VM For MVS Systems Programmers Part 1

Martha McConaghy, Marist College
Mark Post, EDS
Monday, February 12, 2007
Session 9127

About Us

- Martha - 22 year VM systems programmer at Marist College
- Mark - 27 years with GM and EDS, 22 of them as an MVS systems programmer.

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.
  - Illustrate similarities with basic MVS systems and concepts.
  - Explain differences between systems in terms familiar to MVS systems programmers.
  - Provide introduction to VM system administration.

Follow Up Presentations

- Other presentations this week which cover related subjects in more detail:
  - 9107 – 9109 Introduction to VM Hands-on Lab Tue 8am-noon
  - 9125 Virtual Networking with z/VM Guest LANS and Virtual Switch Tue 8am
  - 9115 VM Performance Introduction Tue 1:30pm
  - 9119 z/VM Installation - From Cardboard Box to IPL Tue 3pm
  - 9117 Introduction to VMSES/E for z/VM Wed 9:30am
  - 9118 Maintaining z/VM with VMSES/E lab Wed 11am
  - 9133 Configuring, Customizing and Modifying your VM System without an IPL Wed 3pm
  - 9123 TRACK for z/VM – What's Happening in your Virtual Machine Thur 9:30am
  - 9116 Configuration Tools for z/VM TCPIP Thur 3pm
  - 9136 Automated Linux Guest Monitoring on z/VM using PROP Thur 1:30pm

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! ☺)

- 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 IBM service structure
  - z/OS uses SMP/E, VM uses SES/E – similar concepts
- 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 has now been added to VM
  - Both systems can run native, in an LPAR, or as a guest of VM!
- In the past, VM has sometimes lagged in support of new hardware. This has changed significantly in recent years.

z/VM

The Basics

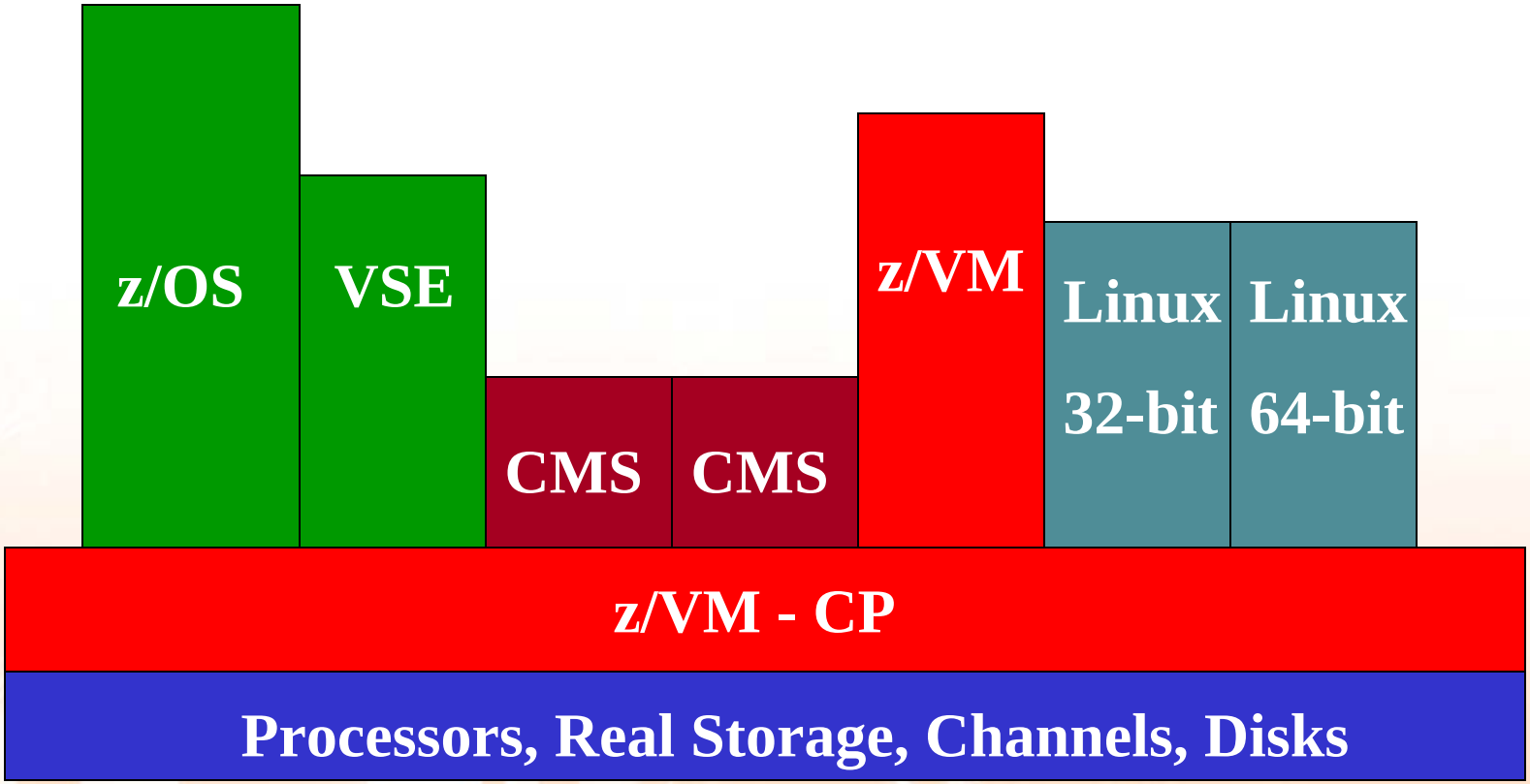
z/VM – The Basics

- 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”



What is a Virtual Machine

Anyway???

Virtual Machine

- Each virtual machine is a separate, individual environment. By default, nothing is shared and there is no communication between them, just like in an LPAR partition.

- What occurs in one virtual machine has no direct impact on other virtual machines running in the same CP. They can indirectly impact each other by tying up CP resources. This is where performance tuning comes in.

Virtual Machine

- Hardware resources are virtualized, and can be dynamically changed by systems programmer (some changes are disruptive)
  - processors
  - memory (storage)
  - NIC (network interface connection)
  - printer, reader, punch
    - May seem archaic, but very effective for transferring data between virtual machines
- Disk space provided by combination of:
  - Minidisks (cylinders allocated as a virtual device)
  - Dedicated or Attached DASD volumes
  - Shared File System directory
  - Byte File System (VM's version of UNIX System Services' HFS)

System (CP) Directory

- All “users” of VM are virtual machines. Whether it is a z/OS system or a CMS user, the characteristics of the virtual machine are defined in the CP Directory.
- All virtual machines are created by defining them in the CP Directory. There is nothing really comparable to this in z/OS. Perhaps the closest thing is defining a user in RACF or ACF2.
- A few of the same types of things are defined, such as the name of the user and password. Otherwise, they are very different.

System (CP) Directory

- Directory starts out as a CMS text file listing all virtual machines and their definitions. It is compiled into an object form and written to disk by the DIRECTXA command.
- CP cannot run if the directory is damaged or corrupted, so it is extremely important. Be sure to back it up as part of your disaster recovery plans.
- Directory is capable of containing 20,000 or more virtual machine definitions. Most systems are much smaller.
- As the directory gets larger, it is harder to manage and easy to screw up. Overlaying minidisk allocations is especially dangerous with a large directory.
- Products like DIRMAINT (IBM) can help you manage the directory.

System (CP) Directory

- Many of the virtual machines defined in the CP directory will, in turn, run operating systems within their environment.
- These guests, such as z/OS or Linux systems, will contain their own users. They are created using the processes of each guest system. The CP directory has no impact on this. Users defined to a guest z/OS or Linux system will have no awareness of CP at all.

Sample Directory Records

- **USER** – defines name of virtual machine, along with password, virtual storage size and privilege classes. It's always the first record in a virtual machine definition.
- **IPL** – defines the virtual address of the "boot" disk or SYSRES of the system to be IPL'd in the virtual machine. Just like the LOAD option in an LPAR definition. It could also point to a NSS (Named Saved System), like CMS.
- **CONSOLE** – defines the virtual address of the device that will act as a console for the system running in the virtual machine. The type of console is also defined. For a z/OS guest, this would be the address of the z/OS system console.
- **SPECIAL** - defines a virtual device to the virtual machine. This is often used for things like virtual terminals, virtual NICs to connect to a guest LAN or VSWITCH.

Sample Directory Records

- **MDISK** – defines the virtual address, size and location of a minidisk that will be owned by the virtual machine. The size of the minidisk can be as small as 1 cylinder (or track for FBA devices) or as large as the physical size of the volume. Minidisks cannot span physical volumes.
- **LINK** – allows the virtual machine to have access to a minidisk owned by another virtual machine. Read/only, Read/Write and Multi/Write are the options. Multi/Write can be dangerous as it allows 2 machines to write to the same disk at the same time. **(No GRS for serialization!)**
- **DEDICATE** – gives sole control of the device to the virtual machine. CP will have no more involvement in the device while the virtual machine is logged onto the system. Used often to give full disk volumes to a guest system.

Simple Virtual Machine Definition

```
USER LXURMM password 64M 64M G
   IPL 195
   MACHINE ESA
   SHARE RELATIVE 2000 RELATIVE 5000 LIMITHARD
   CONSOLE 0009 3215 T LXMON
   SPECIAL 3000 HIPER 3 SYSTEM WEBDMZ2 0
MDISK 0195 3390 601 2000 MARB1A MR
```

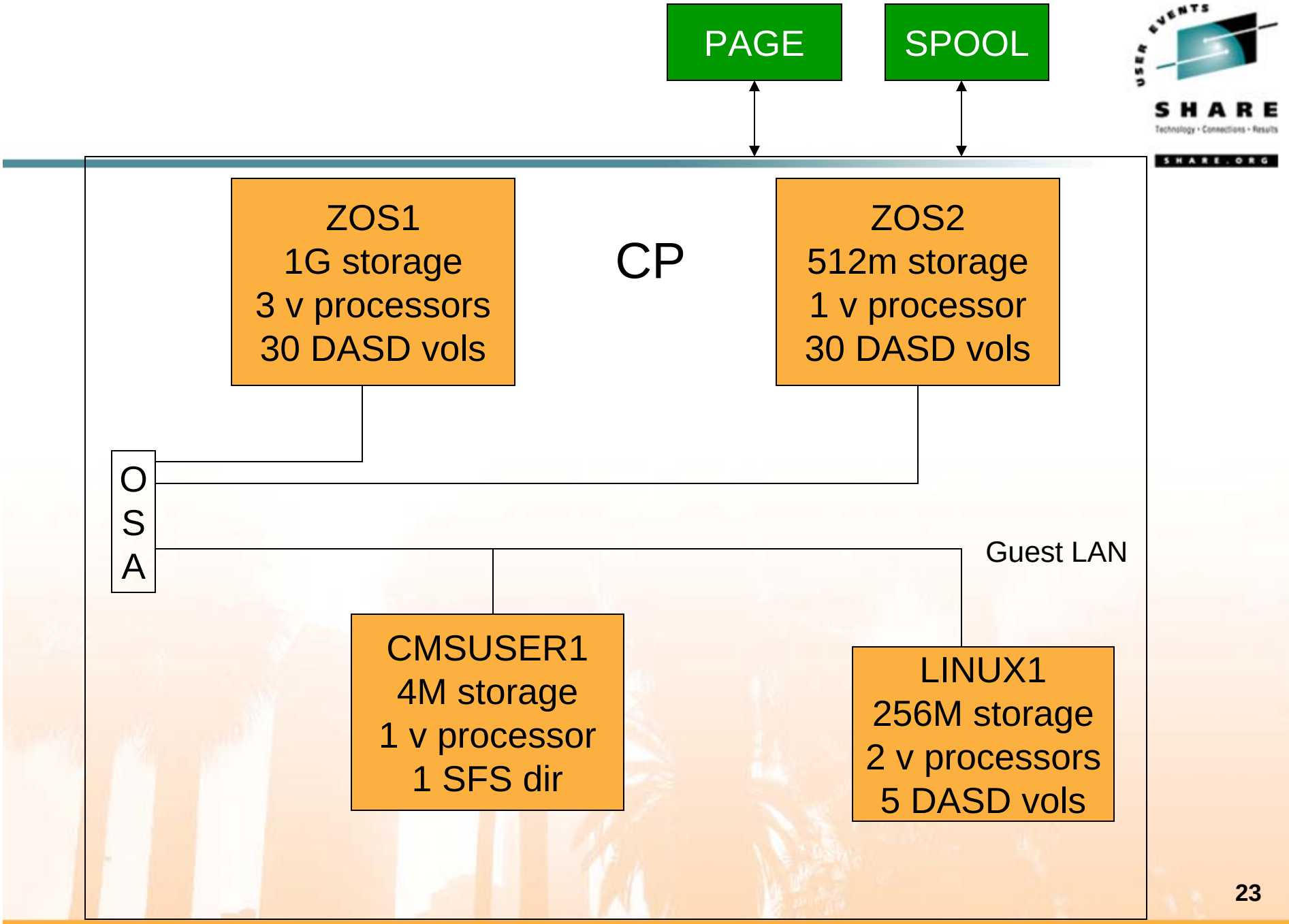
Basic Needs of

A Happy

CP

System Disk Space

- Both z/OS and CP require PAGE and SPOOL space.
  - VM does not have discrete datasets for either.
  - The space is spread over multiple volumes which are reserved using ICKDSF. While not recommended, either can be combined on the same volume with each other, or other data.
  - **SPOOL** is used for temporary files waiting to be printed, like MVS. However, VM uses it for numerous other purposes too.
  - **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.


PAGE
SPOOL
ZOS1
1G storage
3 v processors
30 DASD vols
CP
ZOS2
512m storage
1 v processor
30 DASD vols
O
S
A
Guest LAN
CMSUSER1
4M storage
1 v processor
1 SFS dir
LINUX1
256M storage
2 v processors
5 DASD vols

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 culmulative.
- 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.

Its no good if you can't

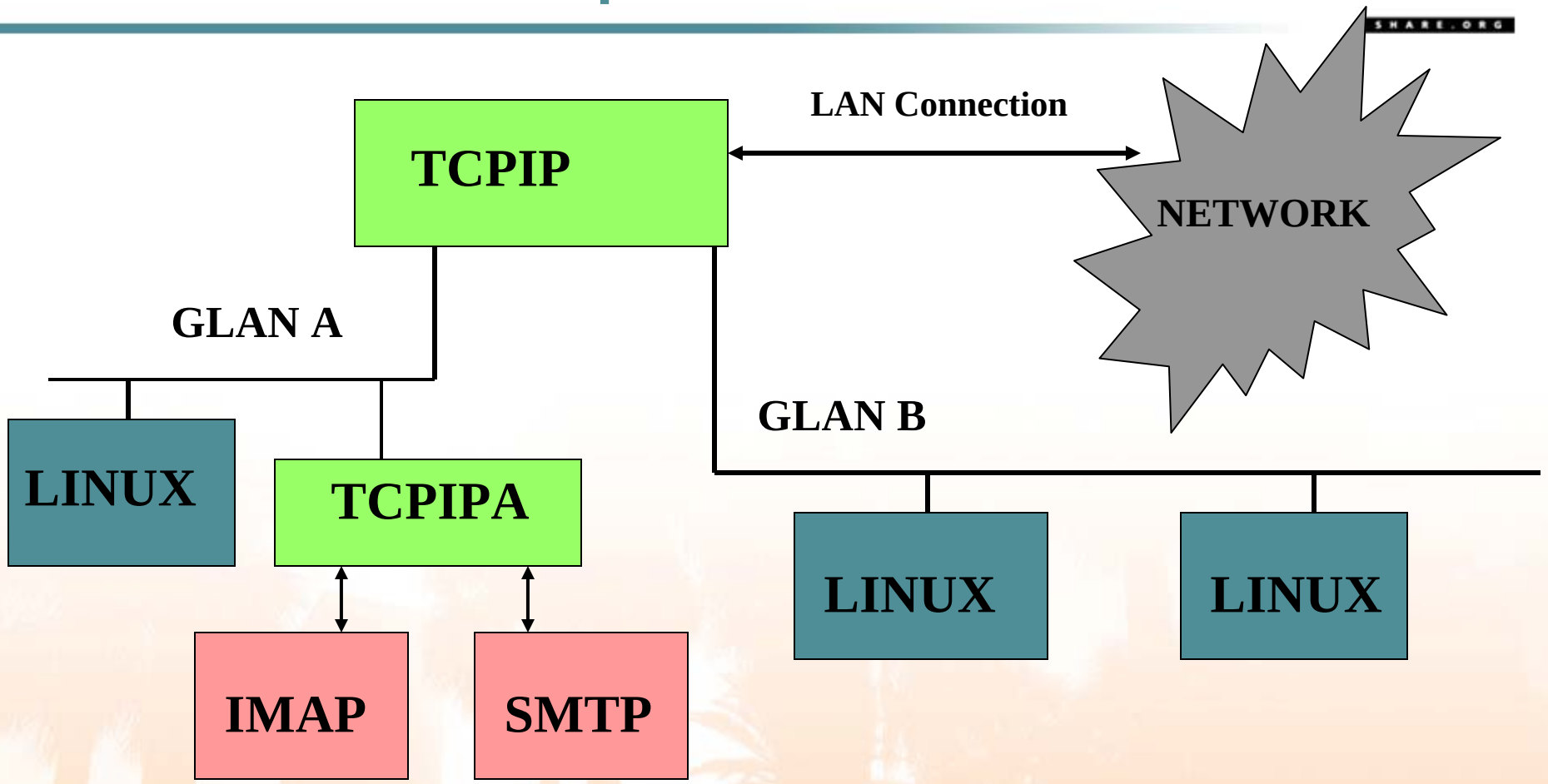
SHARE it with someone!

Network Connectivity

Network Connectivity

- Real:
  - VM supports same network hardware that z/OS supports, i.e. Open System Adapters (OSAs), CLAWs, CTCs, etc.
  - Both TCP/IP and VTAM are available. TCP/IP comes with z/VM and is the de facto standard. VTAM is a separate product.
- Virtual:
  - Guest LAN is a “virtual Ethernet or HiperSocket LAN”, unique to z/VM.
    - Allows for creation of simple to elaborate “networks” within a single z/VM system.
  - Virtual switch works with OSA to emulate real network Layer 2 or Layer 3 switching capabilities.
    - Allows virtual machines to join real network VLANs, using native network routing, security and protocols.
  - Both technologies developed to help VM systems support running hundreds of Linux guests. However, they can be used with MVS as well.
  - IUCV, virtual CTCs still supported but much more cumbersome.

Guest LAN - Example



VSWITCH - Example