A STUDY OF INTRODUCTION OF THE VIRTUALIZATION TECHNOLOGY INTO OPERATOR CONSOLES

T. Ohata, M. Ishii
SPring-8, Hyogo, Japan

ABSTRACT

We introduce a virtualization technology into the control system as operator consoles. The virtualization technology is a hot topic for server computing. The virtualization technology enables to consolidate a lot of server computers to a few host computers and reduces system maintenance cost drastically. Recently a dozen of commercial and non-commercial virtualization products are proposed. We evaluated several virtualization products on the point of view on performance. The response time of the network communication between distributed computers on MADOCA control framework was measured. We selected Xen as a virtualization environment, since it has enough performance and a function of high availability. We discuss requirements of future computer for virtualization environments and showed a practical side of introducing the virtualization technology into beamline control system as operator consoles.

INTRODUCTION

Virtualization technologies have gotten a lot of attention recently to simplify management of computers. A proliferation of computers leads to an increasing of unwanted maintenance tasks. Virtualization technologies can consolidate many existing computers onto fewer servers. By introducing a server virtualization we can reduce computers and its maintenance costs.

In SPring-8, about 50 beamlines are operating. Each beamline has at least a front-end VMEbus system and workstation as an operator console. The workstation is set on each beamline that is located around the storage ring with 1.5 kilometers circumference. The maintenance on distributed computers is not easy. In the case of a certain computer failure, the downtime of the control system on a beamline might need about one hour. And we must keep complete independency of computing environment between beamlines. The server virtualization is better way to solve them.

Virtualization technologies are not novel technology. It is originated from the mainframe [1, 2], which was able to run many virtual machines (VMs). Each VM has an independent computer resource separated real hardware and can run an individual operating system instance. In this decade, the performance of modern computers advanced rapidly and they have become inexpensive. The distributed computing of large amount of computers became a main technique by speeding up the network. However, the frequency of the trouble increases in proportion to the number of the computer although the reliability of the recent computer became high. An explosive increase in the number of computers pushed up the total cost including the management cost in recent years. The virtualization technology has revived as a method of breaking such a situation. We can now obtain virtualization environment on a general-purpose server computer. This gives lowering frequency of machine failures and reduction of total cost. In this paper, we described the introduction of the virtualization technology as operator consoles, and showed possibilities of a real no-stop control system.

VIRTUALIZATION TECHNOLOGIES

There is various approach to implement the virtualization from a proprietary product to open sources. Figure 1 shows some typical structures of virtualization.

One is based on hardware emulation as shown in figure 1-(a). CPU, Disks and Network interface card (NIC) are emulated by the software layer. VMware and VirtualPC [6] are corresponding to this category. QEMU [7], Bochs [8] and Plex86 [9] known as open source products emulated the instruction set of x86 architecture completely. The overhead is extremely large, they cannot be used practically. However, this approach has a merit that the original operating system of Windows, Solaris, Linux, etc. can run without any modifications.

As another approach an independent kernel can be implemented on the user process. This is one of the approaches of emulation. User-Mode-Linux (UML) [10] and Cooperative Linux (coLinux) [11] are included in this category. This approach is different from the previous one at the emulation level. Because this emulator focuses on specific guest operating system, corresponding kernel with special patch is required. Figure 1-(b) shows the conceptual structure.

Figure 1-(c) shows another implementation of the virtualization. This is based on multiplex of physical resources. To supervise granularity of computer resources a management program is implemented on firmware or on a layer of software. They are called hypervisor or virtual machine monitor(VMM). It is a technology that was brought up by the mainframe, and it is adopted into a lot of virtualization products of major UNIX vendors and Xen [12] of an open source product. A specially patched kernel is needed in most case.

Application programs of “jail” on FreeBSD and “chroot” on Linux are deeply related to a virtualization technology. As shown in figure 1-(d) operating system builds up barriers between server applications and makes other application spaces invisible. Virtuozzo [13] and Solaris container [14] are included in this category.

Our requirements for the virtualization environment are as follows. Supporting an UNIX-based operating system. An individual operating system on VM is independent rigidly and any failures on one of the operating system should not influence another. Moreover, we needed high availability (HA) and enough processing performance after many computers are integrated. And it was important that the high cost performance would be realized.

We narrowed down the candidate of virtualization environment to VMware, UML, Solaris container and Xen. These products can be easily obtained in the market and from open source projects. Although the products of IBM and HP can achieve what we want, they are too expensive. The support from the manufacturer can be expected of VMware and Solaris container. Excluding VMware these are free except hardware and the maintenance costs.

VMware 4.5 Workstation

VMware has some lineups of product. We tested only the basic model, VMware Workstation 4.5, on the Linux operating system. Stability of VMware has increased on VMware very much compared with earlier version. And the management function is improved. Debian Linux 3.1 (kernel-2.6.8) was used for both host and VM environment for the test.

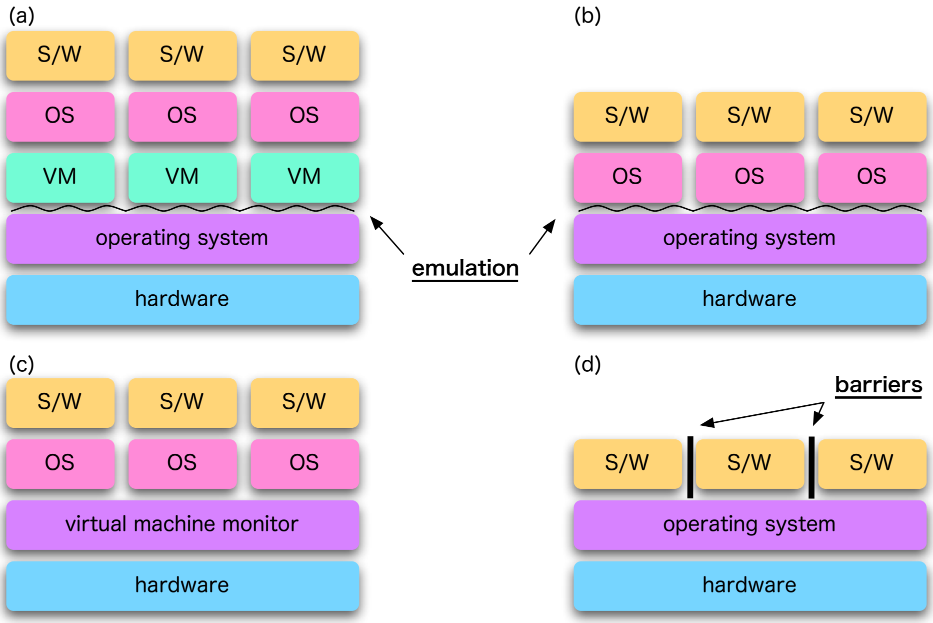


Figure 1: Structures of typical virtualization technologies.

User Mode Linux (UML)

UML is a somewhat older project of virtualization now. This has an advantage in which a virtual environment can be easily constructed. As shown in the name, UML supports only Linux. And basically it runs on x86 architecture. For the test kernel-2.4.26-3um was used on VMs. And Debian Linux 3.1 (kernel-2.6.8) was used as a host operating system.

Solaris container

Solaris container is a standard function of Solaris 10. Both sparc and x86 architectures are supported. It is an extension from "zone" on the previous version of Solaris. Solaris is acknowledged as robust and trust operating system. A new scheduler, Fair Share Scheduler (FSS), has been introduced to control running state of virtual operating system (zones). The FSS will prevent from locking of a CPU by a certain zone. The best advantage of Solaris container is that the patch management is easy. The enforcement of the patch to host operating system effects all zones. Moreover, zone can be pinned to specific CPU by standard function of Solaris. This function is effective under the symmetric multiple processor (SMP) environment. Standard kernels of Linux are not supporting a function of pinning to CPU yet.

Xen-2.06

As mentioned above, Xen is based on the resource management technology. A performance of nearly original of hardware can be obtained because there is no overhead by emulation. The resource management layer implemented by software is called virtual machine monitor (VMM). We need some correction to kernel of an operating system to run on the VMM. Xen has a bid advantage at the point of HA. Virtual operating system running on the host computer can migrate to other host dynamically. Other products do not support migration function. Xen has FSS and function of pinning to CPU as same as Solaris container.

PERFORMANCE MEASUREMENT

We examined whether virtual technology has enough performance for our operator consoles. A response time of the MADOCA [15] control framework was measured as an indicator of performance. The MADOCA is used for all of the control system in SPring-8. Figure 2 shows schematic diagram of the measurement. The components such as networks and VME were constructed to fit real environment of SPring-8. As a host computer of VMs Dell Power Edge 2650 Dual Xeon 3.0GHz was used with 6 GByte main memories. Each host operating system of Solaris 10 and Linux were installed into individual internal hard drive with Ultra-160 SCSI interface. All virtual environments can take a root filesystem as a single image of file. Image files of VMs were put on external storage connected by NFS. Throughput between storage and

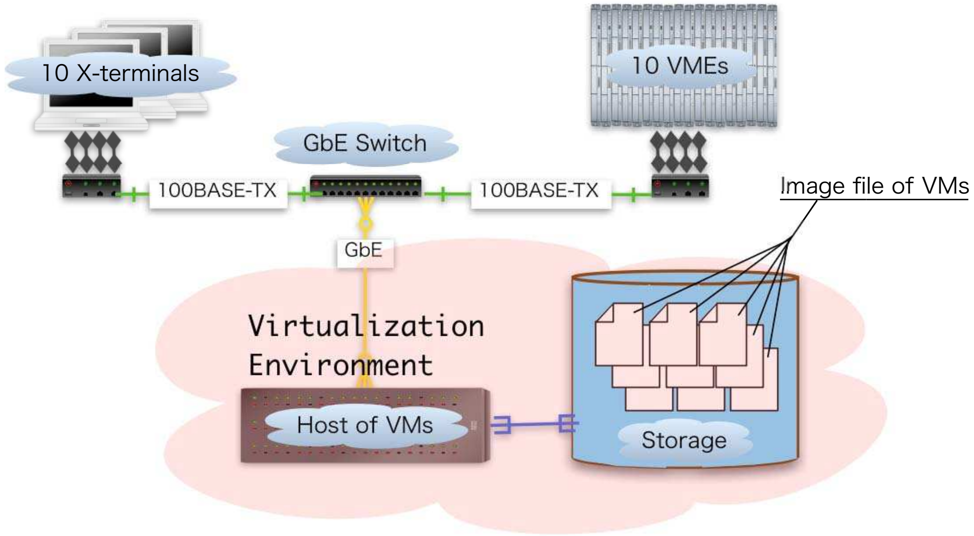


Figure 2: Test bench of the performance measurement.

host computer reaches 60~65MB/sec by using Gigabit Ethernet (GbE). A resource configuration of each VM sets to 256MB memory area and 8GB storage space. In Solaris container and Xen, VMs were equally assigned to each CPU. We tested two kernel versions to compare the influence of the scheduler in Xen. One is kernel-2.6.8, and another is kernel-2.4.30. Because this version of Xen does not support Physical Address Extension (PAE), we didn't use PAE at others.

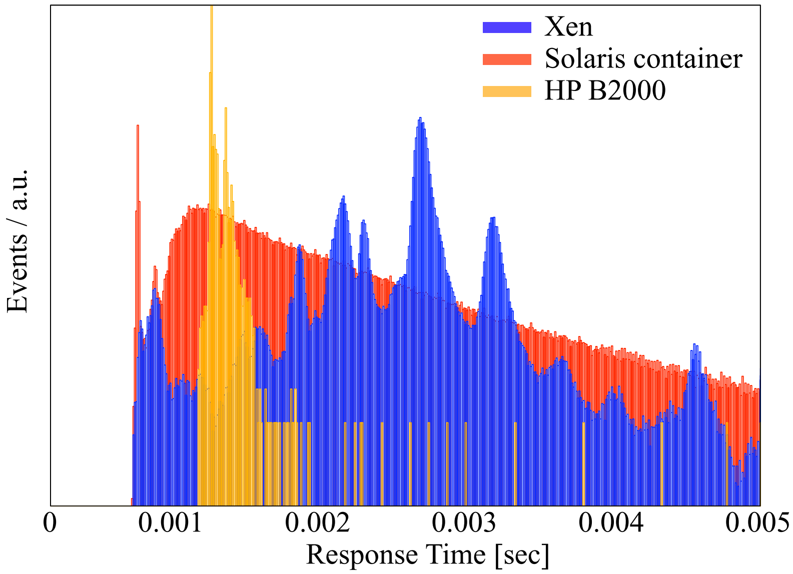


Figure 3: The result of response time measurements with 10 VMs.

We measured a response speed between MADOCA application running on VMs and corresponding VMEs. MADOCA uses ONC-RPC network communication protocol with fixed data size of 256 bytes. The packet size of one communication becomes about 700 bytes with overhead of both the RPC header and Ethernet frame header. X-terminals are used for operator consoles of VMs. Figure 3 shows the measurement results at 10 VMs of Solaris container and Xen virtual environment. The result in HP workstation used now is also shown for comparison. A horizontal axis is time and a vertical axis is the normalized number of events. Performances differ with virtual environments. The response time of maximum, minimum, average and standard deviation are shown in Table 1. The performance of UML and VMware is so good. In VMware under the SMP environment, we have seen the case where acquisition of system time was not correct. The performance of Solaris container and Xen are exceeding that of HP workstation in case of maximum, minimum and standard deviation.

The dependency of average response time and number of VMs was shown in figure 4. In the case of 6 VMs are running simultaneously, the average response time of Solaris container and Xen are as same as that of HP workstation. The differences were not found between the kernel versions on Xen. It is difficult to consolidate of a lot of operator consoles by using VMware and UML, because the processing speed falls rapidly with the increase in a number of VMs.

REQUIREMENTS TO FUTURE COMPUTER

From the plot of Solaris container and Xen in figure 4, a saturation point can be seen. It seems that a certain resource reached the limit on virtualization environment. CPU loads, network

Table 1: Statistics of performance measurement on each virtual environment under 10 VMs. The unit of the table is a second.

	Max.	Min.	Ave.	SD.
VMware	0.794572	-	0.005179	0.005288
UML	1.708498	0.001765	0.007655	0.014526
Solaris container	0.058907	0.000571	0.001931	0.001542
Xen	0.095544	0.000575	0.002649	0.001279
HP B2000	0.131388	0.001041	0.001256	0.001889

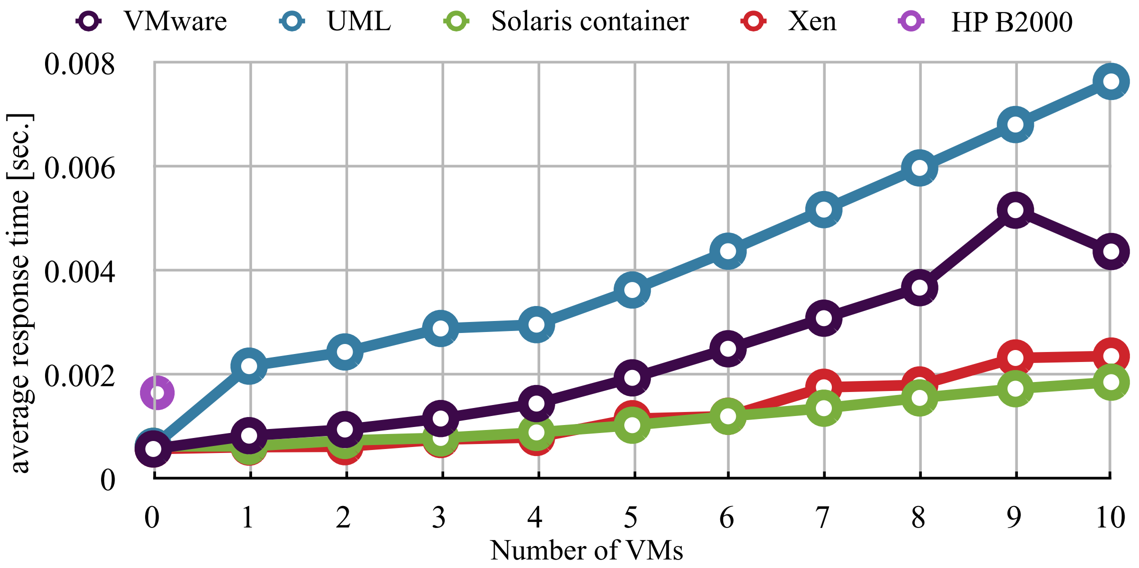
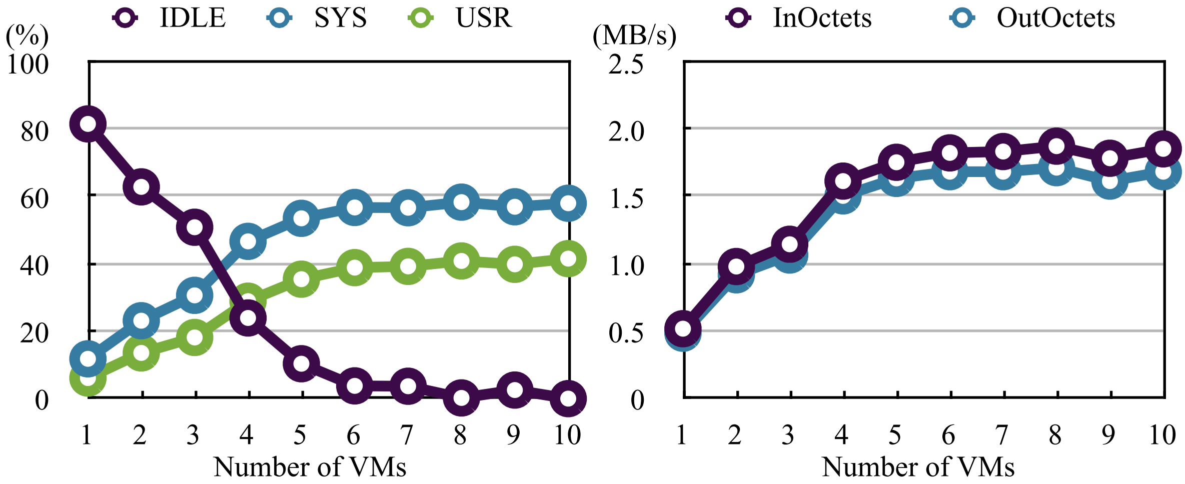


Figure 4: Comparison of VM number dependancy of the average response time on each virtualization environment.



traffic and page fault frequency of host machine during response measurement on Solaris container was shown in figure 5. From the result, the network has no problem for the performance deterioration. This is consistent with the estimation from the control message size of MADOCA. CPU load and page fault frequency saturate when 4~5 VMs are running. It seems that the page fault, which increased rapidly wastes CPU time, and it makes performance lowering. In order to solve the problem, it is necessary to increase the throughput of CPU or to remove the cause of a page fault. Because each VM is assigned 256MB memories, many VMs consume the free address space of x86 architecture. It causes a miss hit of Translation Lookaside Buffer (TLB) and swap out of a running memory. Setting larger page size on PAE or

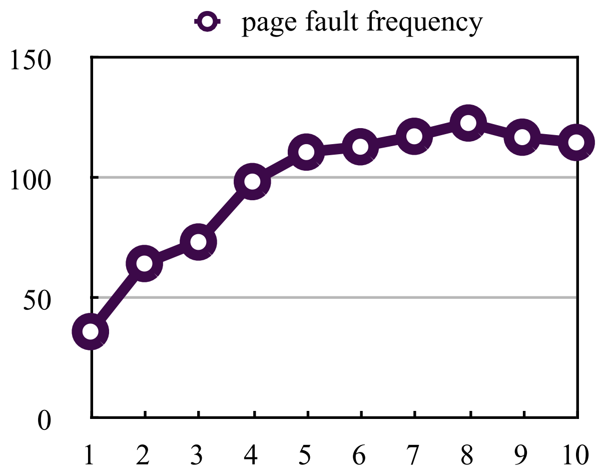


Figure 5: Stastistics of host computer of the virtualization. a) shows CPU loads, b) shows network traffic and c) shows page fault during performance measurement.

on 64-bit architecture (x86-64) is considered to be the good solution. These architectures can use a vast address space of 64GB. The Processing performance of each CPU core is enough to manage a few VMs. Even if something poor CPU, a computer with large number of CPU core is attractive.

INTRODUCTION INTO CONTROL SYSTEM

Finally, we installed the virtualization environment into one of a SPring-8 beamline. Xen was selected, since it has live migration function with HA. From a performance standpoint, Solaris container is better than Xen. However Solaris container cannot move VMs to other hosts. It means the redundant configuration is difficult. A very expensive no stop server computer of the enterprises class is needed for HA for Solaris.

As an experimental introduction, a pair of Dell Power Edge 750 is used for Xen host computer with redundant configuration. And Dell Power Edge 2650 is used to share image files of VM as a NFS server. These are connected each other by GbE network. We can move VMs to the other without any stops of VMs if primary computer will be down. We can maintain a host computer at any time if it is need. The switchover time of the live migration was about a few hundred milli-seconds. The user was able to continue operation during switchover.

Present configuration of virtualization environment has some single failure points. First, the unexpected death of host computer will stop all VMs because the function of migration of Xen can only behave manually. We are studying a cluster configuration of Xen. The single system image (SSI) cluster such as OpenSSI will be ideal solution to improve redundancy. Secondly, a redundant storage is most important. If the storage system breaks down, all VMs will be stopped. In near future, we are considering installing any of the following storage system; a storage area network (SAN), an iSCSI or a network-attached storage (NAS).

CONCLUSION

We studied several virtualization technologies to introduce as operator consoles. All the tested virtualization environments kept operating stability during the test more than half a year. We selected a open source virtualization technology, Xen, from it’s high availability function. The installation of the Xen was done at the summer stop period of SPring-8 as an operator console for the beamline control. And it operates stably until today. The virtualization environment on the recent server computer has enough performance and stability. We have a plan to add more beamlines as new VMs. And we will upgrade to a redundant computer system with no single point of failures.

In a cost side, about 50 HP workstations are installed now. When these are replaced by a virtualization environment, about 75 percent of total cost can be saved.

REFERENCES

[1] G. M. Amdahl, G. A. Blaauw and F. P. Brooks, Jr., IBM J. Res. Develop. Vol. 8 No. 2 (1964).
[2] http://en.wikipedia.org/wiki/Hypervisor
[3] http://www-03.ibm.com/servers/eserver/iseries/lpar/
[4] http://h71028.www7.hp.com/enterprise/cache/257389-0-0-0-121.aspx
[5] http://www.vmware.com/
[6] http://www.microsoft.com/windows/virtualpc/default.mspx
[7] http://www.qemu.org/
[8] http://bochs.sourceforge.net/
[9] http://plex86.sourceforge.net/
[10] http://user-mode-linux.sourceforge.net/index.html
[11] http://www.colinux.org/
[12] http://www.xensource.com/
[13] http://www.swsoft.com/en/products/virtuozzo/
[14] http://www.sun.com/software/whitepapers/solaris10/grid_containers.pdf
[15] R.Tanaka S. Fujiwara, T. Fukui, T. Masuda, A. Taketani, A. Yamashita, T. Wada and W. Xu, “Control System of the SPring-8 Storage Ring”, Proc of ICALEPCS’95, Chicago, USA, (1995) p.201