfaces.

3.2 Survey of available IMS clients

IMS clients or end user equipment are entities through which users interact with IMS entities and perform signalling procedures and service invocation. IMS clients may range from clients that support registration and a voice call only up to clients that support voice/video calls, messaging and presence, IPTV, and etc. The UE connected to the IMS testbed should support AKAv1/2-MD5 authentication, IMS SIP signalling (PRACK, precondition), and voice calls. Advanced IMS clients should support voice and video calls, messaging and presence, address books, advanced security functionalities and they should be NAT friendly. Up to now, there are several open source projects that have been working on IMS client development. Some of them have been already closed, some of them are still active. Table 1. provides a feature analysis of six currently identified IMS clients that are available under some of open source or free licenses and that are suitable for IMS testbed usage for different scenarios. Those are Boghe for Windows, iDoubs for iOS and Mac X OSs, IMSDroid for Android, myMonster and IMS Communicator, all of them are operating system independent as they are developed in Java. The UCT IMS client is intended for Linux users only.

Boghe is an IMS client for windows OSs which supports Rich Communication Suite (RCS). RCS is managed by the GSM Association (GSMA) with a goal to provide a set of fully interoperable rich services to be used inside the fixed-mobile environment and to speed up the evolution of mobile phone communication towards service rich communication. To become a RCS compliant, a client must at least support Enhanced Address Book (defined by the OMA), Enhanced Messaging (OMA), Content Sharing (GSMA) and File Transfer (OMA) services. A Boghe IMS client is based on a Doubango RCS framework which is shared with co-projects developing IMS clients for iOS and Android. A client provides user friendly GUI supporting comfortable configuration, many types of audio and video codecs. Comparing to other IMS clients, Boghe

IMS client feature analysis. Table 1.

IMS clients	Boghe IMS/RCS client	iDoubs	IMSDroid	myMonster TCS	UCT IMS client	IMS Communicator
www	code.google.com/p/boghe	code.google.com/p/idoubs	code.google.com/p/imsdroid	www.monster-the-client.org	uctimsclient.berlios.de	imscommunicator.berlios.de/
License	GPLv3	GPLv3	GPLv3	free, own	GPLv2	LGPL
Last release	2.0.91.683	2.0.184	2.0.481	0.9.19 TCS	1.0.13	0.70.605
Date	24.2.2012	20.9.2011	28.2.2012	2009	13.2.2009	8.7.2007
Operating system	Win	iOS,Mac OS X	Android	Multiplatform (Java)	Linux	Multiplatform (Java)
Multiple accounts	No	No	No	Yes	No	No
Registration	AKAv1/v2-MD5, MD5, basic	AKAv1/v2-MD5, MD5, basic	AKAv1/v2-MD5, MD5, basic	AKAv1/v2-MD5, MD5	AKAv1/v2-MD5, MD5	AKAv1-MD5, MD5
IMS Signalling	Yes	Yes	Yes	Yes	Yes	Yes
SigComp	Yes	Yes	Yes	No	No	No
Audio	Yes PCMA, PCMU, GSM, AMR-NB-OA, AMR-NB-BE, iLBC, Speex-NB	Yes G.722, G.729AB, AMR-NB, iLBC, GSM, PCMA, PCMU, Speex-NB, Speex-WB, Speex-UWB	Yes G729AB1, AMR-NB, iLBC, GSM, PCMA, PCMU, Speex-NB	Yes PCMA, PCMU, MCA	Yes GSM, PCMA, PCMU, MP2	Yes JMF codecs
Video	Yes MP4V-ES, Theora, H264, H263/+/++	Yes VP8, H264, MP4V-ES, Theora, H263/+/++	Yes VP8, H264, MP4V-ES, Theora, H.263, H.263-1998, H.261	Yes H.263, MP2T, H.263-,MPV, MP4V-ES	Yes -	Yes JMF codecs
Messaging/Chat	Page / Session mode	Page / Session mode	Page / Session mode	Page / Session mode	Page mode	-
Presence	Yes OMA, IETF	Yes OMA, IETF	Yes OMA, IETF	Yes OMA, IETF	Yes -	Yes -
XDMS/XCAP	Yes	Yes	Yes	Yes	Yes	No
Security	TLS, IPSec	-	TLS, IPsec	-	No	IPSec
IPv6	Yes	Yes	Yes	Yes	Yes	Yes
NAT Traversal	Yes STUN/TURN	Yes STUN/TURN	Yes STUN/TURN	No	No	Yes STUN
P-CSCF discovery	Fixed, DNS NAPTR+SRV	Fixed, DNS NAPTR+SRV	Fixed, DNS NAPTR+SRV	Fixed, SRV DNS	Fixed	Fixed
Other features	Enhanced Address Book, Content Sharing, File Transfer, Image share	-	File Transfer, Image Sharing, Video Sharing	File transfer, Group list management	IPTv (RTSP)	-

is the most recent initiative (released in 2011). The project is alive, the latest released version of the client is 2.0.85. Clients iDoubs (iOS, Mac X) and IMSDroid (Android) are very similar to a Boghe client, they are produced by the same developing team using the same Doubango developing framework [17] and they share most of the functionalities with a Boghe client.

The myMonster TCS IMS client is a feature rich client, based on the Telco Communicator Suite (TCS). TCS is a java based telecommunication framework from the myMONSTER toolkit (Multimedia Open InterNet Services and Telecommunication EnviRonment) developed at the Fraunhofer Institute FOKUS. The myMONSTER TCS supports 3GPP IMS, GSMA RCS and OMA service enabler access as Presence, XML Document Management (XDM), SIMPLE IM-based instant messaging and other functions. A client provides a user friendly GUI, it can be configured quite easily and supports multiple accounts. The latest released version is 0.9.19 from 2009 for Windows mobile, Windows and Linux OSs. Since that time, the project has not been actively evolving.

The UCT IMS client is developed by the University of Cape Town, South Africa. UCT IMS client is intended for Linux users only and it is intended especially for developers. It contains a message debug window and two preconfigured testing accounts for Alice and Bob. The latest version is 1.0.13 released in 2009. Again, since that time it has not been actively evolving. However, the support for an IPTv service with RTSP provides an interesting feature.

The IMS communicator is a Java based IMS client (JAIN), which does not require installation. Originally, it was developed by the PT Innovacao to support IMS service testing and IMS service development. The GUI interface is simple, however, client configuration is quite comprehensive and requires deeper knowledge. To make configuration more friendly, it provides a simplified configuration wizard. A client contains hard coded DNS resolving, and without java files modification a client is useless. The IMS Communicator has not been updated since 2007, and the project itself is not active anymore. Active tests executed over an IMS testbed show the ability to use vanilla IETF SIP clients for simple voice/video call scenarios.

For the reasons mentioned above, we recommend to use Boghe, followed by the myMonster TCS as the most feature rich and promising IMS clients.

4. Deploying an IMS testbed

To popularize IMS technology as an advanced learning platform, an enabler technology for a future protocol and services analyses, research and experimentations, we have analyzed the possibilities of using an open source platform for building up an IMS testbed. The IMS platform should provide a living environment that may be used for learning master degree courses offered by our department. It also should allow an access to the IMS technology and provide basic IMS services accessible from IMS clients. The testbed should be established with low costs based on open source software whenever possible. The IMS testbed should be open and extendible with new and innovative services and communications components.

Having used open source components mentioned above, we built our laboratory prototype environment which fulfills all these requirements. Even though in our research we are focusing on a Kamailio SIP server [14], the topology simulates two IMS providers and it is based on the both open source IMS projects. The first provider is based on the Kamailio IMS; the second one uses the Open IMS Core. The IMS testbed consists of core IMS entities only (CSCFs and FHoSSs). Our tesbed is realized with two servers (even though one of them is required only) and several software IMS clients connected over a LAN network. The first server is the departmental DNS server which keeps DNS NAPTR and SRV records required for the both virtual IMS providers. An example of the DNS configuration for virtual provider called IMS1 (DNS FQDN domain ims1.sip.uniza.sk) is described in Fig. 4.

```
$ORIGIN ims1.sip.uniza.sk.
@                NAPTR 10 100 "S" "SIP+D2U" "" _sip._udp.pcscf
@                NAPTR 20 100 "S" "SIP+D2T" "" _sip._tcp.pcscf
pcscf            IN   A          158.193.139.40
_sip.pcscf       IN   SRV  0  0  5060 pcscf
_sip._udp.pcscf  IN   SRV  0  0  5060 pcscf
_sip._tcp.pcscf  IN   SRV  0  0  5060 pcscf
icscf            IN   A          158.193.139.41
_sip             IN   SRV  0  0  5060 icscf
_sip._udp        IN   SRV  0  0  5060 icscf
_sip._tcp        IN   SRV  0  0  5060 icscf
scscf            IN   A          158.193.139.42
_sip.scscf       IN   SRV  0  0  5060 scscf
_sip._udp.scscf  IN   SRV  0  0  5060 scscf
_sip._tcp.scscf  IN   SRV  0  0  5060 scscf
hss              IN   A          158.193.139.43
presence         IN   A          158.193.139.37
```

Fig. 4 DNS records required for an IMS testbed

Thanks to Xen server virtualization technology, the second server is hosting all eight virtual servers with CSCFs and HSSs IMS control entities (Fig. 5). As the OS platform, 32 bit linux Debian OS release Squezze was used, and all testbed components are based on open source software (Kamailio IMS, Open Core IMS, Mysql, Apache, Boghe, Monster IMS clients, RTPproxy). To support a call establishment behind the Network Address Translator (NAT), the RTPProxy media proxy is used.

Over the testbed, we have provided several testing interconnection scenarios for different basic services such as audio/video calls, file transfers and messaging, different IMS clients, and simulating intra and inter domain connectivity. The signalling message exchange have been captured, analyzed and evaluated to proof the interconnectivity. Some scenarios demonstrated the applicability of “vanilla” IETF SIP clients which were able to register into an IMS domain and to make simple calls. The quality of IMS client implementations differs and the tests of interoperability show that

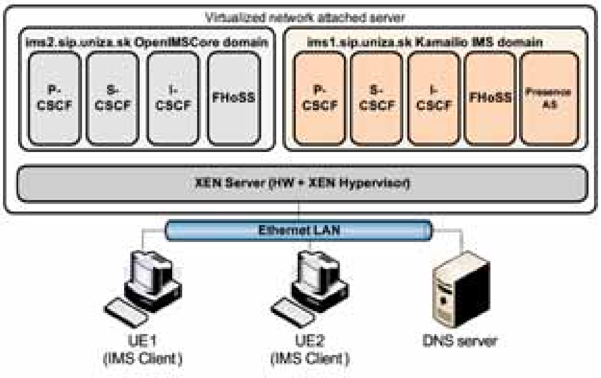
Virtualized network attached server
ims2.sip.uniza.sk OpenIMSCore domain
P-CSCF
S-CSCF
I-CSCF
FHoSS
ims1.sip.uniza.sk Kamailio IMS domain
P-CSCF
S-CSCF
I-CSCF
FHoSS
Presence AS
XEN Server (HW + XEN Hypervisor)
Ethernet LAN
UE1 (IMS Client)
UE2 (IMS Client)
DNS server

Fig. 5 IMS testbed scenario

while the basic connectivity is usually successful, the interoperability of clients using advanced services is not. Similar results are declared in [18]. In these cases a deeper structural testing and diagnostic is needed. We can recommend the Boghe IMS client following by Monster TCS as the most appropriate IMS clients. The testbed is possible to be extended and to be redundant with the deployment of more S-CSCF and P-CSCF servers. Both IMS solutions provide basic IMS feature set, with different UE authentication methods, as Digest-AKAv1/v2-MD5, Diget-MD5, SIP Digest. Additionally, both solutions support transport (Transport Layer security – TLS) and network layer (IP Security – IPSec) security mechanisms. As a result of our experiments, several articles have been published at the web site of the Department of InfoCom Networks [16].

4.1 Advanced services and other extendibility

The development of new IP services is one of the assets of IMS. We have shown that the both IMS solutions can be extended with new components providing new features and services, for which other open source solutions may be used. Our testbed was simply extended with a presence service based on the Kamailio SIP server that was used as the Presence AS. Kamailio provides an embedded XCAP server which supports the IETF SIMPLE messaging and presence standards. The Kamailio SIP server may be used as a standalone AS server as it supports Lua programming language for service creation. We identified other open source projects such as Mobicent Open Source Cloud Communications framework [19] (it consists of many subprojects as the Presence service for OMA SIMPLE, JAIN servers, SIP servlet server, media server and so on), the OpenVCS for video conferencing [20], the UCT Advanced IPTv for IPTV service [21], popular B2BUA as Asterisk and FreeSWITCH as media servers.

Tests of SIP based presence and messaging documents different level of standards adoption. The SIMPLE page mode and presence status indication work successfully. More advanced services such as the session mode which uses MSRP protocol or contact lists (address book) services (XCAP with XDM) reached poor results.

5. Conclusions

The deployment of the both open source IMS solutions with basic functionalities is quite easy and fast. It can be deployed with relatively low costs (apart from the costs required for a hosting hardware platform). Documentation is publically available; this, of course, accelerates research possibilities of universities, operators and vendors and enables the development of new IMS applications and testing new communications components. We tested the deployment of IMS testbeds which will be used in our learning and scientific practice. We also compared the two IMS frameworks; the Open IMS Core provides a bigger range in terms of installed solutions and users. This may be a result of its exclusive position for years. The Open IMS Core has simpler installation procedure than the Kamailio IMS. It has more detailed information database spread around the world describing many interesting extendibility solutions. However, the Open IMS core has not been actively evolving. Moreover, technological improvements of an underlying Kamailio SIP Server component against the SER SIP server 2.0, which is used as the basement of CSCF IMS entities, are notable. The technological advance and improvements of Kamailio SIP (actually, version 3.2.2), on which the Kamailio IMS was built, provides better future prospects. The Kamailio SIP server is a viable open source solution with many developers, testers and many installations. The number of new Kamailio IMS based installation will grow in a near future and will exploit a rich feature set of integrated solutions, based on Kamailio IMS and Kamailio AS. Even though the vision of open source projects is always difficult to be provided, the vitality of the Kamailio project is very promising.

References

[1] 3GPP TS 23.228, IP Multimedia Subsystem (IMS); Stage 2
[2] 3GPP TS 22.228, Service requirements for the Internet Protocol (IP) Multimedia core network Subsystem (IMS); Stage 1
[3] 3GPP TS 23.002: Network Architecture
[4] ETSI ES 282 001, NGN Functional Architecture
[5] SULTAN, A.: *Overview of ETSI TISPAN's Next Generation Network Release 1,* The definition of NGN, ETSI TISPAN, 2009[6] ETSI ES 282 007, TISPAN: IP Multimedia Subsystem (IMS): Functional Architecture.
[7] ETSI TS 123 517, IP Multimedia Subsystem (IMS); Functional architecture

[8] ROSENBERG, J., SCHULZRINNE, H., CAMARILLO, G., JOHNSTON, A., PETERSON, J., SPARKS, R., HANDLEY, M., SCHOOLER, E.: *SIP: Session Initiation Protocol,* RFC 3261, July 2002

[9] The Open IMS Core Project, http://www.openimscore.org

[10] The Kamailio IMS extensions, NG Voice, http://www.ng-voice.com/

[11] WEIK, P., WAHLE, S. , CARVALHO DE GOUVEIA, F. , MAGEDANz, T.: *Experiences on the Establishment and Provisioning of NGN/IMS Testbeds - The FOKUS Open IMS Playground and the related Open Source IMS Core,* 11th Intern. Conference on Intelligence in Service Delivery Networks (ICIN) 2007, Bordeaux, France, 2007

[12] The Open IMS Core entities documentation, http://www.openimscore.org/docs/ser_ims/index.html

[13] The NGN Lab EU initiative, http://www.ngnlab.eu/

[14] Kamailio – The Open Source SIP server, http://www.kamailio.org

[15] BOCK, C.: *Kamailio IMS extensions, 10 Years SER Conference,* Berlin, September 2, 2011

[16] Network information library knowledge portal, http://nil.uniza.sk/; http://nil.uniza.sk/ngnims/

[17] Doubango 3GPP IMS/RCS framework, http://doubango.org/

[18] WAITING, D., GOOD, R., SPIERS, R., VENTURA, N.: *The UCT IMS client, Testbeds and Research Infrastructures for the Development of Networks & Communities,* 5th Intern. Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities and Workshops, 2009, pp. 1–6, 2009

[19] Mobicents – Open Source Cloud Communications, http://www.mobicents.org/sip-presence/intro.html

[20] OpenVCS – Open Source Video Conferencing Server, http://code.google.com/p/openvcs/

[21] SPIERS, R., GOOD, R., MARSTON, R., VENTURA, N.: *The UCT IMS IPTV Initiative.* In Proc. of the 2009 Third Intern. Conference on Next Generation Mobile Applications, Services and Technologies, pp. 503–508, 2009.