VIRTUAL MACHINE TECHNOLOGY IN GRID COMPUTING

Marko Dimitrijević, Vančo Litovski
Faculty of Electronic Engineering Niš

Abstract: *This paper presents the concept of virtual machine utilization in grid computing. Virtual machines have several advantages that qualify them as promising technology in computing grids. Basically, virtual machines provide possibility to run multiple independent operating systems on same physical machine, bringing better security, resource control and efficient hardware utilization.*
Regarding number of different computer nodes running in single grid site, dynamical deployment of virtual machines is also significant improvement.

Keywords: *virtualization, virtual machines, grid computing.*

1. INTRODUCTION

Virtualization refers to the abstraction of computer resources. There are many virtualization concepts and techniques; in this paper we will refer to *platform virtualization*, particularly *full virtualization* which separates an operating system from the underlying platform resources, i.e. hardware platform.

Full virtualization is technique that implements a *virtual machine* environment that provides a complete simulation of the underlying hardware resources: processor cores, operating memory, storage capacity, network connectivity, etc. It can even provide simulation of hardware that physically does not exist (e.g. the disk image can be mounted as physically present hard disk or optical drive). The result is a system in which all software capable of execution on the raw hardware can be run in the virtual machine. In the context of full virtualization, it particularly refers to operating system.

Virtualization is performed on a given hardware platform by *host software* (or *host operating system*), which creates a simulated computer environment, a virtual machine, for its *guest operating system*. There are no special requirements for guest operating system, and it runs as it was installed on a raw hardware platform. The number of guest operating systems running on the same host is limited only by the host's hardware capabilities.

The host software is often denoted as virtual machine monitor, or *hypervisor*. Hypervisors are classified in two types: type 1 (*native, bare metal*) and type 2 (*hosted*) hypervisors.

A type 1 hypervisor is software that runs directly on a given hardware platform (as an operating system control program). A guest operating system thus runs at the upper, second level above the hardware (Figure 1). The classic bare metal hypervisor was CP/CMS, developed at IBM in the 1960s, ancestor of IBM's current z/VM. Actual bare metal hypervisors are Oracle VM, VMware ESX Server, LynuxWorks LynxSecure, L4 microkernels, Green Hills Software INTEGRITY Padded Cell, VirtualLogix VLX, TRANGO, IBM POWER Hypervisor (PR/SM), Microsoft Hyper-V, Xen, Citrix XenServer, Parallels Server, ScaleMP vSMP Foundation and Sun Logical Domains Hypervisor. A variation of this is embedding the hypervisor in the firmware of the platform, as in the case of Hitachi's Virtage hypervisor. KVM software, which transforms a complete Linux kernel into a hypervisor, is also bare metal hypervisor.

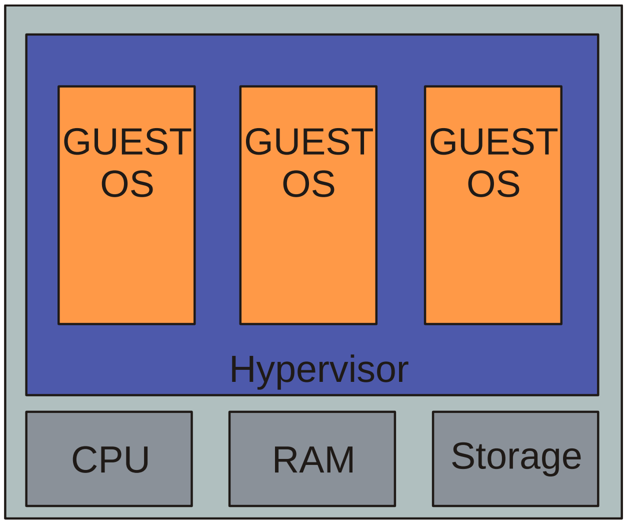

Fig. 1 Type 1 hypervisor

A type 2 hypervisor is software that runs within a classic operating system environment. A guest operating system thus runs at the third level above the hardware (Figure 2).

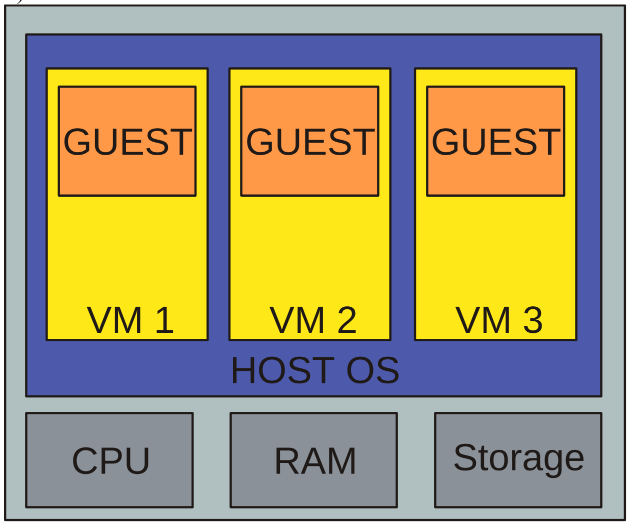

Fig. 2 Type 2 hypervisor

Examples of hosted hypervisor are VMware Server (formerly known as GSX), VMware Workstation, VMware Fusion, QEMU, Microsoft Virtual PC and Microsoft Virtual Server products, Sun (formerly InnoTek) VirtualBox, Parallels Workstation and Parallels Desktop for Apple Mac. In this case, hypervisor is dependent from host operating system.

2. COMPUTING GRID

The development of Internet and WAN links of great capacity led to a computational grid. The intention was to associate to the electrical power grid. In the same way electrical power could be obtained from power grid, computational power should be obtained *on demand* from a network of *providers*, potentially belonging to the entire Internet [1]. In the beginning, this notion has been strictly scientific and academic; but as the Internet, it became widely accepted and popular. One of the most common definitions says that a computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities providing on-demand access to computing, data, and services [2]. Basically, grid computing intends to provide access to resources using wide area connections; it can be determined as cooperation of geographically distributed computer systems (clusters) where user jobs can be executed (Figure 3).

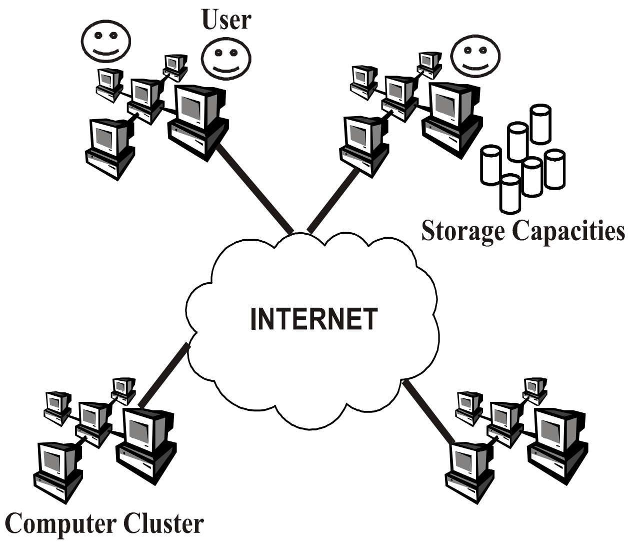

Fig. 3 A Grid structure

Grid computing is suitable for intensive calculations that require significant processing power, large operating memory and throughput, as well as storage capacity. The simulations of integrated electronic circuits are paradigmatic example of these calculations [3, 4].

The role of the software component is to provide distributed services for job submission and management, file transfer, database access, data management and monitoring (Figure 4). They also ensure security in multi-user environment using certificates. The software part consists of two layers: operating system and middleware.

The hardware part of computational grid infrastructure can be extremely heterogeneous. It consists of a number of different computer clusters containing various numbers of nodes consisting of various types of processors, amounts of memory, LAN and WAN connectivity and mass-storage capacity.

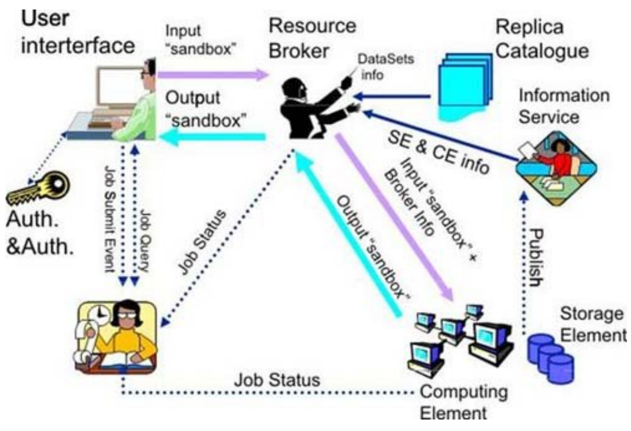

Fig. 4 Grid services

A single cluster (Figure 5) has many advantages over classic supercomputer: it is inexpensive, flexible, easy to use, easy for maintenance and highly stackable. The total price of computer cluster is more than ten times lower then dedicated supercomputer with similar computing power, and after amortization cluster nodes can be used as single personal computers. A cluster is also highly stackable: it can be easily extended by adding additional node or demoted by subtracting one.

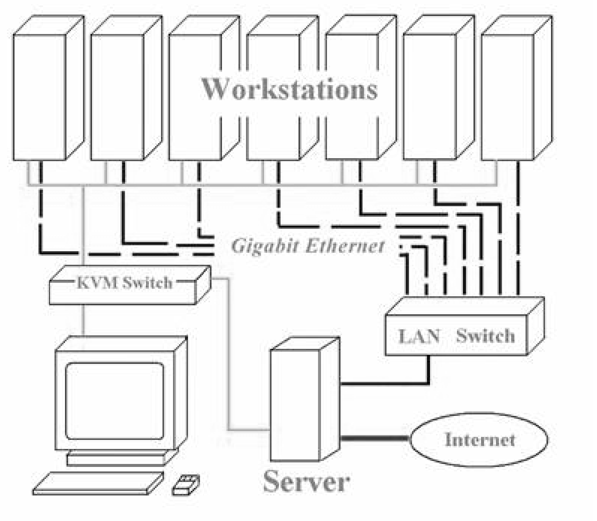

Fig. 5 Organization of the computer cluster

Each cluster, or in grid terminology site, employs some mandatory services: computing element (CE), storage element (SE), site information service (site BDII), monitoring box (MON) and user interface (UI). It also employs a number of worker nodes (WNs). Traditionally, these services are implemented at different nodes, or simply installed in some combination on the same machine (e.g. CE and BDII on the first node, SE on the second, MON on third, etc.)

We can define *service granularity*, which presents number of services per operating system. In the first case previously described, service granularity is one service per operating system.

3. VIRTUALIZATION ADVANTAGES

Virtual machines allow dynamic sharing of resources and multiplexing of services onto hardware at the granularity of a single service per operating system session, supporting unique service virtual machine configuration and isolation from other services sharing the same physical resources, unlike classic operating systems where many services run at the same operating system, sharing the same hardware resources [5].

3.1. Compatibility

Virtual machines support compatibility at the level of binary code, thus there is no need for recompilation or dynamic re-linking to port applications to a virtual machine. Furthermore, the legacy support provided by classic virtual machines is not restricted to applications: entire legacy environments – hardware support, the operating system, application interfaces and applications are supported.

3.2. Customization and resource control

Virtual machines can be highly customized without requiring system reboot. Some of the resources used by a virtual machine can be customized dynamically at instantiation time, such as memory and disk sizes, number of processor cores as well as system software parameters – operating system version and kernel configuration. Furthermore, multiple independent operating systems can coexist in the same hardware. In a grid environment it becomes possible to provide virtual machines that satisfy individual service requirements.

It is also possible to implement mechanisms to limit the resources utilized by a virtual machine at run time by implementing scheduling policies at the level of the hypervisor. Opposite to typical multitasking operating systems, where resource control mechanisms are applied on a per-process basis, classic virtual machines allow fine-grained resource control and allocation. Furthermore, resource control policies can be established dynamically. Dynamic resource control is important in a grid environment for two reasons: it allows a provider to limit the impact that a remote user may have on resources available for a local user and it enables a provider to account for the usage of a resource. Resource control mechanisms based on virtual machines are particularly important in computing grid since they can be applied to legacy applications, unlike standard grid accounting solutions like APEL or based on Java.

3.3. Security, service and user workspace isolation

Regarding the intention to provide access to resources using wide area connections (WAN) to different users belonging to different administrative domains, potentially whole Internet, grid security is an important issue [6, 7].

Grid security is based on *Grid Security Infrastructure* (GSI). This is a security model where resource providers – grid sites – trust the integrity of user applications. It restricts the usage of grid to cases where mutual trust can be established between providers and users. This trust is established using enhanced *public key infrastructure* (PKI); all users and services must have valid certificate, signed by certification authority, which identify them in communication. Resource mapping and user access control is implemented using *virtual organizations* (VO). Each computing grid user belongs to virtual organization.

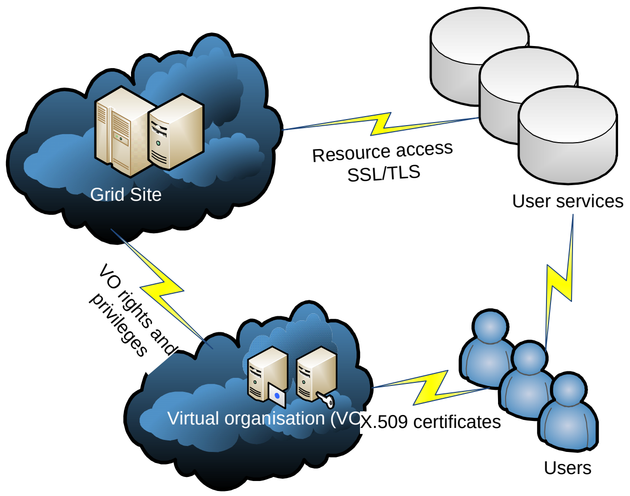

Fig. 6 GSI security model

If users are to submit jobs to computational grid without trust relationship, the integrity of a computation may be compromised by a malicious resource and the integrity of the resource may be compromised by a malicious user. To address these concerns, a computing grid system must enable the execution of unverified applications within a secure, isolated environment, called *sandbox* [8]. Essentially, the impact of the running application must be restricted to the sandbox, thereby protecting the underlying host machine. Depending on the application's requirements, access to resources on the host machine can be selectively allowed from within the sandbox. Moreover, while security attacks and faults can occur within the sandbox, its framework must guarantee that these vulnerabilities do not affect the underlying execute machine.

Classic virtual machines achieve stronger software security than a conventional multitasking operating system approach if redundant and independent mechanisms are implemented across the hypervisor and the host operating system. In case where grid users have access to classic virtual machines, it is more difficult for a malicious users to compromise the resource than it is a case in conventional multitasking operating system, because they must be able to break two levels of security, the hypervisor and the operating system. Furthermore, for malicious user, it will be very difficult to realize that the hypervisor level even exists. Virtual machine security services can work even if an attacker gains complete control over the guest operating system.

Implementing virtual machines, complete service and user workspace isolation is achieved. There is no interference between services and/or users, and eventual security incident affects only one service or user. In this case, the system recovery is much easier, using native virtual machine's ability to make system snapshots and boot from read-only images. Virtual machine features also provide the ability to reconstruct an attack in arbitrary detail.

3.4. Privileges

In classic multitasking and multiuser operating systems (such as Linux/UNIX systems), critical system operations are reserved to a privileged user – the system administrator. These operations are restricted to a trusted user because they can compromise the integrity of the operating system, resources and other users. In grid environment, special privileges such application installation and management are supported using *roles*.

In many cases the need to secure operating system leads to restrictive approach in determining which operations are critical and consequently privileged. The result is limitation in usage of the operating system. For example, the mount command is typically privileged, thus not accessible by ordinary non-privileged users. This prevents malicious users from gaining unauthorized access to local resources, but also disallows legitimate-use cases – a user who wishes to access remote data from network shared partition. When classic virtual machines are implemented with granularity of one – each user or service has a dedicated virtual machine, these requirements can be less restrictive. There are no users/services sharing the same virtual machine and the integrity of the resource underlying the host operating system – virtual machine – is independent from the integrity of the shared physical machine. If necessary it is then possible to grant system administrator privileges to unverified grid applications because the actions of malicious users are confined to their virtual machines.

3.6. Site independence

An application running on a virtual machine is decoupled from the system software of the underlying host machine. This ensures that an unverified user or application can only compromise the virtual machine and not the underlying physical resource. Virtual machines also enable fine-grained resource allocation for grid jobs. It is hence feasible to restrict the memory, network, disk size, and even the processor time allocated to a given virtual machine. Furthermore, the use of virtual machines allows the target execution environment for a grid application to be completely customized, thereby enabling support for jobs with special requirements like root access or legacy dependencies. VMs also enable process migration without requiring any modification of the grid application. A virtual machine guest presents a consistent runtime software environment – regardless of the software configuration of the virtual machine host. This capability is very important in a grid environment: combined with the strong security and isolation properties of virtual machines, it enables cross-domain scheduling of entire computation environments (including OS, processes, and memory/disk contents of a virtual machine guest) in a manner that is decoupled from site-specific administration policies implemented in the virtual machine hosts. A virtual machine can be instantiated on any resources that are sufficiently powerful to support it because it is not tied to particular physical resources. Furthermore, a running virtual machine can be suspended and resumed, providing a mechanism to migrate a running machine from resource to resource.

4. CONCLUSION

Virtual machines prove to be a promising platform for sandboxing in computing grids, providing resource control at coarser granularity, legacy application accounting, fault isolation, customized execution environments and support for process migration.

In this paper, we have presented some arguments for the usage of virtual machines in grid computing regarding security, flexibility, resource control and compatibility.

5. REFERENCES

[1] I. Foster, C. Kesselmann, S. Tuecke, “The Anatomy of the Grid, Enabling Scalable Virtual Organizations”, International J. Supercomputer Applications, 2001

[2] I. Foster, C. Kesselmann, “The Grid: Blueprint for a New Computing Infrastructure”, 1998

[3] M. Dimitrijević, B. Anđelković, M. Savić, V. Litovski, “Gridification and Parallelization Of Electronic Circuit Simulator“, INDEL 2006 Conference, Banja Luka, Bosnia and Herzegovina 2006, pp. 95-100

[4] B. Anđelković, M. Dimitrijević, V. Litovski, “Using Grid Computing In Parallel Electronic Circuit Simulation”, Proc. Of 16th Electronics 2007 Conference, Sozopol, Bulgaria, 2007, Book 4, pp. 109-114

[5] R. J. Figueiredo, P. A. Dinda, J. A. B. Fortes, “A Case For Grid Computing On Virtual Machines”, Proceedings of the 23rd International Conference on Distributed Computing Systems, IEEE Computer Society, Washington, DC, USA, 2003, pp. 550

[6] A. Chakrabarti, A. Damodaran, S. Sengupta, “Computer Grid Security, A Taxonomy”, IEEE Security & Privacy, Vol. 6, No 1, January 2008.

[7] M. Dimitrijević, “Network Level Computing Grid Security”, in Serbian, LII Conference of ETRAN 2008, Palić, Serbia, 2008, pp. XX1.1-1-4

[8] S. Santhanam, P. Elango, A. Arpaci-Dusseau, M. Livny, “Deploying Virtual Machines as Sandboxes for the Grid”, Proceedings of the 2nd conference on Real, Large Distributed Systems – volume 2, San Francisco, USA, 2005, pp. 7 - 12

Садржај: *У овом раду је представљен концепт употребе виртуалних машина у рачунарском гриду. Виртуелне машине имају неколико предности које их квалификују као перспективну технологију која се може применити у компјутерском гриду. Виртуелне машине омогућавају покретање више независних оперативних система на истом хардверу пружајући могућност боље безбедности, контроле ресурса и ефикаснију расподелу ресурса. Имајући у виду велики број различитих нодова у оквиру једног сајта, динамичка расподела виртуелних машина такође представља значајан напредак.*

TEHNOLOGIJA VIRTUELNIH MAŠINA I RAČUNARSKI GRID

Марко Димитријевић, Ванчо Литовски