The Very Basics of z/VM

Brian K. Wade, Ph.D.
and many other authors

bkw@us.ibm.com

Revision 2011-07-19 BKW

Credits

- People who contributed ideas and charts:
  - Alan Altmark
  - Bill Bitner
  - John Franciscovich
  - Reed Mullen
  - Brian Wade
  - Romney White
- Thanks to everyone who contributed!

Introduction

- We'll explain basic concepts of System z:
  - Terminology
  - Processors
  - Memory
  - I/O
  - Networking
- We'll see that z/VM *virtualizes* a System z computer:
  - Virtual processors
  - Virtual memory
  - ... and so on
- Where appropriate, we'll compare or contrast:
  - PR/SM or LPAR
  - z/OS
  - Linux

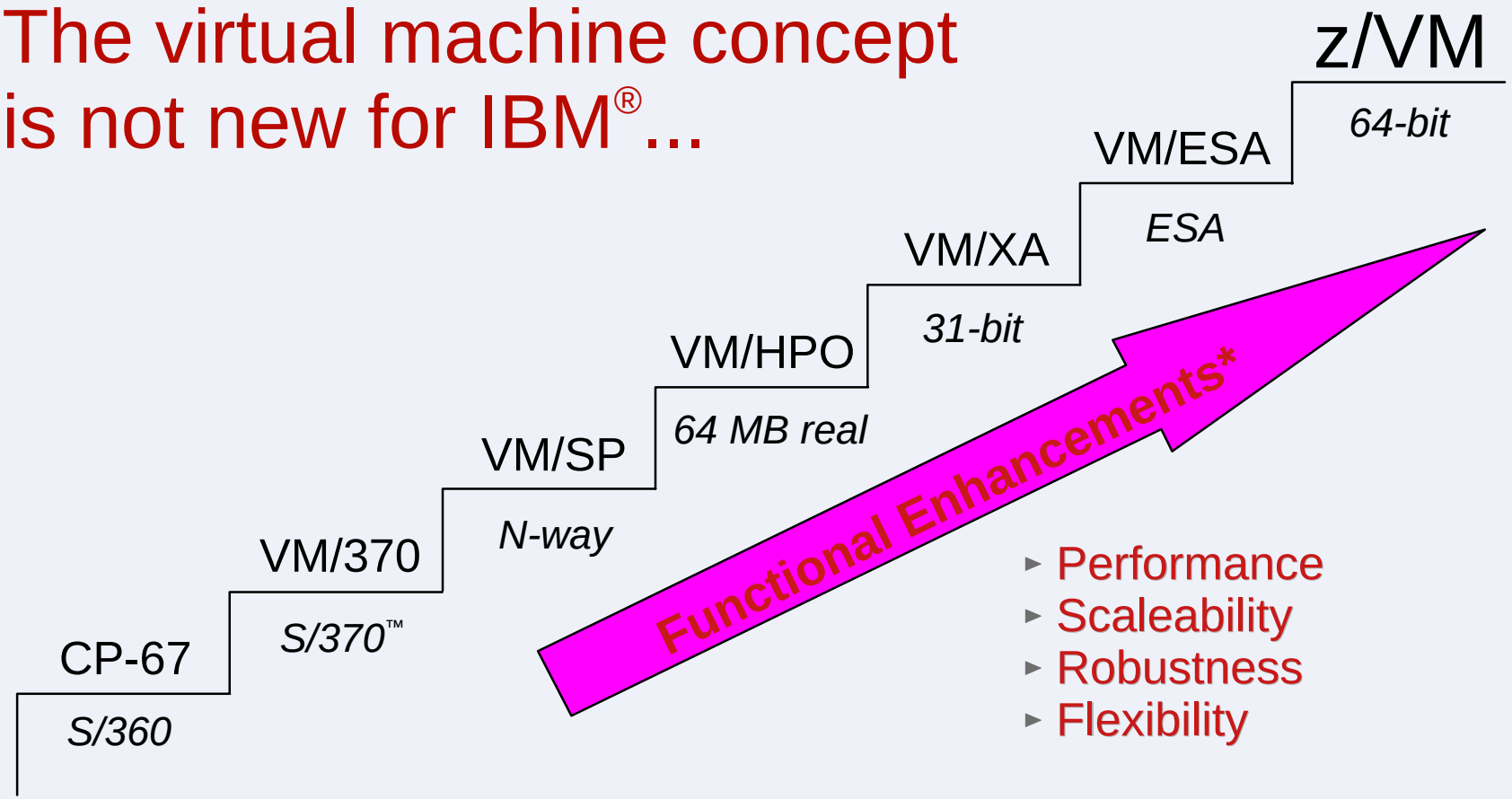
Terminology

System z Architecture

- Every computer system has an *architecture*.
  - Formal definition of the hardware's capabilities, as perceived by software.
  - It's the hardware's functional specification
  - It's what the software can expect from the hardware
  - It's *what the hardware does*, not how it does it
- IBM's book z/Architecture Principles of Operation defines System z hardware
  - Instruction set
  - Processor features (registers, timers, interruption management)
  - Arrangement of memory
  - How I/O is to be done
- Different *models* implement the architecture in different ways.
  - How many processors there are
  - How the processors connect to the memory bus
  - How the cache is arranged
  - How much physical memory there is
  - How much I/O capability there is
- z900, z800, z990, z890, z9 EC, z9 BC, z10 EC, z10 BC, z196, and z114 are all *models* implementing z/Architecture.

IBM Virtualization: Evolution

The virtual machine concept is not new for IBM®...



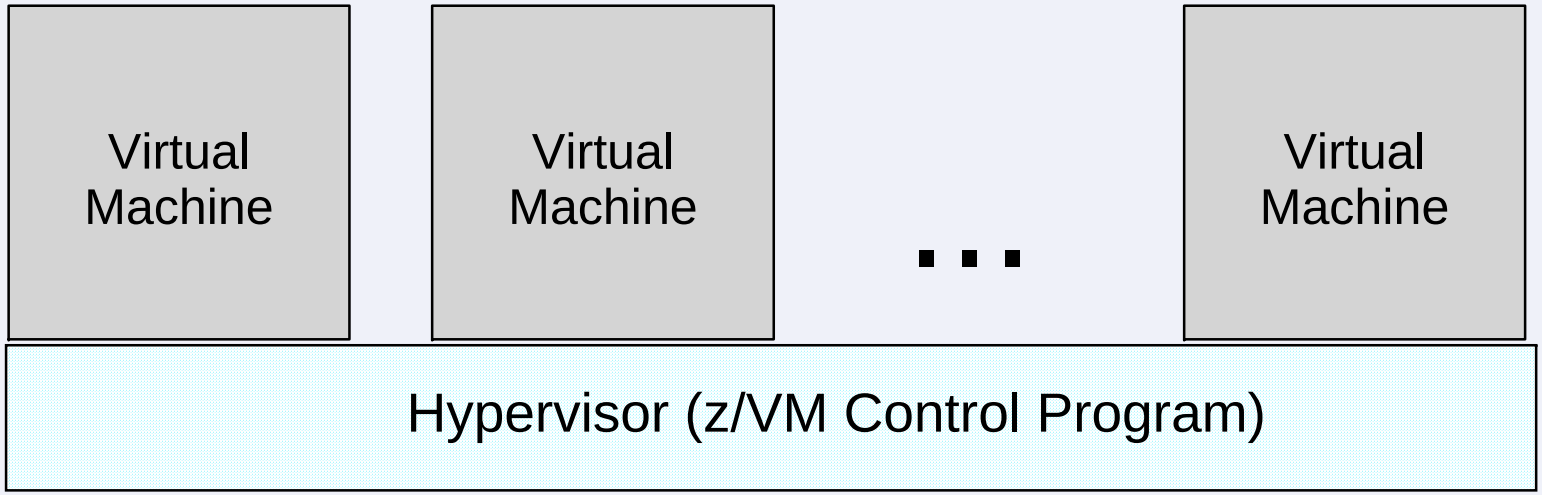
* Investments made in hardware, architecture, microcode, software

System z Parts Nomenclature

Intel, pSeries, etc.	IBM and System z
Memory	Storage Central storage, central, or CSTORE Expanded storage, expanded, or XSTORE XPRAM
Disk, storage	DASD (Direct Access Storage Device)
Processor	Processor, CPU (central processing unit), engine, PU (processing unit), IP (instruction processor), core, CP (central processor), IFL (Integrated Facility for Linux), zAAP (zSeries Application Assist Processor), zIIP (zSeries Integrated Information Processor), ICF (Integrated Coupling Feature), IFA (Integrated Facility for Applications), SAP (system assist processor), IOP (I/O processor)
Computer	CEC (central electronics complex)

Virtual Machines

What: Virtual Machines

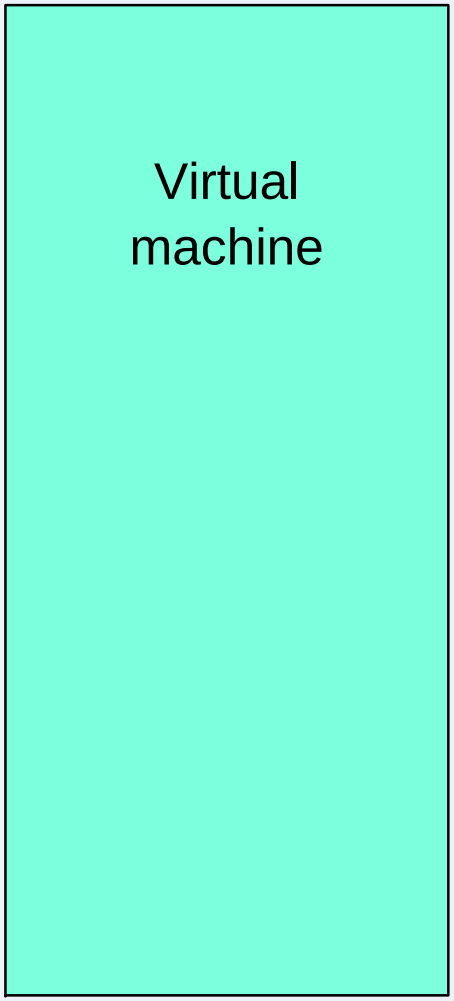


A **virtual machine** is an execution context that obeys the architecture.

The purpose of z/VM is to **virtualize** the real hardware:

- Faithfully replicate the z/Architecture Principles of Operation
- Permit any virtual configuration that could legitimately exist in real hardware
- Let many virtual machines operate simultaneously
- Allow overcommittment of the real hardware (processors, for example)
- Designed for many thousands of virtual machines per z/VM image (I have seen 40,000)
- Your limits will depend on the size of your physical System z computer

What: A Virtual Machine



z/Architecture

512 MB of memory

2 processors

Basic I/O devices:
- A console
- A card reader
- A card punch
- A printer

Some read-only disks

Some read-write disks

Some networking devices

We permit any configuration that a real System z machine could have.

In other words, we completely implement the z/Architecture Principles of Operation.

There is no "standard virtual machine configuration".

How: VM User Directory

Definitions of:

- memory
- architecture
- processors
- spool devices
- network device
- disk devices
- other attributes

```
USER LINUX01 MYPASS
CLASS G
STORAGE 512M
MAXSTORAGE 1024M
MACHINE ESA 2
IPL 190 PARM AUTOCR
CONSOLE 01F 3270 A
SPOOL   00C 2540 READER *
SPOOL   00D 2540 PUNCH  A
SPOOL   00E 1403 A
NICDEF 500 TYPE QDIO DEVICES 3 LAN SYSTEM MYLAN
LINK    MAINT 190 190 RR
LINK    MAINT 19D 19D RR
LINK    MAINT 19E 19E RR
MDISK   191 3390 012 001 ONEBIT MW
MDISK   200 3390 050 100 TWOBIT MR
```

Getting Started

- IML
  - Initial Machine (or Microcode) Load
  - Power on and configure processor complex
  - *LOGON* or *SET MACHINE*
  - Analogous to LPAR *image activation*
- IPL
  - Initial Program Load
  - Like *booting* a Linux system
  - z/VM lets us *IPL* a system in a virtual machine
  - Analogous to LPAR *load* function

How: CP Commands

- CP DEFINE
  - Adds to the virtual configuration somehow
  - CP DEFINE STORAGE
  - CP DEFINE PROC
  - CP DEFINE *{device} {device_specific_attributes}*
- CP ATTACH
  - Gives an entire real device to a virtual machine
- CP DETACH
  - Removes a device from the virtual configuration
- CP LINK
  - Lets one machine use another's disk device (read-write, read only)
- Changing the virtual configuration after logon is considered normal

Processors

What: Processors

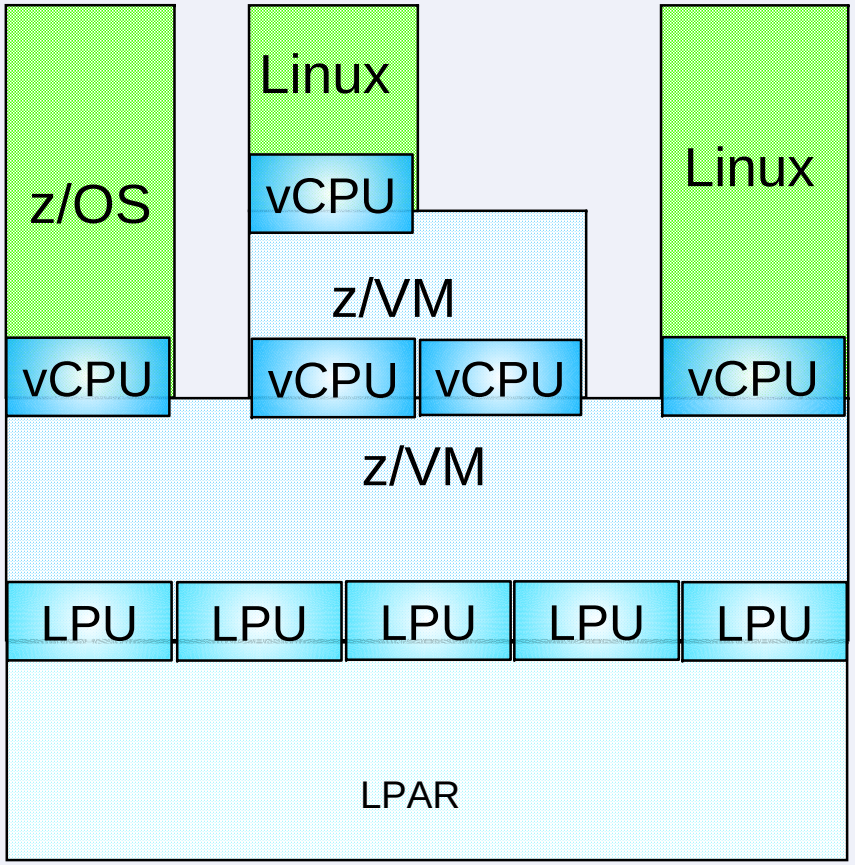
- Configuration
  - Virtual 1- to 64-way
    - Defined in user directory, or
    - Defined by CP command
  - We call these dispatchable units *virtual processors*
  - A real processor can be dedicated to a virtual machine

- Controls and Limits
  - Scheduler selects virtual processors according to apparent CPU need
  - "Share" setting - prioritizes real CPU consumption
    - Absolute or relative
    - Target minimum and maximum values
    - Maximum values ("limit shares") either hard or soft
  - Share for virtual machine is divided among its virtual processors

Phrases Associated with Virtual Machines

- In VM:
  - *Guest:* a system operating in a virtual machine, also known as *user* or *user ID* or *userid*
  - *Running under VM:* running a system as a guest of VM
  - *Running on (top of) VM:* same as *running under VM*
  - *Running second level:* running VM as a guest of VM
- In relationship to LPAR (partitioning):
  - *Logical partition:* LPAR equivalent of a virtual machine
  - *Logical processor:* LPAR equivalent of a virtual processor

What: Logical and Virtual Processors



How: Start Interpretive Execution (SIE)

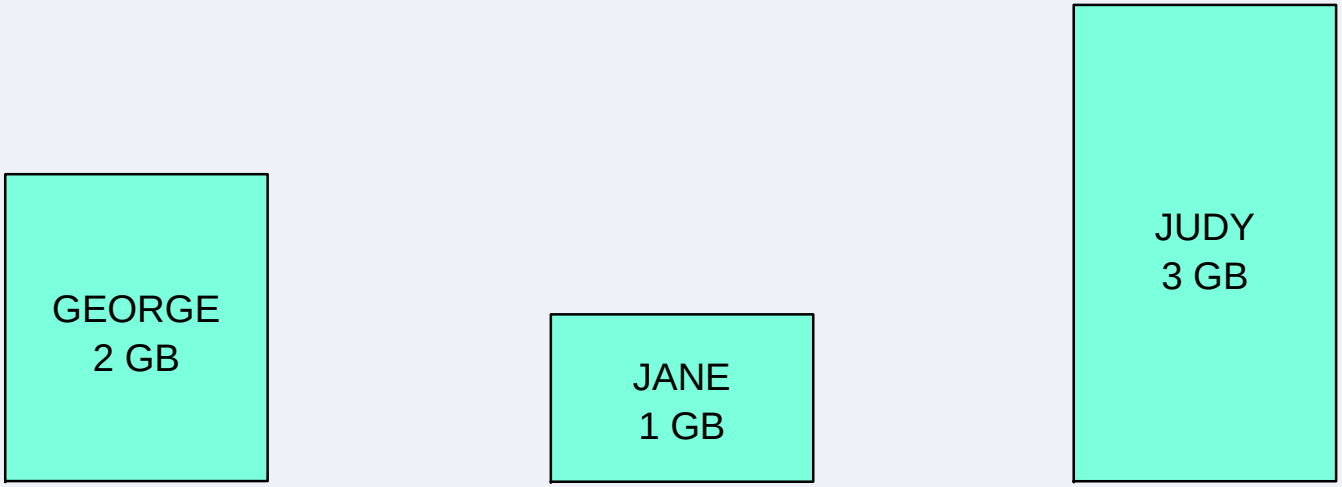
- SIE = "Start Interpretive Execution", an instruction
- z/VM (like the LPAR hypervisor) uses SIE to "run" virtual processors
- Our processors contain special hardware (registers, etc.) to make SIE fast
- SIE has access to:
  - A control block describing the virtual processor state (registers, etc.)
  - The Dynamic Address Translation (DAT) tables for the virtual machine
- z/VM gets control back from SIE for various reasons:
  - Page faults
  - I/O channel program translation
  - Privileged instructions
  - CPU timer expiration (dispatch slice end)
  - Other, including CP asking to get control for special cases
- CP can also shoulder-tap SIE from another processor to end SIE

How: Scheduling and Dispatching

- VM
  - *Scheduler* determines priorities based on share setting and other factors
  - *Dispatcher* runs a virtual processor on a logical processor
  - Virtual processor runs for (up to) a *minor time slice*
  - Virtual processor keeps competing for (up to) an *elapsed time slice*
- LPAR hypervisor
  - Uses *weight* settings for partitions, similar to share
  - Dispatches logical processors on real engines
- Linux
  - *Scheduler* handles prioritization and dispatching of processes
  - Process runs for a time slice or *quantum*

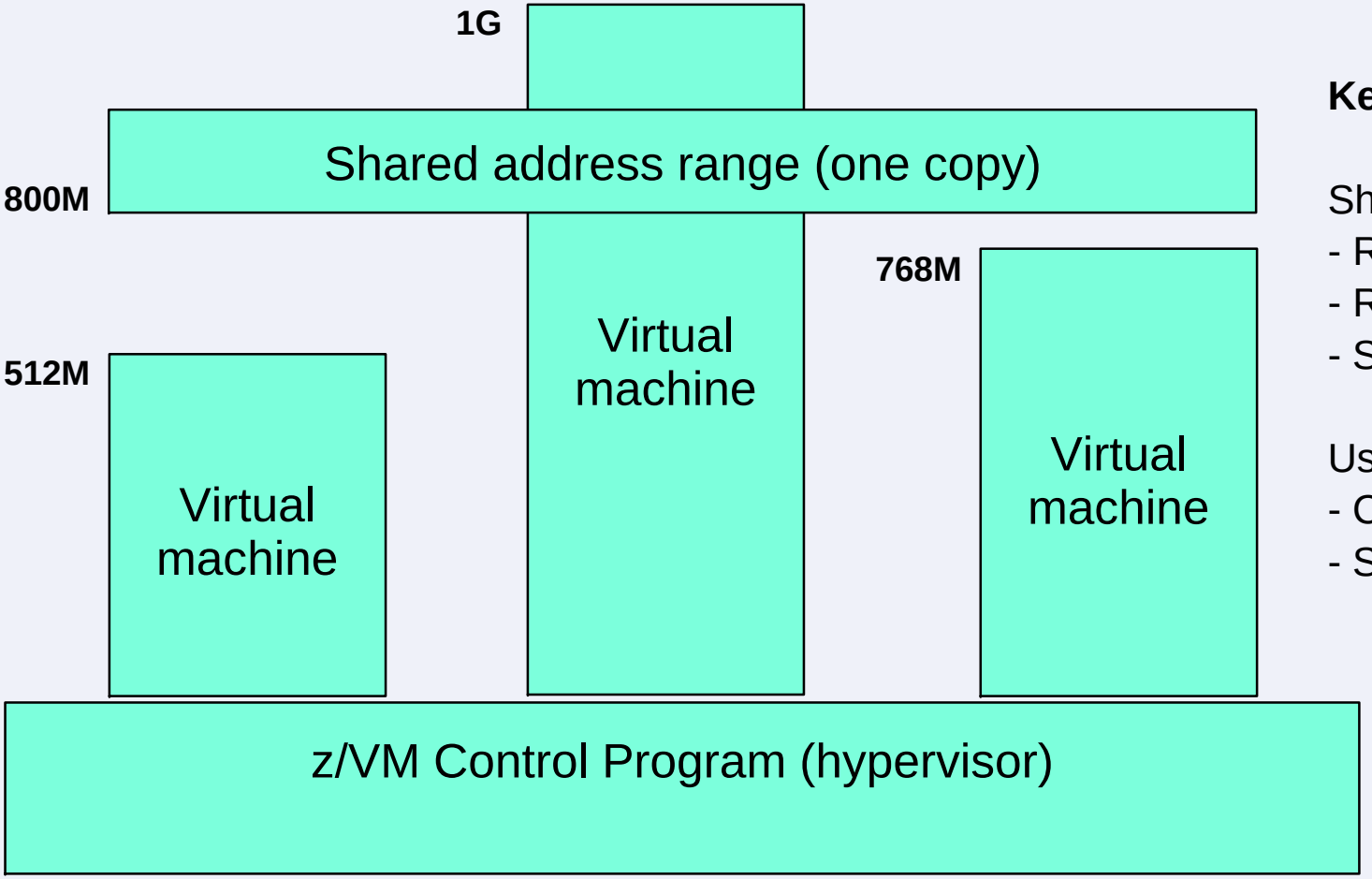
Memory

What: Virtual Memory



- Configuration
  - Defined in CP directory entry or via CP command
  - Can define storage with gaps (useful for testing)
  - Can attach expanded storage to virtual machine
- Controls and Limits
  - Scheduler selects guests according to apparent need for storage and for paging capacity
  - Guests that do not fit criteria are placed in the *eligible list*
  - Can *RESERVE* an amount of real storage for a guest's pages

What: Shared Memory



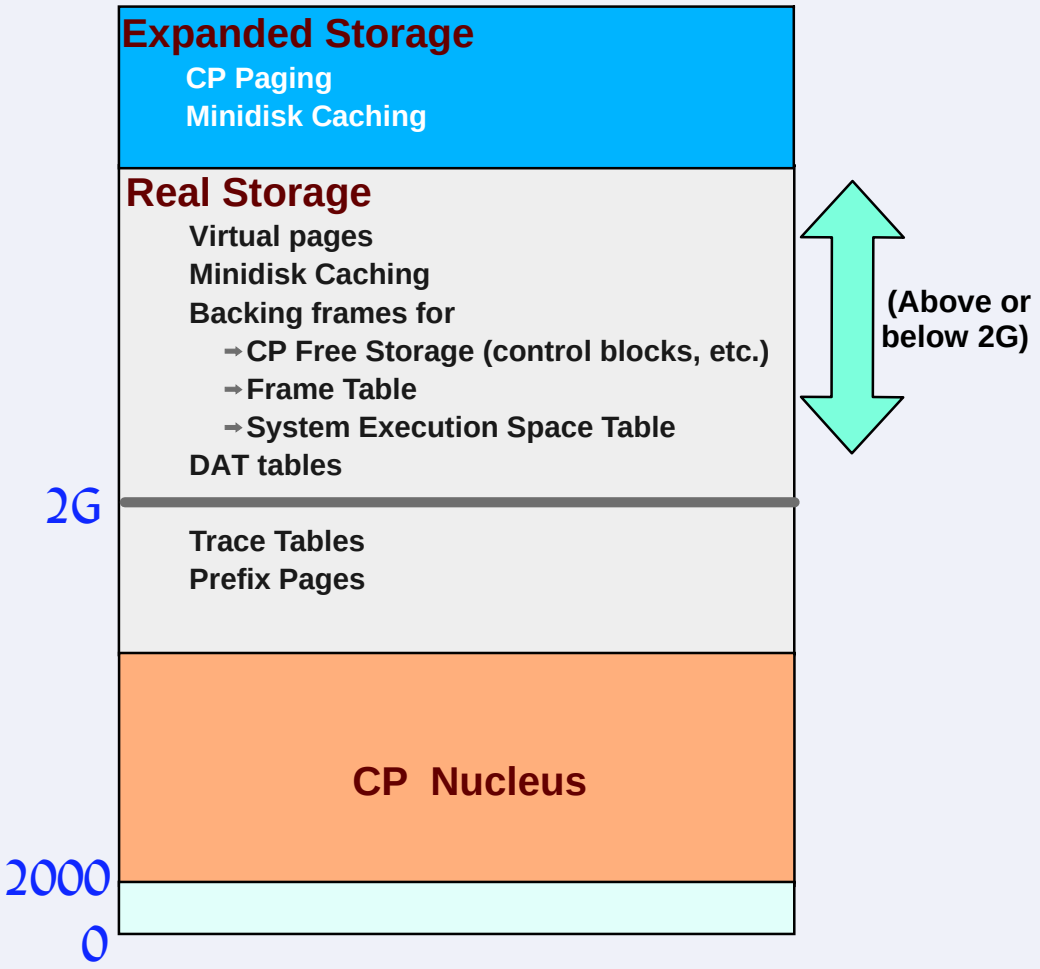
Key Points:

Sharing:
- Read-only
- Read-write
- Security knobs

Uses:
- Common kernel
- Shared programs

More: Layout of Real Storage

z/VM 6.1



How: Memory Management

- VM
  - Demand paging between central and expanded
  - Block paging with DASD (disk)
  - Steal from central based on LRU with reference bits
  - Steal from expanded based on LRU with time stamps
  - Paging activity is traditionally considered normal
- LPAR
  - Dedicated storage, no paging
- Linux
  - Paging to "swap disks" (aka "swap extents")
  - Traditionally considered bad

I/O Resources

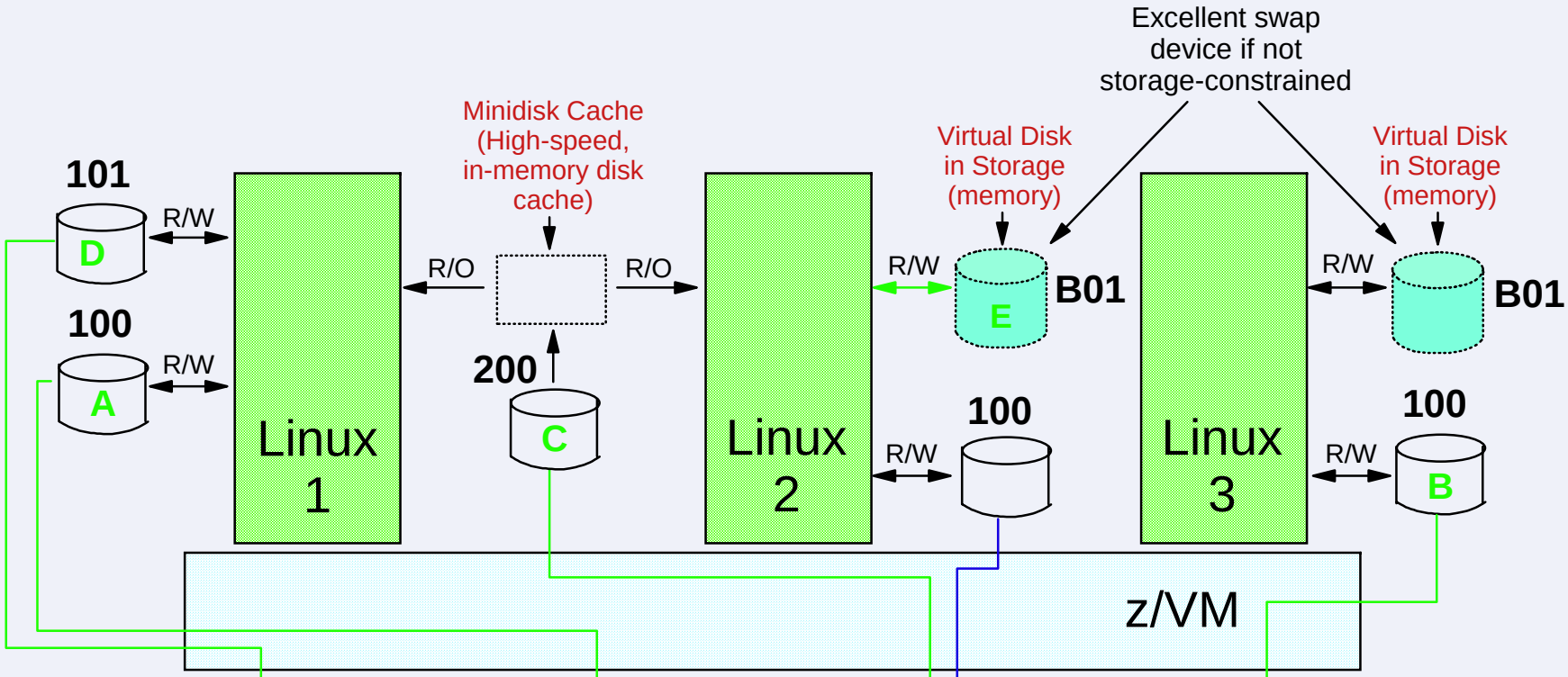
What: Device Management Concepts

- *Dedicated* or *attached*
  - The guest has exclusive use of the entire real device
- *Virtualized*
  - A slice (in time or in space) of a real device
  - E.g., DASD or crypto
- *Simulated*
  - No real hardware (all smoke and mirrors)
  - Virtual disks, virtual NICs, virtual channel-to-channel adapters (CTC)
- *Emulated*
  - Provide a device of one type using real hardware of a different type
  - E.g., CP can emulate FBA using SCSI LUNs

What: Device Management Concepts

- Vocabulary
  - *RDEV:* real device, or control block representing it
  - *VDEV:* virtual device, or control block representing it
  - *Subchannel:* Hardware-managed control block representing device in I/O operations
- Controls and Limits
  - Indirect control through share setting
  - Real devices can be "throttled"
  - Channel priority can be set for a virtual machine
  - MDC fair share limits (can be overridden)

What: Virtualization of Disks



TDISK: on-the-fly disk allocation pool

Minidisk: z/VM disk allocation technology

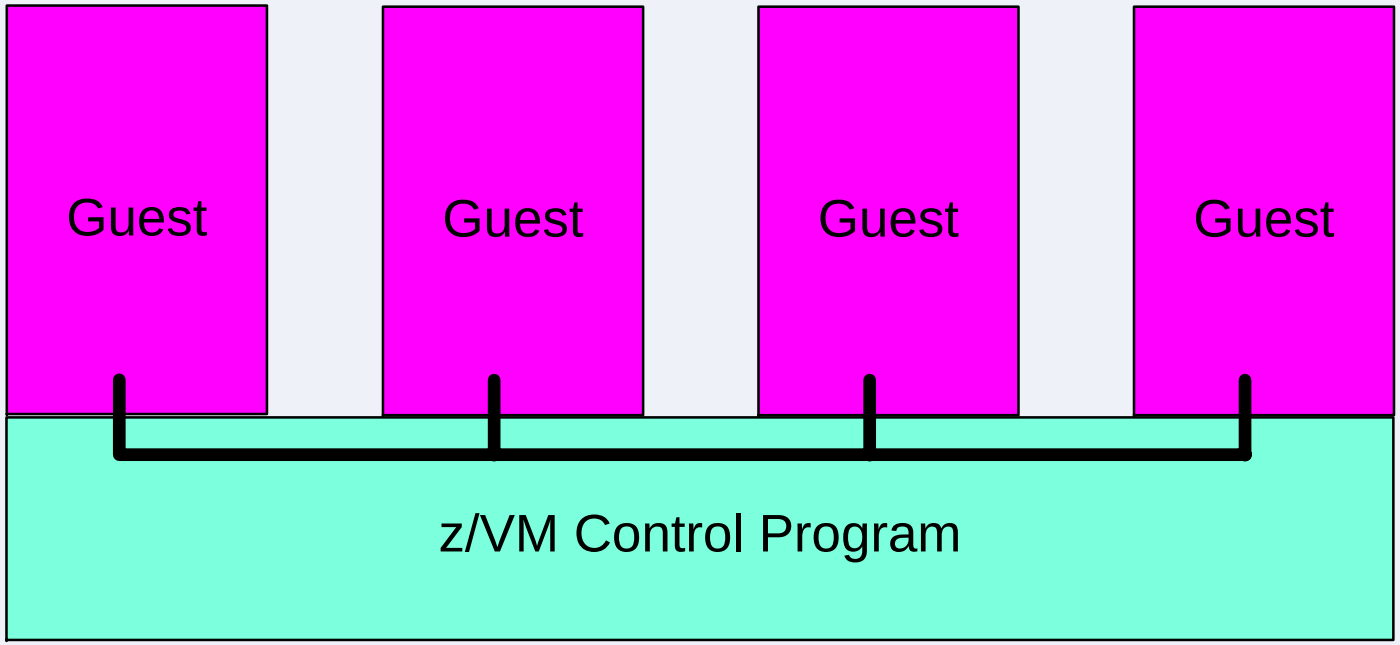
Notes:
R/W = Read/Write
R/O = Read Only

What: Data-in-Memory

- Minidisk Cache (MDC)
  - Caches reads of non-dedicated disks (minidisks)
  - Write-through cache
  - Great performance
  - Lots of tuning knobs
- Virtual Disk in Storage (VDISK)
  - Like a RAM disk, but pageable
  - Volatile (of course)
  - Appears to be an FBA disk
  - Can be shared among virtual machines
  - Plenty of knobs here too

Networking

What: Virtual Networking



- Create a simulated LAN segment, called a *guest LAN,* inside z/VM
- Give the guest a simulated network adapter, called a *guest LAN adapter*
- *Couple* the fake adapter to the fake LAN
- At this point the guest is on a network

Thinking Outside the Box

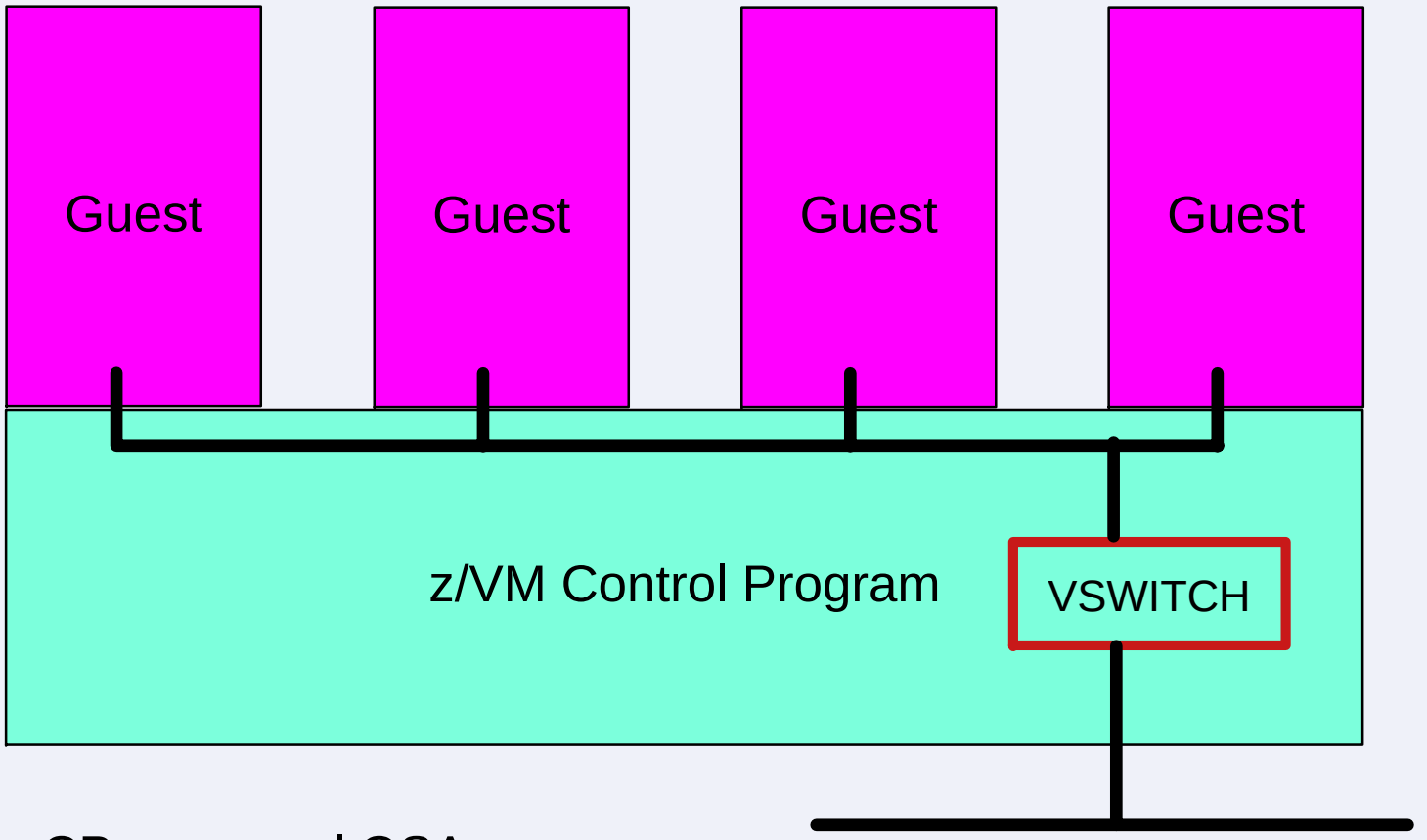
- System z does have real networking hardware
  - *Open Systems Adapter (OSA):* a genuine LAN adapter
  - *HiperSocket:* a LAN adapter that talks only from one partition to another
  - *Channel-to-channel adapter (CTCA):* a point-to-point link between System z machines

- Who's going to own the real hardware?

  - The z/VM Control Program: the *Virtual Switch*
    - CP itself owns a real Open Systems Adapter
    - CP does the data switching functions
      - Layer 2 (Ethernet frames) or layer 3 (IP packets) switch
    - The guests are effectively on the external network

  - A guest: a *virtual machine router*
    - A specific guest owns a real networking device (OSA, HiperSocket, CTCA)
    - Said guest is also coupled to one or more guest LANs
    - This guest performs IP packet routing functions
    - IP subnet boundaries
    - Linux guest or VM TCP/IP stack

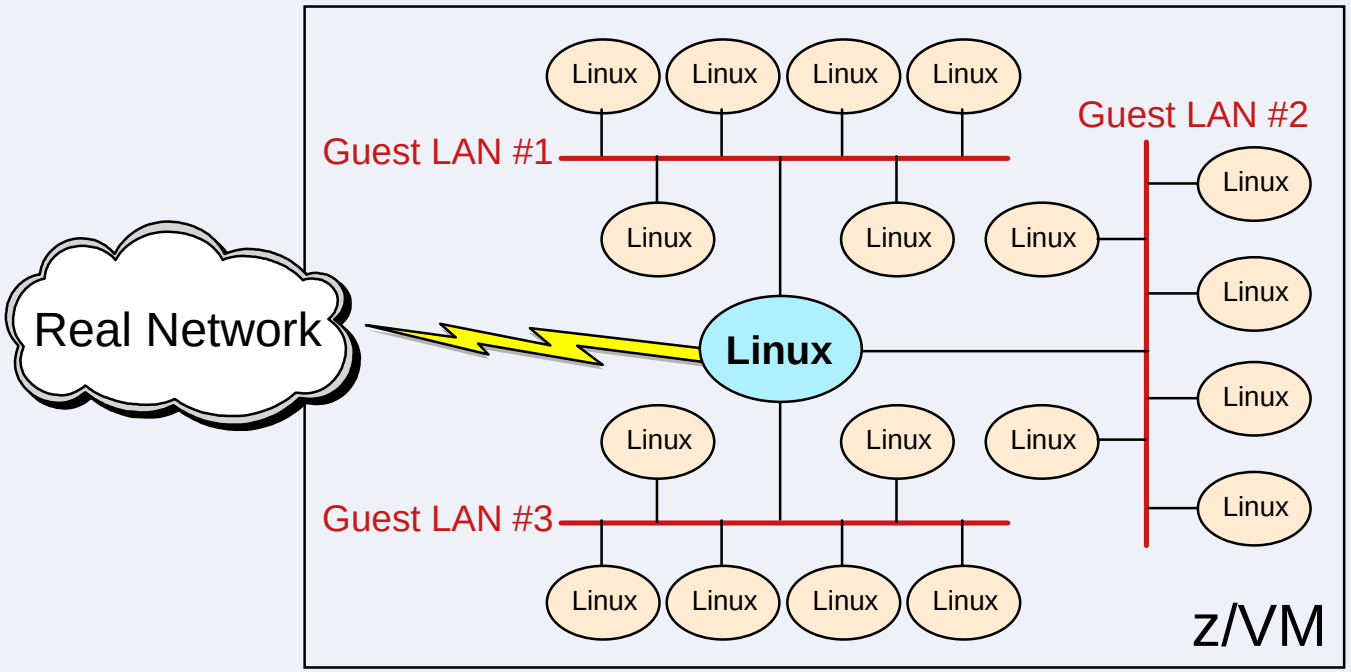
What: Virtual Switch



- CP owns real OSA
- Guests are on the external LAN

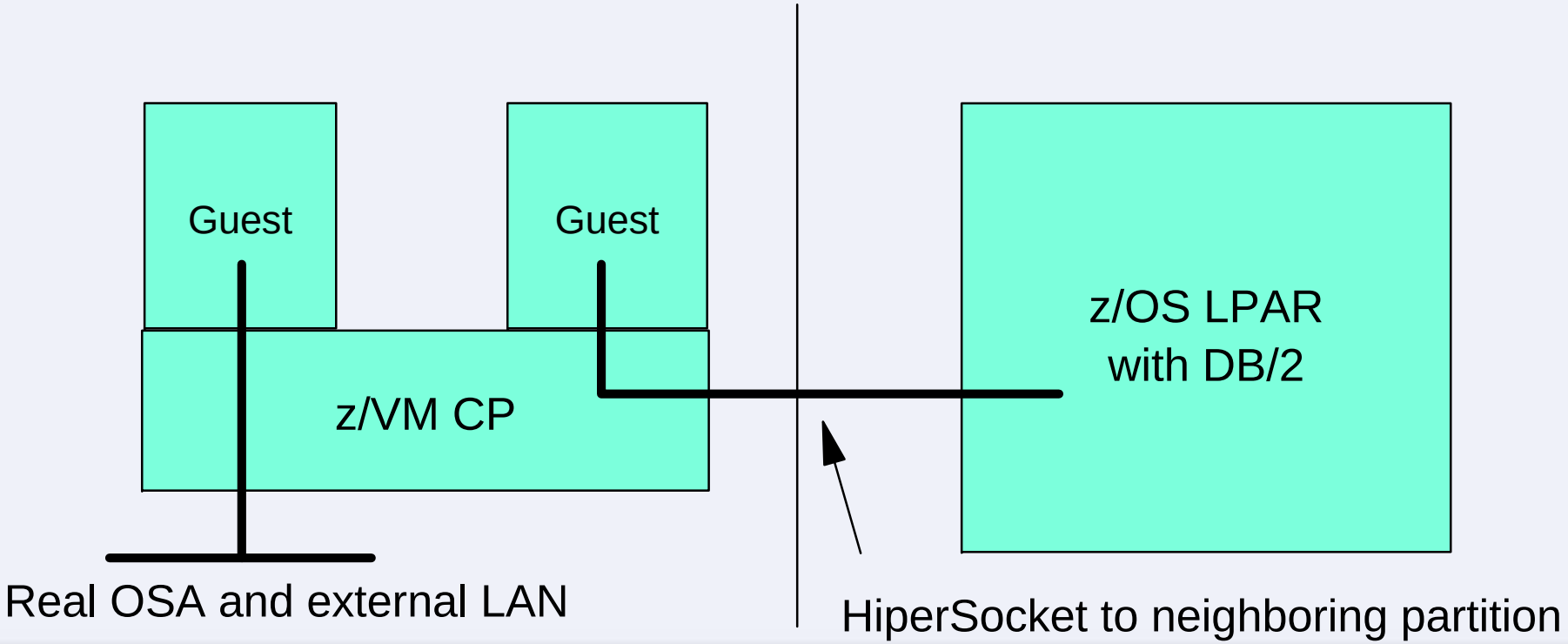
What: Virtual Machine Router

- One or more guest LANs
  - Each guest LAN is its own IP subnet
- Anointed guest is IP router
  - Is coupled to all the guest LANs
  - Also owns a piece of real network gear (OSA or HiperSocket, usually)



There's Always Real Hardware

- Dedicate or attach a real OSA to the guest
- Dedicate or attach a real HiperSocket to the guest
- Dedicate or attach a real CTCA to the guest



Beyond Virtualization

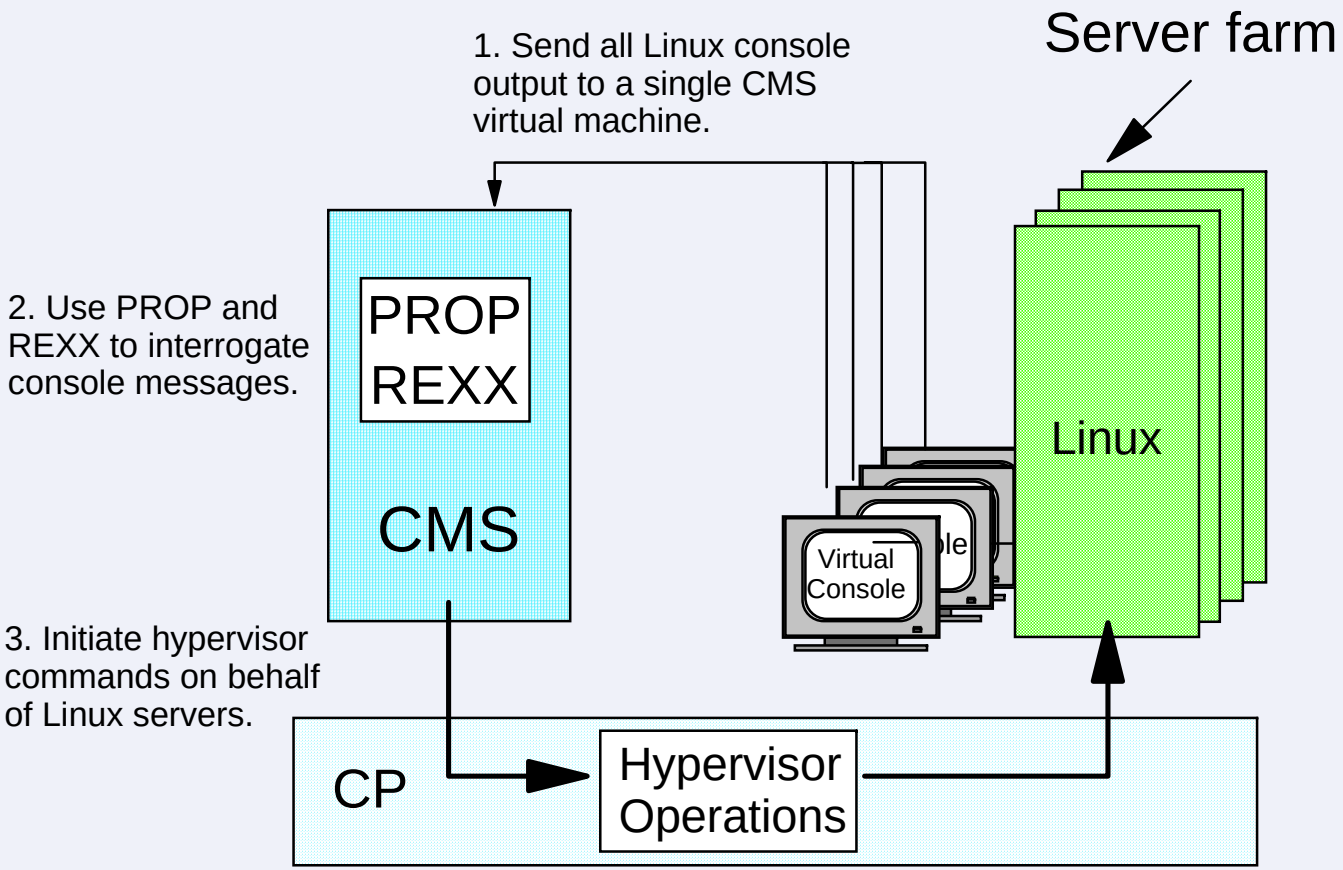
What: Other Control Program Interfaces

- **Commands**
  - Query or change virtual machine configuration
  - Debug and tracing
  - Commands fall into different *privilege classes*
  - Some commands affect the entire system
- **Inter-virtual-machine communication**
  - Connectionless (VMCF) or connection-oriented (IUCV) protocols
  - These pre-date TCP/IP, guest LANs, etc.
- **System services**
  - Guest has an enduring connection to CP via IUCV
  - Guest and CP exchange information on the IUCV connection
  - Guest is in a long-term relationship with CP (performance, accounting, security)
- **Diagnose instructions**
  - These are really programming APIs (semantically, procedure calls)
  - Large number of "entry points" provides many callable functions for guests

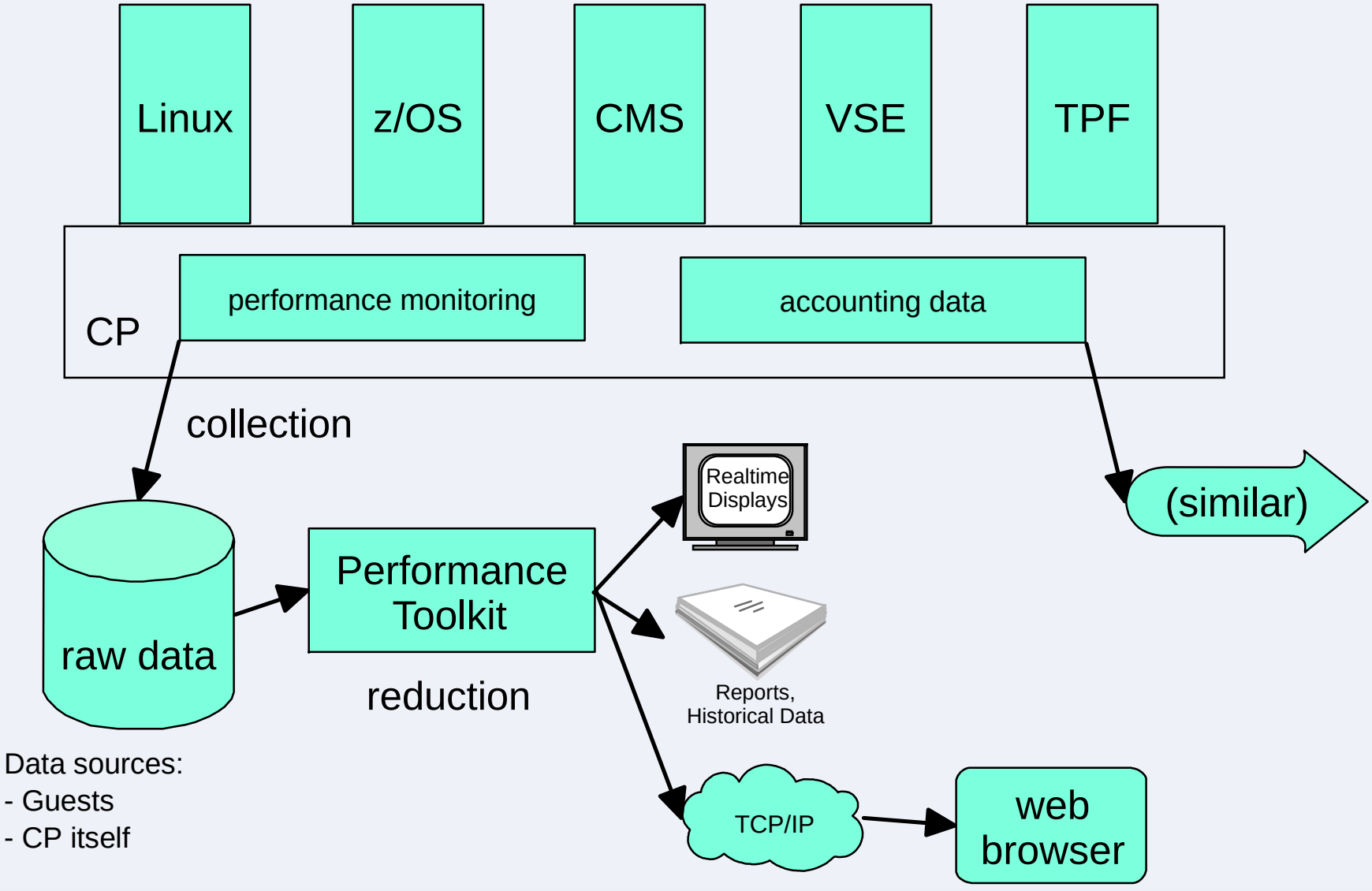
What: Debugging a Virtual Machine

- Tracing a virtual machine's execution
  - CP TRACE command: can trace
    - Instructions (specific opcodes, specific addresses)
    - Storage references or alterations
    - Register alterations
    - Use of various address spaces
    - ... on and on and on ...
  - Single-step through execution, or run and collect trace to a spool file
  - Trace points can trigger other commands
- Display or store into guest memory
  - Helpful especially in conjunction with tracing
  - Can select address space to display or alter
  - Options for translation: ASCII, EBCDIC, System z opcodes (disassemble)
  - Locate strings in memory

What: Programmable Operator



What: Performance and Accounting Data



References

- VM web site: www.vm.ibm.com
- Publications there:
  - www.vm.ibm.com/pubs/
  - Follow the links to the latest z/VM library
  - Of particular interest:
    - z/VM CP Command and Utility Reference
    - z/VM CP Planning and Administration
    - z/VM CP Programming Services
    - z/VM Performance
- z/Architecture Principles of Operation:
  http://publibz.boulder.ibm.com/epubs/pdf/dz9zr006.pdf
- Materials related to this presentation:
  - Narration: http://www.vm.ibm.com/devpages/bkw/vmbasics.ppt (**161 MB**)
  - z/Journal: http://www.vm.ibm.com/devpages/bkw/vmbasics.pdf (880 KB)
- IBM Systems Journal, vol. 30, no. 1, 1991
    - Good article on SIE
    - http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5387504

End of Presentation

Question and Answer Time