Exploiting virtual machine technology for the development and test of z/OS system solutions

Prologue –

z/VM presents a unique approach to computer operating systems. It provides each user with an individual working environment known as a *virtual machine*. The virtual machine simulates the existence of a dedicated real machine, including processor functions, storage, and input/output (I/O) resources.

Operating systems and application programs can run in virtual machines as guests. For example, many customers run multiple Linux images on the same z/VM system that is supporting various applications and users. As a result, development, testing, and production environments in this case can share a single physical computer. Having this virtualization capability why not employ z/OS Guests as runtime environments for your development and testing needs(?).

The virtual machine capability of z/VM allows you to**:**

- Test programs that can cause abnormal termination of real machine operations and, at the same time, process Development work. The isolation that is provided for a virtual machine enables system-oriented programs and teleprocessing applications, for example, to be tested on the virtual machine while production work is in progress, because this testing cannot cause abnormal termination of the real machine.
- Test a new operating system release. A new release of an operating system can be generated and tested at the same time that the existing release is performing production work. This enables the new release to be staged for production quickly. The ability to operate multiple operating systems concurrently under z/VM may enable an installation to continue running programs that operate only under a back-level release (programs that are release-sensitive and uneconomical to convert, for example) concurrently with the most current release.
- Test a new operating system. The existing operating system can be used to process production work concurrently with the generation and testing of a new operating system. Experience with the new system can be obtained before it is used on a production basis, without dedicating the real machine to this function.

System Efficiency

z/VM's guest support helps you to run systems at lower cost with greater efficiency. For example:

Performance benefits

Guest systems may see performance improvements by exploiting z/VM features. For example, both virtual disk in storage and minidisk cache allow guests to avoid real I/Os by using data in storage and caching techniques.

Reduced hardware cost

Guest systems can share devices such as channels, printers, and DASD, which z/VM efficiently manages. Indeed, z/VM adds new value to such devices merely by the way it manages them. A good example is z/VM's minidisk support, which allows one real disk to function as if it were several smaller disks (such as multiple IPL-able minidisks).

And because z/VM simulates some devices (such as CTC adapters), your installation may be able to avoid buying real ones.

Operations management

With PROP (z/VM's PRogrammable OPerator) you can cut your console traffic substantially. That saves time and reduces errors.

Note: The programmable operator facility increases the efficiency of z/VM system operation and allows operation of systems in a distributed processing environment. The facility intercepts all messages and requests directed to the z/VM operator virtual machine and handles them according to programmed actions. Some messages are simply recorded for future reference, others are acted upon programmatically, and others are sent to a real operator to handle. The programmable operator facility runs in a CMS virtual machine. Although it can run in any virtual machine, because of its programmed capability to log, handle, or redirect messages, it is most commonly run in the CP system operator's virtual machine.

z/OS can run as a guest operating system under z/VM. Using the simulation capabilities of z/VM, it makes it easier to test new hardware and software features on z/OS by running it as a guest under z/VM. Virtual machines can be duplicated very quickly using the cloning facilities available in z/VM. So this makes test and development faster.

When z/OS systems are running as guests of z/VM, it is easy to share devices such as tape drives among multiple systems. For example, the **MULTIUSER** operand for the **DEDICATE** control statement in the z/VM USER Directory is one way to serially share a real tape drive with multiple guest systems. z/OS is often hosted under z/VM at disaster recovery (DR) sites where the virtualization of device numbering means that few, if any, changes need to be made to the z/OS guests.

Recovery management
You can build guest virtual machines to simulate systems at your organization's other sites. So, z/VM can become a disaster-recovery backup site. When you analyze the z/OS environment, remember that you have two operating systems running in a single processor. Both z/VM and z/OS are vying for the basic system resources, such as processor, I/O, storage, and paging. Each is generating its own accounting information, and each is supplying its own performance information.

Remember that z/OS is unaware that it is running as a guest under z/VM. What the z/OS guest thinks is real-time is actually the time-of-day clock and processor timer facility. Elapsed time as measured by the time-of-day clock is accurate. The guest's virtual processor timer (VPT) runs whenever the guest is dispatched or is in a voluntary wait state. It does not run if the guest is in a CP wait state. Thus, when z/VM dispatches another virtual machine and later dispatches the z/OS guest, z/OS does not realize it had stopped running.

Note: z/OS does not support FBA DASD. You can run a mixed DASD configuration of FBA and CKD on a z/VM with a Z/OS guest system. However, the z/OS guest is restricted to using only CKD DASD, just as it would be if it were running natively.

If the installation has very little DASD, then configuring the environment to share as many volumes as possible is very beneficial.

System residence volume
A single copy of the z/OS system residence volume can be shared among multiple test images to reduce the need for DASD space on each image.

- A single master copy of the system residence is set up and maintained by one particular z/OS virtual machine, referred to in the following discussion as the ZOSMAINT machine.
- Each test virtual machine has read-only links to this DASD. This implies that each virtual machine sees exactly the same level of code as other virtual machines, provided that no STEPLIBs or other methods of overriding system code are used. Further, when updates are applied to the system residence volume, all users will see those changes at next IPL.

PARMLIB Structure
In a shared-DASD environment, each image needs to have configuration information that is specific to that instance of z/OS (for example, TCP/IP profiles so that each image gets a unique IP address). Shared PARMLIBs can be used to contain the bulk of the configuration information, with support team members being the only users permitted to alter this configuration information, and only on the ZOSMAINT machine. However, each user tends to have to make

minor changes to his or her z/OS instance, so a PARMLIB hierarchy can be implemented to establish a private PARMLIB for each image as the first one to be checked.

These private PARMLIBs can reside on DASD that is private to the particular image, or on DASD that is shared R/W by all images. PARMLIB search order is specified in the LOADxx member used at IPL time.

Recommendations when running z/OS as a guest of z/VM:

- Define the z/OS master console on a different control unit from the z/OS log on console. Otherwise, you may experience console lockouts.
- For any z/OS virtual machine, there is the choice as to whether to SET SVC76 CP or to SET SVC76 VM.
    - **SET SVC76 CP** lets z/VM collect all error records in one location. It also lets CP translate your virtual machine's storage addresses into host addresses, and your virtual machine's I/O device addresses into real device numbers.
    - **SET SVC76 VM** sends the error records to z/OS's SYS1.LOGREC data set, and address translation does not occur.

      Note:
      SET SVC76 CP is convenient because all error records are stored in one location, but since the error records are not written to SYS1.LOGREC, z/OS error recovery may be adversely affected.

- For detailed information on SET SVC76 see the z/VM: CP Commands and Utilities Reference.
- z/VM simulates the hardware load parameter facility for virtual machines. This lets the Z/OS virtual machine operator pass a load parameter (up to 8 bytes of data) when using the IPL command to load an alternative nucleus.
- z/OS cannot operate in an XC virtual machine[‡].

When z/OS IPLs, WTOR messages appear. If the operator does not answer each of these messages promptly, then z/OS might consume all of the resources it is authorized to use. This can cause inconvenience throughout the system.

Here are a couple of ways to avoid this problem:

- Limit the resources to which your z/OS guests have access. For example, you might logically partition the processor and use LPAR resource capping.
- If you prefer to IPL z/OS manually, be certain that a responsible operator monitors the console and responds to the IPL messages correctly and promptly.

Console Definitions

You have two kinds of virtual console to consider when running z/OS under z/VM:

- The log on console is the virtual machine console. Use this console to enter CP commands.
- The z/OS operator console communicates with z/OS. Use this console to enter Z/OS commands.

You can use two separate terminals as your log on and z/OS operator console, or you can use one terminal for both.

[‡]z/VM uses extensions to the interpretive-execution facility to provide the Enterprise Systems Architecture /Extended Configuration (ESA/XC) virtual machine architecture. ESA/XC is an architecture unique to virtual machines. Because it exists mainly to provide services to application programs in virtual machines, ESA/XC architecture does not have a native-mode equivalent. ESA/XC architecture lets virtual machines share multiple data spaces. An XC virtual machine can access one or more data spaces of another virtual machine if so authorized. This is extremely useful for applications that require one or more virtual machines to serve many users.

If you use separate terminals, your directory entry should look like this:

```
CONSOLE 01F
DEDICATE CC0 CC0
```

The console at virtual address 01F is your log on console. From there, you can log onto your guest virtual machine and IPL the guest. You can then disconnect the log on terminal. The terminal at real address CC0 is now your z/OS operator console.

If you want to use the same terminal as your log on and z/OS operator console, create a PROFILE EXEC that automatically sets up a 3270 console for you.

Your log on console is at virtual address CC0. The 3270 specification lets the z/OS guest share a locally attached terminal controlled by CP. The z/OS guest can use the terminal in full-screen mode, while CP shares the terminal and uses it as a line device.

Crypto Definitions

To run Integrated Cryptographic Service Facility (ICSF) applications, z/OS guests must be configured to use the z/VM guest support for crypto. z/OS guests may use the following types of cryptographic hardware:

- The IBM CP Assist for Cryptographic Function (CPACF), which is available on the processor board
- One or more IBM optional cryptographic features, with PCI-X adapters that can be individually configured as a secure coprocessor or as an accelerator for SSL.

Virtual Multiprocessing

z/OS can run in a multiprocessor (MP) environment and can have many more virtual processors defined than are available in the real system. For the sake of performance, however, you should not define more virtual processors than the hardware offers.

With z/VM you can run virtual machines in certain multiprocessing environments. A real multiprocessing environment is created using an *n*-way processor complex: a two-way processor complex consists of two processors sharing a common real storage; a three-way processor complex consists of three processors sharing a common real storage; and so on.

To set up a virtual multiprocessing environment, your installation does not have to have a multiprocessor complex. You can do virtual multiprocessing even if your real processor is a uniprocessor. With a uniprocessor, however, you cannot dedicate a real processor to a virtual machine.

Problem Determination

System problems appear in the usual way when you run z/OS under z/VM. Abends, wait states, loops, and incorrect results are typical symptoms of system problems.

When you run z/OS under z/VM, any of these system problems may occur with either z/OS or z/VM in control of the processor.

Error Recording and Analysis

Two sources of information for analyzing system problems are the SYS1.LOGREC data set and the CP error recording virtual machine (EREP).

CP Dumps

Whenever an abend occurs in CP, the module HCPDMP produces a system abend dump. A system abend dump occurs whenever the system operator presses the PSW restart key. When you obtain a system abend dump (also called a CP dump)**:**

- Use the CP SET DUMP command to direct an abend dump to an output device.
- The system operator can produce a system abend dump by performing a PSW RESTART.

z/OS Dumps
z/OS provides many service aids for collecting information on system problems. For details, see *Z/OS Diagnosis: Tools and Service Aids*.

SVC Dumps
SVC dumps for z/OS are a useful source of information about z/OS problems. At times, they provide clues to what appear to be hardware problems.

Remember, z/OS does not know that it is running under z/VM. Certain CP instruction simulation errors may appear to the z/OS guest as hardware problems.

For very difficult hardware-like errors, you may have to ZAP the z/OS system to load a disabled wait PSW. This will allow you to dump the Z/OS system using the Z/OS stand-alone dump.

Stand-Alone Dumps
Creating a stand-alone dump of your Z/OS guest is not very much different from creating a “hands-on” dump

Why Parallel Sysplex under z/VM?

There are some very attractive advantages to z/VM that you should consider for running a Parallel Sysplex under z/VM for test, development, and training purposes.
The main advantages are:

- Isolation - z/VM is ideal for providing a testing environment. You can do just about anything you like in a z/OS virtual machine running under z/VM, without the fear that you might impact anything else. You can break operating systems, CF links, and CFs without going near the real hardware or touching the Hardware Management Console (HMC). Given the nature of the things that we do in a test environment, this provides a great degree of comfort, and permits the users of the system the flexibility to do anything they want without the fear that this might impact the production system.
- Flexibility - One of the really impressive features of z/VM is the self-contained, virtual environment it provides for you. You can break CF links, CFs, and operating systems, all with a z/VM command. As a result, you can set up a training environment for Parallel Sysplex on z/VM and use it from anywhere that you can log on to z/VM. There is no requirement that the training should be done in the computer area. In fact, if you have dial-in facilities, you can conceivably do all of this from home.
- Reduced resource requirement - When running in the logical partition (LPAR) mode, each operating system image and each CF requires its own dedicated processor storage. However, if you are running under z/VM, the storage of each virtual machine is paged in and out by z/VM as required, allowing you to run with a significantly smaller amount of processor storage. Obviously, the additional processor storage requirement depends on the way the system is used (that is, when and how intensive).

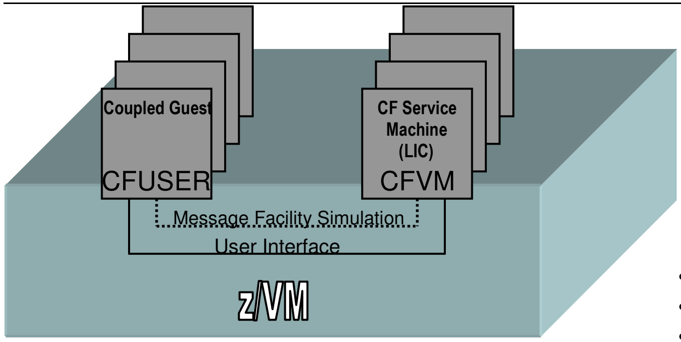
z/OS Parallel Sysplex under z/VM

In addition to being able to run z/OS systems as guests under z/VM, you can simulate advanced *Coupling Facility* (CF) and *message facility* (MF) functions.

z/VM guest coupling provides for the simulation of one or more complete Parallel Sysplexes within a single z/VM system image. The intent is to provide a preproduction testing platform for a coupled-system installation, and for training people.

The z/VM simulated environment is not intended for production use because its single points of failure negates the intent of the Parallel Sysplex® environment.

Continued…



Another reason for running a z/OS Parallel Sysplex under z/VM could be to perform disaster recovery testing, which may be a very cost-effective solution.

The z/VM guest coupling simulation support consists of three components:

- Coupling Facility Service Machines
- Coupled guests
- Simulated Message Facility.

Planning to set up a Parallel Sysplex under z/VM
To set up a Parallel Sysplex under z/VM, you have the choice of using an existing z/VM system, if there is one already, or use a dedicated z/VM LPAR to run the Parallel Sysplex.

The advantages of setting up a separate z/VM LPAR just for the Parallel Sysplex are that it provides more isolation and a more secure environment. The disadvantages are that it takes more resources than running everything under a single z/VM LPAR and there is one more z/VM system to manage. Select the best method based on your environment and requirement.

– – –