Poster ID: THP038

technische universität dortmund

First Experience with the Introduction of Virtualization Techniques into the DELTA Control System

D. Schirmer, P. Towalski, O. Kopitetzki*

Center for Synchrotron Radiation (DELTA) – TU Dortmund University – D-44221 Dortmund – Germany

* Now at VARIAN Medical Systems Particle Therapy GmbH – D-51429 Bergisch Gladbach – Germany

Introduction

In computer science virtualization is a framework of dividing the resources of a computer into multiple execution environments by applying one or more concepts or technologies such as hardware and software partitioning, time-sharing, partial or complete machine simulation and many others. Virtualization technologies (VTs) are in use since many years for several usage scenarios. The list of reasons for and benefits of virtualization is rather long [6]. The main motivations for the implementation of VT at DELTA are:

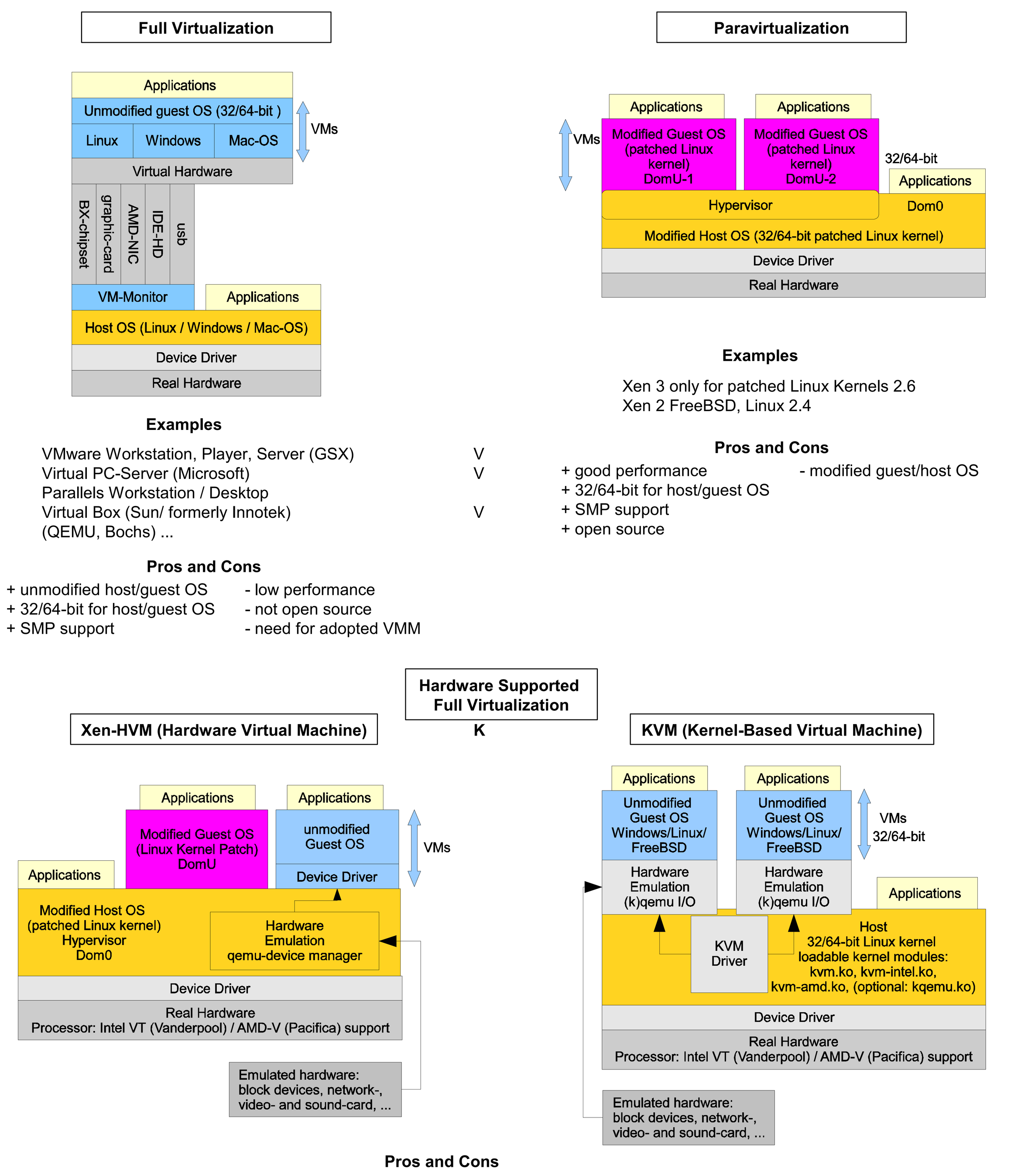
Server consolidation:
VT allows to move multiple separated servers onto a single physical host with performance and fault isolation provided inside the virtual machine (VM) boundaries. Hence, the number of physical machines can be reduced and the workload of several under-utilized processors can be optimized.

Legacy systems support:
VT enables legacy applications and operating systems to run on newest hardware without major upgrades.

Test and development agility:
VT offers the possibility to run multiple OSes simultaneously on the same hardware and provides a secure, isolated 'sandbox' for running untrusted applications. In this way it serves as a development and test environment.

Network simulation:
Virtualization can also be used to simulate complex networks of independent real and/or virtual computers.

Virtualization Technologies



Full virtualization:
The virtual machine simulates enough hardware to allow an unmodified guest OS to be run in isolation. Typical examples are Parallels Workstation/Desktop for Mac (http://www.parallels.com/de), VirtualBox (http://www.virtualbox.org), Virtual PC (http://www.microsoft.com/windows/virtual-pc), VMware Workstation/Server (formerly GSX Server)(http://www.VMware.com), QEMU (http://www.qemu.org) or Bochs (http://bochs.sourceforge.net). This approach leaves the guest and host system untouched, but for the price of markedly lower performance due to the emulation layer.

Paravirtualization:
The virtual machine does not necessarily simulate hardware, but instead or in addition offers a special API that can only be used by modified guest OS, e.g. Xen [9]. This concept offers good performance but suffers from the drawback that guest and host OS have to be modified for every kernel release.

Hardware-assisted virtualization:
The hardware provides architectural support (e.g. AMD-V and Intel VT) that facilitates building a virtual machine monitor and allows guest OSes to be run in isolation. Examples are: KVM (Kernel-based Virtualization Monitor [4] and Xen-HVM (http://www.xen.org). This concept does not need OS modification but on the other hand, it requires recent VT compatible hardware and is limited to x86 hosts, so far. Furthermore, except for CPU instructions and memory management, some hardware must be simulated which reduces the I/O performance in such cases

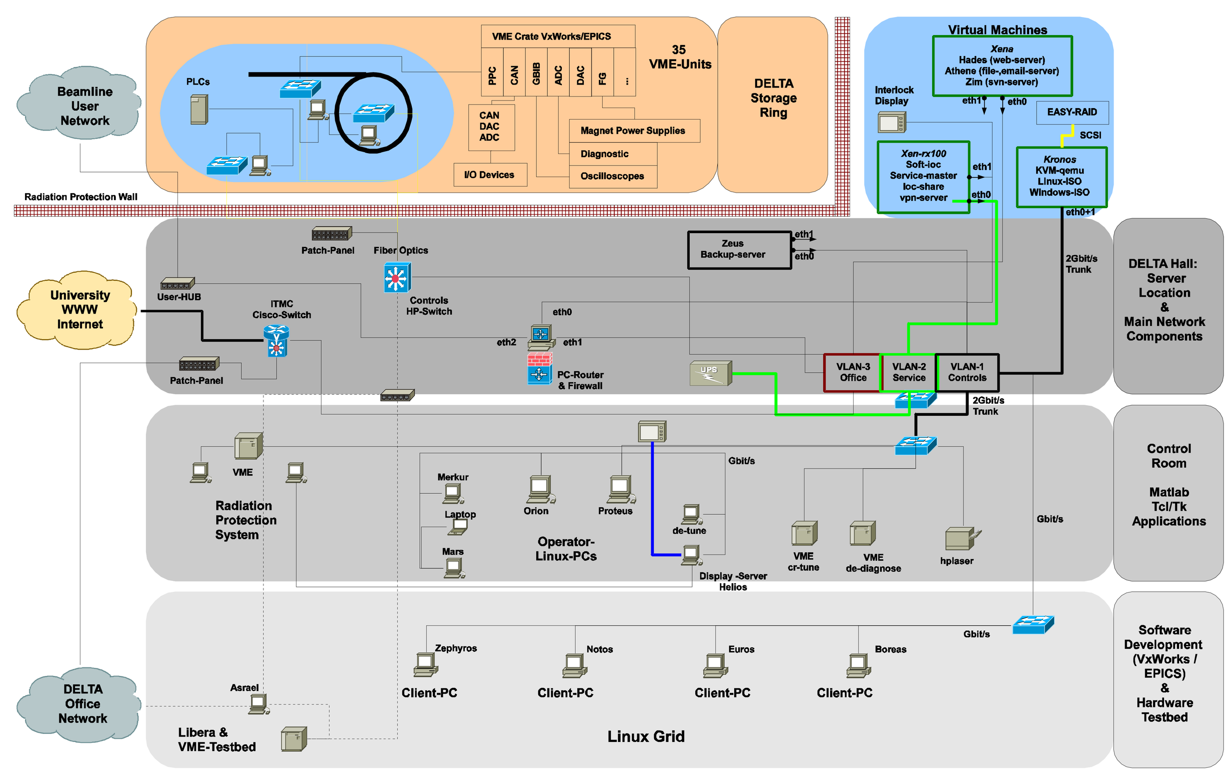
References

[1] S. Khan et al., PAC 2009, Vancouver, Canada.
[2] T. Weis et al., RuPAC 2006, Novosibirsk, Russia.
[3] P. Barham et al., Xen and the Art of Virtualization, Univ. of Cambridge, SOSP~2003.
[4] KVM Homepage:http://www.linux-kvm.org
[5] D. Schirmer et al., Status of the DELTA Control System, Proceedings of the 11th ICALEPCS'07, Knoxville, USA.
[6] A. Singh, An Introduction to Virtualization, kernelthread.com, January 2004.
[7] Wikipedia VT comparison: http://en.wikipedia.org/wiki/Comparison_of_platform_virtual_machines
[8] A.Shah, Deep Virue, Kernel based virtualization with KVM, ISSUE~86, January 2008.
[9] Wikipedia: http://en.wikipedia.org/wiki/Virtual_machine

Abstract

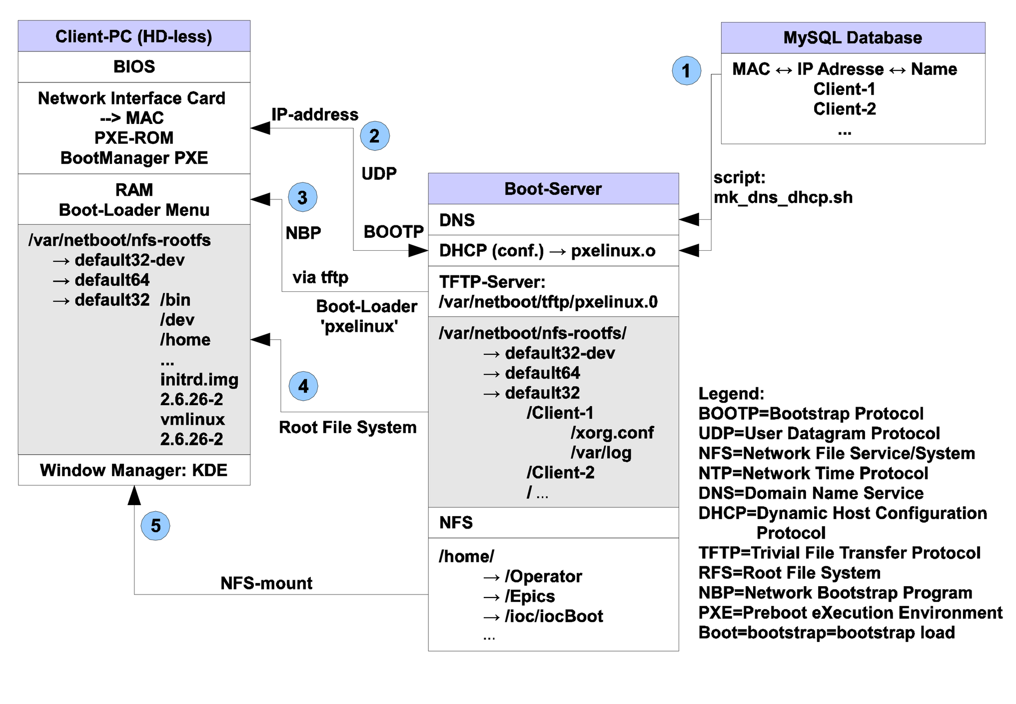
After many years of operation the client/server-architecture of the EPICS-based control system [5] at the synchrotron light source DELTA [1,2] has been modernized. Due to successive augmentation with additional computers for dedicated tasks, the previous topology grew in the course of time. As a result, the maintenance effort increased while the efficiency of the processor load decreased. Here, the introduction of virtualization methods offers a way out. After comparison studies, two different implementations of virtualization technology concepts were put into action, Xen [3] and KVM [4].

Control System Topology



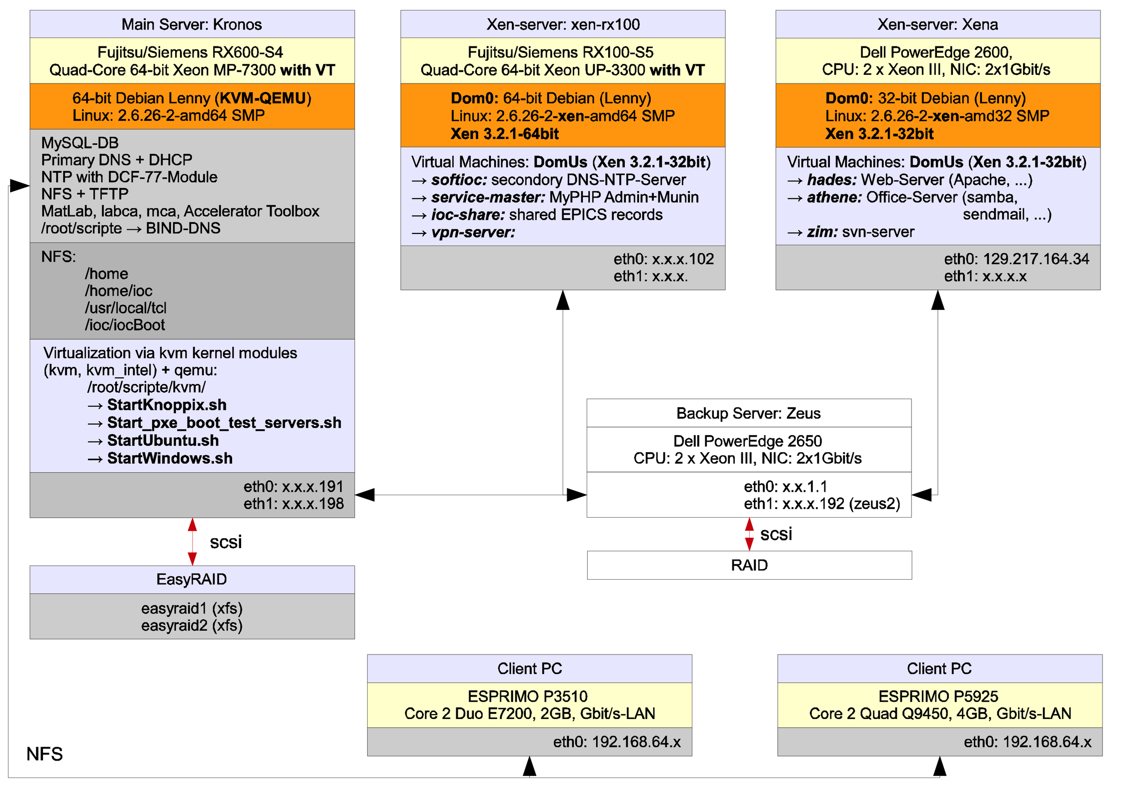
The computer network at Delta is divided in three main domains:
Office net: All bureau PCs (mostly running Windows or Linux OSes) and corresponding peripheral components are connected to this 100baseTX/FX-net. Access to the Internet is submitted by a gateway switch which is in the area of authority of the IT Media Center (ITMC) of the TU Dortmund University.
Beamline user net: Due to security reasons and to decouple machine and beamline control, all computers for synchrotron radiation beamline operation are subsumed in this domain.
Machine control system net: This subnet itself is separated again in four locally separated areas:
1. Testbed: Linux-Client-PCs for VxWorks and EPICS software development as well as for I/O-hardware tests. The PCs can also form a Linux-Grid which allows load balancing for higher CPU-performance.
2. Control Room: Operator client PCs for machine control and monitoring, the radiation protection system as well as some VME-EPICS-servers providing diagnostic data are located in this area. The network bandwidth has been upgraded to 1-2Gbit/s.
3. Machine hall: Outside the radiation protection wall all main servers and active network components are concentrated at the central network wiring node.
4. Machine Accelerators: Inside the radiation protection wall the majority of VME-IOCs (VxWorks, EPICS), PLCs and some dedicated client PCs are distributed. Accses to a variety of different I/O-devices is obtained by CAN, GPIB or serial field bus systems.
Secure data transfer between all these networks is an very important issue. Thus, we implemented a dedicated harddisk-less router-PC, processing a highly configurable firewall software (http://www.devil-linux.org) which is supplemented with a virtual LAN.

Client Boot Concept



All novel clients are equipped with PXE-able (Preboot eXecution Environment) network interface cards (NICs) and can boot their OS from a central server providing a common Linux kernel and the associated root file system (rfs). All client MAC-addresses and corresponding IP-numbers are stored in a MySQL database. The DHCP-server receives the latest MAC-IP-name-mapping from the database by performing an update script manually or via a cronjob **(1)**. Then, at client boot time, the pxe-boot-manager connects to the DHCP-server **(2)** which allows booting the Linux boot loader by uploading the file 'pxelinux' using the trivial file transfer protocol (TFTP)**(3)**. Afterwards the kernel can be loaded **(4)** and common directories are mountable via NFS **(5)**.
Up to now, three releases of common kernels with the related rfs are selectable at boot time (production system 32bit, development system 32bit and 64bit). Client dependent log files and special hardware issues (e.g. Xconfig) are stored in client-specific directories on the central server, too. All other common files, like home-directories, EPICS data, KDE configurations or GUI-applications are mounted via NFS. Since all control system applications (e.g. Tk/Tcl- and GUI-scripts, MatLab) are running locally on the client's CPU and RAM, the processor load is well distributed and the central server is not overloaded. Sufficient data transfer bandwidth is supplied by a Gbit/s network. Thus the look, feel and behavior of all clients as well as their specific configurations are managed from one central location. This standardization has simplified the client maintenance substantially.

Server Consolidation



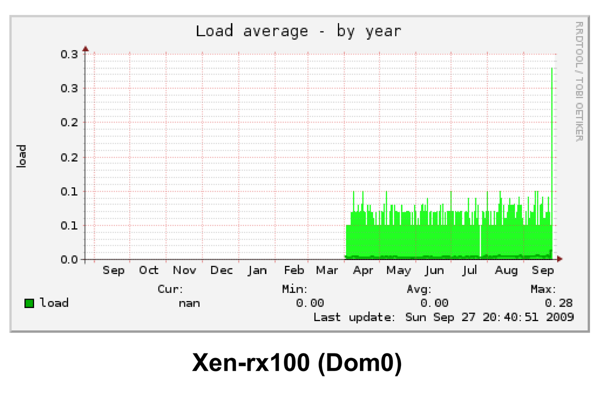
As for the clients a similar situation emerged for the servers. In the course of time the demand for additional software services increased continuously (servers like: samba, file, email, web, boot, database, MatLab, backup, monitoring, soft-iocs, svn, vpn). In parts, this need could only be satisfied by the integration of additional computers which in return produced additional maintenance effort. After many years of continuous operation the probability of age-related server-failures has increased and it was necessary to replace main parts of the hardware. Limited place in the conditioning server cabinet gave further reason for a server consolidation and forced us to look for alternative solutions. Here, VT offers a way out to reduce the number of physical machines, to optimize the hardware exploitation of the servers and as a sideline affords plenty of other advantages (e.g. issues like: portability, live host/guest migration, scalability, backup and disaster recovery, snapshots).

Decision Criteria for DELTA

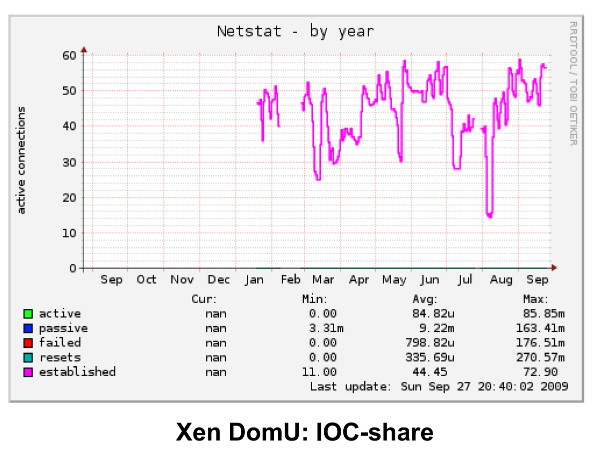
Up to now, the DELTA control system operates 32-bit programs exclusively, all running on Linux based OSes. A step by step migration to 64-bit applications is strongly aspired. Therefore, VT at DELTA has to support both bit architectures in the guest and in host OS, respectively. Since the newly installed servers are equipped with recent VT-compatible CPU-types and much memory, it is reasonable that VT exploits CPUs with a x86-32/64 architecture (Intel-VT/AMD-V) including SMP (symmetric multiprocessing) support. Furthermore, a good guest to host speed ration especially for network traffic and block device I/O throughput as well as high log-term stability is also a matter of concern. In addition, because of economical reasons, we aim for an well maintained open source solution. This wish list lead us to the setup of two Xen- and one KVM-based servers. All servers (KVM and Xen) are running a stable release of the Debian Linux distribution ('Lenny') (http://www.debian.org/) based on kernel version 2.6.26-2 in combination with Xen version 3.2.1. We implemented one 32-bit Dom0-server hosting three 32-bit VMs (DomUs: web-, email-, svn-server) and one 64-bit Dom0-server hosting four 32-bit Vms (DomUs: soft-ioc, service-master, ioc-share, vpn-server). The KVM-based 64-bit host serves, among other system services like NTP and NFS, also as a server for the MySQL database and MatLab. Additionally, it is the boot server for all Linux client PCs and the VME-IOCs. Several virtual guest systems (Windows, Knoppix, Ubuntu) are stored as iso-image files on the host and can be launched via prepared start up scripts. In that way the KVM-based server was also an ideal test bed for our novel client boot concept.

First Experience

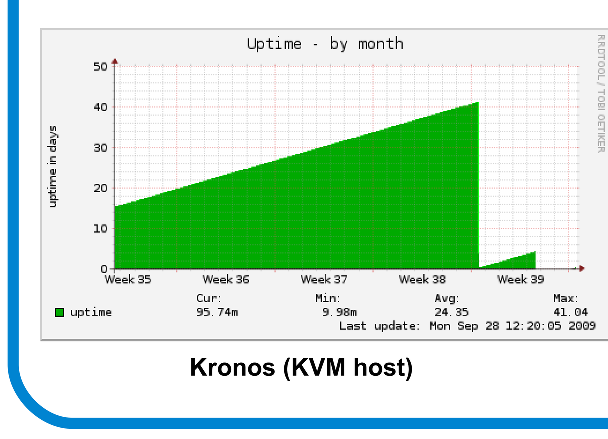
The stability of the VT-system depends strongly on a properly patched Linux kernel version for the Dom0 and DomUs, respectively, in combination with a well adopted Xen version (http://wiki.xensource.com/xenwiki/XenDom0Kernels/). Since the introduction of VT at DELTA was done in a very pragmatic manner, intensive performance and stability test were not carried out, yet. Instead, basic 'ping flood test' (test for lost packets) and long term monitoring with tools like Munin (http://munin.projects.linpro.no/) showed no substantial limitation in I/O performance. We observed nearly native network transfer rates in the DomUs compared to the Dom0. No crashes in the virtual guest and hosts OSes have been observed over many weeks of continuous operation. The stable Debian release 5.0 ('Lenny') in cooperation with Xen 3.2.1 seemed to be a good choice. Nevertheless, kernel forward patches allow an upgrade to Xen 3.4.1 which offers some interesting new features (e.g. improved support for USB-devices and power management). Concerning our KVM-based server which is in the test phase, we have to gain further experience with this rather new technology.



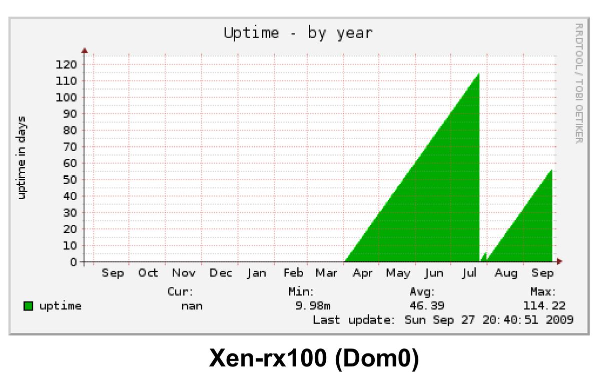
Xen-rx100 (Dom0)



Xen DomU: IOC-share



Kronos (KVM host)



Xen-rx100 (Dom0)