



S H A R E
Technology • Connections • Results



VM For MVS Systems Programmers, Part 1

Martha McConaghy, Marist College
Mark Post, Novell

Monday, March 2, 2009
Session 9127

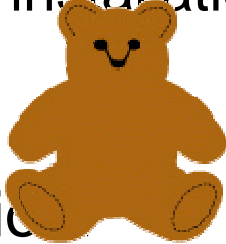
About Us



- Martha – 24+ years VM systems programmer at Marist College
- Mark - 27 years with GM and EDS, 22 of them as an MVS systems programmer. Now, a Linux guru with Novell.



Don't Panic!

- VM really isn't scary, just different. Remember, our mascot is the teddy bear!
 - Its so easy, the entire installation summary is a total of 2 sheets of paper!
- Purposes of presentation
 - Brief introduction to basic concepts of VM with focus on system administration.
 - Illustrate similarities with basic MVS systems and concepts.
 - Explain differences between systems in terms familiar to MVS systems programmers.

Follow Up Presentations

- Other presentations this week which cover related subjects in more detail:
 - 9102 The Very Basics of z/VM – Concepts and Terminology Mon 9:30am (available on proceedings)
 - 9115 VM Performance Introduction Mon 1:30pm (available on proceedings)
 - 9107 – 9108 Introduction to VM Hands-on Lab Tue 8am-10:30
 - 9117 Introduction to VMSES/E for z/VM Thur 8:00am
 - 9118 Maintaining z/VM with VMSES/E demo Thur 9:30am
 - 9124 Experiences using z/VM VSWITCH Tue 1:30pm
 - 9129 z/VM Security and Integrity Wed 3pm
 - 9133 Configuring, Customizing and Modifying your VM System without an IPL Tue 1:30pm
 - 9134 Dynamically Managing Hardware I/O Configuration using VM Tue 3pm

Agenda



- **Part 1:**
 - **Introduction to and comparison of basic concepts**
 - **What is a hypervisor and what about all the “virtual” stuff?**
 - **Caring for a VM system (maintenance, system datasets, etc.)**
 - **System configuration concepts**
- **Part 2:**
 - Applications and guests
 - CMS vs. TSO
 - File Editors
 - Common Tasks

We will answer questions as time allows.....

Kissin' Cousins....

- VM and z/OS have a lot in common:
 - Both have roots going back to the 1960's when they were developed.
 - Both developed to run on IBM hardware.
 - Both have progressed through the familiar IBM architecture evolution:
 - 360
 - 370
 - XA
 - ESA
 - zSeries
 - System z9
- The early versions of PR/SM were really a subset of VM implemented in microcode. (See, you've been running VM all along! 😊)

Hardware Support

- Both systems run on the same IBM hardware
 - Real device addressing is the same
 - IOCP/IODEF and dynamic I/O definitions used by both systems
 - HCD support is available for VM
 - Both systems can run native, in an LPAR, or as a guest of VM!
- Differences are important too:
 - VM can run on IFL engines, z/OS cannot.
 - ZIIP and ZAAP engines are not used by VM. However, it can virtualize them for z/OS guests.

Support Structure

- VM and z/OS share similar support structure:
 - PTFs
 - Service packages (PUT tapes/RSU tapes)
 - Release and version levels have similar meaning
 - Both are supported by similar IBM service structure
 - z/OS uses SMP/E, VM uses SES/E – similar concepts
 - PTF is received and applied. Target module is rebuilt.
 - Each VM component is built using a different method. SES/E “hides” that by providing common tool which handles most of the differences.
 - Most components can be tested in virtual environment prior to being put into production. Second level z/VM systems are very useful for this purpose.

z/VM – The Basics

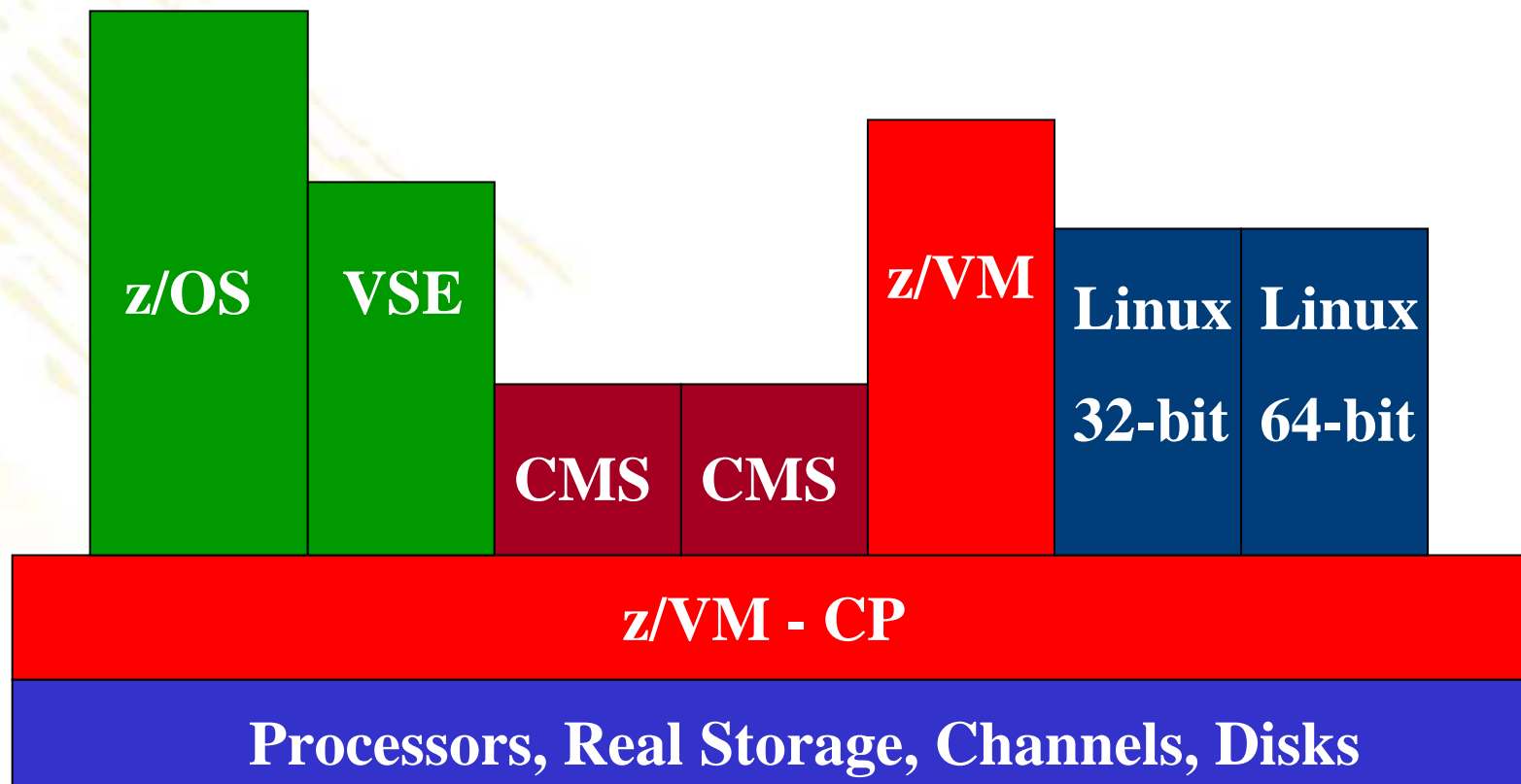


- z/VM operating system comprised of 2 main components:
 - Control Program CP (hypervisor – creates the virtual environments)
 - CMS (file system, editor, programming tools, etc.)
- Other components used for specific purposes:
 - GCS (run environment for certain products)
 - TCP/IP (provides network connectivity)
 - VM SES/E (system maintenance)
 - AVS (APPC/VM VTAM support)
 - TSAF (Transparent Service Access Facility, an APPC enabler)
 - RSCS (Remote Spooling Communication Subsystem)
 - Various priced support features

CP – the Hypervisor

- **CP** controls the real resources of the machine, i.e. memory, processor, storage, etc.
- **CP** allocates resources to “users” as needed so they are **unaware** of the presence of each other or of CP itself (**hypervisor**). Each user appears to have an entire real machine to themselves, hence the name “**virtual machine**”.
- A “**virtual machine**” which contains another operating system, such as MVS, VSE or Linux is called a “**guest**”.
- The “cost” to do this virtualization is very low. Handshaking with processor microcode helps keep things going fast!

VM - "Piece of Cake"



“Levels”



- **First Level CP** – the hypervisor that controls real resources of processor within the LPAR. The resources for all users are controlled by this system.
- **Second Level** – the system running in the virtual machine. This could be z/OS, Linux or z/VM. Most resources appear real to this level, but are probably virtual. Some resources, like tape drives, may be real and dedicated to a specific virtual machine or shared between several.
- z/OS virtualizes memory for different tasks running on it. (Remember the “V” in MVS?) However, it does not have the concept of levels. This is a hypervisor concept.

Basics of Running Hypervisor

- Smooth operation of First Level CP is critical to smooth operation of all guests.
 - Key performance indicators must be monitored for problems.
 - Resources must be balanced between guests, i.e. relative shares must be adjusted as circumstances change over time.
 - For example, the workload on a particular guest system may increase over time as more users or services are added. The priority for this guest may have to be increased to ensure good response for its users.
 - Conversely, a guest's demands for resources may have to be limited to keep it from swamping other guest systems.
 - This is similar to workload management on z/OS, but the tools are very different.
 - Performance management sessions give great information on how to balance resources between guest systems.

Basics of Running Hypervisor

- Key resources for CP are:
 - Expanded processor memory for initial paging space
 - Ample disk space for “old” page migration
 - Ample disk space for spool files
 - SPOOL has many uses in VM, its not just for print files.
 - CP PARM disks – proper maintenance and backup is essential
 - CP Directory – proper management is a MUST! (Too many system admins have paid the price of messing up the directory....ouch!)
 - NSS files for CMS and other support components.
- Most system admin work involves maintaining these resources.

Basics of Running Hypervisor – System Space



- Both z/OS and CP require PAGE and SPOOL space.
 - VM does not use discrete datasets for either. The space is allocated on volumes using ICKDSF utility. Multiple volumes normally used to spread I/O load.
 - **SPOOL** is used for temporary files waiting to be printed, like MVS. Also used for saved systems, save segments, NLS segments, TRSOURCE traces, etc.
 - **PAGE** is used for Auxiliary Storage. Similar to MVS, but the process for determining what pages to put out on disk is very different.
 - z/VM still benefits a lot from having some expanded storage defined for CP paging.

System Configuration



- Basic System Requirements
 - z/VM does not have a SYS1.PARMLIB, per se. (You had to have seen that coming, surely 😊).
 - It does have a CMS text file named SYSTEM CONFIG, which serves a similar purpose. It is read at IPL time and contains customization parameters such as:
 - System owned disk volumes
 - Name of system
 - Time zone definition
 - Console addresses
 - Warm start and checkpoint disk areas
 - SYSTEM CONFIG file is used for a lot of important parameters
 - **CP** system code is compiled into a single module, called the “CP nucleus”, similar to what is contained in SYS1.NUCLEUS. The name of the module is CPLOAD MODULE.

System Configuration



- Config file and the CP load module are stored on special minidisk called a **PARM disk**
- System PARM disk is a CMS formatted disk that can be read by CP. Each z/VM system comes with 3 PARM disks predefined:
 - PARM1 is the primary disk, read first by CP
 - PARM2 is intended as a backup to PARM1. At IPL time, CP can be instructed to read it instead of PARM1. Used if there is a problem with the contents of PARM1.
 - PARM3 can be a third backup or used as a place to hold local programs and exits.
- PARM disks also contain Logo definition files.
 - Similar to VTAM USSTAB screens, but usually much prettier. 😊
 - Also easier to define.

System Operator



- A special virtual machine which takes control of system after IPL. Numerous CP messages are directed to the Operator, similar to master console on MVS.
- It is usually defined in the CP Directory with all CP privileges. It is similar to “root” in Linux, but does not have access to all file systems.
- The Operator virtual machine is designated in the SYSTEM CONFIG file.
- Console messages from other virtual machines can be directed to the Operator. It can be used to filter messages and take action, similar to how Netview works on MVS.
 - PROP (Programmable Operator, comes with VM)
 - VM:Operator (CA)

System Security



- VM has several levels of native security. The most basic level is the privilege “class”. This controls what commands can be issued from a virtual machine.
 - As supplied by IBM, privilege class A has the most powerful commands, class G the least powerful. Classes can be redefined to fit local site’s needs.
 - Multiple classes can be assigned to a virtual machine.
 - Privilege classes only control commands issued to CP. They do not provide any special access to file systems or to applications running in virtual machines. (Class A is not the same as “root” access in UNIX.)
- Virtual machines each have separate passwords, controlled by the directory.
- Access to real resources of system controlled by privilege class and/or definitions in the directory.

System Security



- Minidisks can have passwords to allow other virtual machines read and/or write access.
 - Not a good idea to write to a minidisk that someone else is already using in write mode.
 - Rules based security is available with additional products such as:
 - VM:Secure (CA)
 - RACF (IBM)
 - ACF2/VM (CA)
 - Top Secret (CA)
- Applications and systems running in virtual machines may also have their own security, which is separate from VM. For example, a guest MVS system may run RACF, but underlying VM does not.

System Maintenance



- Service comes in three forms:
 - Release
 - RSU (Recommended Service Upgrade)
 - PTF (Program Temporary Fix)
- New releases are installed as a new system and then migrated to production.
- RSU (similar to PUT service) is a collection of PTFs. RSU's are normally cumulative.
- Individual PTFs can be applied to fix specific problems between RSUs.
- All service applied using VMSES/E – similar to SMP/E. Concepts are similar, however the tracking is not as detailed.