
Chapter 7. The Value of Virtualization

Virtualization provides valuable support for running large numbers of Linux images on the mainframe. IBM mainframe virtualization technology is found in the z/VM product and in logical partitioning (LPAR). These technologies provide the infrastructure that enable, on one hand, numerous Linux images that use only small amounts of CPU, and, on the other hand, a few Linux images that consume large amounts of CPU to be hosted on a single mainframe. Such deployments might save your company money.

z/VM is the zSeries Virtual Machine operating system. This chapter explains why z/VM is ideally suited for large-scale Linux deployments and server consolidation. z/VM's ability to run hundreds of Linux images on a single machine opens a multitude of business possibilities. Because the powerful mainframe virtualization technology is little understood outside the mainframe world, this chapter should be especially useful for readers not familiar with the mainframe.

In this chapter, we discuss these questions:

- What is z/VM, and how did its virtualization technology develop?
- How does Linux run on z/VM?
- What are the possibilities and benefits of running Linux under z/VM or on an LPAR?

7.1 What is z/VM?

To better understand what is going on today, it is sometimes helpful to revisit the past. In this section, we examine what drove the creation of z/VM and why it plays such a key role in Linux on the mainframe.

Virtual machine (VM) technology development started with the IBM System/360 in the mid-1960s. One reason for developing it was the need for testing facilities. Many software programmers could thus test new programs on the virtual hardware simultaneously.

Another factor driving VM development was the need for multiuser systems. Computers were large and costly. Thus, a need arose for an operating system environment that allowed multiple users to use one machine simultaneously in real-time.

To support testing and multiple users, researchers developed the unique concept of *virtual machines*. In the virtual machine model, the hardware resources of a computer system are

managed by a *hypervisor*. VM's hypervisor is called *Control Program (CP)*. Users' activities are managed by an operating system. When users log on to VM, CP creates a unique guest virtual machine for each user. Depending on the guest definition, CP then allows the user to start up an operating system within that guest.

The first commercial VM product offering from IBM, in 1972, was the VM/370 product. Apart from CP, it included a special-purpose operating system called *CMS*, the *Conversational Monitoring System*. CMS has been a component of VM products ever since the delivery of VM/370. The purpose of the CMS operating system is to provide multiple users with a powerful yet simple interactive interface that performs extremely well. Performing well means sub-second response time for hundreds of CMS instances on a single machine. The design for simplicity extends to all aspects of CMS, including the command set and file system. The CMS that offers the VM community a rich set of programming tools and utilities can help manage and automate a virtual Linux server farm running on the current VM product, z/VM.

Guests on z/VM are zSeries operating systems in their own right. The CMS operating system today is different from the other guests in that it no longer runs native on the hardware. The use of CMS has changed from being VM's only interactive end user operating system to also being a home for utility-like functions. For example, TCP/IP or performance management functions can be run in a guest of their own and can be used by other guests.

Similarly, in a Linux-on-the-mainframe environment with many Linux guests, CMS is where scripts get run to automatically restart failing images, or manage switch-over to a hot standby. A system administrator uses CMS to create new images (or CMS is the programmatic interface where the request for the creation of a new Linux image is routed).

Over time, IBM has made investments in hardware, architecture, and microcode, as well as in the VM product itself, to enhance the virtualization technology available with each successive line of mainframe computers.

7.2 How Linux can run on z/VM

At its core, the z/VM operating system is a hypervisor. The hypervisor can present virtual copies of the underlying hardware resources that it controls to operating systems running on a virtual machine. Users can run multiple images of other operating systems as “guests” of the hypervisor, sharing the same single set of real CPUs and I/O facilities, as shown in Figure 7-1.

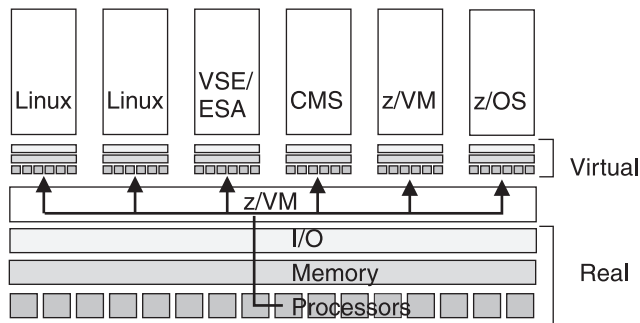


Figure 7-1. Virtualization technology with z/VM. Resources can be virtualized and presented to the user as if each user had his or her own machine.

Any operating system that can run on an S/390 or z/Architecture mainframe can also run under the control of z/VM, that is, in a z/VM-created virtual machine. z/VM faithfully represents the complete mainframe architecture to the guest operating system.

Like any other operating system that runs on the mainframe architecture, *Linux runs on z/VM without any further changes.*

7.3 What does z/VM provide?

z/VM is a mainframe operating system. Linux on the mainframe can particularly benefit from what z/VM provides:

- The hypervisor.

z/VM's CP component is the hypervisor that manages resources for its guests. The hypervisor includes all the emulation code and an extremely efficient scheduler. The fact that hundreds of users can work concurrently and expect sub-second response times shows the efficiency of the scheduler.

- The CMS operating system.

CMS has a simple user interface and is used by system administrators to manage both z/VM itself and its guests. It is also used as a utility operating system for various server functions such as TCP/IP.

- A large set of systems management tools.

In the z/VM directory, system administrators can quickly and easily change the definitions of a guest or create a new guest. z/VM has both a user interface and a command line interface that can be driven by scripts to control the entire environment. z/VM's system administration facility includes functions that help the user create and

manage multiple Linux images. Functions include assigning disks, starting and stopping Linux images, and more. System and hardware resources can be allocated among multiple Linux guests, which can also be managed by z/VM.

A scripting language (REXX) allows for programmatic response to events; for example, you can use the programmable operator (PROP) to filter messages from the z/VM operator console. You can specify a REXX program (called a *PROP exit*) to be invoked whenever a guest logoff message appears. The message includes the name of the guest that terminated. The exit could then take whatever action was appropriate, such as restarting the guest. There are also extensive logging capabilities for performance tracking or debugging.

z/VM has a large, active user community (much like the Open Source community) that provides tools and scripts for managing a z/VM environment.

- Security and integrity.

z/VM is built on the mainframe architecture that highly values those attributes. We examine z/VM security and integrity in Chapter 8.

To understand the difference of z/VM compared to other virtualization techniques, we need to take a look at some technical issues.

Importantly, z/VM allows its guests to take advantage of the underlying mainframe architecture. In the interest of achieving good performance, z/VM guests run as much as possible as if the hardware were real. Only when z/VM must run privileged instructions for the guest does it take control from the guest.

In 1985, IBM announced and delivered a microcode assist for its 3081, 3083, and 3084 processor models called *Start Interpretative Execution (SIE)*. The SIE assist reduced the number of instructions VM had to execute on behalf of a guest operating system. With SIE, the processor (in interpretative execution mode) could handle more guest I/O instructions and associated interrupts. Less VM intervention was required, thus significantly improving guest performance.

The distinguishing factor of z/VM is that all resources can be virtualized. CPU, memory, disks, and network communications can be virtual (see Chapter 10). Additionally, certain hardware that is supported on some mainframe models can sometimes be virtualized on other models. For example, it is possible to use virtual HiperSockets on machine models that do not support real HiperSockets (such as G5, G6, and Multiprise 3000 servers).

In summary, the guest operating system believes that it owns and controls the hardware. CP allows the guest control until an instruction occurs that CP must interpret. From the outside observer's perspective, there is a "duet" of the CP and the guest.

7.4 What is logical partitioning?

Once VM was generally available in 1972, some companies used it to partition machines in order to run different workloads in different partitions. It turned out that for some companies, a simpler partitioning scheme would be just as useful. As a result, some fundamental virtualization concepts were included into the architecture. These concepts became the hardware hypervisor *Processor Resource/Systems Manager (PR/SM)* and the *logical partitions (LPARs)* into which PR/SM could divide a machine. A zSeries 900 or 800 mainframe can run in one of two modes: basic mode or LPAR mode. In basic mode, the entire machine is under the control of a single operating system. In LPAR mode, you can logically divide the machine into partitions so that multiple operating systems can run concurrently.

LPAR partitioning helps companies to exploit the resources of the machine better than if there were no partitioning. LPAR allows companies to have a single machine support both production and test requirements. Initially, companies would bring up the test partition only at off-peak times.

The PR/SM hypervisor of the zSeries 900 can manage up to 15 LPAR “guests.” The CPU and channel paths are the two resources that can be logically shared. Central storage must be physically partitioned and assigned to each active partition.

Although the virtualization technology that is best known in the context of Linux on the mainframe is z/VM, there are some occasions where you might consider using an LPAR:

- You might want to take advantage of the cost savings available with Integrated Facility for Linux (IFL). The IFL feature CPUs can be assigned only within an LPAR that is dedicated to Linux, or Linux running on z/VM (see 21.4, “Integrated Facility for Linux (IFL)”).
- You might want to have just a few Linux images. Then you might not require the ability of z/VM to handle hundreds of images.
- You might require the certified security offered by LPAR. Typically, you might want to run a few Linux images each in their own LPAR (Figure 7-2). An independent authority has certified LPAR to confirm that LPARs are isolated from one another (see 23.1.1, “LPAR certification”). Depending on your corporate audit policy, a firewall might be an instance where you want an individual Linux in its own LPAR.

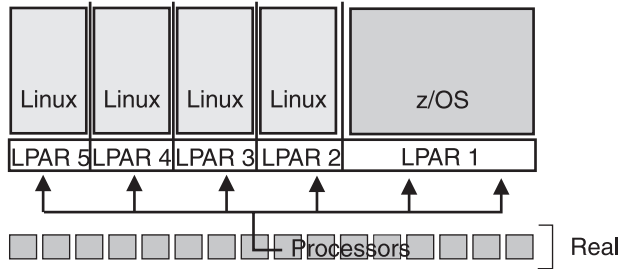


Figure 7-2. Two LPARs, one running z/OS, the other running Linux

Companies can utilize their mainframe with more flexibility using LPARs than using native mode. However, LPAR provides far less flexibility and function than what is available with z/VM.

7.5 Why run Linux on z/VM?

The interest in Linux on the mainframe can be attributed to three things:

- As a source of new technology for mainframe users
- As an opportunity to reduce complexity when deploying an integrated server environment
- As a possibility to host large server farms

z/VM is ideally suited to hosting an integrated Linux server environment, letting companies run applications across multiple images on a single hardware platform. Information technologists can exploit the benefits of having numerous images, each running only one application, and yet limit the costs associated with having unique hardware for each image, as shown in Figure 7-3.

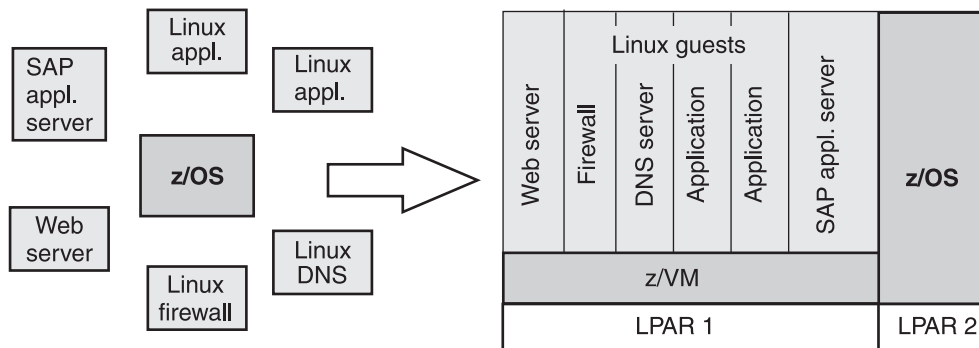


Figure 7-3. Application images can be brought closer to the back-end z/OS (integrated environment)

Many years of product development have made z/VM ideally suited today for Linux deployments. Through the years of product development, z/VM has been designed to support:

- Production systems. Typically, these are single-image operating systems or applications consuming large amounts of CPU, for example, VSE/ESA, or SAP R3 application servers.
- Interactive users. z/VM can support hundreds of CMS users.
- Test environments. These environments can be used for migration, or to test different operating system levels before deploying one.

7.5.1 Why z/VM is ideally suited for hosting large application Linux images

Even today, companies running large z/OS or VSE/ESA systems as guests of z/VM typically have only a few images (maybe just two: one for production and one for testing). The demand on z/VM is to maximize the amount of work that the z/OS or VSE/ESA guests can perform. In the past, this required VM developers to be very efficient in their support of new technologies such as large real memory, advanced disk storage systems, and processor facilities. The goal for supporting guest systems is to give each guest what it needs and get out of the way.

A characteristic of the z/VM operating system that benefits a Linux server environment is the high degree of resource sharing that is achieved with VM's CP. z/VM facilitates sharing processor capacity, main memory, disk devices, channel adapters, network adapters, and practically anything you can attach, connect, or plug into your mainframe among the virtual machines. Note that z/VM allows defining of guests dynamically; no static partitions need to be created ahead of time.

At the same time, if a guest needs sole access to a processor, device, or network resource, that resource can be dedicated to the guest. A hallmark of z/VM is its flexibility in supporting a wide variety of computing requirements. System administrators can mix and match shared resources with dedicated resources on the same z/VM system, and even in the same guest, changing the landscape of the virtual machine configurations, if needed, in a matter of minutes, not hours or days.

7.5.2 Why z/VM is ideally suited to Linux server consolidation

CMS workloads in the past were typically multi-image, interactive workloads. Many companies were deploying hundreds of CMS guests on their VM systems and were asking IBM for enhancements to support that workload. Today, Linux workloads tend to be spread across multiple images and be highly interactive, such as Web-serving workloads. Because of the similarities between Linux workloads and traditional VM workloads, the provisions made to accommodate these workloads also work well for Linux.

The refinements made to VM's scheduler and dispatcher over the course of decades are equally suited for a Linux server environment. This kind of environment requires the facilities to host a large number of images with an unpredictable set of workloads that must meet stringent response-time objectives.

The appeal of z/VM's resource sharing becomes obvious when one considers the amount of wasted resources that typically exist in a discrete server farm environment. It is not unusual for individual servers to have, on average, a utilization rate as low as 3%. Servers with such low utilization are, for example, domain name servers (DNS) where data does not have to be looked up often. A company can have many DNSs. ISPCCompany, for example, could have a DNS for each client of its outsourcing business, which adds up to hundreds over time. In general, utilization rates depend on many factors, including:

- The function provided by the application
- Its I/O requirements
- The networking bandwidth
- The number of users or clients who access the application
- The level of technology available in the server itself

Many network infrastructure servers, such as DNSs and firewalls, experience short bursts of activity followed by periods of inactivity. When a server is inactive, its resources are unavailable to other servers that could benefit from additional processor capacity, real memory, network or I/O bandwidth, or disk space. Purchasing server resources to handle application

peaks compounds this problem. Processor speed and technology, memory requirements, and all other tangible capital costs associated with a server application are generally measured by how much capacity an application needs when user demand is high. To provide any less resource means potentially falling short of the computing capacity which the application needs at critical times. Unfortunately, this only compounds the waste of system resources when the server is idle or underutilized. (See 3.3.2, “Utilize CPU power.”)

In a z/VM world, a server running in a virtual machine can grow or shrink its consumption of available system resources based on application demand. For example, processor cycles that would otherwise be wasted during times of low utilization are available for other images running on the same z/VM system. When resource requirements in a guest exceed expectations, z/VM lets a guest get more cycles or be backed by more real storage, provided no one else with higher priority needs it. Using CP commands, a system administrator can allocate more processor capacity, disk storage, or network bandwidth to the virtual machine when it needs it, not days later. This is possible provided that the virtual resources are available. (It usually takes days to provision a real server to replace the one that no longer meets the customer's need for computing power.)

Figure 7-4 shows many servers with low utilization being consolidated on a mainframe under z/VM.

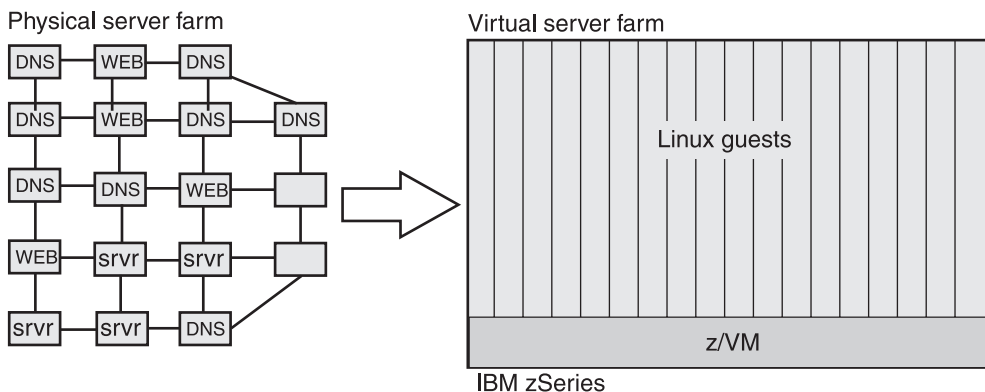


Figure 7-4. Many servers with low utilization can be consolidated on z/VM

There are other costs associated with separate, seldom used servers. When a server goes idle, it still requires energy, floor space, and other environmental costs such as cooling to maintain its connection to the network and be ready for its next transaction. A guest on z/VM does not represent incremental environmental costs, whether active or idle. The environmental cost of a z/VM system is fixed for a given configuration of hardware and wastes resources only if the server farm in aggregate becomes idle or has low utilization. Studies

have shown that environmental cost savings can be realized when running a multi-image server environment on z/VM versus a discrete system solution.

7.6 Summary

On a zSeries machine, using LPAR and z/VM for Linux deployments presents a multitude of possibilities not available anywhere else. There can be substantial hardware and floor space savings from using virtualization technology on a zSeries machine. Virtualization technology gives you a flexible server farm environment that can be easily managed and changed in minutes with a few keystrokes.

The purpose of z/VM's efficient virtualization technology is to:

- Allow operating systems to run almost as if they were running on the hardware.
- Maximize the logical sharing of the physical resources such as CPU and central storage.

Much of the appeal of running Linux on the mainframe comes from the ability to run tens to hundreds of Linux images on a single machine.