Open-Source Software Virtualization Tools for Programmable Modular Router Implementation and Testing

(Extended Abstract)

Francesco Apollonio, Carlo Caini, Walter Cerroni, Daniele Lacamera, Carla Raffaelli
DEIS - University of Bologna
viale Risorgimento, 2 - 40136 Bologna, Italy
e-mail: f.apollonio@ldlabs.org, {carlo.caini,walter.cerroni}@unibo.it, root@danielinux.net, carla.raffaelli@unibo.it

I. PROGRAMMABLE MODULAR ROUTER

Advanced transport technologies offer large capacities in both core and access telecommunication networks. This opportunity can be fruitfully exploited to meet the requirements of new network service markets which strongly promote emerging bandwidth-demanding applications, such as large data transfers, immersive video-conferencing, high-definition TV and cinema services. At the same time, new pervasive applications as well as more traditional services – like voice-over-IP, instant messaging, web browsing and e-mail – are increasingly being delivered to mobile network terminals through broadband wireless access. To smartly support the emerging needs, networks of the future should exploit different available technologies and be able to manage a growing range of services on top of a heterogeneous infrastructure by means of integrated management and control planes.

Both network and node concepts must be significantly revised to allow this evolution. In particular, the network should be considered as a programmable system, where a customer is capable of setting up specific network layer functions when required, similarly to the programming of general purpose computers for processing functions [1], [2]. This approach can take advantage of an open, modular router architecture [3] based on emerging standard for node design and network management, such as IETF ForCES [4], [5] and NETCONF [6] respectively. Each node module implements a specific router functionality, with distinction among data, control and management planes. A further module decomposition in the forwarding plane takes into account hardware dependent and independent functions.

This idea is empowered by the possibility to run each module on a different virtual context and to configure router functions in a completely distributed, dynamic and platform-independent way through smart management tools and interfaces, as illustrated in Fig. 1. The use of open-source software virtualization tools to implement a programmable modular router architecture is one of the main contributions of the University of Bologna research unit to the Italian national project titled “Software router to Improve Next-Generation Internet” (SFINGI) [7], funded by MIUR from 2011 to 2013.
The outcome of the study is intended to show the potential benefits that a programmable network based on a virtualized modular architecture is able to offer to next generation Internet service and infrastructure providers. The impact of the proposed solution on router performance in terms of throughput and processing overhead will also be assessed by means of a multi-node virtual test-bed, according to the scheme shown in Fig. 2. This test-bed will be essentially based on software virtualization tools, which then represent a key instrument for the development of this research work.

II. SOFTWARE VIRTUALIZATION TOOL FOR IMPLEMENTATION AND TESTING

VirtualBricks [8] is one of the software tools that will be considered for the aforementioned purposes in the SFINGI project. It is intended to be the virtualization solution for the GNU/Linux platforms. In brief it is a front-end for the management of QEMU/KVM virtual machines (VMs) and VDE virtualized network devices (switches, channel emulators, etc.). It can be used to manage either isolated VMs, or testbeds consisting of many VMs interconnected by VDE elements. In particular, it allows the user to design and manage complex virtual networks, where the most important channel characteristics (delay, loss, bandwidth) are emulated by the “wirefilter” VDE tool, which does not require a dedicated VM (all VDE tools run on the host machine).

Moreover, by creating VPNs between two (or more) remote hosts it is possible to build distributed virtual testbeds, i.e. testbeds where the VMs are distributed on multiple hosts. In this case, VirtualBricks allows the user to manage all VMs, i.e. the whole testbeds, from a GUI on the main host machines. Among the wide variety of possible applications we cite the following: test, run (and debug!) different operating systems; provide an accurate tool for network protocols debugging and testing with multiple nodes and link characteristics (both for research an teaching); manage a testbed running reliable experiments before production of distributed solutions; easily connect remote real and virtual systems in the same private network with no need for a separate VPN tool; work toward a flexible solution for production environments (e.g., clouds and distributed server farms virtualization).

Figures 3 and 4 show two screen captures of the VirtualBricks GUI: the former displays the main program window, where all the virtual machines and networks currently running on QEMU/KVM are listed and can be easily managed; the latter shows the “brick” selection window, where the user can choose a new element to be added to the virtual network topology.

REFERENCES