Enabling Physical Interaction with Virtualized Testbeds for Hands-on Networking Courses

Mark Schmidt, Florian Heimgaertner, Michael Hoefling, and Michael Menth
University of Tuebingen, Dept. of Computer Science, Tuebingen, Germany
{mark-thomas.schmidt,florian.heimgaertner,hoefling,menth}@uni-tuebingen.de

Abstract—Hands-on networking courses are an important part in IT education and provide students practical qualification for their later jobs. We present a networking lab testbed that makes use of virtualization while still enabling the course participants to physically interact with cables and devices.

I. MOTIVATION

Hands-on networking courses enable students to improve their practical qualifications by interacting with real networking devices, performing cabling, and configuring services. At the University of Tuebingen we offer two hands-on networking courses: an undergraduate course, and a graduate course. The undergraduate course includes topics like networking basics, forwarding, routing, DHCP, DNS, transport protocols, and e-mail. This course is partly based on the manual of Liebeherr and El Zarki [1]. The graduate course focuses more on advanced and security-related topics like SSL/TLS, firewalls, VPNs, WiFi security, and VoIP.

Originally, we had a lab infrastructure consisting of 3 physical testbeds, each consisting of 6 PCs and 2 Cisco 2514 routers. A logical view of the setup is given in Figure 1. We used this infrastructure many years and over the time the hardware degraded and had to be replaced.

In the process of planning the upgrade, we recognized some shortcomings of physical testbeds. Besides high power consumption and emission of heat and noise, space requirements proved to be a major limitation. Additionally, the number of students interested in the practical networking courses increased. To satisfy the demand, we had to double the number of testbeds which was not possible using the old setup with regard to space and budget.

A common solution to overcome these issues is virtualization. However, a pure virtual environment cannot provide hands-on experience as physical interaction with the components is not possible. Our solution combines the advantages of both approaches. We developed a virtualized testbed as depicted in Figure 2. It comprises 6 PCs and 2 routers running as virtual machines (VMs) on a testbed host, which is a sufficiently powerful server machine. We made the network interfaces of the VMs available on a patch panel of a so-called interconnection cabinet. This allows physical interactions, in particular cabling, which is essential for a practical networking course. This approach has the additional benefit that the server infrastructure can be efficiently reused for computation purposes. Moreover, students learn also about virtualization technology and VLAN when being introduced to the testbed architecture.

We presented our proposed solution as a demo [2] at the ACM SIGCOMM conference and as a technical paper [3] at the ACM SIGITE conference. The audience’s reactions and discussions confirmed the need for such a solution.

This demo differs from [2] as it focuses on the usage of the virtualized testbed rather than on its technical implementation.

II. ARCHITECTURE OF THE VIRTUALIZED TESTBED

The 6 PCs and 2 routers of each testbed are realized as VMs on a single host using the Linux Kernel-based Virtual Machine (KVM) [4]. We use Ubuntu 14.04 LTS [5] as operating system and the libvirt [6] framework to manage and configure the
VMs. The software infrastructure is entirely based on open-source software.

The platform requirements for the testbed host include a 64 bit x86 CPU with hardware virtualization support [7] and an Input/Output Memory Management Unit (IOMMU). For presenting the network interfaces of the VMs to the users as physical ports, we need a network interface card (NIC) with Single-Root I/O Virtualization (SR-IOV) capability and a managed switch supporting VLANs [8].

Figure 3 shows a single testbed. The networking hardware is placed in a standard 19-inch wall-mount cabinet. A managed switch is mounted at the bottom facing backwards. Two panels with DB9 and RJ45 outlets above the switch provide access to the virtual routers. Unmanaged switches and other networking devices for use in the lab exercises are placed on a rack-mount shelf. The topmost rack unit contains a labeled patch panel providing Ethernet ports for the virtualized PCs.

The NIC of the testbed host provides 4 physical ports. Those ports are implemented as separate devices called Physical Function (PF). Each PF can provide up to 7 Virtual Functions (VF). The VFs can be individually passed through to VMs. At the PFs, VLAN tags are inserted into Ethernet frames originating from the VFs, before they are forwarded to the switch. The PFs remove VLAN tags of incoming frames from the switch, and forward the frames to the VF matching the VLAN ID from the tag. The virtualized testbed also allows connecting USB devices to VMs.

The managed switch is configured to (de-)multiplex the VLANs to separate switch ports that are connected to the outlets of the patch panel.

More technical information is available in [3].

III. DEMONSTRATION

We demonstrate the usage of the virtualized testbed by performing a dynamic routing experiment of our undergraduate course adapted from [1]. The experiment shows the effects of changing the physical cabling at the interconnection cabinet.

The topology of the experiment is shown in Figure 4: PC1 can reach PC4 over 4 Ethernet hops.

It is set up by interconnecting the corresponding interfaces on the patch panel with cables after starting the VMs. The network interfaces and routing services are configured on the respective VMs using a graphical console. We add an additional serial link between the two routers, shown in Figure 5, using USB/serial-converters passed through to the VMs. This reconfiguration triggers an update of the routing tables. Although packets take a shorter path of 3 instead of 4 hops, we observe an increased round-trip time between the PC1 and PC4 which is due to the usage of the slow serial link. Thus, this experiment reveals a shortcoming of the hop-count metric for routing.

More technical information is available in [3].

References