Recommendations for Large-Scale Linux Deployment on LinuxONE

Frederik Hartmann
z/VM Chief Product Owner
frederik.hartmann@de.ibm.com

Trademarks

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries. For a complete list of IBM Trademarks, see www.ibm.com/legal/copytrade.shtml: AS/400, DBE, e-business logo, ESCO, eServer, FICON, IBM, IBM Logo, iSeries, MVS, OS/390, pSeries, RS/6000, S/30, VM/ESA, VSE/ESA, Websphere, xSeries, z/OS, zSeries, z/VM

The following are trademarks or registered trademarks of other companies

Java and all Java-related trademarks and logos are trademarks of Sun Microsystems, Inc., in the United States and other countries
LINUX is a registered trademark of Linus Torvalds
UNIX is a registered trademark of The Open Group in the United States and other countries.
Microsoft, Windows and Windows NT are registered trademarks of Microsoft Corporation.
SET and Secure Electronic Transaction are trademarks owned by SET Secure Electronic Transaction LLC.
Intel is a registered trademark of Intel Corporation
* All other products may be trademarks or registered trademarks of their respective companies.

NOTES:

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

References in this document to IBM products or services do not imply that IBM intends to make them available in every country.

Any proposed use of claims in this presentation outside of the United States must be reviewed by local IBM country counsel prior to such use.

The information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

Permission is hereby granted to SHARE to publish an exact copy of this paper in the SHARE proceedings. IBM retains the title to the copyright in this paper, as well as the copyright in all underlying works. IBM retains the right to make derivative works and to republish and distribute this paper to whomever it chooses in any way it chooses.

© 2026 IBM Corporation

Contents

© 2026 IBM Corporation

Recommended Setup

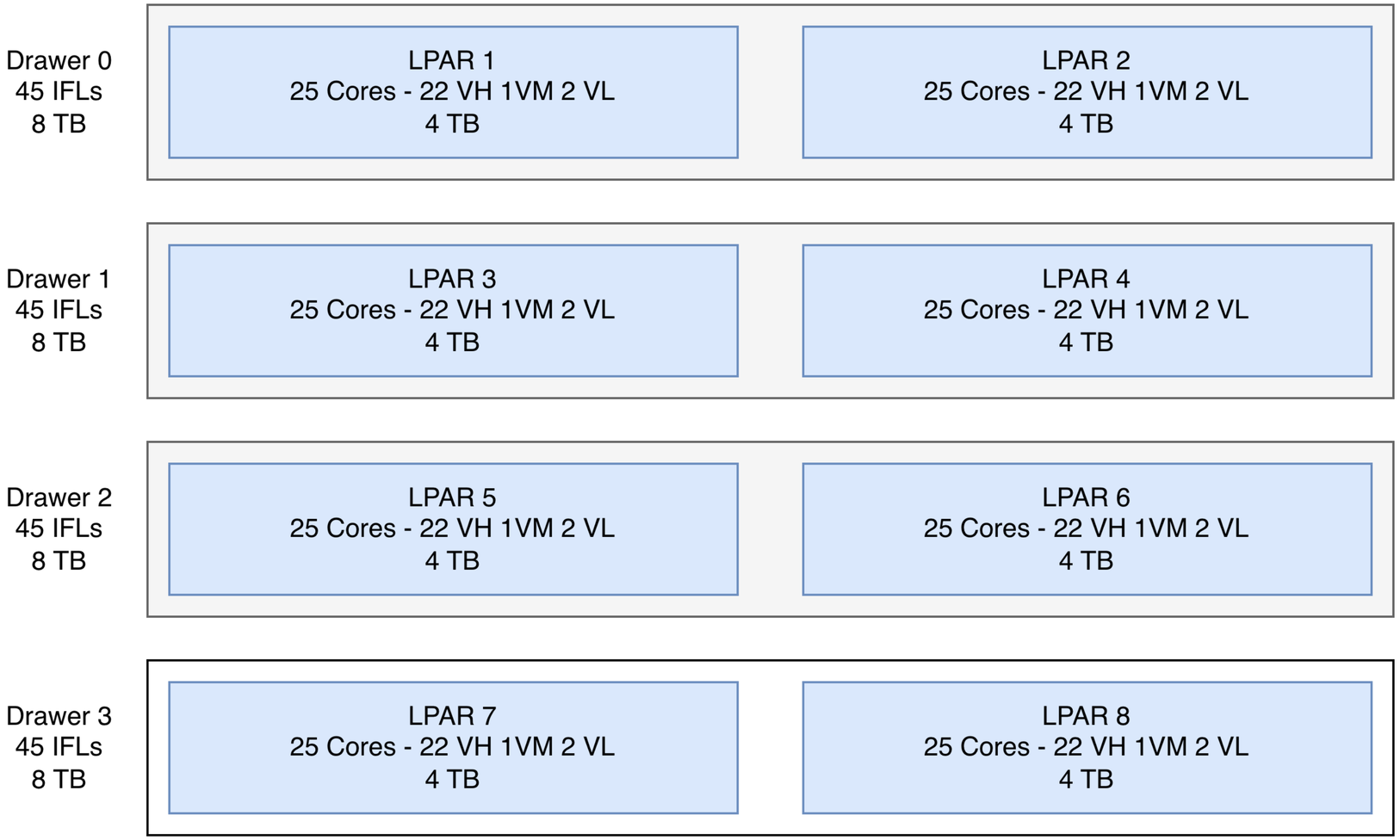
© 2026 IBM Corporation

LPAR Configuration

Max183 CEC
180 IFLs
32TB Memory

- LPARs with identical CPU/Memory proportion
- Split drawers into two or four LPARs

- Don’t change LPAR size, relocate workload



© 2026 IBM Corporation

z/VM Configuration

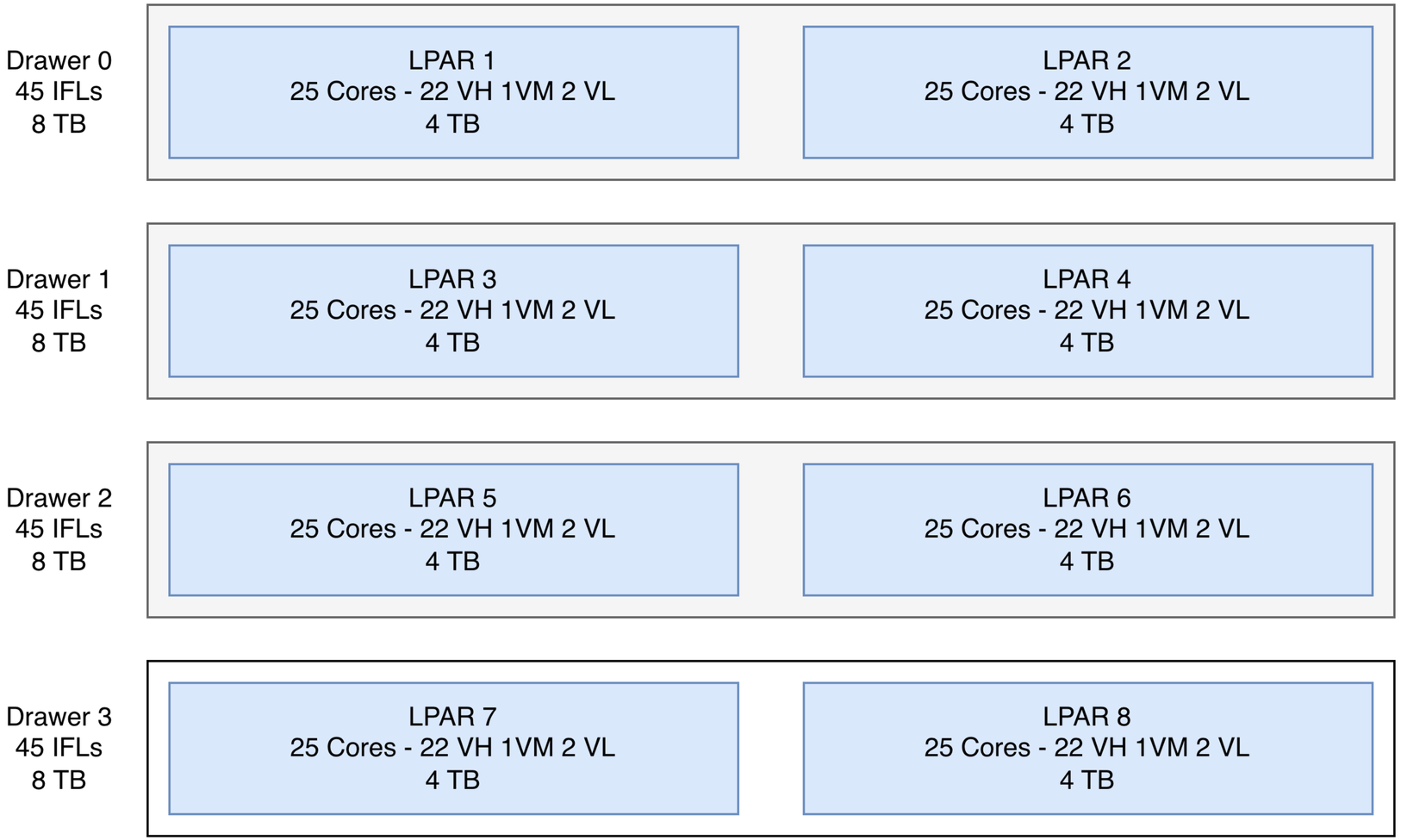
- DASDs for z/VM
- FCP Passthrough for Linux VMs

- MVLAG vSwitch for smaller Linux VMs
- PCIe Passthrough for network for workload consolidators like OCP

- SMT-2 should be enabled by default

- Memory Overcommit should not exceed 1:1.3
- Total CPU overcommit should not exceed 1:10

- The total workload compute requirements should not exceed 90% of LPAR capacity

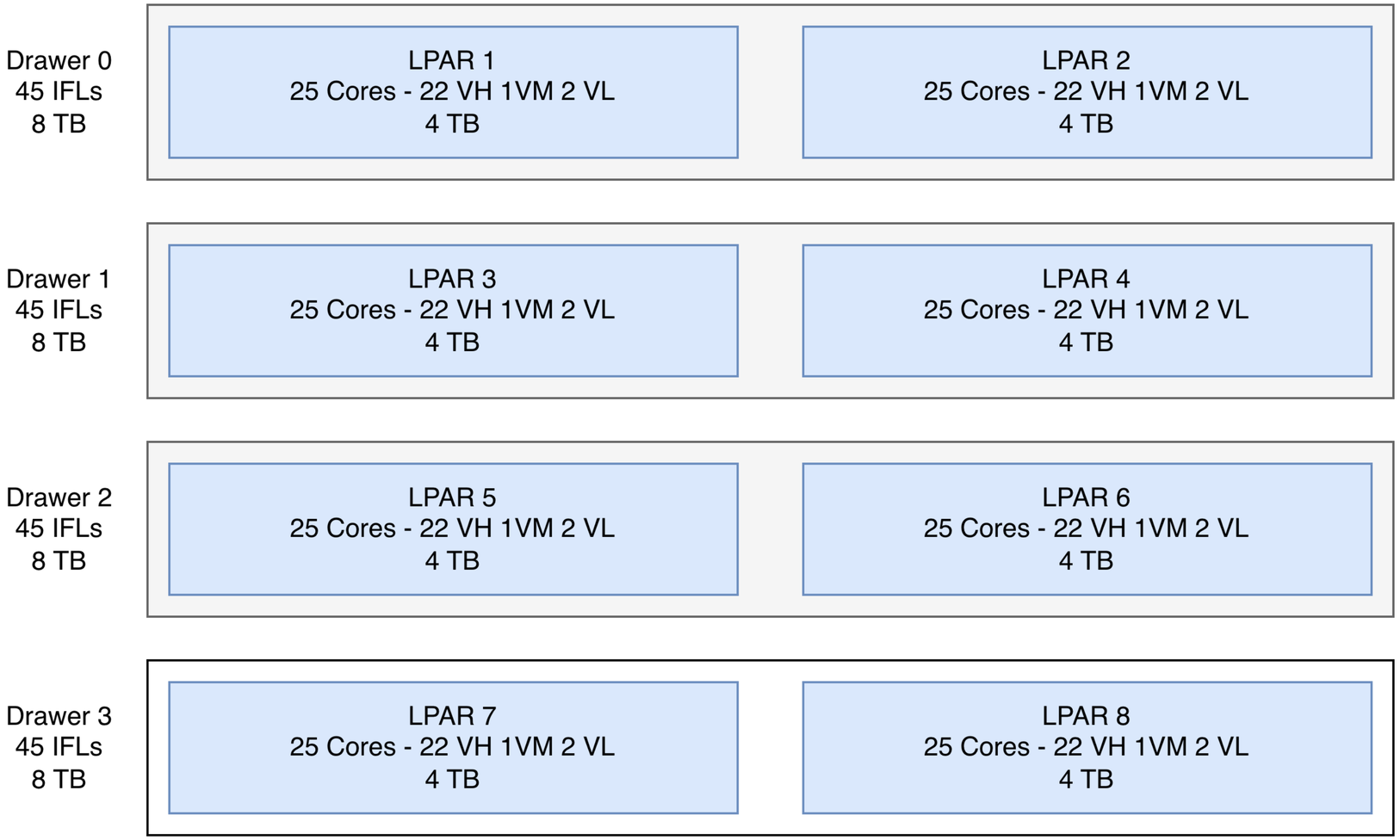


© 2026 IBM Corporation

VM Configuration

- Enough vCPUs to cover workload spikes
- Enough entitlement to cover average workload
- Enough memory to avoid swapping

- Overallocation of vCPUs and memory is hurting performance!



© 2026 IBM Corporation

Overview & Terminology

© 2026 IBM Corporation

Critical Terminology

CEC (Central Electronics Complex) or CPC (Central Processor Complex)

This is the physical machine that consists of the CPUs, memory, PCIe cards and everything else physical that is necessary to build a working mainframe.

This can be called the physical layer, managing and controlling physical devices.

LPAR (Logical Partition)

This is a partition created by PR/SM (Processor Resource/ Systems Manager), a partitioning engine that allows the creation of up to 85 partitions on one CEC.

This can be called the logical layer, providing logical resources to an operating system running in it.

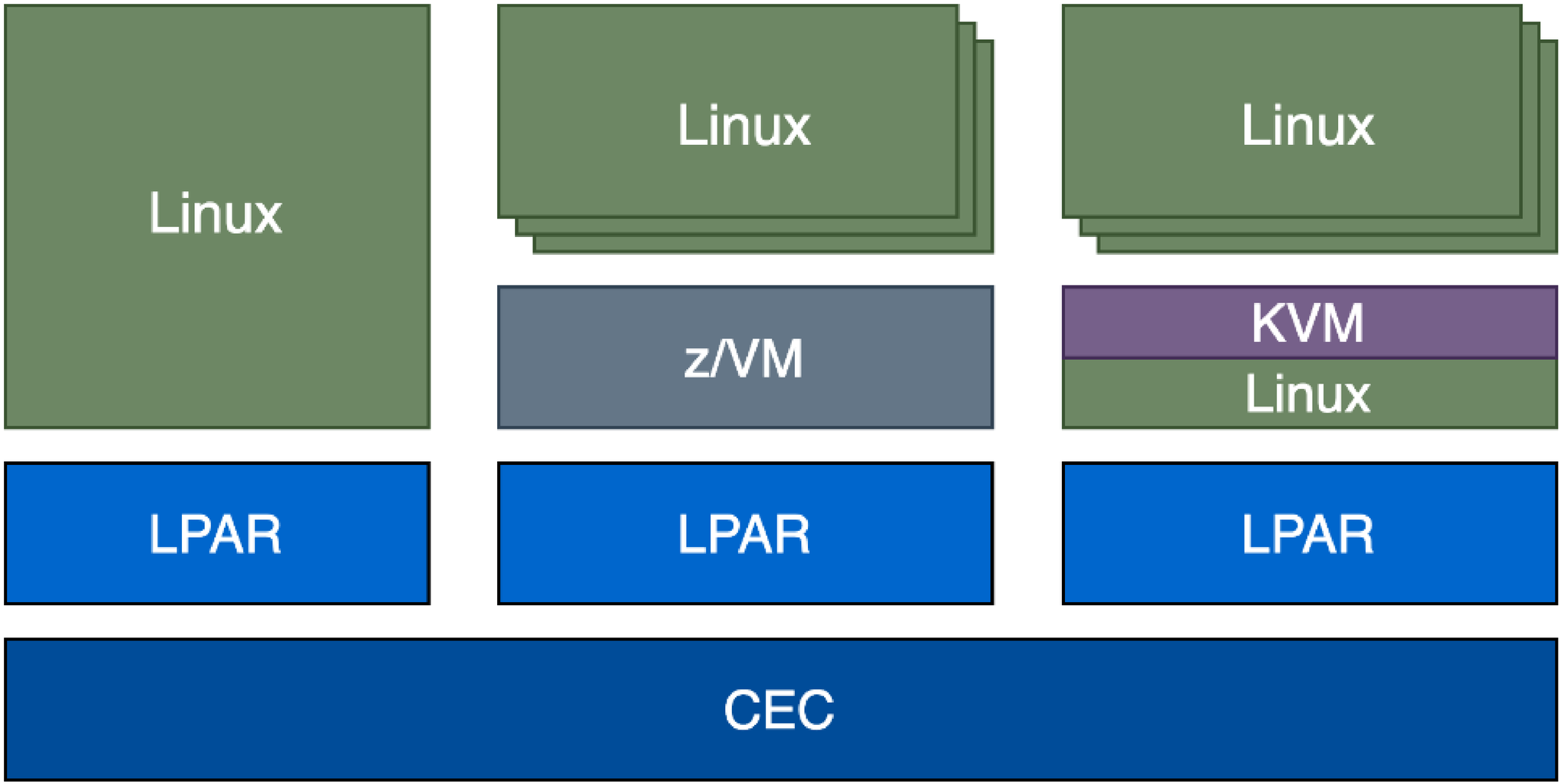
Hypervisor

This optional layer allows the creation and management of virtual machines within one partition, sharing processors, memory and I/O.

This can be called the virtualization layer, providing virtual devices to virtual machines.

© 2026 IBM Corporation

Layer Visualization



Physical Cores

A distinct portion of hardware that executes the z/Architecture® instruction set.

Each physical core in a IBM® LinuxONE system is designed to support two independent instruction streams (“hardware threads”) when operating in SMT-2 mode.

A physical core can be shared or dedicated to one LPAR.

Logical Cores

A logical partition (LPAR) is equipped with a certain number of logical cores. Based on the LPAR definition, these are either dedicated (physical cores assigned for the exclusive use of this LPAR) or shared (dynamically mapped onto a pool of shared cores).

Logical cores are dispatched on physical cores by PR/SM.

Logical Processors

When Simultaneous Multithreading (SMT-2) is active for an LPAR, the operating system or hypervisor running in that LPAR sees each logical core as two logical processors, each of which can execute a z/Architecture program.

When SMT-2 is not active, each logical core in the LPAR acts as a single logical processor.

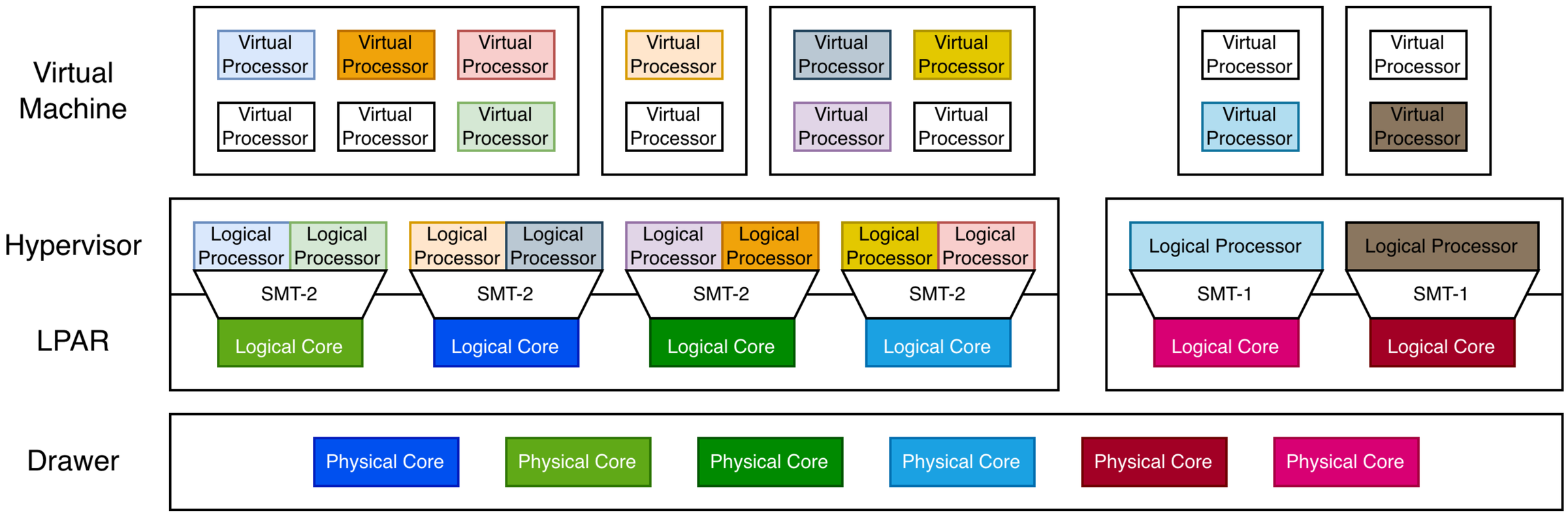
Virtual Processors

A virtual machine (VM) is equipped with virtual processors (vCPUs) defined by its definition in the hypervisor.

vCPUs are dispatched on logical processors by the hypervisor.

© 2026 IBM Corporation

Layer Visualization



Weights, Shares & Entitlement

LPAR Weights

Defines the effective size of an LPAR in relation to all other running LPARs.

They are configured in the HMC and can be dynamically changed.

Significant changes to LPAR weights should not be done too frequently, every change can trigger a replacement of the running LPARs.

z/VM LPARs should be configured in a way to limit the number of VL CPUs to a maximum of 3.

z/VM Shares

Relative shares define the effective size of a virtual machine in relation to all other running virtual machines.

Absolute shares define the effective size of a virtual machine in relation to the size of the LPAR.

Primarily relative shares should be used due to easier management.

Entitlement

Represents the absolute resource entitlement of a partition/virtual machine.

A partition/virtual machine can consume more or less than its entitlement, depending on free capacity on the hypervisor.

Unused entitled capacity can be used by other partitions/virtual machines on the same CEC/hypervisor.

Calculated based on the weights/shares as well as all available system resources.

© 2026 IBM Corporation

Relationship between vCPUs and Entitlement

Entitlement

- Entitlement should be aligned with the expected performance requirements of a virtual machine.
- If an important virtual machine is utilizing 4 logical cores during peak utilization, it should have sufficient shares to result in an entitlement of these 4 logical cores.

- **Entitlement becomes critical if the z/VM LPAR becomes CPU constrained.**

vCPUs

- The number of vCPUs define the theoretical maximum of compute power a virtual machine can use.
- Every vCPU will be visible as a core to the guest operating systems and will require management, monitoring and upkeep.
- A high number of vCPUs lead to an unnecessary number of context switches, loss of cache locality and virtualization overhead.

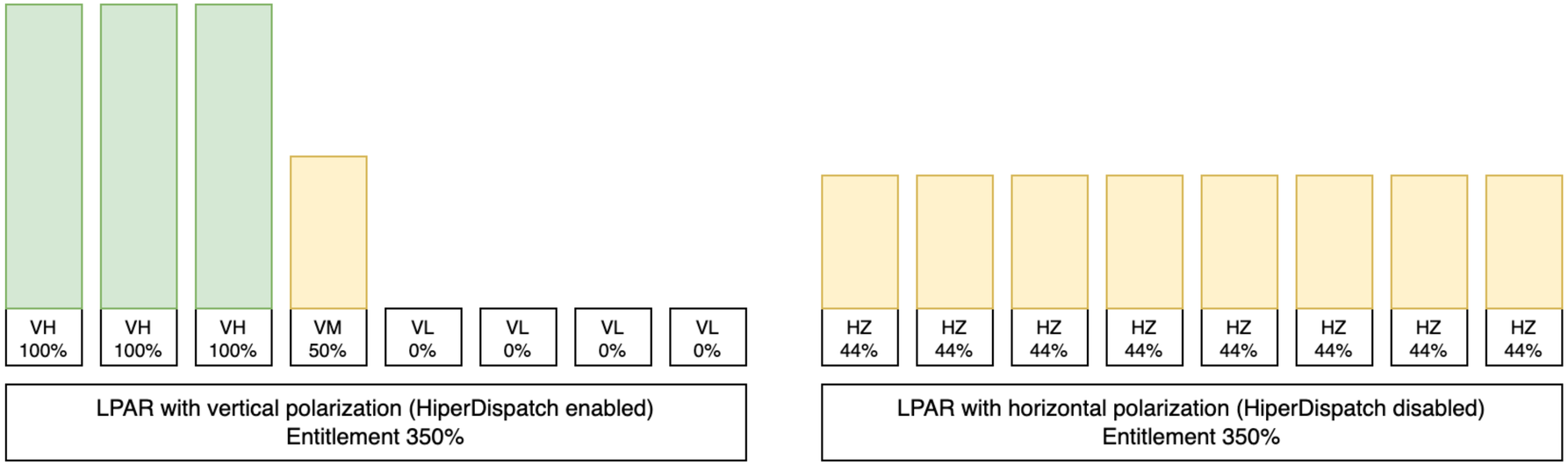
- **The number of vCPUs should be as small as possible and as large as necessary.**

© 2026 IBM Corporation

HiperDispatch

Each LPAR is configured by the operating system to be dispatched by PR/SM either in horizontal or vertical polarization. This configuration is controlled by the operating system running in the LPAR, comparable to the usage of SMT.

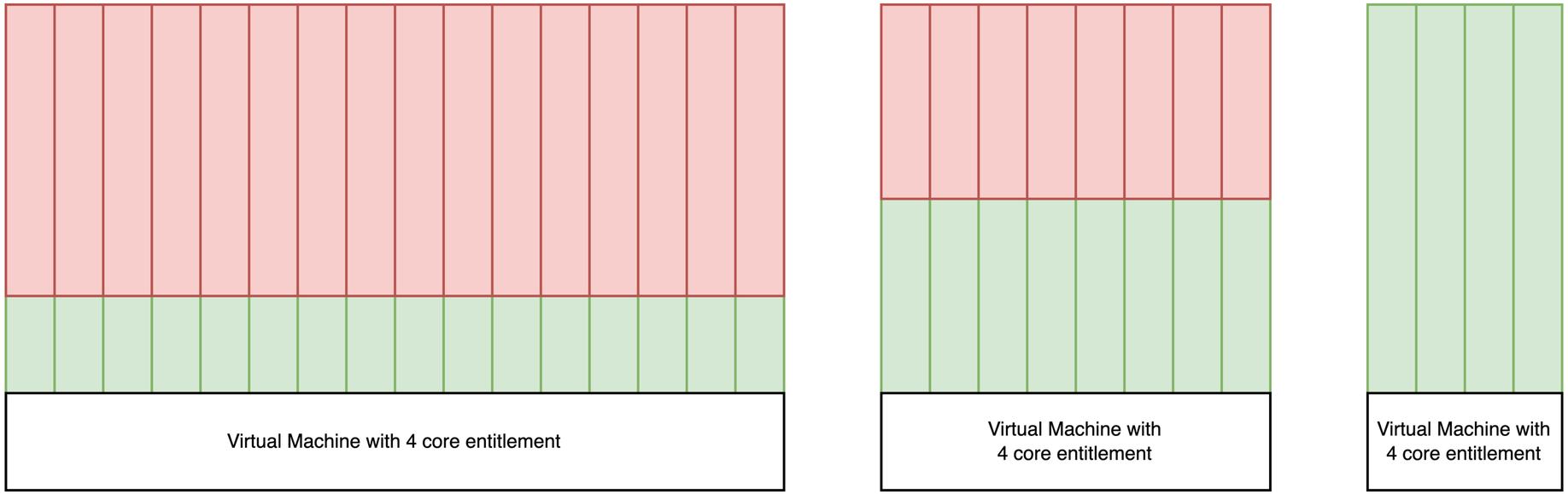
Utilizing vertical polarization is recommended, especially when the total number of logical cores across all shared-processor LPARs significantly exceeds the number of physical cores in the pool of shared cores in the CEC.



© 2026 IBM Corporation

Effective usage of Entitlement

Entitlement and the number of vCPUs should be in a healthy relationship to minimize overhead while providing enough free space to absorb unexpected workload spikes.

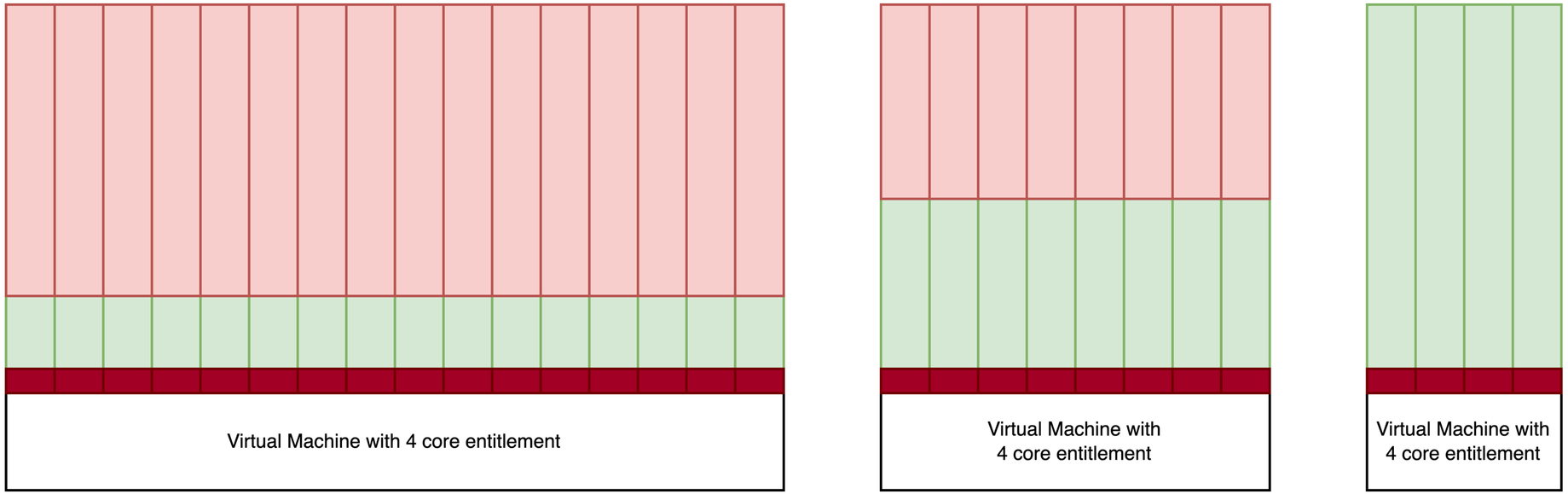


© 2026 IBM Corporation

Overhead & vCPUs

Overhead is disproportionally bad for high vCPU low entitlement environments.

Low entitlement and a high number of vCPUs can create scenarios where regularly vCPUs get undispatched while holding a lock, leading to stalling applications and a high DIAG 9C rate.



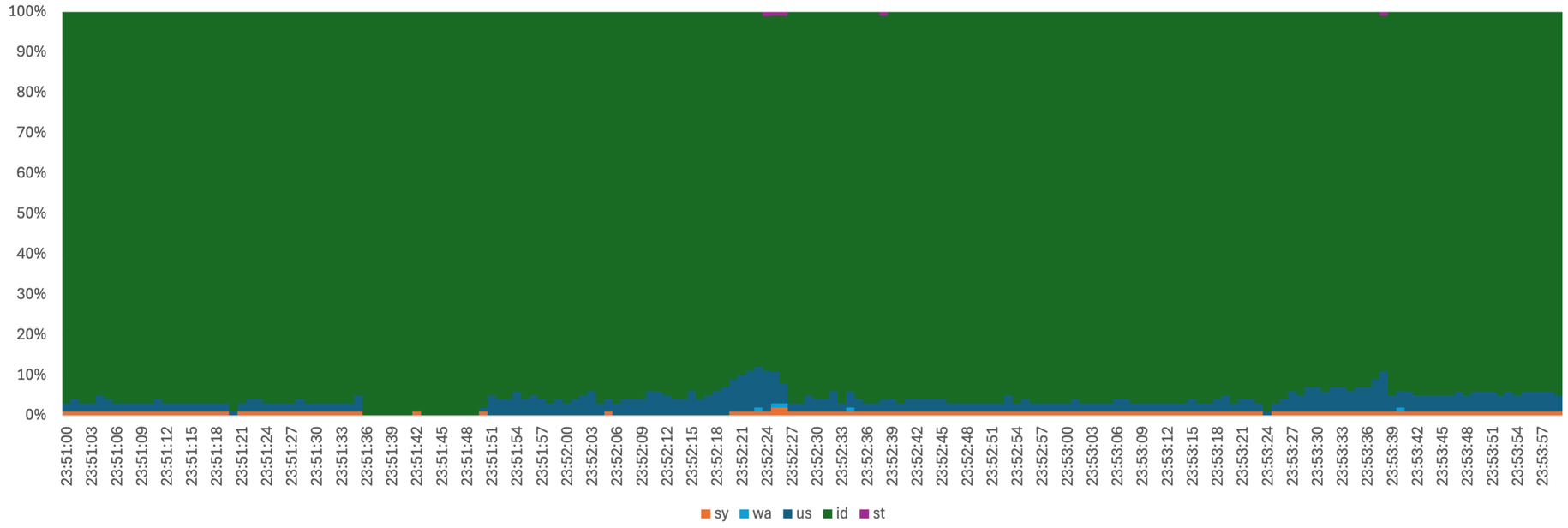
© 2026 IBM Corporation

Low Usage per vCPU

Entitlement and the number of vCPUs should be in a healthy relationship to minimize overhead while providing enough free space to absorb unexpected workload spikes.

The example on the right should have its vCPU count reduced by 50% as a first step.

A good starting point is to observe the high-water mark of average CPU consumption during an interval of the length of the SLA.



© 2026 IBM Corporation

Example Calculation

Shared physical cores: **60**

Name	Cores	Weight	Polarity	Entitlement	# VHs	#VMs or HZs	Ent(VM or HZ)	#VLs	Unusable Ent
PROD 1	15	13	V	1300.0	12	1	100.0	2	0.0
PROD 2	15	13	V	1300.0	12	1	100.0	2	0.0
PROD 3	15	13	V	1300.0	12	1	100.0	2	0.0
PROD 4	15	13	V	1300.0	12	1	100.0	2	0.0
DEV 1	15	4	V	400.0	3	1	100.0	11	0.0
DEV 2	15	4	V	400.0	3	1	100.0	11	0.0
Totals ->	90	60		6000.0	54	6		30	0.0

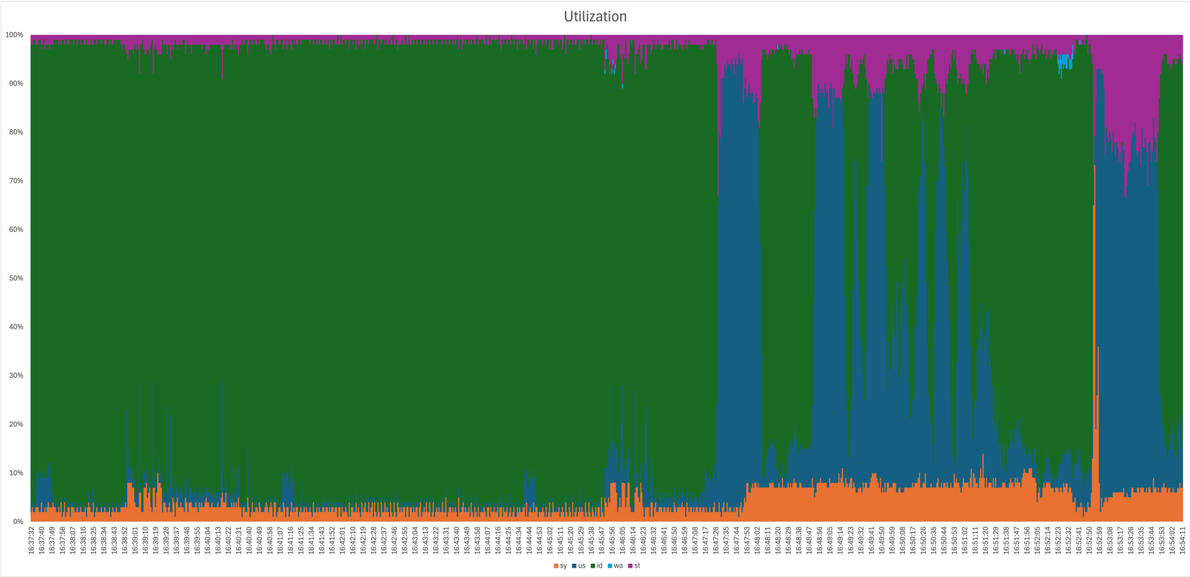
https://www.ibm.com/support/pages/zvm/perf/tips/lparcalc.html

© 2026 IBM Corporation

Linux CPU Usage & Steal

© 2026 IBM Corporation

Linux CPU Measurements



Linux CPU Usage Types

- CPU usage in Linux can be obtained using various tools like top, vmstat, dstat ...

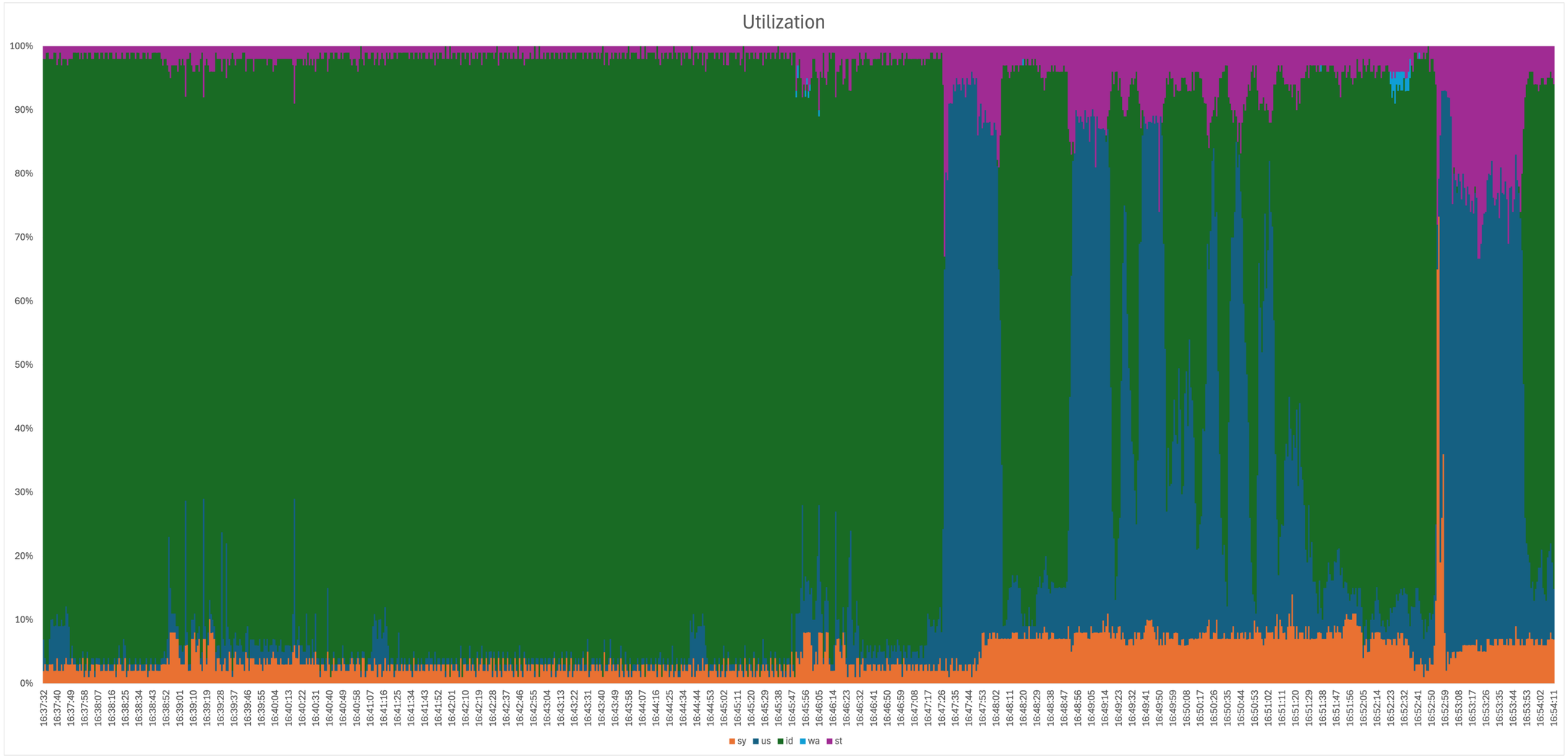
- user time:
  Time spent running non-kernel code.
- system time:
  Time spent running kernel code.
- idle time:
  Time spent idle.
- wait time:
  Time spent waiting for IO.
- **steal time:**
  Time the virtual machine process is waiting on the physical CPU for its CPU time.

sar, vmstat & dstat

- Collecting Linux performance data is of critical importance to validate hypervisor performance and virtual machine sizing
- Most important metrics
  - CPU utilization
  - Memory utilization
  - Network IO
  - Disk IO
  - **Application latency & transaction rates**

© 2026 IBM Corporation

Linux CPU usage & steal



© 2026 IBM Corporation

Linux Steal Time

Steal Time is one of the most important performance metrics to track in any virtualized environment.

In simple terms, steal time is the mismatch between CPU time requested by an operating system and the CPU time provided by the environment it is running in.

Steal Time is a symptom of a performance constraint, not the reason for a performance problem.

Steal time is calculated by the operating system and expressed as a percentage of time “stolen” of the total system CPU resources.

E.g. 20% steal time in a virtual machine with 20 vCPUs would represent the capacity of 4 logical processors unavailable due to outside constraints.

This could result in 8 vCPUs being only available 50% of the time or a uniform loss of 20% over all vCPUs.

Reasons for steal time

- CPU Overcommitment
- Memory Overcommitment
- Hypervisor executing on behalf of VM
- Firmware executing on behalf of Hypervisor or VM

Another source of steal time is work performed by PR/SM or the hypervisor that is not attributable to any one partition or virtual machine. This is usually not a significant source of steal time.

© 2026 IBM Corporation

Memory

© 2026 IBM Corporation

Virtual Memory Overview

Overview

- Absolute/Real Memory is managed in 4K blocks, called pages by z/VM.
- The hypervisor is assigning memory to guests using dynamic address translation tables.
- This guest absolute/real memory is a virtual address space owned by the hypervisor.
- Pages in this virtual address space can be resident in memory or pages out to a paging device.
- If a virtual machine is accessing non-resident memory, the hypervisor will recognize a page fault and fetch the page from storage.

Overcommitment

- Paging introduces a significant delay to the virtual machine accessing non-resident memory.
- Memory access time usually below 10ns, best case SSD access time is usually around 250us, more than 25.000 times slower.
- The hypervisor aims to keep all relevant virtual machine memory resident.
- **Optimization only possible for steady state workload, significant shifts in workload patters can cause excessive paging.**
- **Acceptable overcommit ratio decreases with an increase in real memory size.**

© 2026 IBM Corporation

Linux Memory Usage

used

- Used memory is currently assigned to the virtual address space of a process.
- It should constitute most of the system total memory.

free

- This memory is currently unused and does not contribute to the operation or performance of the system in any way.
- **A significant amount of free memory is a strong indication for an oversized system.**

buff/cache

- Memory used in kernel buffers are essential for system operation and usually are not available for other processes.
- Memory used to cache files or data improve the access times should the file/data be needed by a process.

available

- This usually consists of all free memory as well as significant parts of buffer/cache memory.
- The most significant value to monitor to ensure that a system doesn’t run out of memory.

© 2026 IBM Corporation

Summary

© 2026 IBM Corporation

Summary

- The size of virtual machines have a significant impact on the efficient operation of z/VM.
- High CPU or IO overcommit ratios are possible due to the high flexibility and minimal delays in redistributing IO bandwidth or compute power.
- Memory overcommit must be monitored and managed to ensure smooth operation of all virtual machines.
  - 128GB in memory overcommit requires 256GB in IO to complete in a drastic load change.
    This would take more than 2 minutes with more than 2000MB/sec!
- Steal time is a symptom, not a cause!

© 2026 IBM Corporation

Recommendations

- Almost all virtual machines are oversized, it is human nature to request more than needed just in case.
  Start as small as possible and grow if necessary.

- Efficient virtual machine memory usage is preferrable over memory overcommit, especially in data serving environments.

- Application IO is in almost all cases preferable over paging IO, either from the operating system or from the hypervisor.

- Hypervisor memory overcommit should not exceed the sum of “free” Linux memory.

- Decide how many z/VM LPARs should run on a CEC.
  - Usual recommendations: 2 or 4 per drawer depending on available memory or cores.
  - Configure LPAR weight to limit VL cores to at most 3 per LPAR.

- Workload relocation is preferred over LPAR size modifications.

© 2026 IBM Corporation

Things to avoid!

- LPAR configuration that does not allow for LPARs to be placed without one or more LPARs spanning a drawer boundary
- Defining more virtual processors in a single virtual machine than there are logical processors in the LPAR hosting that virtual machine
- Physical core to logical core overcommitment at a ratio greater than 2x
- Entitled processing power of an LPAR significantly less than its defined logical cores
- Memory overcommitment within an LPAR at a ratio greater than 1.3x

© 2026 IBM Corporation