

Integration of 3G connectivity in PlanetLab Europe

A step of an evolutionary path towards heterogeneous large scale network testbeds

Alessio Botta · Roberto Canonico ·
Giovanni Di Stasi · Antonio Pescapé ·
Giorgio Ventre · Serge Fdida

Received: date / Accepted: date

Abstract Distributed research testbeds play a fundamental role in the evaluation of disruptive innovations for the Future Internet. In recent years, the main research funding agencies have promoted several initiatives aimed at designing and building open, large scale, realistic experimental facilities that could be used to evaluate innovative networking architectures, paradigms and services. In this paper, after discussing the main challenges in building such infrastructures, we present how, by leveraging the concept of federation, we managed to introduce a first degree of heterogeneity into PlanetLab Europe, a European testbed federated with PlanetLab, by providing UMTS connectivity to PlanetLab Europe nodes. Our contribution is just a first step of an evolutionary path whose further developments will lead to next generation large-scale heterogeneous testbeds for supporting experimentally-driven research on the Network of the Future [†].

Keywords Wireless Network Testbeds, UMTS, Virtualization, Usage Models, Performance Evaluation

1 Introduction

The availability of a large scale real-world distributed research infrastructure has been widely recognized as a fun-

damental necessity to drive the evolution of the future Internet [3]. Such an infrastructure should be able to support both research initiatives aimed at developing new incremental evolutions of the current Internet and more disruptive long-term approaches aimed at radically redesigning the basic architecture of the network according to different communication and networking paradigms. Building such an infrastructure is in the agenda of both US and European research funding agencies.

ONELAB is a research project funded by the EC in its FP6, started in September 2006 with “*two overarching objectives: (1) to extend the current PlanetLab infrastructure and (2) to create an autonomous PlanetLab Europe*”. The project has terminated its activities in August 2008, successfully achieving both its goals. PlanetLab Europe is a European-wide research testbed that is linked to the global PlanetLab through a peer-to-peer federation [4]. The objective of extending PlanetLab has been pursued in ONELAB through development of a number of extensions to the PlanetLab architecture, aimed at enriching it with new wireless access technologies, with powerful traffic monitoring instruments and with new emulation capabilities. In this paper we describe in details how we managed to successfully provide 3G connectivity to stationary PlanetLab Europe nodes. We describe the specific usage model we chose for granting access to the UMTS connection within a slice and the technical challenges we had to face to integrate such support in a PlanetLab-based environment. We believe this can be useful for other researchers working on PlanetLab as well as on other distributed testbeds. To provide evidences of the integration of 3G connectivity we also present experimental results, with both TCP and UDP, showing the usefulness of having a commercial UMTS connection available in PlanetLab Europe, and some insights we gained from the analysis of these results. Since our technique is quite general and perfectly integrated into the core of the PlanetLab architecture,

A. Botta, R. Canonico, G. Di Stasi, A. Pescapé, G. Ventre
Università di Napoli Federico II, Napoli, Italy
Consorzio Interuniversitario Nazionale per l'Informatica CINI
Via Claudio 21, 80125, Napoli, Italy
{a.botta,roberto.canonico, giovanni.distasi,
pescape,giorgio}@unina.it

S. Fdida
LIP6, Université Pierre et Marie Curie, Paris 6
104 Avenue du Prèsident Kennedy, 75016 Paris, France
serge.fdida@lip6.fr

[†]Preliminary results within the same framework of this work have been recently published in [1, 2].

we claim that our approach can be also pursued to extend PlanetLab with support for other networking technologies.

The rest of the paper is organized as follows. In Section 2 we illustrate the main challenges we have to face to extend a planetary scale testbed like PlanetLab to overcome some of its limitations. In Section 3 we describe in details the UMTS component we developed for PlanetLab Europe, in the context of the ONELAB project. In Section 4 we illustrate the results of an experimental analysis of a commercial UMTS connection performed using ONELAB nodes. In Section 5 we draw our conclusions and discuss about ongoing efforts aimed at producing large-scale heterogeneous wireless testbeds.

2 Enhancing PlanetLab

As of today, the most relevant large scale distributed testbed for networking research is PlanetLab [5]. PlanetLab is a geographically distributed testbed for deploying and evaluating planetary-scale network applications in a highly realistic context. Nowadays the testbed is composed of more than 900 computers, hosted by about 400 academic institutions and industrial research laboratories. Since its initial deployment in 2003, PlanetLab has become of indisputable importance for the academic research community and industry, which use such infrastructure for the evaluation of a wide range of distributed systems. Examples of scenarios that have been tested on PlanetLab are peer-to-peer systems [6], overlay [7] and content-distribution networks [8], and network measurement frameworks [9]. Figure 1 shows a conceptual view of the current architecture of the PlanetLab testbed, whose node set is the union of disjoint subsets, each of which is managed by a separate authority. As of today, two such authorities exist: one located at Princeton University (PLC) and another located at Université Pierre et Marie Curie UPMC in Paris (PLE).

To run a distributed experiment over PlanetLab, users need to be associated to a *slice*, which is a collection of virtual machines (VMs) instantiated on a user-defined subset of all the testbed nodes. Slices run concurrently on PlanetLab, acting as network-wide containers that isolate services from each other. An instantiation of a slice in a particular node is called a *sliver*. Slivers are Virtual Machines created in a Linux-based environment by means of the VServer virtualization technology [10]. By means of so-called *contexts*, VServer hides all processes outside of a given scope, and prohibits any unwanted interaction between a process inside a context and all the processes belonging to other contexts. VServer is able to isolate services with respect to the filesystem, memory, CPU and bandwidth. However, it does not provide complete virtualization of the networking stack since all slivers in a node share the same IP address and port space [11]. The adoption of VServer in PlanetLab is mainly

motivated by the need of scalability, since up to hundreds of slivers may need to be instantiated on the same physical server [12].

Each testbed authority (PLC and PLE) contains an entity called Slice Authority (SA), which maintains state for the set of system-wide slices for which it is responsible. The slice authority includes a database that records the persistent state of each registered slice, including information about every user that has access to the slice [13].

Testbed authorities also include a so called Management Authority (MA), which is responsible of installing and managing the updates of software running on the nodes it manages. It also monitors these nodes for correct behavior, and takes appropriate action when anomalies and failures are detected. The MA maintains a database of registered nodes at each site. Each node is affiliated with an organization (owner) and is located at a site belonging to the organization.

Over the last few years, PlanetLab's usefulness has appeared to be somewhat jeopardized by some limitations, such as:

1. lack of heterogeneity,
2. limited usage models for resource allocation to experiments,
3. limited manageability at a planetary scale.

In the following subsections we briefly discuss about these limitations and describe the pragmatic approach that we decided to pursue in the ONELAB project to overcome some of them. In section 3 we will describe in details our contribution to the integration of UMTS connectivity to PlanetLab nodes.

2.1 Heterogeneity

One of the main limitations of PlanetLab is the lack of heterogeneity, that jeopardizes the effectiveness of such infrastructure in providing realistic results from the experimentations [1], [14], [15]. Nearly all PlanetLab nodes are server-class computers connected to the Internet through high-speed research or corporate networks. This contrasts with the real Internet in which connected devices also comprise laptops, hand-held computers, cellular phones, network attached storage devices, video-surveillance appliances, as well as other household devices. Besides, all such systems are connected to the Internet through a variety of access technologies, both wired (ADSL, CATV) and wireless (WiFi, GPRS, UMTS, CDMA, etc.). As a consequence, it has also been noted that the behavior of some applications on PlanetLab can be considerably different from that on the Internet [16]. To introduce more heterogeneity into the testbed, several approaches have been pursued. Dischner et al., for instance,

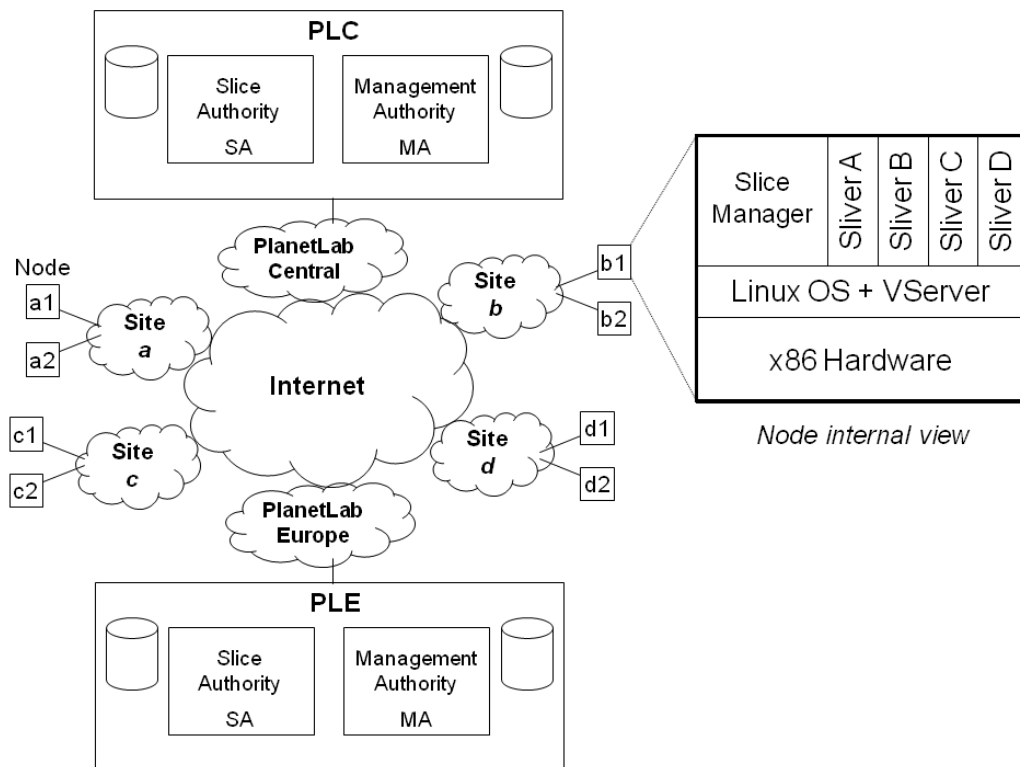


Fig. 1 Conceptual PlanetLab architecture.

have recently proposed an architecture (SatelliteLab) to include into the experiments a variety of non-dedicated devices made available by residential users [17]. Such boxes are not part of the testbed but rather controlled by special PlanetLab nodes. In the context of the ONELAB project, the problem of integrating different kinds of technologies (comprising UMTS, WiMax, wireless ad hoc) jointly with other kinds of components (e.g. programmable emulation boxes) has been addressed. The integration of such a wide range of technologies into the testbed was done by defining specific usage models for each of them. These usage models differ in the level of concurrency that has been allowed and in the techniques that have been used to implement such concurrency.

2.2 Resource allocation and concurrency management

To maximize the benefit/cost ratio, multiple concurrent experiments should be able to use the infrastructure resources at the same time. However, concurrency of resource usage might lead to interference and produce meaningless and not replicable results. This problem has already been addressed in several testbeds, usually by the adoption of proper resource sharing mechanisms to guarantee the isolation of experiments.

The simplest resource sharing mechanism consists in relying on the native CPU and packet schedulers implemented in the nodes of the testbed to let several processes, possibly belonging to different experiments, share both computational and communication resources. Unless special schedulers are adopted, this approach does not provide any isolation among concurrent experiments. Hence, it is viable only if shared resources never get exhausted by any single experiment, thanks to over-provisioning.

A more reliable approach, usually referred to as *slicing*, consists in allocating a subset of physical resources to each experiment. Slicing is usually implemented by means of *virtualization*. Virtualization is a widely used technique in which a software layer creates multiple instances of a logical resource by multiplexing a single physical resource. In a virtualized system, slicing is then obtained by allocating logical instances of the same resource to different slices.

Virtualization of *computing resources* is becoming more and more sophisticated and its efficiency may benefit from the use of specialized features embedded in last generation CPUs. The application of virtualization to manage *communication resources*, on the other hand, is not straightforward and still object of current research.

In particular, nodes of a shared network testbed may need to virtualize the following communications resources,

each of which has its own challenges: (i) link capacity; (ii) network devices as seen by the OS; (iii) routing tables.

Multiplexing the *link capacity* among several virtual network interfaces is a problem typically addressed by network emulation testbeds. This problem may be solved in different ways, at different levels. For some wireless technologies, the multiplexing problem may be addressed at the physical layer, by letting the network interface use several radio channels at the same time (e.g. by continuously switching among two or more channels). This kind of approach is usually referred to as *wireless virtualization* [18], but unfortunately it is not easily applicable to all varieties of wireless technologies. Furthermore, the interference issue poses two important challenges that are not observable in a wired network: (i) coherence, i.e. when a transmitter of one experiment is active, all of the corresponding receivers and potential sources of interference as defined by the experiment should be simultaneously active on their appropriate channels of operation, and (ii) isolation, i.e. when a node belonging to one experiment is receiving some signal pertinent to the experiment, no transmitter of a different experiment within the communication range of the receiver should be active in the same or a partially-overlapping channel.

Virtualization of wired links, on the other hand, is usually performed at higher layers. A MAC-level approach for Ethernet-like switched networks is based on IEEE 802.1Q VLANs (virtual LANs), that can be used to associate outgoing packets with different tags [19]. A properly configured switch may then associate different tags to different shared media segments or point-to-point links.

In PlanetLab, slice creation and resource allocation are decoupled. When a slice is first instantiated on a node, the availability of part of the node's resources may be guaranteed to the slice. Some other virtualized resources are instead granted to each sliver according to the current level of contention, according to a best effort model. Such an approach has been deliberately chosen in the original PlanetLab design. However, this approach is not suitable in heterogeneous testbeds to allocate resources (e.g. a UMTS card, a special NIC based on a programmable FPGA, etc.) whose virtualization is challenging, if not impossible. For these kinds of resources, advance reservation mechanisms may be more suitable to grant exclusive usage to specific slices over limited time intervals.

2.3 Federation for improved manageability

Until July 2007, only one single operations center hosted at Princeton University, known as *PlanetLab Central*, was responsible for the management of all the PlanetLab machines. Peer-to-peer federation [4] between PlanetLab Central (PLC) and PlanetLab Europe (PLE) has been successfully established in the framework of the ONELAB project.

Now, thanks to federation, it is possible for researchers to setup distributed experiments involving nodes associated to either PLC or PLE, so that the two federated testbeds appear to users as a unique seamless infrastructure. As soon as PLE was up and running, a proper migration process was established and implemented, to let ONELAB take over the administration of PlanetLab nodes across Europe. Of course, the first sites to be migrated were those of ONELAB partners. Now that PLE is active, its operation needs coordination of efforts with Princeton researchers and technical staff. Hence periodic coordination meetings are held and common management practices are developed and used to keep the federation in good shape. Both the central management entities and the distributed nodes of the two infrastructures are updated when major software versions are released. More recently, an agreement has been reached to further extend federation to PlanetLab Japan (PLJ), a consortium created to foster the PlanetLab community in Japan.

3 The ONELAB UMTS extension

In order to introduce more heterogeneity into the PlanetLab architecture we added UMTS connectivity to PlanetLab Europe nodes. More precisely, our aim was to provide every node of the testbed with the possibility of using a UMTS interface and to handle the related Point-to-Point Protocol (PPP) connection. While this task can be easy in a common Linux box, it is not as easy in a PlanetLab node. This is because such nodes have a modified kernel and also because very limited privileges are granted to normal users within an unprivileged sliver context.

Our UMTS extension has been initially implemented and validated in a private PlanetLab deployment, the so-called *Private OneLab*, whose resources are provided by and accessible to ONELAB members only. After testing and validating our extension in this private testbed, we have made it publicly available in PlanetLab Europe [20]. In this section we will firstly explain why we decided to integrate UMTS connectivity into PlanetLab and then how we did it.

3.1 Motivations

UMTS, Universal Mobile Telecommunications System, is a third-generation (3G) cellular phone technology standardized by 3GPP¹. This standard has been adopted by several telecom operators all over the world, and it has become largely popular to provide both voice services and ubiquitous high-speed access to the Internet. The number of UMTS users reached 400 millions and it is still increasing [21]. Most recent releases of the UMTS standard support

¹<http://www.3gpp.org/>

both voice traffic and data traffic in a common IP-based network infrastructure. In such releases, also thanks to the introduction of the High Speed Downlink/Uplink Packet Access (HSDPA and HSUPA), data rates can be as high as 14 Mbps downstream and 5.8 Mbps upstream.

The choice of integrating UMTS connectivity in PlanetLab Europe was motivated by different reasons. Firstly, UMTS has become a competitor of other well known access technologies such as WiFi and xDSL. A second reason is related to the applications that are spreading over such network infrastructure. The IP Multimedia Subsystem (IMS) [22], which is being progressively introduced in the UMTS architecture, is triggering the development of new generations of network applications such as presence, conferencing and location-based services. Such applications are attracting an increasing number of users and should be therefore taken into account in a realistic network testbed. An example of emulation-based testbed used to reproduce the behaviour of 3GPP networking scenarios is proposed in [23].

Our main goal was neither emulating a UMTS network nor integrating private small-scale UMTS testbeds into PlanetLab, but rather to allow PlanetLab institutions to equip their nodes with such kind of connectivity by using a Telecom Operator of choice. In principle, this approach allows to perform experiments by using the UMTS connection provided by different networks and to compare the results. To accomplish our task in OneLab we have used two UMTS networks: a private UMTS micro-cell provided by Alcatel-Lucent Italia and located at their 3G Reality Center in Vimercate (Italy), and a commercial connection provided by one of the principal European telecom operators. While the first network is private and could be used only thanks to the collaboration of Alcatel-Lucent Italia, the second is a public UMTS network, that we used without any particular agreement with the operator.

3.2 Usage model

Before even approaching the problem of integrating UMTS support into the PlanetLab software system, we had to define a proper usage model for this specific extension, i.e. a proper model for sharing such a resource (the UMTS connection) in the PlanetLab virtualized environment [1]. To this end, we decided to adopt a policy that allows only one experiment (i.e. slice) at a time to control and use the UMTS interface. This is motivated by two main reasons: (i) the low bandwidth of a UMTS connection would cause high interference between concurrent experiments, and (ii) in order for the experiment to be realistic, the dial-up connection should be set up and torn down just before and after the test respectively.

To foster the worldwide deployability of UMTS in PlanetLab, we chose to support two of the most widely used net-

work interface cards (at the beginning of our work): the Option Globetrotter GT+ 3G card and the Huawei e620card.

The nodes equipped with a UMTS interface are also provided with an Ethernet NIC (Network Interface Controller). The wireless UMTS connection is used for the experiments while the wired Ethernet is used for control data. This is also necessary because the UMTS connectivity provided by the operators often employs firewalls or filters that do not allow to reach the UMTS-equipped host by using terminal services such as *ssh*. When the UMTS connection is established, the default route is left to the Ethernet interface. In order for a slice to use the UMTS connection it is necessary to specify the destinations for which the UMTS connection is required or to explicitly bind to the UMTS interface. The control of the UMTS connection is provided to the users by means of a special *umts* command, which allows to start, stop, and check the status of the connection, as well as to add and delete the destinations previously mentioned. These commands have to perform operations requiring superuser (i.e. root) privileges, which is not allowed from inside a slice. To overcome this problem we use the features provided by *vsys*, as explained in the following section.

3.3 Implementation details

The PlanetLab node operating system is derived from a Fedora Core Linux distribution (Fedora Core 8 as of August 2008) with some patches. The patches are mainly targeted to the virtualization features needed to support the concurrency of the experiments and the related networking issues. To add support for the UMTS interfaces we needed to add both kernel modules and user-space tools.

The kernel modules we had to integrate into the distribution are those related to the management of the PPP connection (*ppp-generic*, *ppp-filter*, *ppp-async*, *ppp-sync-ty*, *ppp-deflate*, and *ppp-bsdcomp*) and those required by the two NICs, i.e. *pl2303* and *usbserial* for the Huawei card, and *nozomi*² for the Globetrotter card. The *nozomi* module required some modifications in order to run with the latest PlanetLab kernel, based on Linux kernel version 2.6.22.

In user space some tools are needed to establish the PPP connection and then to discipline the access to it. The first goal is achieved thanks to the *comgt*³ and *wvdial*⁴ tools. *comgt* is used to register into the operator network. *wvdial* establishes the actual PPP connection. The second goal is achieved by means of the *iproute2*⁵ and *iptables*⁶ tools.

²<http://www.pharscape.org/>

³<http://sourceforge.net/projects/comgt>

⁴<http://freshmeat.net/projects/wvdial>

⁵<http://www.linuxfoundation.org/en/Net:Iproute2>

⁶<http://www.netfilter.org>

Their use is aimed at isolating the slice that requested the UMTS interface (*UMTS slice* in the following), by allowing only that slice to use the UMTS interface. In particular, *iproute2* is used for (1) creating an additional routing table, which contains only a default route that points to the UMTS interface, and for (2) defining the rules that allow only specific packets to be routed through the UMTS interface using the additional table. Such packets are those matching one of the following rules: i) those generated by the *UMTS slice* and directed towards the destinations for which the user requested UMTS interface; ii) those generated by the *UMTS slice* and having source address equal to that of the UMTS interface. Packets generated by the *UMTS slice* are recognized by means of a mark applied with *iptables*, exploiting a feature of the new VNET+ subsystem of Planetlab⁷. Packets generated by other slices will not be able to use the UMTS interface (as they are either not marked or marked differently) and will continue to get routed following the rules contained in the default routing table. In order to enforce the isolation between slices (i.e. allowing only one slice at a time to use the UMTS connection), we also add another rule with *iptables* that drops all the packets not generated by the *UMTS slice* and that are about to go out using the UMTS interface. Such rule is necessary to cope with special cases, such as a user of a different slice that tries to generate packets destined to the other endpoint of the PPP connection or that binds an application to the UMTS interface.

All these user space tools require superuser privileges and, therefore, must run in the root context of a PlanetLab node. For PlanetLab users, however, it is not possible to execute commands with such privileges. To overcome this problem, PlanetLab developers have created a utility called *vsys*⁸ which allows to execute a program with root privileges from inside a slice. Basically, *vsys* creates a pair of FIFO pipes that allow a program in a slice (i.e. the front-end program) to communicate with another program in the root context (i.e. the back-end program). The front-end program receives the commands requested by the user, and pass them to the back-end program, which runs in the root context and can execute programs with superuser privileges. The front-end and the back-end programs we developed implement the following features:

- *start*: check and lock the UMTS interface, set up the UMTS connection, and enforce the routing rules;
- *stop*: tear down the UMTS connection, delete the routing rules, unlock the interface;
- *status*: check the status of the connection;
- *add destination*: add a rule to reach *destination* through the UMTS connection;
- *del destination*: delete the rule for *destination*.

⁷<http://www.cs.princeton.edu/~sapanb/vnet>

⁸<http://www.cs.princeton.edu/sapanb/vsys>

As of now, the software we developed is not provided with accounting capabilities and therefore does not enforce any given limit of time or traffic for the UMTS connection. Some efforts are ongoing, however, in order to add this missing capability, so as to allow the node owner define limits of utilization (e.g. number of hours of connection per slice) or to bill users for their connections.

4 Experimental analysis of a commercial UMTS connection using ONELAB nodes

In this section we show how it is possible to experimentally study the performance of a UMTS link using both PlanetLab nodes and the tools we developed (described in Section 3). For this analysis, we equipped one of our PlanetLab nodes (*onelab00.dis.unina.it*, located in Italy) with a commercial UMTS connection provided by one of the major 3G operator in Italy. We then utilized the tools presented in Section 3 to setup and manage the UMTS connection, and we generated synthetic traffic towards another PlanetLab node (*onelab09.inria.fr*, located at INRIA in France). This allowed us to provide (i) evidences of the work we done in adding 3G connectivity to PlanetLab based nodes and (ii) a characterization of the performance of a UMTS connection. Even if this analysis is far from being a through performance study, it constitutes a use case for the introduced features, providing some interesting insights into the UMTS connections commonly used. In more details, we performed two kinds of experiments: (i) we assessed the performance of the UMTS up-link in non-saturated conditions, i.e. using low-bitrate traffic, with both UDP and TCP; (ii) we measured the the maximum throughput achievable in the uplink direction (i.e. that with the lowest capacity) of a UMTS connection using both UDP and TCP, also evaluating other performance indicators in such saturated conditions. In Figure 2 the setup for the experiments is shown. The characteristics of the traffic we used and the details on the experiments are reported in Section 4.1. To have a performance reference, all the experiments have been performed also using the Ethernet connections of the same machines. This means that, for each experiment, we measured the characteristics of two network paths connecting the same couple of hosts: the first path connects the UMTS interface of *onelab00.dis.unina.it* to the Ethernet interface of *onelab09.inria.fr* (we refer to this path as “*UMTS-to-Ethernet*”); while, the second path connects the Ethernet interface of *onelab00.dis.unina.it* to the Ethernet interface of *onelab09.inria.fr* (we refer to this path as “*Ethernet-to-Ethernet*”). This allowed to pin-point which of the observed behaviors were depending on the UMTS connections, and therefore discover the real causes of the obtained results.

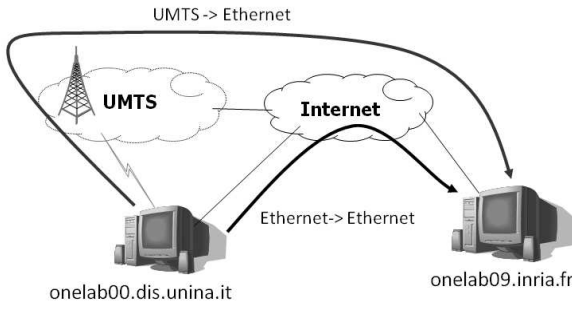


Fig. 2 Network setup for experiments.

4.1 Methodology

Probing traffic was generated using D-ITG (Distributed Internet Traffic Generator) [24], a platform capable to produce traffic at packet level accurately replicating appropriate stochastic processes for both IDT (Inter Departure Time) and PS (Packet Size) random variables (exponential, uniform, cauchy, normal, pareto, ...). D-ITG supports both IPv4 and IPv6 traffic generation and it is capable to generate traffic at network, transport, and application layer. Moreover, it can perform one-way-delay (OWD), round-trip-time (RTT), packet loss, jitter, and throughput measurements. Finally, it is capable to store information about sent and received packets on both the sender and the receiver sides.

We compared the characteristics of the two end-to-end paths by using two traffic classes carried by both UDP and TCP. The first one was obtained by generating a low- and constant-bitrate traffic resembling the characteristics of a real VoIP call using codec G.711 (i.e. 72 Kbps obtained with packet size equal to 80 Bytes of voice samples + 12 Bytes of RTP header = 92 Bytes of payload, and packet rate equal to 100 pps). The other one was obtained by using a 1-Mbps CBR traffic (packet size equal to 1024 Bytes and packet rate equal to 122 pps). The former kind of traffic allows to assess the performance of the UMTS in non-saturated conditions and the feasibility of a voice call over the UMTS, also comparing the achievable quality with that on a high capacity link (“Ethernet-to-Ethernet” path). The latter kind, instead, allows to analyze the behavior of the UMTS up-link in fully saturated conditions (as we will see in the following 1-Mbps is more than double the capacity of the UMTS uplink). All the experimented lasted for 120 seconds and were repeated 20 times, obtaining very similar results.

After the traffic generations ended, we retrieved the log files from the two nodes and we analyzed them by means of ITGDec, another component of D-ITG suite. In this way we obtained the samples of four QoS parameters that are bitrate, jitter, loss, and round-trip time (RTT). Such samples represent the average values calculated over non-overlapping windows of 200 milliseconds. In the following subsections we show the time plot of the samples of such parameters

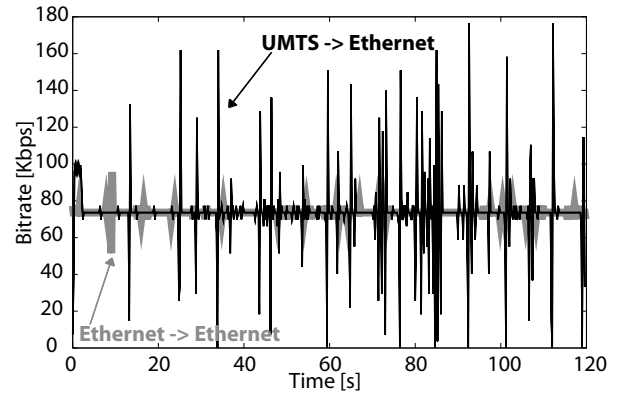


Fig. 3 Bitrate of the UMTS in non-saturated conditions using UDP.

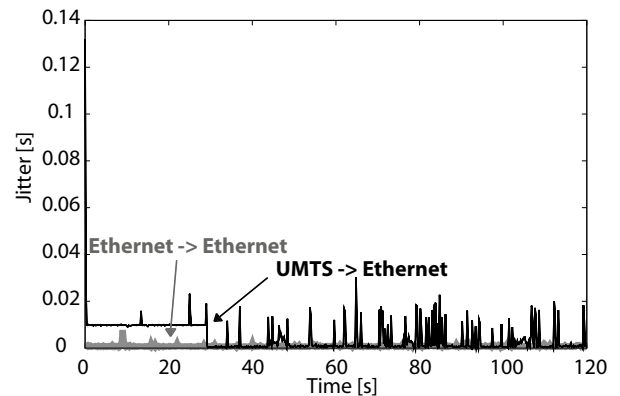


Fig. 4 Jitter of the UMTS in non-saturated conditions (UDP).

obtained with the two traffic classes, using the two network paths, and for both UDP and TCP. For all the tests with TCP, the packet loss results are not reported because we are interested in observing the performance perceived by the applications and, with TCP, the losses are not visible at application layer.

4.2 Results

In the next sections we study the behavior of the UMTS connection in both saturated and non saturated condition with both UDP and TCP.

4.2.1 Non saturated conditions

UDP Figures 3-5 show the time plots of the bitrate, jitter, and round trip time respectively. The packet loss is not reported for this experiment because it was always equal to 0. Each figure contains the results obtained on both “UMTS-to-Ethernet” and “Ethernet-to-Ethernet” paths. As we can see at first look, all the parameters present different behaviors in the two cases.

In more details, we observe that the bitrate of the UMTS connection is more fluctuating than in the Ethernet case even

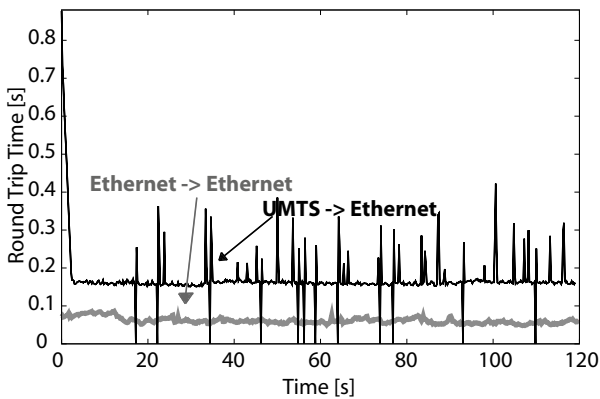


Fig. 5 RTT of the UMTS in non-saturated conditions (UDP).

though, in both cases, the required imposed value is achieved in average. The jitter plots, instead, show the UMTS connection introduces a higher jitter, which is also more fluctuating. It reaches values up to 30 milliseconds, which, however, can be easily compensated and still allow a VoIP communication to be satisfactory for the users [25]. Moreover, we can observe that such high values are obtained only in the first seconds of the communication, after which the jitter is much lower. This behavior will be explained in the following section. In any case, these results have to be taken into account when designing novel applications and systems using UMTS connections for real-time communications. Finally, looking at the round-trip delay, we can observe that the average value is higher for the UMTS connection with respect to the Ethernet one. Moreover, as seen for the jitter, the RTT is more fluctuating on the wireless connection and it reaches values up to 700 milliseconds.

TCP To study the case of non saturated conditions with TCP in a real scenario, we analyze the results we obtained with the VoIP-like flow and TCP. The use of TCP for both voice and video of traffic has highly increased in the last years [26]. This is mostly because, with respect to UDP, TCP can more easily traverse firewalls and NAT boxes, and the bandwidth available on many Internet paths allows to overcome the delays introduced by such protocol.

In Figure 6 we report the throughput obtained with TCP in non saturated conditions. As we can see, the required value (i.e. 72 Kbps) is achieved in average even if the UMTS presents a more spiky trend with respect to the Ethernet. Figure 7 shows the jitter obtained with TCP. It can be noticed that the maximum jitter can be as high as 60 ms. This value, however, is reached only in very few periods while in most cases the values are around 20 ms. This means that, as far as the jitter constrains are concerned, the UMTS connection allows to perform a VoIP call using TCP too.

The RTT of the UMTS in non-saturated conditions observed with TCP is reported in Figure 8. If we compare these values with those obtained with UDP (Fig. 5), we can

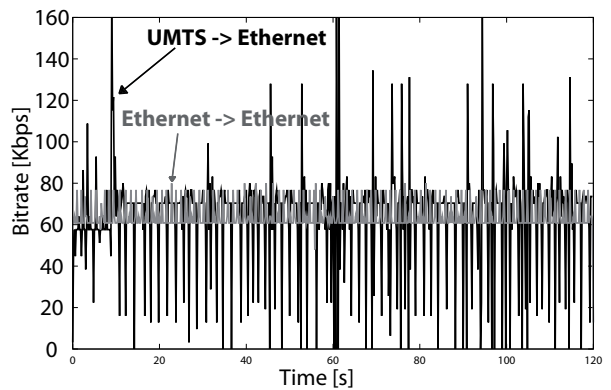


Fig. 6 Bitrate of the UMTS in non-saturated conditions (TCP).

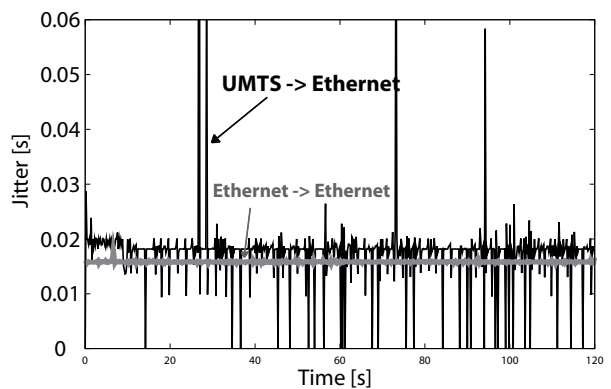


Fig. 7 Jitter of the UMTS in non-saturated conditions (TCP).

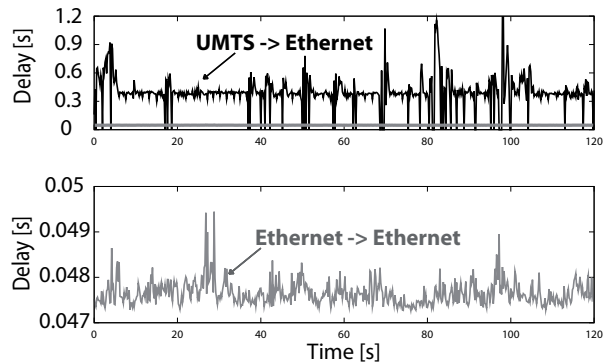


Fig. 8 RTT of the UMTS in non-saturated conditions (TCP).

observe the impact of TCP. Such protocols, indeed, causes RTT values nearly doubled and reaching an average value of about 350 ms and maximum values of more than 1 s: this means that it is very difficult to perform a VoIP call using TCP, at least at a satisfactory quality. On the contrary, looking at the bottom graph of Figure 8, we can observe that on the *Ethernet-to-Ethernet* path the RTT never reaches 50 ms, which allows to perform multimedia communications with a good quality [25].

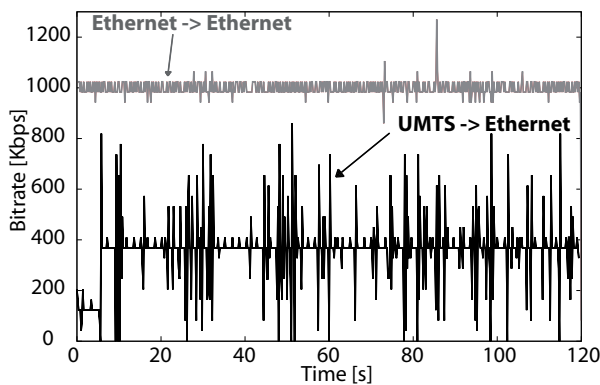


Fig. 9 Bitrate of the UMTS in saturated conditions (UDP).

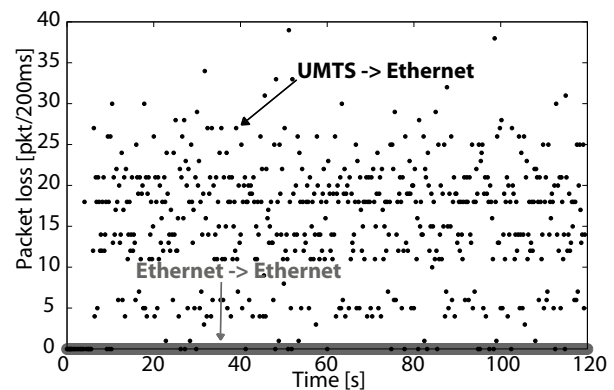


Fig. 11 Loss of the UMTS in saturated conditions (UDP).

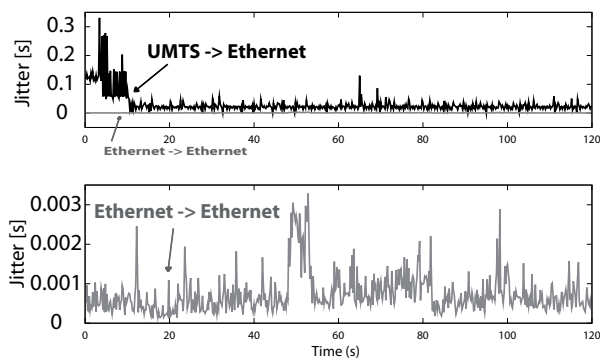


Fig. 10 Jitter of the UMTS in saturated conditions (UDP).

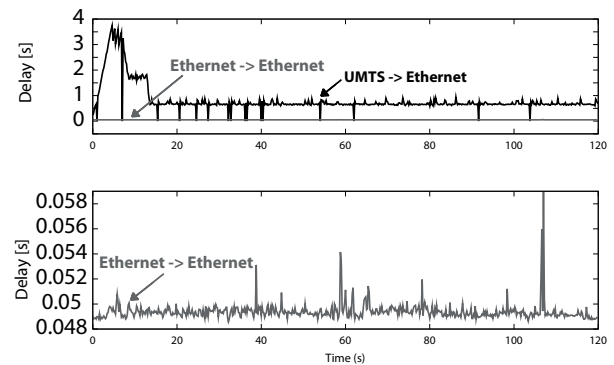


Fig. 12 RTT of the UMTS in saturated conditions (UDP).

4.2.2 Saturated conditions

UDP Figures 9-12 show the bitrate, jitter, loss, and round trip delay we obtained on both end-to-end paths. This is to show the differences between the characteristics of these two network connections in the case of a 1-Mbps flow, which clearly saturates the up-link of the UMTS connection but not the Ethernet. As a first consideration, we observe that the bitrate of the UMTS reaches a maximum value of around 400 Kbps. Such value is lower than the required one, and it is therefore representative of the maximum capacity of the up-link of the UMTS connection. The jitter, packet loss, and round-trip delay plots show the very low performance achieved by the UMTS connection. Such results are justified by the fact that such connection is operating in saturation, and therefore all the QoS parameters are heavily affected. In particular we can see that the jitter reaches values larger than 200 milliseconds, which makes a real time communication nearly impossible. This is also confirmed by the values of the RTT which can be as large as 3 seconds.

Besides, looking more closely at the figures we observe that for the UMTS connection there is a first time period in which the situation is much worse than in the rest of the time. In particular, if we look at Figure 9 we can see that in the first 5 seconds the achieved bitrate is about 150 Kbps.

After that time, instead, the bitrate is more than doubled. This is due to an adaptation algorithm implemented by the UMTS network, which allocates the network resources to the users in a *on-demand* fashion⁹ [27]. The change of link characteristics has also an effect on the other parameters, which have an increment of the performance after the first 5 seconds.

TCP In Figure 13 we report the bitrate obtained with TCP in saturated conditions. In such case we are generating a rate that is higher than the capacity of the UMTS link. Therefore, as shown in this figure, the imposed bitrate is never achieved, while the average obtained value is around 400 Kbps and the plot is very spiky. In the *Ethernet-to-Ethernet* case, instead, the requested rate is achieved and the trend is smooth.

The obtained jitter is reported in Figure 14. As we can see, for the *UMTS-to-Ethernet* case, the maximum value is about 100 ms and spikes are present for the entire duration. Instead, for the *Ethernet-to-Ethernet*, the trend is very smooth and the values lower than 5 ms.

⁹Basically, a dedicated channel is assigned to the mobile station if necessary. Otherwise, a shared channel is used.

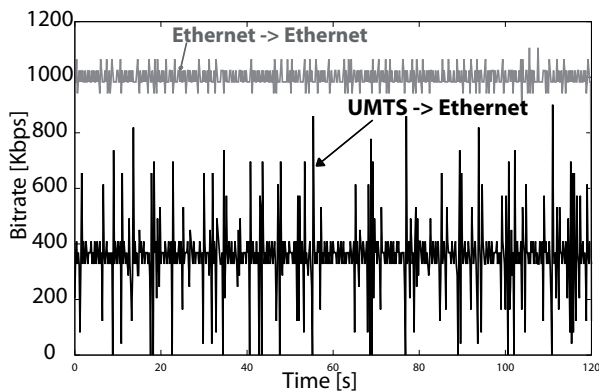


Fig. 13 Bitrate of the UMTS in saturated conditions (TCP).

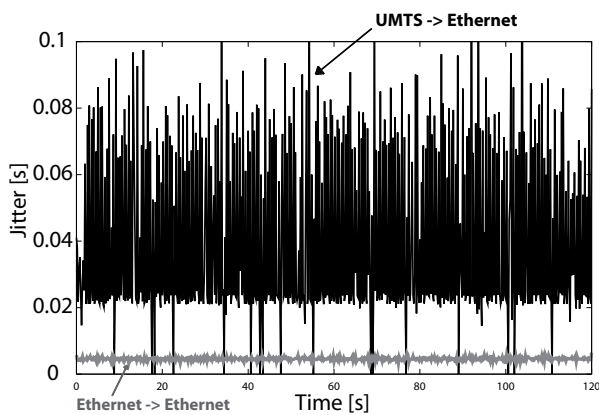


Fig. 14 Jitter of the UMTS in saturated conditions (TCP).

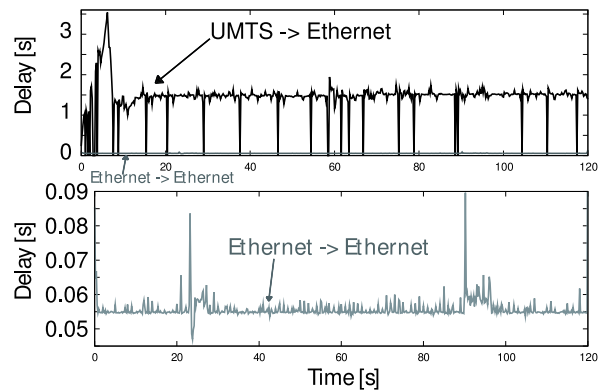


Fig. 15 RTT of the UMTS in saturated conditions using TCP.

Finally, in Figure 15 the RTT values are reported. For the *UMTS-to-Ethernet* path the average of values is high (about 1.5s). This makes it impossible to have satisfactory communications.

5 Conclusions

Distributed research testbeds have become very important for the research community as they support Internet-scale

experimentations and represent strategic resources for the study and the analysis of the Future Internet. PlanetLab represents the most notable example of such infrastructures and, despite its wide use, it still lacks of heterogeneity, especially in the access networks of the nodes, which translates into non highly realistic scenarios for the experiments.

In this paper we presented the main challenges in building distributed research testbed infrastructures and then the efforts we made to solve the problem of integrating UMTS connectivity into a PlanetLab based testbed. We detailed the problems we encountered and the solutions we found. We also presented a case study illustrating a possible application of the features we introduced.

Our contributions, as well as other extensions developed in the context of the OneLab project, are just first steps of an evolutionary path that is still far to be completed. In such a process, a key role is played by *federation*. While so far federation has been mainly used for homogeneous testbeds, the next challenge is to further extend the concept of federation across heterogeneous testbeds. Federation between PlanetLab and EMULAB is currently being investigated in the context of the GENI initiative [28]. Federation of heterogeneous testbeds is also addressed in the context of the ONELAB2 project, a follow-up of ONELAB funded by the EC in its 7th Framework Programme and started in September 2008. In particular, the authors of this paper are currently investigating possible models of integration of local wireless testbeds, based on the ORBIT management framework (OMF) [29] into PlanetLab-based testbeds [30].

6 Acknowledgments

This work has been funded by the European Projects “OneLab” (FP6 IST-2005-034819) and “OneLab2” (ICT FP7 IP 224263).

References

1. Roberto Canonico, Salvatore D’Antonio, and Giorgio Ventre. Extending the planetlab usage model for distributed heterogeneous research testbeds. In *2nd International Workshop on Real Overlays & Distributed Systems, ROADS’07*, Warsaw, Poland, July 2007.
2. Alessio Botta, Roberto Canonico, Giovanni Di Stasi, Antonio Pescapé, and Giorgio Ventre. Providing UMTS connectivity to PlanetLab nodes. In *Proceedings of the 3rd International Workshop on Real Overlays & Distributed Systems, ROADS’08*, Madrid, Spain, December 2008.
3. Tom Anderson, Larry Peterson, Scott Shenker, and Jonathan Turner (Eds.). Report of NSF Workshop on Overcoming Barriers to Disruptive Innovation in Networking. Technical Report GENI Design Document 05-02, GENI, January 2005.
4. Stephen Soltesz, David Eisenstat, Marc Fluczynski and Larry Peterson. On the design and evolution of an architecture for testbed

- federation. In *Paper presented at the 2nd International Workshop on Real Overlays & Distributed Systems, ROADS'07*, Warsaw, Poland, July 2007.
5. Brent Chun, David Culler, Timothy Roscoe, Andy Bavier, Larry Peterson, Mike Wawrzoniak, and Mic Bowman. PlanetLab: An Overlay Testbed for Broad-Coverage Services. *ACM SIGCOMM Computer Communication Review*, 33(3):00–00, July 2003.
 6. A. Rowstron and P. Druschel. Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems. *Lecture Notes In Computer Science*, 2218:329–350, 2001.
 7. L. Subramanian, I. Stoica, H. Balakrishnan, and R.H. Katz. OverQoS: offering Internet QoS using overlays. *ACM SIGCOMM Computer Communication Review*, 33(1):11–16, 2003.
 8. Limin Wang, Vivek Pai, and Larry Peterson. The Effectiveness of Request Redirection on CDN Robustness. *SIGOPS Oper. Syst. Rev.*, 36(SI):345–360, 2002.
 9. R. Sherwood and N. Spring. Touring the internet in a TCP sidecar. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pages 339–344. ACM New York, NY, USA, 2006.
 10. Linux-VServer. (Online, accessed 12 Nov. 2009) <http://linux-vserver.org/>.
 11. Sapan Bhatia, Marc Fiuczynski, Andy Bavier, and Larry Peterson. Lightweight os support for a scalable and robust virtual network infrastructure. In *2nd International Workshop on Real Overlays & Distributed Systems, ROADS'07*, Warsaw, Poland, July 2007.
 12. Stephen Soltész, Herbert Pötzl, Marc E. Fiuczynski, Andy Bavier, and Larry Peterson. Container-based operating system virtualization: a scalable, high-performance alternative to hypervisors. *SIGOPS Oper. Syst. Rev.*, 41(3):275–287, 2007.
 13. L. Peterson, S. Muir, T. Roscoe, and A. Klingaman. Planetlab architecture: An overview. *PlanetLab Design Note PDN-06-031*, available at <https://www.planet-lab.org/doc/pdn>, 2006.
 14. Himabindu Pucha, Y. Charlie Hu, and Z. Morley Mao. On the impact of research network based testbeds on wide-area experiments. In *Proceedings of IMC '06, the 6th ACM SIGCOMM conference on Internet measurement*, Rio de Janeiro, Brazil, December 2006.
 15. S. Banerjee, T.G. Griffin, and M. Pias. The Interdomain Connectivity of PlanetLab Nodes. In *Passive and Active Network Measurement*, volume 3015 of *Lecture Notes in Computer Science*, pages 73–82. Springer, 2004.
 16. J. Ledlie, P. Gardner, and M. Seltzer. Network Coordinates in the Wild. In *Proceedings of NSDI, Cambridge, MA, April, 2007*.
 17. Marcel Dischinger, Andreas Haeberlen, Ivan Beschastnikh, Krishna P. Gummadi, and Stefan Saroiu. Satellitelab: adding heterogeneity to planetary-scale network testbeds. *SIGCOMM Comput. Commun. Rev.*, 38(4):315–326, 2008.
 18. Sanjoy Paul and Srinu Seshan. Virtualization and Slicing of Wireless Networks. Technical Report GENI Design Document 06-17, GENI Wireless Working Group, September 2006.
 19. Daniel Herscher, Steffen Maier, and Kurt Rothermel. Distributed emulation of shared media networks. In *Proceedings of the 2003 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS 2003), Montreal (Canada)*, pages 226–233, 2003.
 20. OneLab SVN repository: UMTS component (Online, accessed 12 Nov. 2009). <http://svn.onelab.eu/planetlab-umts-tools/trunk/>.
 21. UMTS Forum: UMTS Forum and NGMN Alliance confirm cooperation agreement (june 2008). (Online, accessed on 12 Nov. 2009) <http://www.umts-forum.org/content/view/2463/174/>.
 22. G. Camarillo and M.A. Garcia-Martin. *The 3G IP Multimedia Subsystem (IMS): Merging the Internet and the Cellular Worlds*. John Wiley & Sons, 2006.
 23. Miguel Gómez Rodríguez, Fermín Galán Márquez, and Emilio J. Torres Mateos. A 3gpp system architecture evolution virtualized experimentation infrastructure for mobility prototyping. In *TridentCom '08: Proceedings of the 4th International Conference on Testbeds and research infrastructures for the development of networks & communities*, pages 1–10, ICST, Brussels, Belgium, Belgium, 2008. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).
 24. Antonio Pescapé Alessio Botta, Alberto Dainotti. Multi-protocol and multi-platform traffic generation and measurement. In *INFOCOM 2007 DEMO Session, May, 2007*.
 25. Ralf Steinmetz. Human perception of jitter and media synchronization. *IEEE Journal on Selected Areas in Communications*, 14(1):61–72, Jan 1996.
 26. D. Bonfiglio, M. Mellia, M. Meo, N. Ritacca, and D. Rossi. Tracking down skype traffic. *IEEE INFOCOM 2008. The 27th Conference on Computer Communications.*, pages 261–265, April 2008.
 27. A. Barbuzzi, F. Ricciato, and G. Boggia. Discovering parameter setting in 3g networks via active measurements. *Communications Letters, IEEE*, 12(10):730–732, October 2008.
 28. University of Utah and Princeton University Proposal. Statement of Work: Exploring Federation of Testbeds with Diverse Models. Technical report, 2008.
 29. Thierry Rakotoarivelo, Max Ott, Guillaume Jourjon, and Ivan Seskar. OMF: A control and management framework for networking testbeds. In *ROADS'09, 4th Workshop on Real Overlays and Distributed Systems*, October 2009.
 30. G. Di Stasi, S. Avallone, and R. Canonico. Integration of OMF-based testbeds in a global scale networking facility. In *Proceedings of ICST QShine 2009, the Sixth International ICST Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness*, November 2009.