Citation for the published Conference paper:

Title:
Testbed for Advanced Mobile Solutions

Author:
Maria Apell, David Erman, Adrian Popescu

Conference Name:
4th ERCIM Workshop on eMobility

Conference Year:
2010

Conference Location:
Luleå

Testbed for Advanced Mobile Solutions

Maria Apell, David Erman, and Adrian Popescu

Dept. of Communication and Computer
School of Computing
Blekinge Institute of Technology
371 79 Karlskrona, Sweden

Abstract.
This paper describes the implementation of an IMS testbed, based on open source technologies and operating systems. The testbed provides rich communication services, i.e., Instant Messaging, Network Address Book and Presence as well as VoIP and PSTN interconnectivity. Our validation tests indicate that the performance of the testbed is comparable to similar testbeds, but that operating system virtualization significantly affects signalling delays.

1 Introduction

The vision in network evolution comprises technology convergence, service integration and unified control mechanism across wireless and wired networks. These networks are expected to provide high usability, support for multimedia services, and personalization in a Service Oriented Architecture (SOA). Subscribers demand to be able to move between networks and at the same time have access to all subscribed services and applications regardless of the access technology. The key features are user friendliness and personalization as well as terminal and network heterogeneity. The main objective is to setup a testbed, where we carry out research and develop new solutions for the next generation mobile communications. Network convergence, i.e., using the same infrastructure for mobile and fixed networks, represents an important and long time desired advance in the delivery of telecom services. With the Internet Protocol, telecommunication systems started to migrate from circuit-switched to packet-switched technologies. The IP Multimedia Subsystem (IMS) originally specified for mobile systems has been adopted and extended by Telecommunication and Internet converged Services and Protocols for Advanced Networking (TISPAN) to deliver multimedia services to both mobile and fixed networks. The migration of networks to SOA allows resource sharing, reduced cost and shorter time to market. In [1] the authors discuss this migration of existing telecommunication applications into SOA and the techniques used are described. The authors in [2] and [3] describe how an open source based testbed can be used to create new services through service components. Focus is on the expected increase in terms of complexity and the importance of the testbed being open to new components, new technologies as well as new concepts and paradigms that enable the constant process of evolving. Similar service-oriented testbeds are discussed in [4, 5]. The authors in [6] argue for the need for real-life network, to measure the realistic performance and existing services in a testbed. To be able to run real-life scenarios our testbed is connected to PSTN. One driving technical enabler for this is virtualization. One of the main benefits of server virtualization is the ability to rapidly deploy a new system. Building and installing systems on a virtual platform is an important resource saver. Deploying new services and scaling those that already exist is faster once virtualized due to the intrinsic ability of virtualization to rapidly deploy configurations across devices and environments.

The challenge for measuring IMS performance is not necessarily at the protocol level but rather the different types of services that the network is supposed to support. A traditional Voice over IP (VoIP) network handles voice and video. An IMS network network handles voice and video but also support fixed and mobile services simultaneously. Therefore, testing in an IMS environment is more about the interaction of services rather than how well individual protocols function. In [7], The European Telecommunications Standards Institute (ETSI) has produced a Technical Specification document covering the IMS/NGN Performance Benchmark. This document contains benchmarking use-cases and scenarios, along with scenario specific metrics and design objectives. The framework outlines success rate, average transaction response time and retransmissions as the main metrics to report on for each scenario. Our paper reports on the metric for transaction response time for a subset of the scenarios defined.

In [8], the authors analyse the IMS Session Setup Delay (SSD) in CDMA2000 Evolution Data Only wireless systems. Using simulations, measurements and comprehensive analysis, the authors argue that the IMS SSD must be decreased to be a viable option for the growing needs of future services and applications. The authors of the study in [9] identify that the delay in the Serving Call Session Control Function (S-CSCF) is the main contributor to the call processing delay. In [10] the authors show that self-similar properties emerge in Session Initiation Protocol (SIP) signalling delays, modelling the SSD by using a Pareto distribution. Munir *et al* present in [11] a comprehensive study of SIP signalling and particularly identify the registration procedure as the main contributor to the signalling delay and networking traffic [12]. The authors propose a lightweight alternative registration procedure to alleviate these issues.

The rest of the paper is as follows: in section 2 we describe the architecture of our testbed. Section 3 discusses the validation procedure for the testbed and in section 4 we present initial measurement results. Section 5 concludes the paper.

2 Testbed Architecture

In this section the architecture of our testbed is described. The software used and the configuration of the nodes in the testbed is discussed.

The testbed is part of the EU EUREKA Mobicome – Mobile Fixed Convergence in Multi-access Environments project – and interconnects three sites; Blekinge Institute of Technology (BTH), HiQ [13] and WIP [14].

Signalling traffic is considered to be an important type of network traffic and lost signalling messages or congestion can have a devastating impact on all services that rely on signalling sessions. The core functionality of the IMS is built on SIP, the Internet Engineering Task Force (IETF) standardized protocol for the creation, management and termination of multimedia sessions on the Internet. The services provided by this testbed are expected to increase in terms of complexity, and it must be ensured that the testbed is capable of meeting the requirements. In addition, it should be taken into account that the utilization of the services will increase too, which results in higher load on the testbed. A test environment was created for the testbed and a test plan was developed and executed. Initially, three standardized measurements were performed to get an indication of how well the testbed performs in the management of existing services compared to other existing platforms. The test environment has been set up with the ability to meet changing requirements and test objectives.

2.1 Software architecture and configuration

Each node in the testbed has identical software, including several open source technologies to form an IMS network. The system consists of several IMS entities, where the core components are the Call Session Control Functions (CSCFs) and the lightweight Home Subscriber Server (HSS). In the IMS architecture there are three different types of CSCFs: Proxy Call Session Control Function (P-CSCF), S-CSCF and Interrogating Call Session Control Function (I-CSCF). Each entity performs its own task. The P-CSCF is the entry point to the IMS network for all IMS and SIP clients. The S-CSCF is the main part of the IMS Core and performs session control services for User Equipment (UE) and acts as registrar for them. Finally, the I-CSCF is a SIP proxy, which is the entry point in the visited network to the home network. These entities play a role during registration and session establishment and combined they perform the SIP routing function. The Home Subscriber Server (HSS) is the main data storage for all subscriber and service related data of the IMS Core [15].

IMS services can broadly be categorized in three types: services between user equipments through the IMS core (where there is no need for an Application Server (AS)), services between user equipment and AS and services that require two or more ASs to interrogate. Services provided by the IMS Core are basic VoIP, video sharing etc, while Presence and Instant Messaging are examples of services that require an AS. To manage personal profiles an XML Document Management Server (XDM Server) is needed together with the AS that handles the service for which a personal profile should be created. Our testbed handles all categories. Basic call and video sharing services are provided by the IMS Core while Presence, Network Address Book and Instant Messaging are provided through ASs. Personal profiles for these services are managed using an XDM Server together with the ASs.

All components of the testbed run several open source software systems: Focus Open IMS Core [16], Opensips [17] and OpenXCAP [18]. Focus Open IMS Core (OIC) is one of the largest and most well-documented IMS-related open source projects. It is installed on each system to provide IMS functionality. Opensips is a SIP Proxy that includes application-level functionalities including both Instant Messaging and Presence. OpenXCAP acts as an XDM Server to manage personal profiles and does also provide support for the Network Address Book. The components of OIC can be deployed in tiers and run on separate servers. The P-CSCF is usually the entity that is first placed on a separate server to protect the core and distribute the load. The testbed currently runs all CSCFs on the same server, while the ASs currently run on dedicated servers. One node in our testbed is running in a virtualized environment.

The hardware used is based on servers featuring Intel Core II duo, 2.66 GHz processors and 8 GB RAM. The servers are running a Linux 2.6 kernel with a user environment based on Ubuntu and Debian. The choice of operating system was decided based on the recommendation from the software vendors. The virtualized environment is running Linux VServer, which provides multiple Linux environments running inside a single kernel [19].

2.2 Interconnection and Topology

IMS environments contain several potential interconnection points, including connections to other IMS environments, various access networks, the PSTN as well as application services not provided in the IMS network (such as SMS).

In order to interconnect two IMS systems, each I-CSCF should recognize the other domain as a trusted network and each HSS should recognize the other domain as a visited network. DNS resolution between the networks is important as the servers running on each network must be able to resolve the domains of the other networks. The interconnections between the systems make it possible for users from different IMS networks to establish sessions with each other and the configuration of the visited and trusted network gives the users a possibility to use the services even when they visit another IMS network [20].

Users connected to different IMS networks that are interconnected in the same way as in the testbed, experience the setup procedure as for one homogenous network only. When a subscriber in one IMS network initiates a session with a subscriber in another IMS network, the CSCF recognizes that it does not serve the subscriber of the destination address. The S-CSCF also recognizes that it is interconnected with the IMS network that is serving the destination domain and the initiation message is forwarded to it.

It is possible for an IMS subscriber to access IMS services even while they are roaming in another network. The User Agent Client (UAC) receives address information to the entry point (P-CSCF) in the visited network, usually via DHCP. After authorization with this P-CSCF in the visited network, the user can then access services provided by its home IMS system. All requests from the visiting user will initially be sent to the P-CSCF in the visited network, which will forward the request through the visited network to the home IMS network via the I-CSCF in the home IMS system.

Two of the testbed systems are connected to the PSTN via SIP trunks to an Internet and telecommunication service provider in Sweden. OIC is configured with information about the interconnection with the PSTN and to match phone numbers with users in the IMS network by adding a public identity with a tel Uniform Resource Identifier (URI) containing the phone number to the IP Multimedia Private Identity (IMPI) of a user.

2.3 Call routing

When a user in network A wants to start a session with a user in network B, User Equipment (UE) A generates a SIP INVITE request and sends it to the P-CSCF it is registered with. The P-CSCF processes the request, e.g., verifies the originating users's identity before forwarding the request to the S-CSCF. The S-CSCF executes service control, which may include interactions with ASs and, based on the information about user B's identity in the INVITE from UE A, the entry point of the home network of user B is determined. The I-CSCF receives the request and contacts the HSS to find which S-CSCF is serving user B and then forwards the request to this S-CSCF. The process in the S-CSCF that handles the terminating session may include interactions with ASs but eventually it forwards the request to the P-CSCF. The P-CSCF checks the privacy and delivers the INVITE request to user B. UE B then generates a response, which traverses back to UE A following the route that was created on the way from UE A (i.e., UE B → P-CSCF → S-CSCF → I-CSCF → S-CSCF → P-CSCF → UE A)(fig.1).

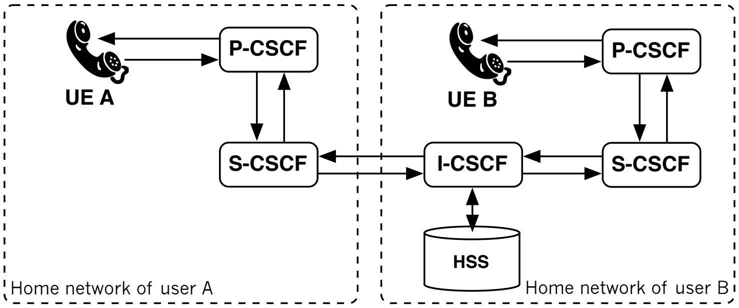


Fig. 1. Call routing between networks.

3 Testbed validation

The initial tests performed on the testbed are described in this section and the associated metrics and test scenarios are defined.

Initially, the main task of our testbed is to provide VoIP services. In a VoIP network voice and signalling communication channels are separated. Signalling sessions are mainly administered by a server, while the media stream is created point-to-point between users. SIP is a text-based signalling protocol with similar semantics to HTTP and SMTP, which is designed for initiating, maintaining and terminating interactive communication sessions between users. Such sessions include, e.g., voice, video, chat. The measurements presented in this paper focus on the signalling part given that there are standardized metrics (section 3.1) that can be performed and compared with other existing platforms.

SIP defines several components, including the following:

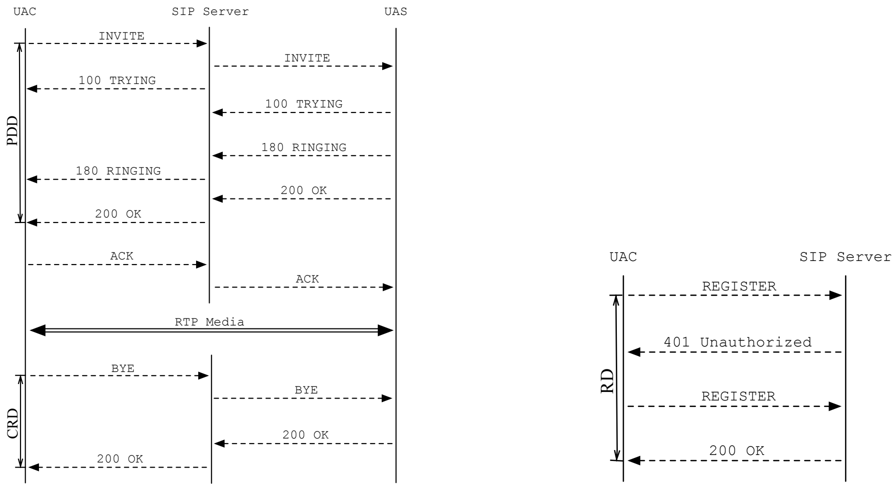
- User Agent Client (UAC): Client in the terminal that initiates SIP signalling.
- User Agent Server (UAS): Server in the terminal that responds to the SIP signalling from the UAC.
- User Agent (UA): SIP network terminal (SIP telephones, or gateway to other networks), contains UAC and UAS.

3.1 Metric definitions

A SIP call setup is essentially a 3-way handshake between UAC and UAS, as shown in figure fig. 2(a). The core methods (as defined in [12]) and responses in a call setup are INVITE (to initiate a call), 200 OK (to communicate a successful response) and ACK (to acknowledge the response). 100 TRYING means that the request has reached the next hop on the way to the destination and 180 RINGING indicates that the server which the UAS is connected to is trying to alert the UAS. When the receiver side picks up the phone the 200 OK is sent and the caller side responds with an ACK. The call is then considered as established and media transfer can take place. The release of the call is made by the BYE method and the response 200 OK to this message indicates that the call is released successfully.

Related to the call flow in fig. 2(a), and the Technical Specification by ETSI [21], the following metrics are defined:

1. Register Delay (RD): Time elapsed between when the UAC starts the registration procedure by sending a REGISTER message and when it receives the messages that the authentication was successful (time between when the UAC sends the initial REGISTER and when the UAC receives the 200 OK) (fig. 2(b)).
2. Post Dial Delay (PDD): This is the time elapsed between when the UAC sends the call request and the time the caller hears the terminal ringing (The time from when the UAC sends the first INVITE to reception of corresponding 180 RINGING) (fig. 2(a)).
3. Call Release Delay (CRD): This is the time elapsed during the disconnection of a call. It is measured between when the releasing party hangs up the phone and when the call is disconnected (the time between when the UAC sends a BYE and when it receives the response, 200 OK) (fig. 2(a)).



(a) Message flow for call setup and teardown.

(b) Register message flow.

Fig. 2. Signalling flows.

3.2 Measurement setup and execution

For the tests and measurements Hewlett-Packard SIPp [22], a free and open source SIP test tool and traffic generator was used. SIP call flows can be customized using XML files, and SIPp can provide statistics from running tests. In order to make our measurements, XML files for both the UAC and the UAS were created. The UAs, both the UAC and the UAS, are running on separate hosts for the duration of the test.

SIP works with either TCP or UDP as transport protocols but most SIP-based networks are using UDP. This means that SIP must provide the logic for retransmission of lost packets. The SIP retransmission mechanism is defined in RFC 3261 [12]. The simplest type of UAS is a stateless UAS that does not maintain transaction state. It replies to requests normally, but discards any state that would ordinarily be retained by a UAS after a response has been sent. It does not for example send informational responses (1xx) such as 100 TRYING and 180 RINGING [12]. The PDD metric depends on the informational response 180 RINGING and therefore the UAS used in the tests must be stateful. It will send 180 RINGING after receiving an INVITE and it will retransmit the following 200 OK if it is lost. In general, UAC retransmits all messages, however that is not necessary for these tests. The only message the UAC will retransmit in this tests is the BYE message, to ensure that all connected calls are also disconnected.

One user is provisioned on each system and a data file with information about this user is saved on the host where the UAC is running. All tests use the same scenario files, but the UAC uses a different data file for each system. Data files contain information about users and specific information about each system. The UAS have an identical setup in all the tests. Two scenario files are created for the UAC, one for registering with the OIC and another to setup a call with the UAS via the OIC and after 4 s starts the teardown of the call. For the UAS one scenario file is created to listen and provide responses to the SIP messages sent by the UAC for the call setup and teardown.

The tests run 10,000 iterations of each scenario. A program starts the first scenario to register as a subprocess, and when this subprocess has ended the second scenario (call setup and teardown) starts as a second subprocess. After the second subprocess has finished, the program pauses for 4 s before it starts a new iteration. The default retransmission time, (T1) is 500 ms, which is an estimate of the maximum round trip time, and the value of 64×T1 is the default transaction timeout timer [12]. This means that the pause between two iterations should be 32 s to ensure that the previous iteration has ended. It was not deemed necessary in this test as the second subprocess for call setup and teardown can not be started until the registration process ends by receiving a response on the second REGISTER. Similarly, the subprocess for call setup and call teardown can not complete until the UAC has received a response to the BYE that was sent to initiate the teardown. The call setup scenario is different from the registration scenario in that messages, which are not necessary to provide functionality, such as informational messages, are sent. The UAC will not wait for these messages before it proceeds, which means that there could be some outstanding messages in the system after the UAC has finished. Therefore a pause is needed between two iterations.

Each iteration creates two files as a result of the tests, one file per scenario. The files contain all messages sent to and from the UAC, including timestamps. The test procedure is verified when the file is parsed. Each file must contain the correct number of messages, which also have to arrive in the right order. We also verify that no messages related to a previous iteration reached the UAC in a subsequent iteration. Before the pause was introduced, up to 10 % of the messages arrived out of order, mainly in the test between two sites. However, an introduction of a 32 s pause would mean that each test would take a very long time to complete. A shorter pause was chosen as a compromise between the rate of out of order messages and total test time. The behaviour after the pauses were introduced is described in tab. 1.

The nodes in the testbed are:

- System A, BTH 1: Non virtualized environment.
- System B, BTH 2: Virtualized environment.
- System C, WIP: Non virtualized environment.
- System D, HiQ: Non virtualized environment.

The conditions for the two nodes BTH 1 and BTH 2 are identical. The hardware is identical and they are located in the same place, connected to the same switch. This switch also connects the UAC and the UAS. Systems C and D are located in two company sites in Karlskrona, Sweden. System D is not part of our study as it did not have a suitable networking infrastructure available. System C is part of the study, but the main focus was on System A and B. System B was tested in two different configurations, namely both with a vserver enabled kernel and with a non-vserver enabled kernel where we refer to the latter as System B2.

Table 1. Data from tests.

Node	Started	Completed	Discarded
System A	10,000	8,454	1
System B	10,000	7,046	1,266
System B2	10,000	9,856	1
System C	10,000	4,906	9

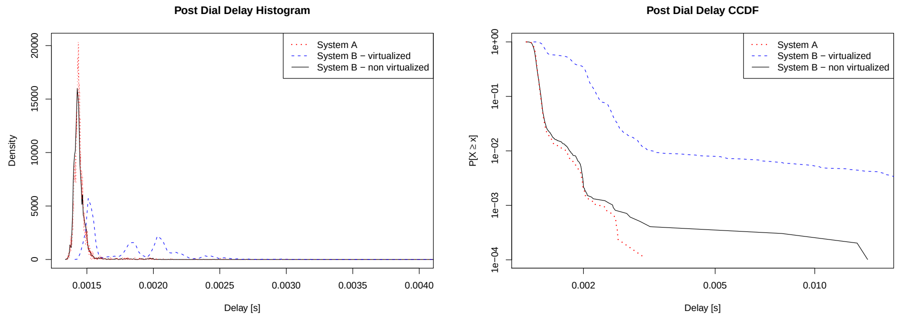
Only data from successful call setups and teardowns are included in the analysis. The files were excluded for System A, B2 and C due to failed call attempts as a result of unsuccessful registration attempts. 180 RINGING was missing in 1,260 files for System B, and 6 files contained failed call attempts, they were all excluded from the analysis. If the initial INVITE from UAC fails, no file will be created for this attempt. This explains the number of files created in System A, B and B2. In System C OIC stopped serving calls. This was preceded by two failed registration attempts in succession, which explains the even lower number of completed calls in this scenario.

4 Measurement Results

In this section we discuss the results of our tests. The main purpose of these tests was to perform standardized measurements to get an indication of how well the testbed performs in the management of existing services.

There were distinct differences in the test results. As the results turned out to distinguish between the non-virtual system, System A, and the virtual system, System B, the latter system was reconfigured into a non-virtual environment. The same tests were performed again to assess whether if the virtualization had an impact on the results or not. In order to simplify the comparison of the results, we focused on Systems A and B when analyzing the test results. As both test setups are essentially identical the results are directly comparable, and easily plotted in the same graph.

The histogram in fig. 3(a) shows that the distribution of the PDD are very similar in the non-virtual systems and that there are long tails on all the PDD distributions. This is even more pronounced in the Complementary Cumulative Distribution Function (CCDF) (fig. 3(b)). The tail is longer in the virtualized environment, which indicates that we can expect higher values of the PDD here. Arbitrary processing time has previously been modeled as Pareto distributed, making the appearance of heavy tails unsurprising [23].



(a) Histogram for the Post Dial Delay.

(b) CCDF for the Post Dial Delay.

Fig. 3. Measurement results.

The test results from System C followed the same pattern as for the non-virtual systems with a longer delay, which is explained by having a path over more network elements with greater distance. The distance between the UAs and System C is 9 IP hops and the histogram for the PDD peaks at a delay of 0.07 ms. Following the data from this result, RD and CRD were also analyzed. These metrics follow the same pattern as PDD with a similar measure of the delays. During our tests the Digest MD5 authentication method was used instead of a more complex authentication method used in [11], which may explain why we do not observe the same phenomenon with RD having higher values than PDD.

Previous work identified the S-CSCF as the main contributor in the call processing delay and the call setup time was modeled using a Pareto distribution [9]. The long tails on the PDD distributions in fig. 3(b) indicates that our testbed behaves in a similar fashion. Eaven heavier tails are to be expected when requests traverse longer links, due to the self-similar nature of network traffic [24].

The Internet and telecommunication service provider that provides the testbed with interconnection to PSTN also provides us with Call Detail Record (CDR) for one week's worth of calls, around 200,000 CDRs in each direction. From this data we calculate the average time between the INVITE is sent to the UAS and the callee picks up the phone to be 12 s making so the PDD is negligible in comparison.

5 Conclusions

In the paper we presented an implementation of a service testbed, intended for research on advanced mobile services in the future Internet, together with measurement results from the testbed.

The tests showed that the distribution of the PDD is very similar for the non-virtual systems and that there is a long tail in the distribution in both cases.

The long tail is expected, given the large number of various processing stages a request passes before being completed. Previous work [9] discussed the same scenario. Further testing is needed, where each entity in the system is analyzed under load and the behaviour of the distributions studied. The future work will follow the framework outlined in [7], and cover additional test scenarios and metrics.

Our validation tests indicate that the performance of the testbed is comparable to similar testbeds. The type of virtualization used in this tests significantly affects the PDD, both in terms of higher delays and larger delay variation. One factor behind higher delay in the virtualized scenario could be that the debugging in OIC is enabled during all tests. If the speed of writing data to the disk is affected by the virtualized environment, we expect changes to the PDD when the debugging level is reduced or disabled. To further investigate this, information needed for the tests can be cached in the main memory in order to minimize writing to disk, and see how this affects the PDD.

Another factor contributing to the delays is the CPU scheduler, which can be replaced by a scheduler that is optimized for virtual environments. There are several options for virtualization besides the Linux VServer, e.g., XEN and VMware. We will therefore evaluate the virtualization solutions as well.

Acknowledgments

The authors gratefully acknowledge the support of The Swedish Governmental Agency for Innovation Systems, VINNOVA, for the work presented in this paper. The work was done as part of the EU EUREKA project MOBICOME.

References

1. R. Chen, V. Shen, T. Wrobel, and C. Lin, “Applying SOA and Web 2.0 to Telecom: Legacy and IMS next-generation architectures,” *e-Business Engineering, 2008. ICEBE ’08. IEEE International Conference*, pp. 374 – 379, 2008.
2. T. Y. Chai, T. L. Kiong, L. H. Ngoh, X. Shao, L. Zhou, J. Teo, and M. Kirchberg, “An IMS-based testbed for service innovations,” *Next Generation Mobile Applications, Services and Technologies, 2009. NGMAST ’09. Third International Conference*, pp. 523 – 528, 2009.
3. T. K. Lee, T. Y. Chai, L. H. Ngoh, X. Shao, J. Teo, and L. Zhou, “An IMS-based testbed for real-time services integration and orchestration,” *Services Computing Conference, 2009. APSCC 2009. IEEE Asia-Pacific*, pp. 260 – 266, 2009.
4. T. Mecklin, M. Opsenica, H. Rissanen, and D. Valderas, “ImsInnovation-Experiences of an IMS testbed,” *Testbeds and Research Infrastructures for the Development of Networks & Communities and Workshops, 2009. TridentCom 2009. 5th International Conference*, pp. 1–6, 2009.
5. M. Tsagkaropoulos, I. Politis, and T. Dagiuklas, “IMS evolution and IMS test-bed service platforms,” *Personal, Indoor and Mobile Radio Communications, 2007. PIMRC 2007. IEEE 18th International Symposium*, pp. 1 – 6, 2007.

6. C. Balakrishna, “IMS experience centre a real-life test network for IMS services,” *Testbeds and Research Infrastructures for the Development of Networks & Communities and Workshops, 2009. TridentCom 2009. 5th International Conference*, pp. 1 – 8, 2009.
7. ETSI, “Telecommunications and Internet Converged Services and Protocols for Advanced Networking (TISPAN); IMS/PES Performance Benchmark,” Feb. 2010. [Online]. Available: http://www.etsi.org
8. M. Melnyk, A. Jukan, and C. Polychronopoulos, “A Cross-Layer Analysis of Session Setup Delay in IP Multimedia Subsystem (IMS) With EV-DO Wireless Transmission,” *Multimedia, IEEE Transactions on*, vol. 9, no. 4, pp. 869 –881, Jun. 2007.
9. S. Pandey, V. Jain, D. Das, V. Planat, and R. Periannan, “Performance study of IMS signaling plane,” *IP Multimedia Subsystem Architecture and Applications, 2007 International Conference*, pp. 1 – 5, 2007.
10. I. Kuzmin and O. Simonina, “Signaling flows distribution modeling in the IMS,” *EUROCON 2009, EUROCON ’09. IEEE*, pp. 1866 – 1869, 2009.
11. A. Munir and A. Gordon-Ross, “SIP-based IMS signaling analysis for wimax-3g interworking architectures,” *Mobile Computing, IEEE Transactions*, vol. 9, no. 5, pp. 733 – 750, 2010.
12. J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, “SIP: Session Initiation Protocol,” RFC 3261 (Proposed Standard), Jun. 2002, updated by RFCs 3265, 3853, 4320. [Online]. Available: http://www.ietf.org/rfc/rfc3261.txt
13. HiQ, “HiQ.” [Online]. Available: http://www.hiq.se/
14. WIP, “WIP.” [Online]. Available: http://www.wip.se/
15. M. Poikselkä and G. Mayer, *The IMS: IP Multimedia Concepts and Services.* Wiley, Jan 2009.
16. Fraunhofer FOKUS, “Open IMS Core.” [Online]. Available: http://www.openimscore.org/
17. The OpenSIPS Project, “OpenSIPS.” [Online]. Available: http://www.opensips.org/
18. AG Projects, “OpenXCAP.” [Online]. Available: http://www.openxcap.org/
19. The Linux-VServer community, “Linux VServer.” [Online]. Available: http://www.linux-vserver.org/
20. ETSI, “Telecommunications and Internet Converged Services and Protocols for Advanced Networking (TISPAN); IP Multimedia Subsystem (IMS) Functional Architecture,” November 2008. [Online]. Available: http://www.etsi.org
21. ETSI, “Quality of service (QoS) measurement methodologies,” January 2002. [Online]. Available: http://www.etsi.org
22. HP invent, “SIPp.” [Online]. Available: http://www.sipp.sourceforge.net/
23. W. Leland and T. Ott, “Load-balancing heuristics and process behavior,” *ACM SIGMETRICS Performance Evaluation . . .* , Jan 1986.
24. V. Paxson and S. Floyd, “Wide area traffic: the failure of Poisson modeling,” *IEEE/ACM Transactions on Networking*, vol. 3, no. 3, pp. 226–244, 1995.