z/VM CPU Pooling and ILMT

Romney White (for John Franciscovich) – romneyw@us.ibm.com

IBM z Systems

Architecture and Technology

SHARE Orlando, August 2015

Trademarks

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries.

BladeCenter*	FICON*	OMEGAMON*	RACF*	System z9*	zSecure
DB2*	GDPS*	Performance Toolkit for VM	Storwize*	System z10*	z/VM*
DS6000*	HiperSockets	Power*	System Storage*	Tivoli*	z Systems*
DS8000*	HyperSwap	PowerVM	System x*	zEnterprise*	
ECKD	IBM z13*	PR/SM	System z*	z/OS*	

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.
Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.
Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.
IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.
ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.
Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.
Linear Tape-Open, LTO, the LTO Logo, Ultrium, and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and
Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
OpenStack is a trademark of OpenStack LLC. The OpenStack trademark policy is available on the OpenStack website.
TEALEAF is a registered trademark of Tealeaf, an IBM Company.
Windows Server and the Windows logo are trademarks of the Microsoft group of countries.
Worklight is a trademark or registered trademark of Worklight, an IBM Company.
UNIX is a registered trademark of The Open Group in the United States and other countries.

* Other product and service names might be trademarks of IBM or other 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.

This information provides only general descriptions of the types and portions of workloads that are eligible for execution on Specialty Engines (e.g., zIIPs, zAAPs, and IFLs) ("SEs"). IBM authorizes customers to use IBM SE only to execute the processing of Eligible Workloads of specific Programs expressly authorized by IBM as specified in the "Authorized Use Table for IBM Machines" provided at www.ibm.com/systems/support/machine_warranties/machine_code/aut.html ("AUT"). No other workload processing is authorized for execution on an SE. IBM offers SE at a lower price than General Processors/Central Processors because customers are authorized to use SEs only to process certain types and/or amounts of workloads as specified by IBM in the AUT.

Agenda

- IBM software pricing methodologies
- Brief review of z/VM scheduling options
- Overview of CPU Pooling in z/VM V6.3
- Update to IBM License Metric Tool (ILMT) 9.0.1
- Software Pricing with CPU Pooling
- Use case examples
- CPU Pooling with IBM z13 and SMT
- Summary and References

z Systems Software Pricing Objectives

- Price-to-value
- Flexibility to run software where it is most efficient
- Capability to predict software charges
- Help with cost of new applications
- Flexibility to pay for software based on workload requirements

Pricing Metrics for z/VM IPLA Products

- z/VM V5 and V6 and certain z/VM related products have pricing based on the number of engines
  - *Engine-based Value Unit* pricing allows for a lower cost of incremental growth with additional engine-based licenses purchased
- Most IBM middleware for Linux is also priced based on the number of engines
  - The number of engines is converted into *Processor Value Units* (PVUs) under the Passport Advantage® terms and conditions
- z/VM 6.3 (with APAR) allows *CPU pooling*
  - *ILMT enhancements* enable using ILMT with pooling

Limiting Single Guests

- Existing LIMITHARD option of SET SHARE command bounds guest processor resource consumption
  - SET SHARE *userid* RELATIVE 2000 ABSOLUTE 40% LIMITHARD
    - RELATIVE 2000 defines entitlement: guest is allotted 20 times as much processor resource as the default (RELATIVE 100) user
    - ABSOLUTE 40% LIMITHARD sets the cap: guest cannot consume more than 40% of the processor resource on the z/VM system (e.g., 2 IFLs in a 5-IFL VM partition)
- Applies to processor resource of type where the guest is dispatched
- Scheduler divides limit evenly among virtual CPUs in a virtual MP
  - Omits stopped vCPUs (e.g., via *cpuplugd*)

Limiting Single Guests …

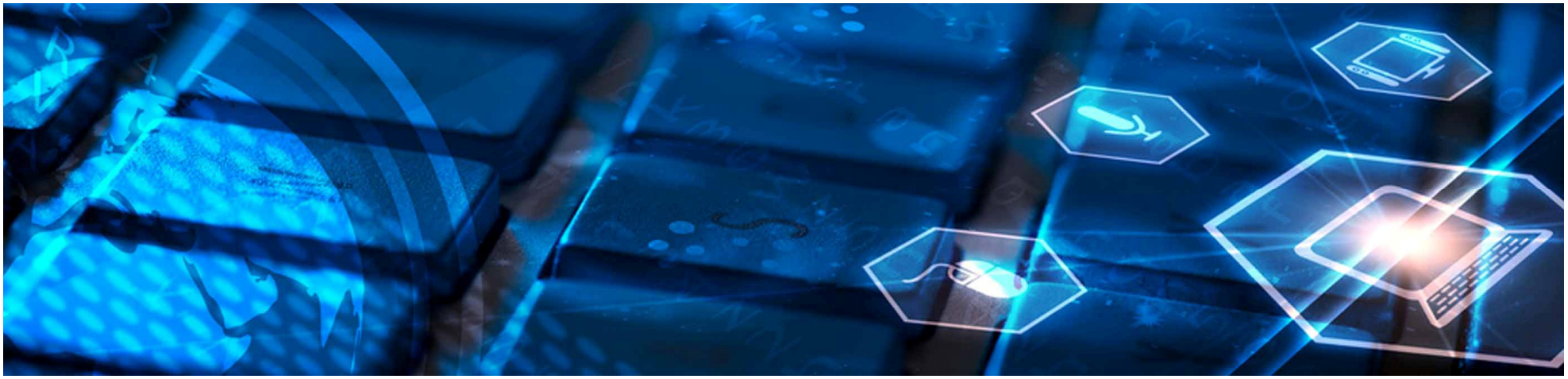
- **SET SHARE LIMITHARD** can be used to
  - Prevent "runaway" virtual machines
  - Limit consumption by less important virtual machines (e.g., test)
  - Help to ensure department budgets are not exceeded
  - Control resources available to contracting clients (service bureau)

- Drawbacks
  - Change in number of logical processors (Capacity on Demand, VARY PROCESSOR ON/OFF) affects actual limit imposed
  - Imposed at the individual guest level
    - Limiting a set of guests may require over-limiting the individuals
  - Not recognized as a means of limiting capacity for IBM sub-capacity software licensing purposes

Environment Information Interface

- New interface allows guest to capture execution environment
  - Processor configuration and capacity information
  - Various Levels: Machine, logical partition, hypervisor, virtual machine
- New unprivileged instruction Store Hypervisor Information (STHYI)
- Includes support for CPU Pooling
- Exploited by ILMT 9.0.1 for sub-capacity pricing of Linux on System z middleware
- Support details:
  - z/VM 6.3 with APAR VM65419 (included in RSU 1501)

CPU Pooling with z/VM V6.3



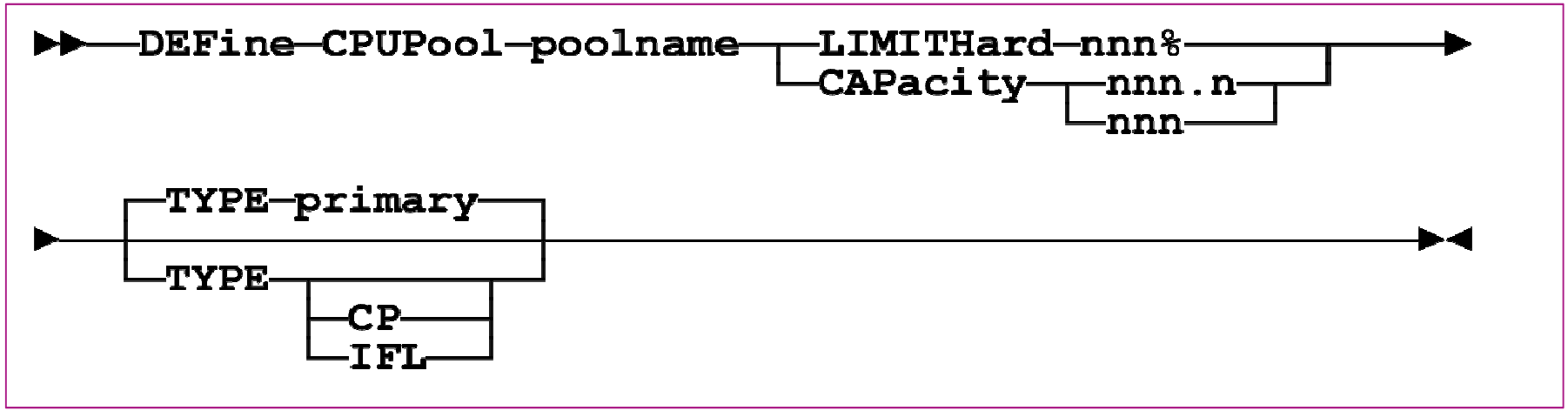
- Create a pool of processor resources available for a group of virtual machines in a z/VM system
- Allows capping of processor utilization for a set of guests to better balance resource utilization
- Allows Live Guest Relocation (LGR) as long as both definitions are compatible
  - Pools are defined and managed independently on each SSI member system
- Available with z/VM V6.3 and APAR VM65418 (in RSU 1501)

Flexible Pool Configuration

- Define named CPU pools with associated capacity
  - Number of CPUs of particular type (CP, IFL)
  - Percentage of CPUs of particular type
- Associate guests with CPU pools
- Limit aggregate guest consumption to pool capacity
  - Coexists with individual guest LIMITHARD setting; both limits enforced
  - Otherwise, resource allotted to group members on demand (“first come, first served”)
- Allows overcommit – no restriction on number of pools or aggregate capacity
- New Environment Information Interface obtains pool capacity information
  - Eliminates manual configuration of data collection

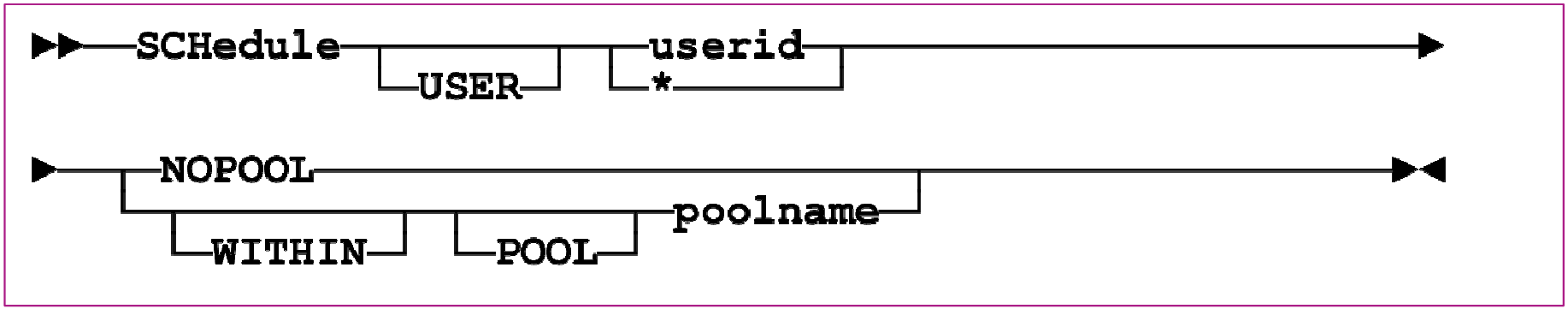
Defining CPU Pools

- Use the DEFINE CPUPOOL command to define named pools
  - Define for a particular TYPE of core (CP or IFL)
    - Default is primary core type (IFL in an IFL-only partition, otherwise CP)
  - CAPACITY – number of CPUs' worth of processing power
    - Limit recognized for sub-capacity licensing purposes
    - Can overcommit (i.e., Sum of CPUPOOL CPUs > Logical processors)
  - LIMITHARD – % of system CPU resources of that type
    - Same enforcement mechanism as SET SHARE LIMITHARD
    - Does not qualify for sub-capacity licensing



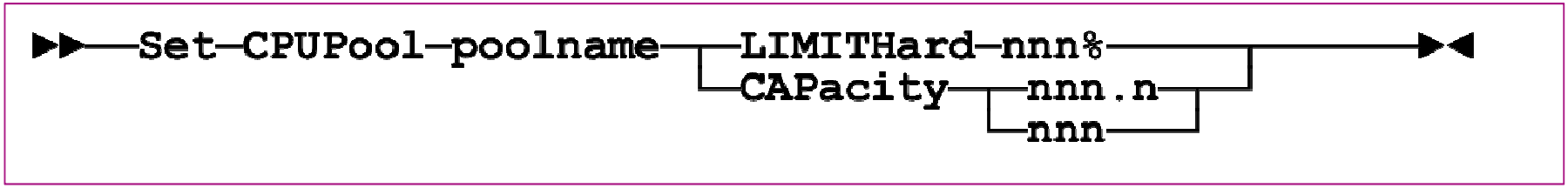
Enrolling Virtual Machines in Pools

- Assign a guest to or remove it from a CPU pool with the SCHEDULE command
  - Specified CPU pool must be already defined
  - Type of CPU in specified CPU pool must match the guest's primary CPU type
    - CPU affinity must be on for the guest
    - If guest already assigned to a CPU pool it is removed from that pool and added to the specified pool



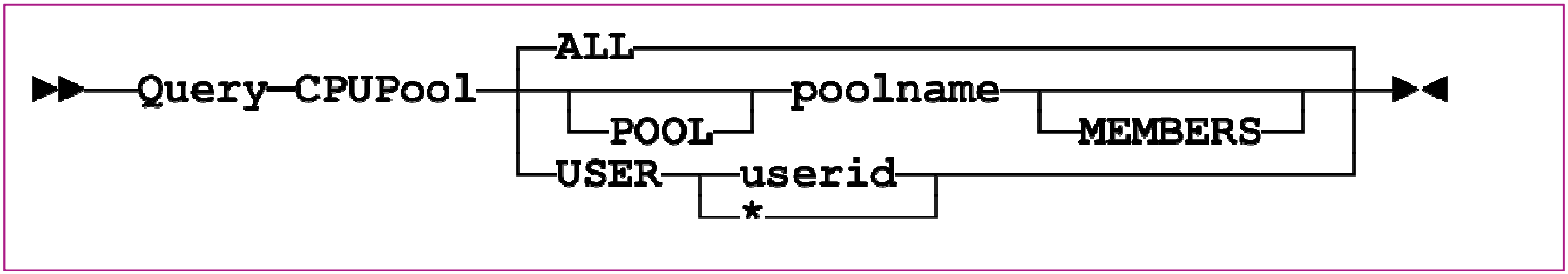
Changing Pool CPU Allocation

- Limits can be changed with the **SET CPUPOOL** command



Displaying CPU Pool information

- Use **QUERY CPUPOOL** to see information about the pools defined on your system



Displaying CPU Pool Information

- **Display all pool definitions**

```
query cpupool all
CPU pool     Limit      Type     Members
LINUXP2     8.0 CPUs     IFL           0
CPPOOL10     12 %         CP           8
LINUXP3      30 %        IFL          20
LINUXP1     2.5 CPUs     IFL           6
```

- **Display one pool definition and member names**

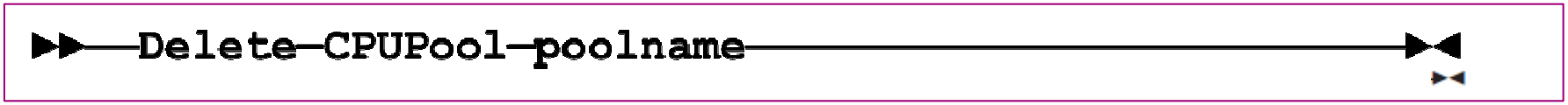
```
query cpupool linuxp1 members
CPU pool Limit          Type    Members
LINUXP1  2.5 CPUs        IFL           6
The following users are members of CPU pool LINUXP1:
D70LIN12 D79LIN03 D79ADM D79LIN10 D79LIN07
D79LIN04
```

- **Display user's pool name**

```
query cpupool user d79adm
User D79ADM is in CPU pool LINUXP1
```

DELETE CPUPOOL

- Use DELETE CPUPOOL to delete a pool definition
- Pool must be empty
  - Use SCHEDULE … NOPOOL first to remove each member



Automating CPU Pool Management

- Complication
  - At VM IPL, no pools are defined (not remembered from prior IPL)
  - Cannot add users to pool until it is defined
- Solutions

1. COMMAND statements in directory definition of OPERATOR or AUTOLOG1

```
USER OPERATOR . . .
   . . .
   COMMAND DEFINE CPUPOOL WEBSPH CAPACITY   5  TYPE IFL
   COMMAND DEFINE CPUPOOL DB2    CAPACITY   3  TYPE IFL
   COMMAND DEFINE CPUPOOL QADEPT LIMITHARD 10% TYPE CP
```

…Or include CP DEFINE commands in AUTOLOG1's PROFILE EXEC

2. COMMAND statements in virtual machine definitions to place them into pools when they log on

```
USER WASPROD1 . . .
   . . .
   COMMAND SCHEDULE * WITHIN POOL WEBSPH
```

Single System Image Considerations

- CPU pools are defined and managed independently on each member of an SSI cluster

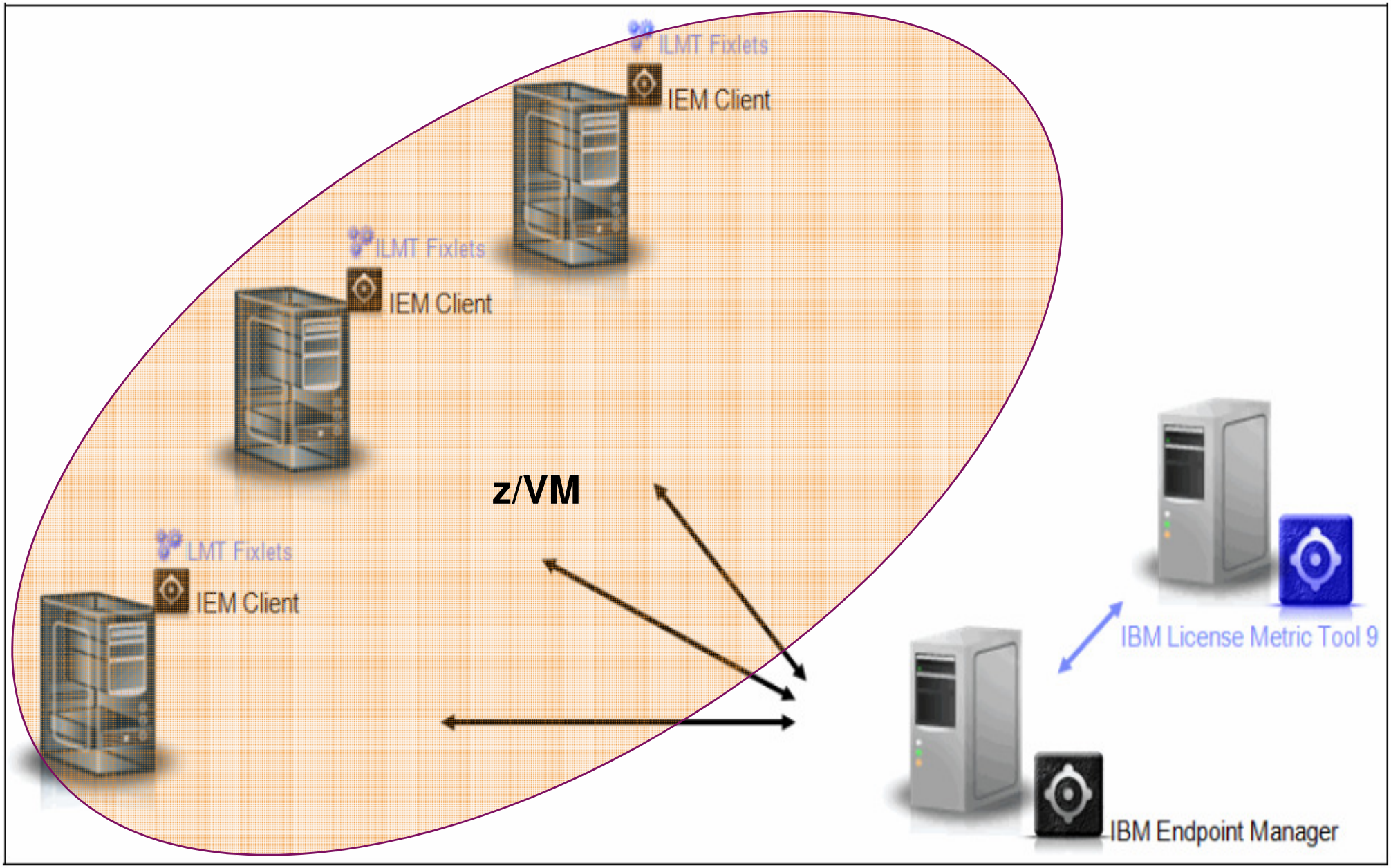
- A guest in a CPU pool can relocate to another system if a CPU pool with the same name and type is defined on the target system
  - Need not have the same limits

- Administrator is responsible for adjusting pool limits if needed
  - May affect software license requirements

Track License Requirements with IBM License Metric Tool

- IBM License Metric Tool (ILMT) is a no-charge tool used to determine PVU licensing requirements
- New Linux interface exploited by ILMT to assess software license requirements
  - Invokes z/VM Environment Information Interface
- Ability to track CPU pools available in ILMT 9.0.1, August 12, 2014
  - Improvements also made to reduce CPU overhead incurred with ILMT
- Using ILMT you are only charged for the CPU pool capacity assigned to Passport Advantage PVU-based software

ILMT Architecture Overview

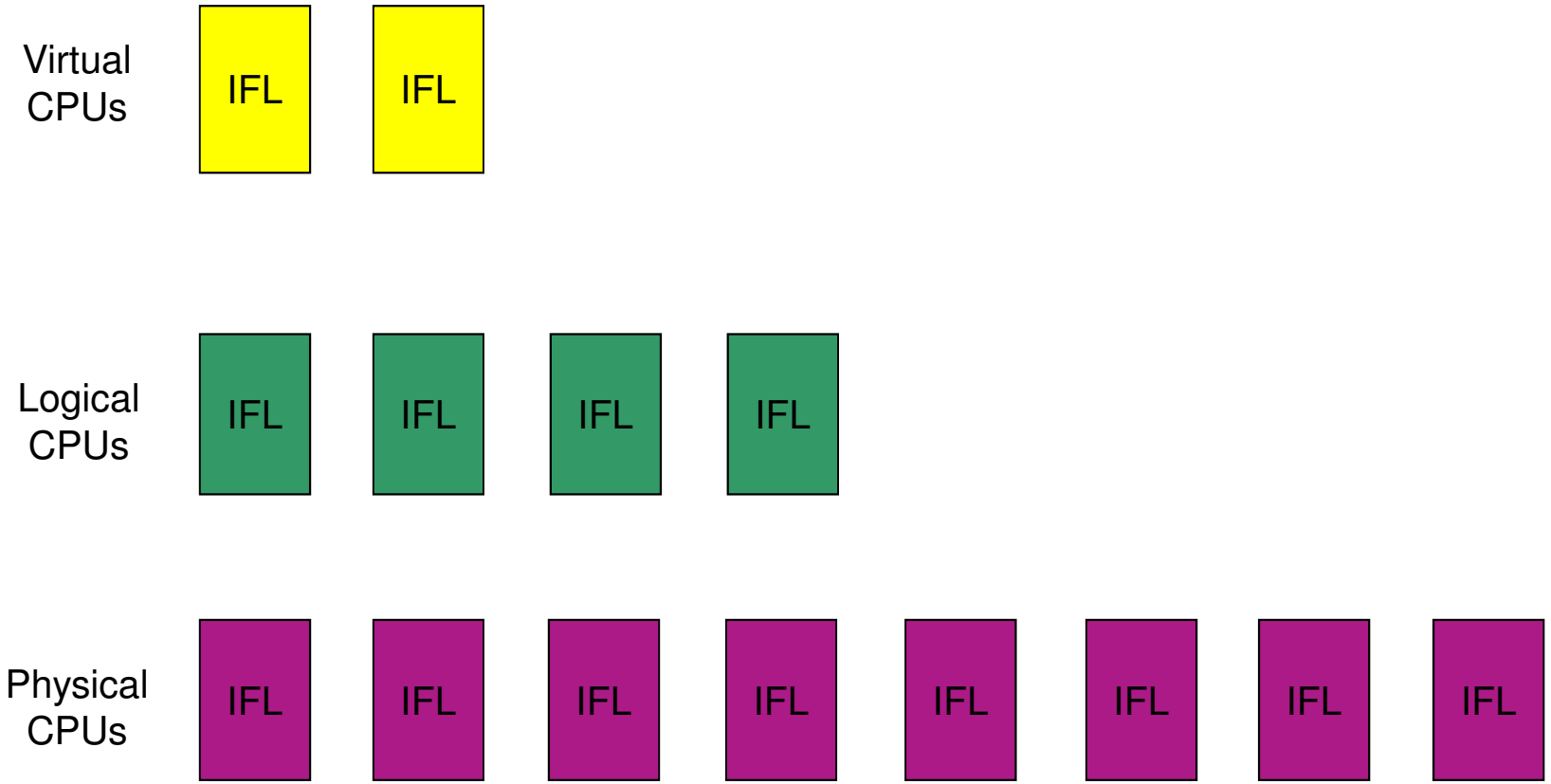


Software Licensing Key Points

- IBM's two Software Categories are z Systems software and Distributed software (entitlements are not interchangeable)
- Value Units (VUs) are used to license z Systems IPLA software and Processor Value Units (PVUs) are used to license Distributed Passport Advantage software
- Distributed Sub-Capacity Terms require customers to keep track of the maximum processor capacity available to a program
  - IBM License Metric Tool calculates this
  - Customers run the tool and retain the reports
- When running z/VM virtual machines or LPARs, a customer is required to obtain licenses for the real hardware resources actually available to each program, not necessarily for all the resources
- PVUs are based on the processor family, for example
  - IFL on z114 is 100 PVUs while IFL on zEC12 is 120 PVUs
  - See IBM pricing expert for details
- On the z13, licensing granularity is one core's worth of processing power
  - No thread-based licensing

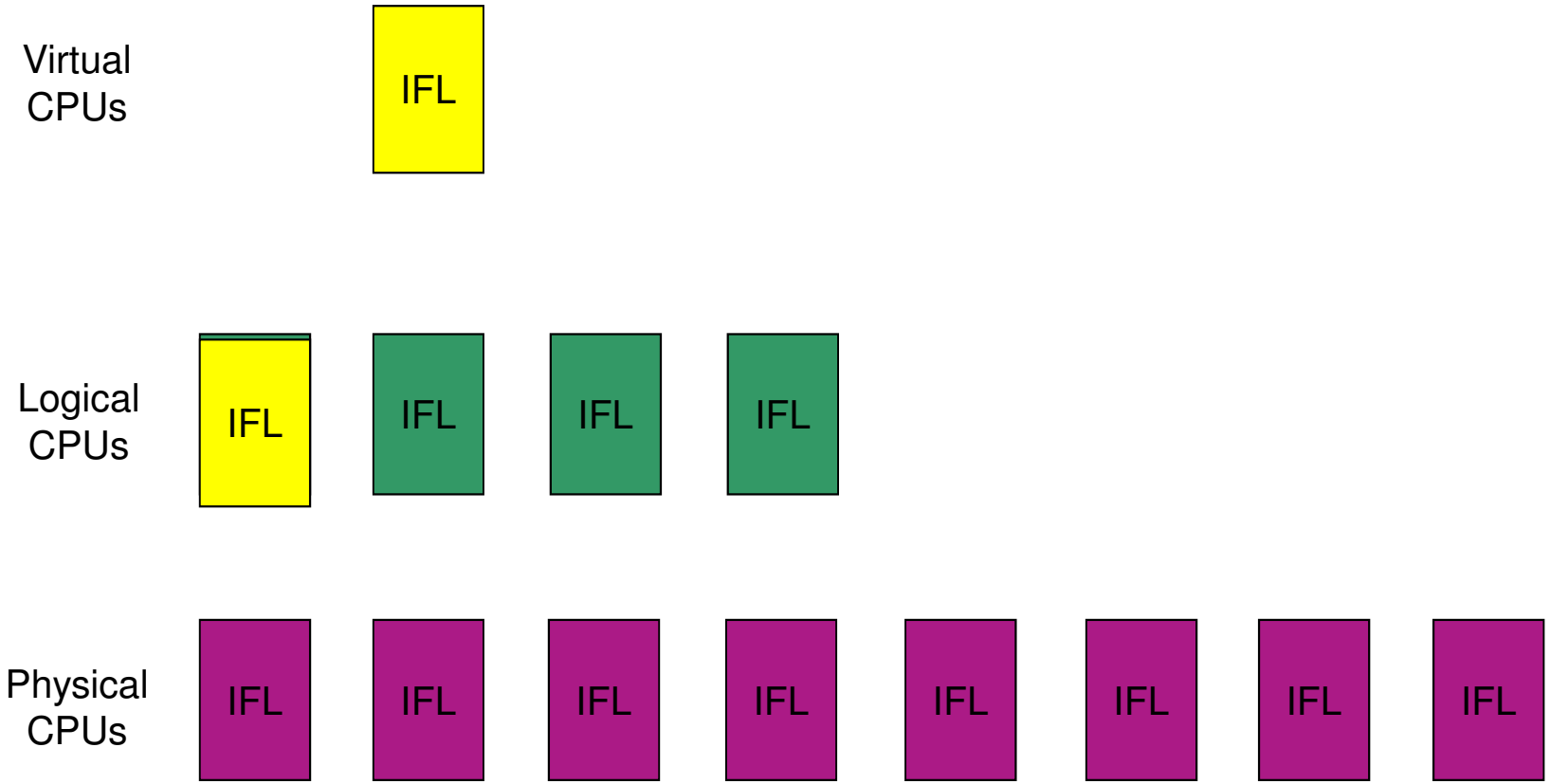
Linux Guest Software Pricing Without CPU Pooling

Pricing rule for products in z/VM guests: The lower of the sum of the virtual engines available to guests running a product or the engine capacity of the z/VM LPAR from which the guests obtain their resources.



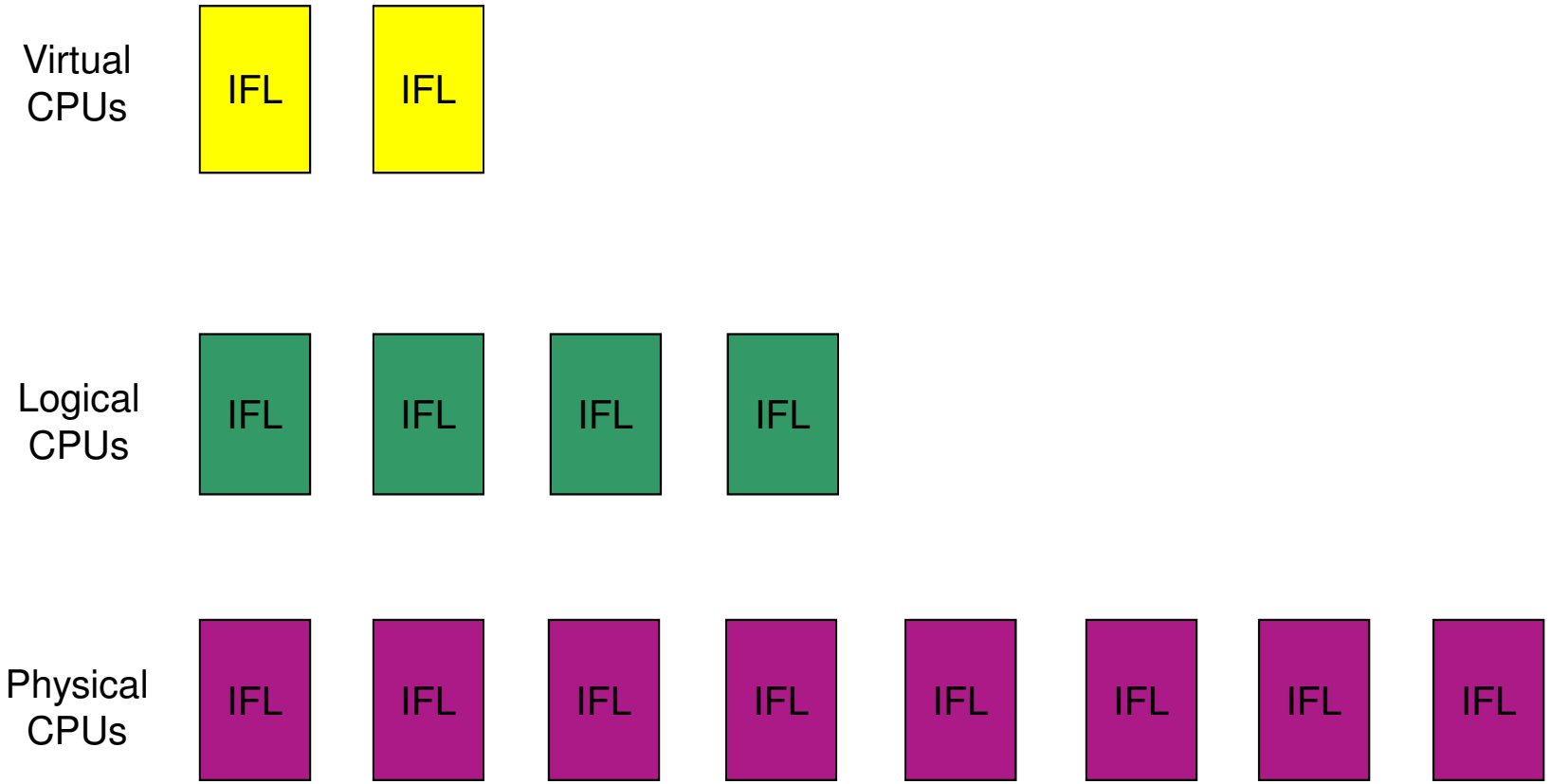
Linux Guest Software Pricing Without CPU Pooling

Pricing rule for products in z/VM guests: The lower of the sum of the virtual engines available to guests running a product or the engine capacity of the z/VM LPAR from which the guests obtain their resources.



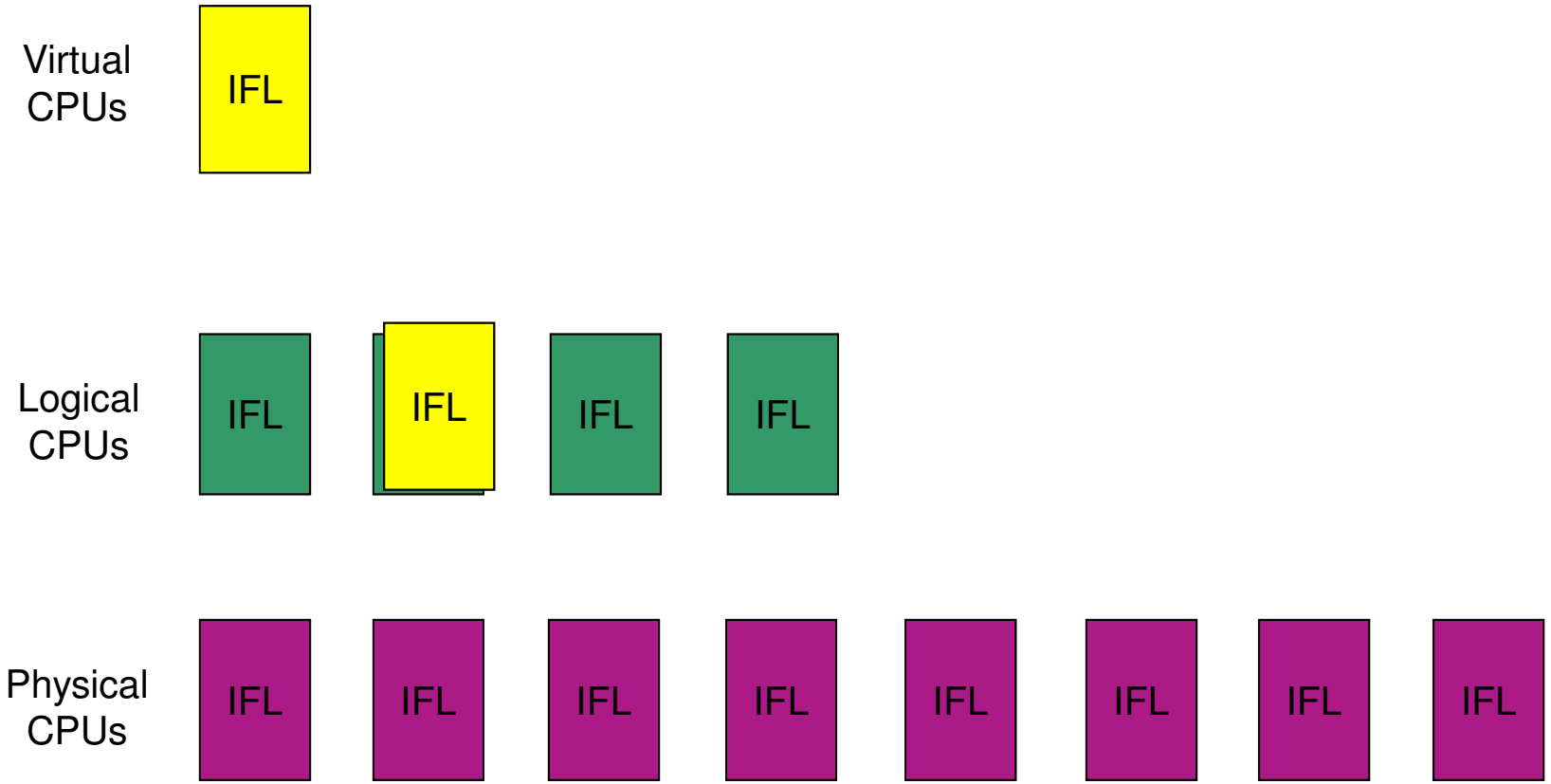
Linux Guest Software Pricing Without CPU Pooling

Pricing rule for products in z/VM guests: The lower of the sum of the virtual engines available to guests running a product or the engine capacity of the z/VM LPAR from which the guests obtain their resources.



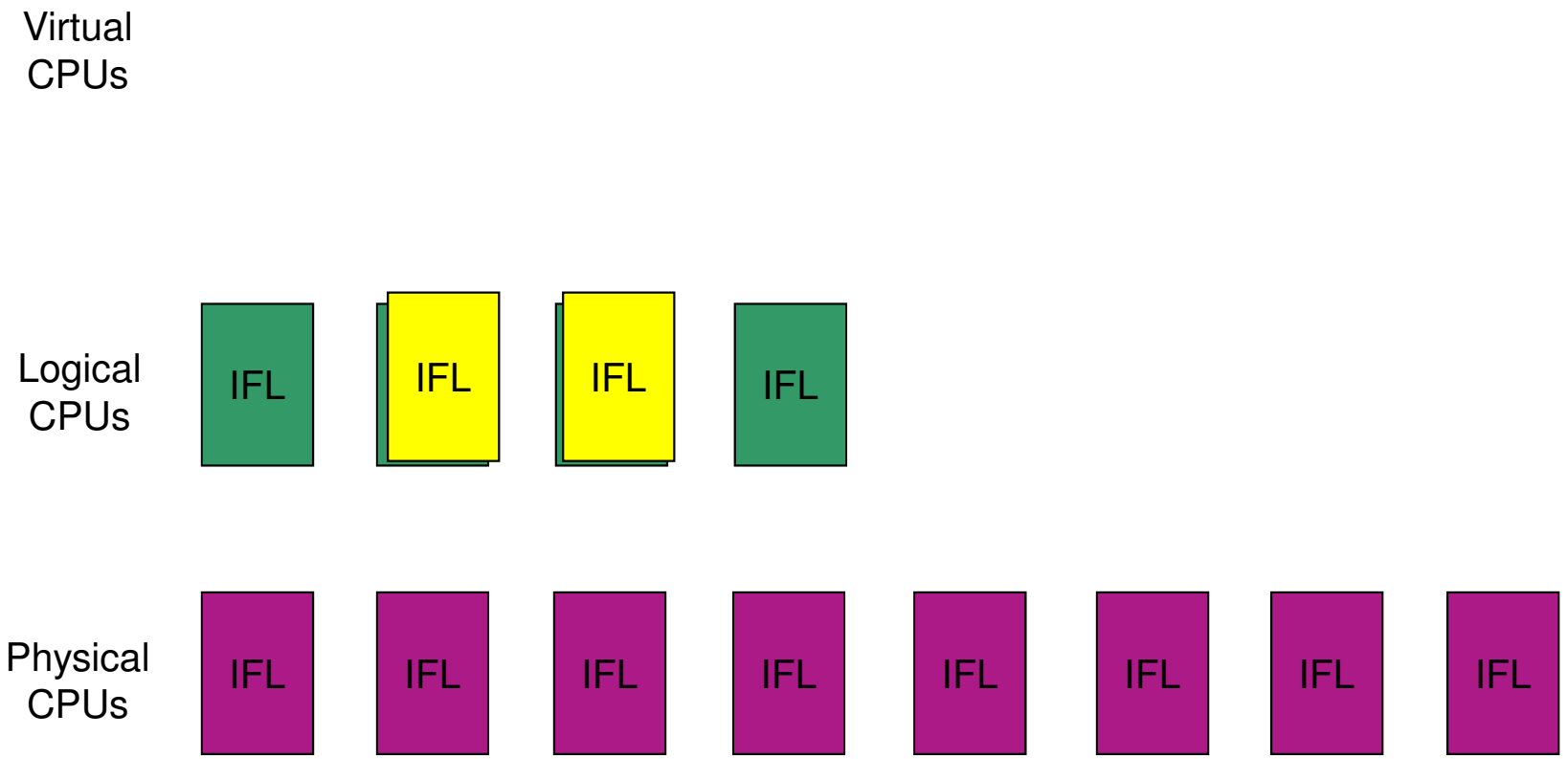
Linux Guest Software Pricing Without CPU Pooling

Pricing rule for products in z/VM guests: The lower of the sum of the virtual engines available to guests running a product or the engine capacity of the z/VM LPAR from which the guests obtain their resources.



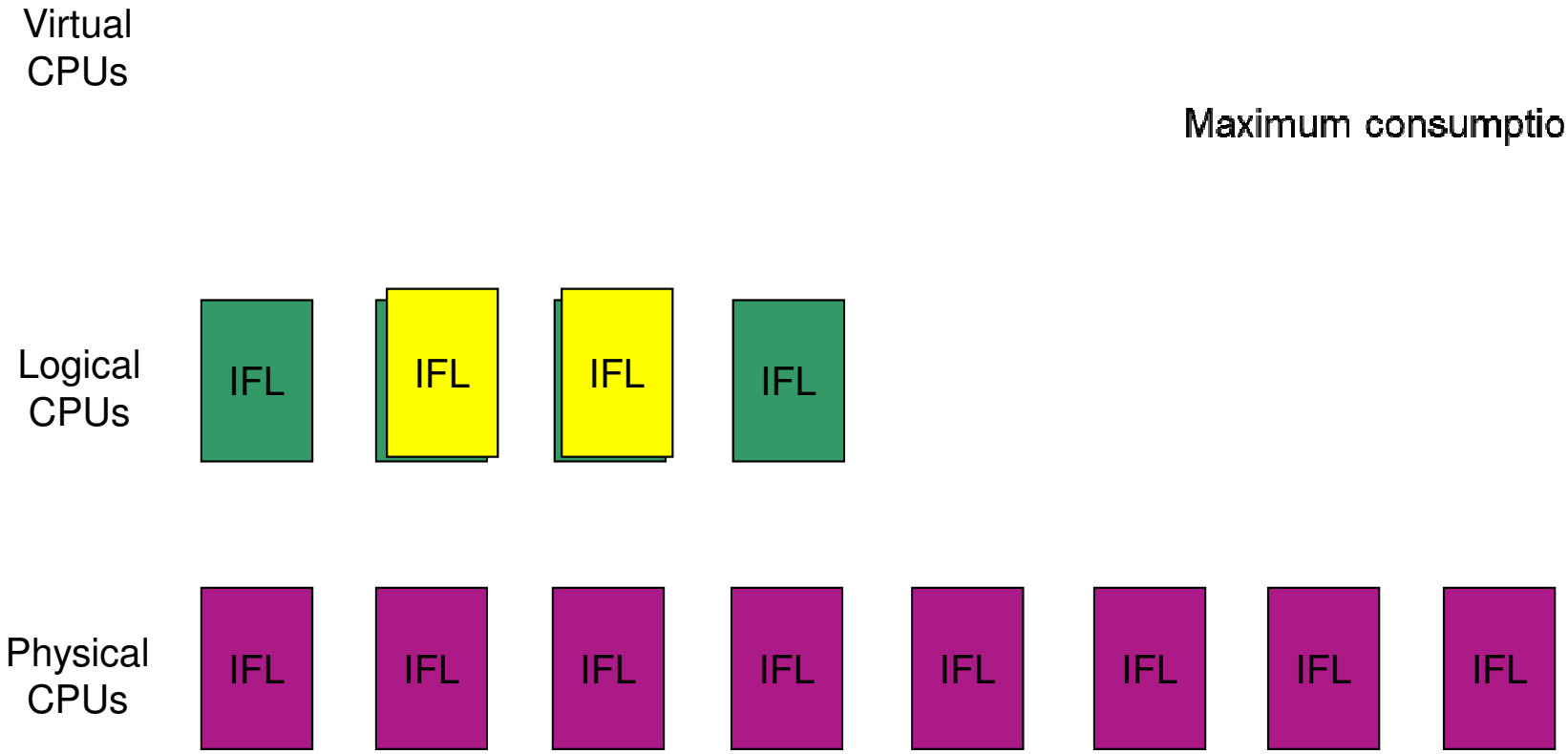
Linux Guest Software Pricing Without CPU Pooling

Pricing rule for products in z/VM guests: The lower of the sum of the virtual engines available to guests running a product or the engine capacity of the z/VM LPAR from which the guests obtain their resources.

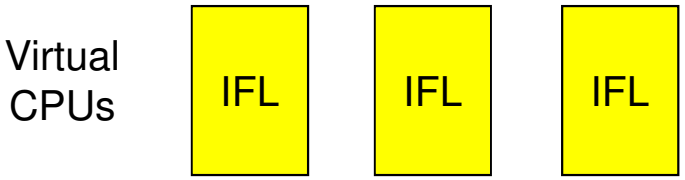


Linux Guest Software Pricing Without CPU Pooling

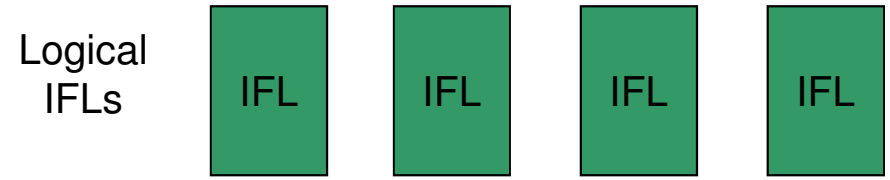
Pricing rule for products in z/VM guests: The lower of the sum of the virtual engines available to guests running a product or the engine capacity of the z/VM LPAR from which the guests obtain their resources.

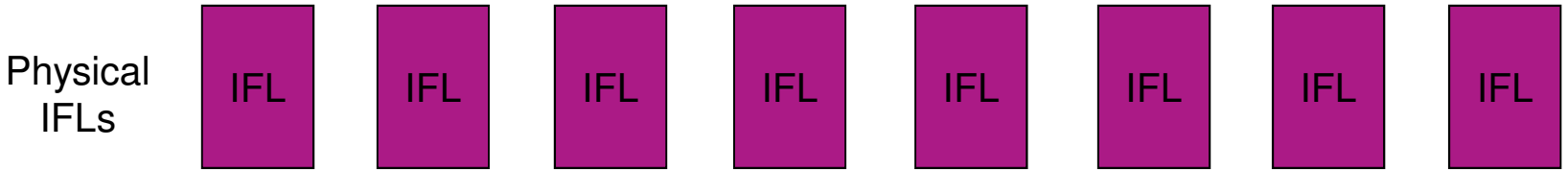


Linux Guest Software Pricing With CPU Pooling

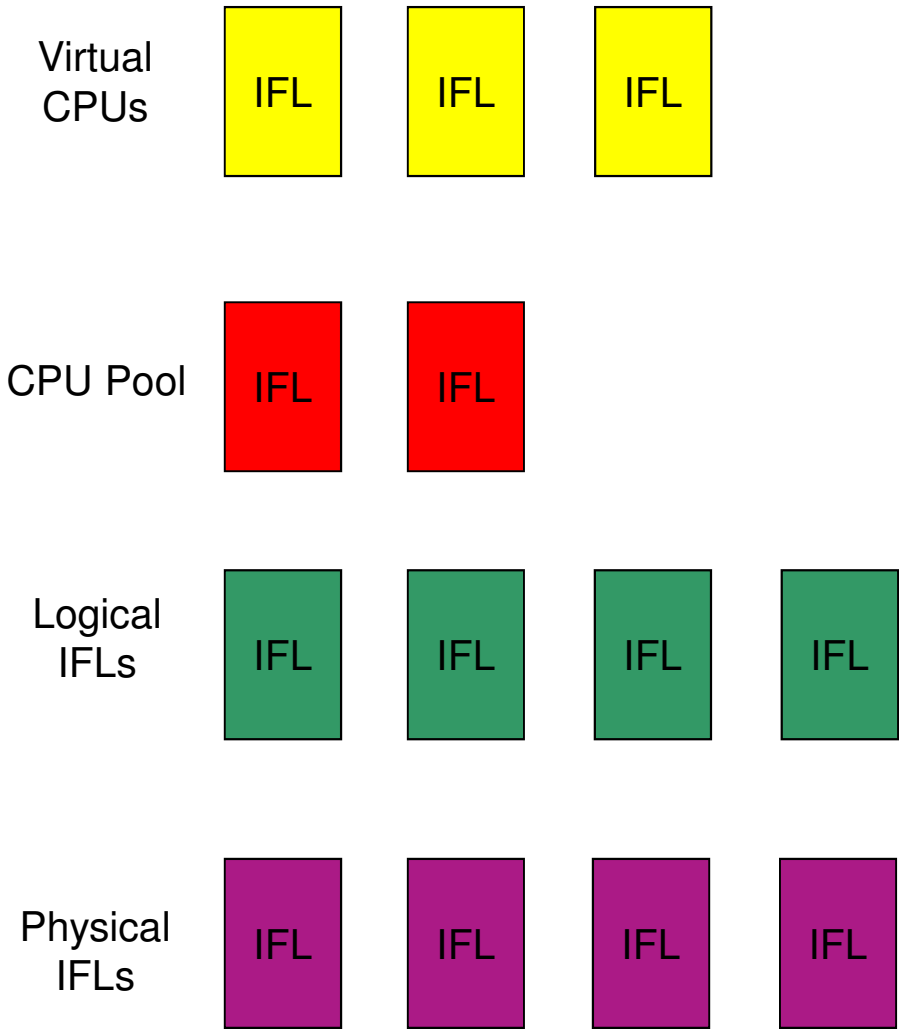


Pricing rule for products in z/VM guests: The lowest of the sum of the virtual engines available to guests running a product, the engine capacity of the (CAPACITY) CPU pool to which the guests are assigned, or the engine capacity of the z/VM LPAR from which the guests obtain their resources.



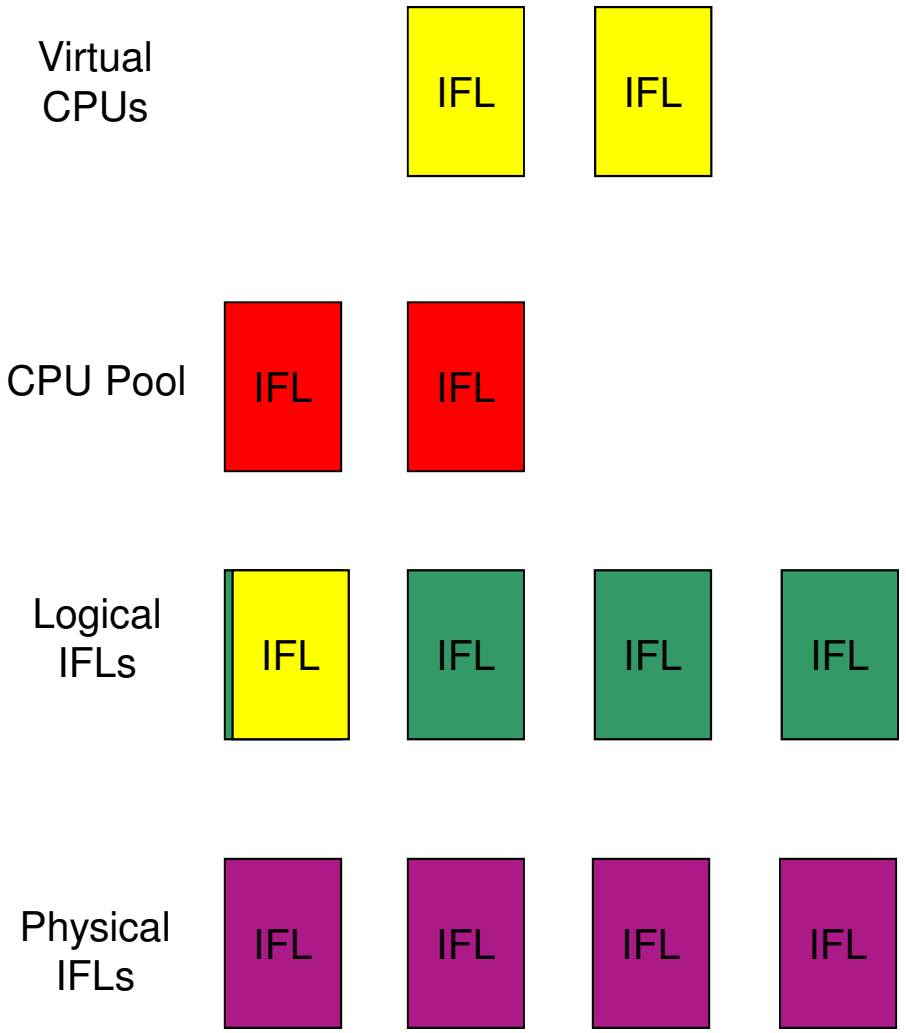


Linux Guest Software Pricing With CPU Pooling



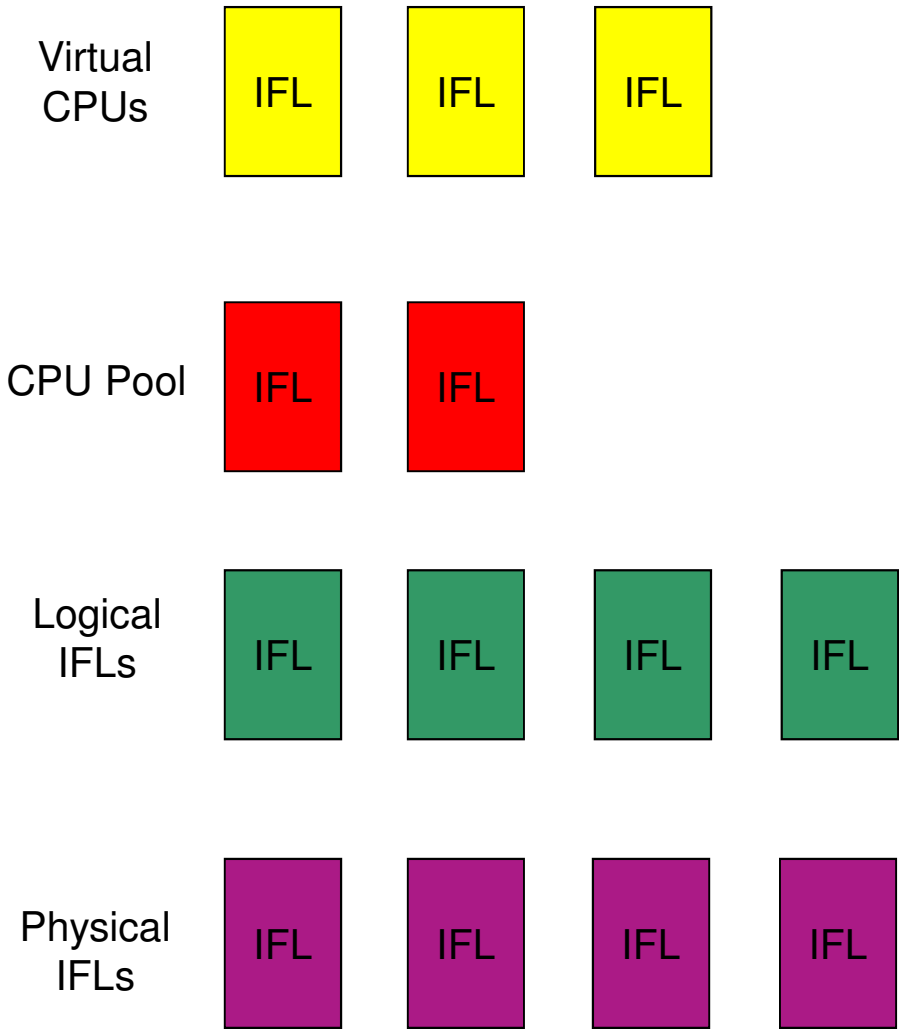
Pricing rule for products in z/VM guests: The lowest of the sum of the virtual engines available to guests running a product, the engine capacity of the (CAPACITY) CPU pool to which the guests are assigned, or the engine capacity of the z/VM LPAR from which the guests obtain their resources.

Linux Guest Software Pricing With CPU Pooling



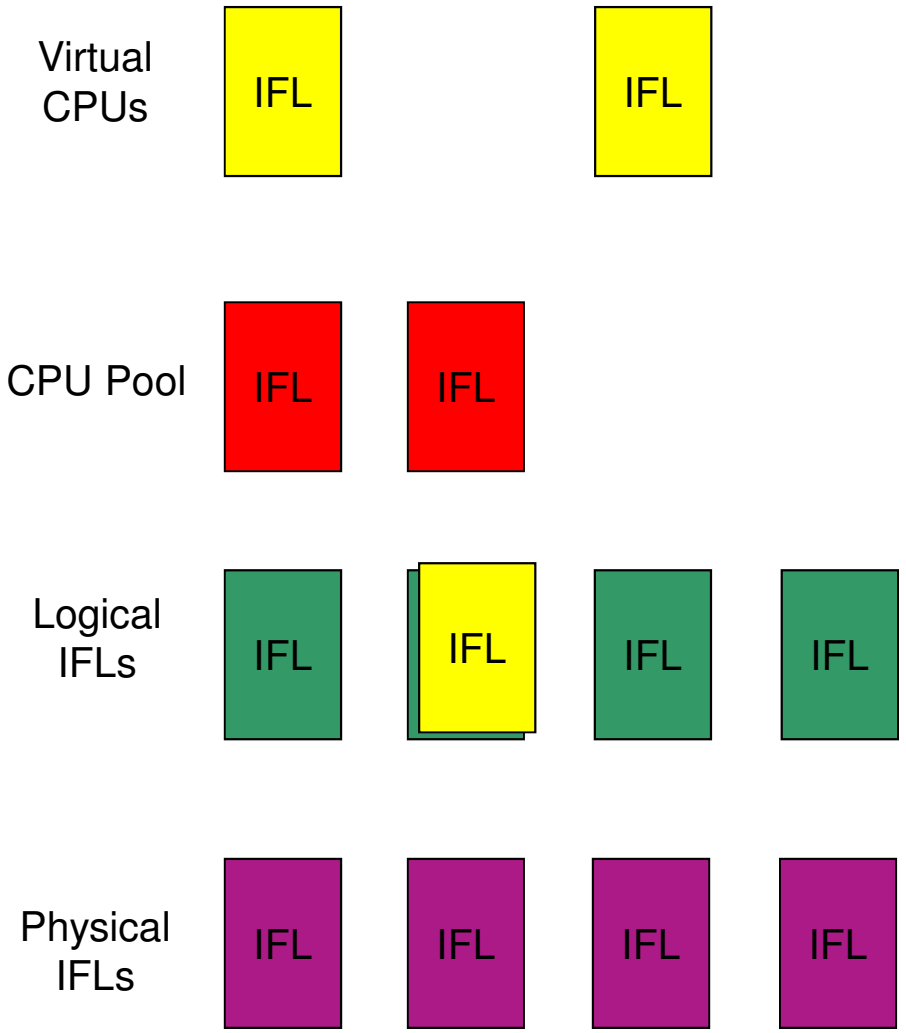
Pricing rule for products in z/VM guests: The lowest of the sum of the virtual engines available to guests running a product, the engine capacity of the (CAPACITY) CPU pool to which the guests are assigned, or the engine capacity of the z/VM LPAR from which the guests obtain their resources.

Linux Guest Software Pricing With CPU Pooling



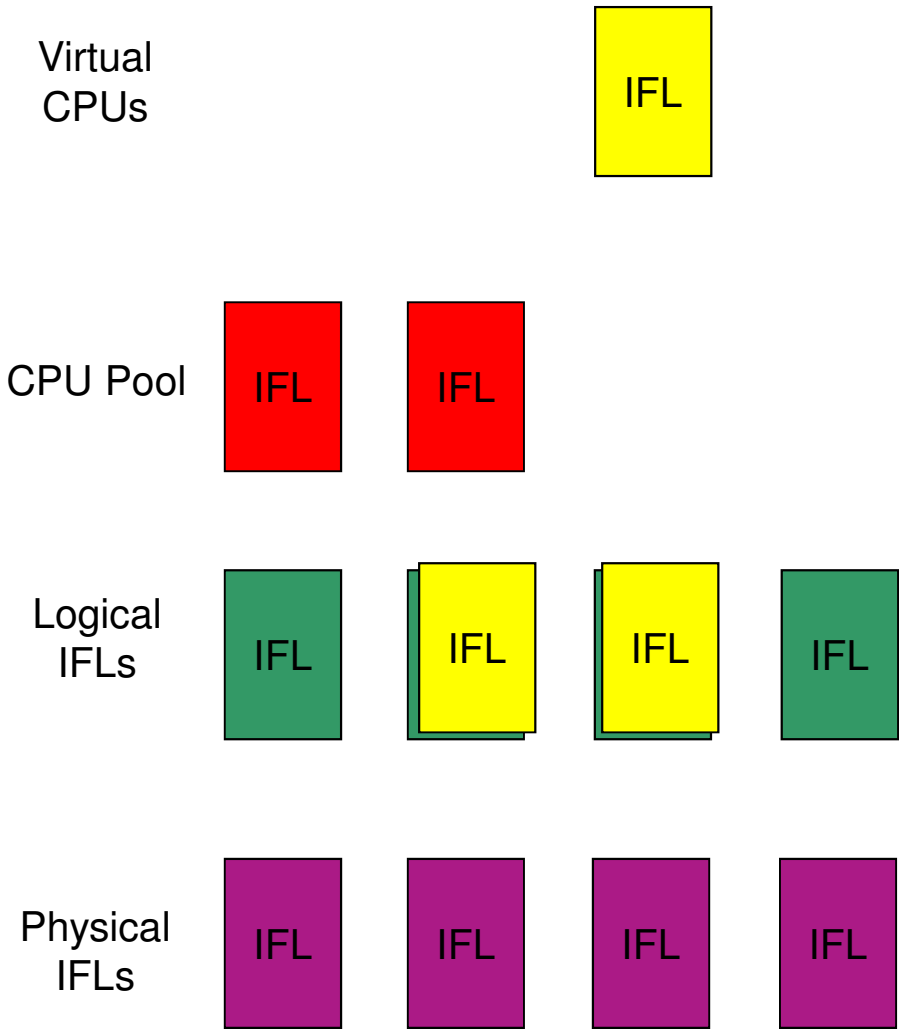
Pricing rule for products in z/VM guests: The lowest of the sum of the virtual engines available to guests running a product, the engine capacity of the (CAPACITY) CPU pool to which the guests are assigned, or the engine capacity of the z/VM LPAR from which the guests obtain their resources.

Linux Guest Software Pricing With CPU Pooling



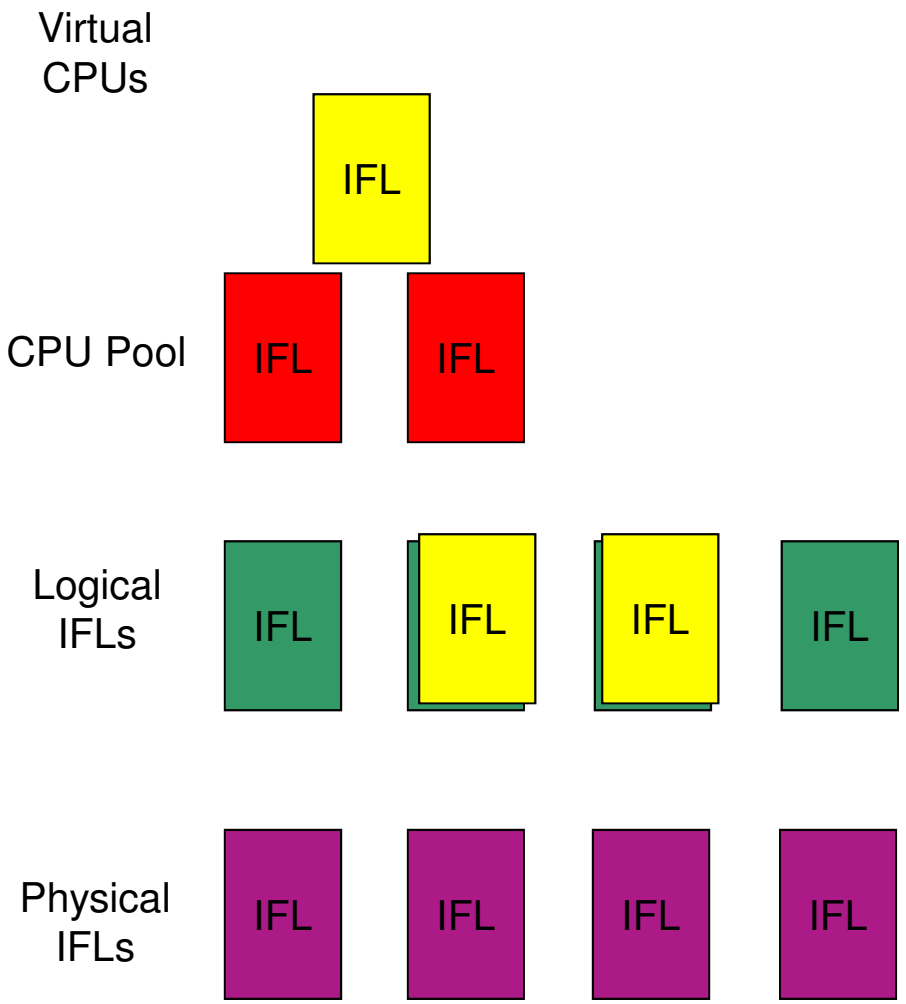
Pricing rule for products in z/VM guests: The lowest of the sum of the virtual engines available to guests running a product, the engine capacity of the (CAPACITY) CPU pool to which the guests are assigned, or the engine capacity of the z/VM LPAR from which the guests obtain their resources.

Linux Guest Software Pricing With CPU Pooling



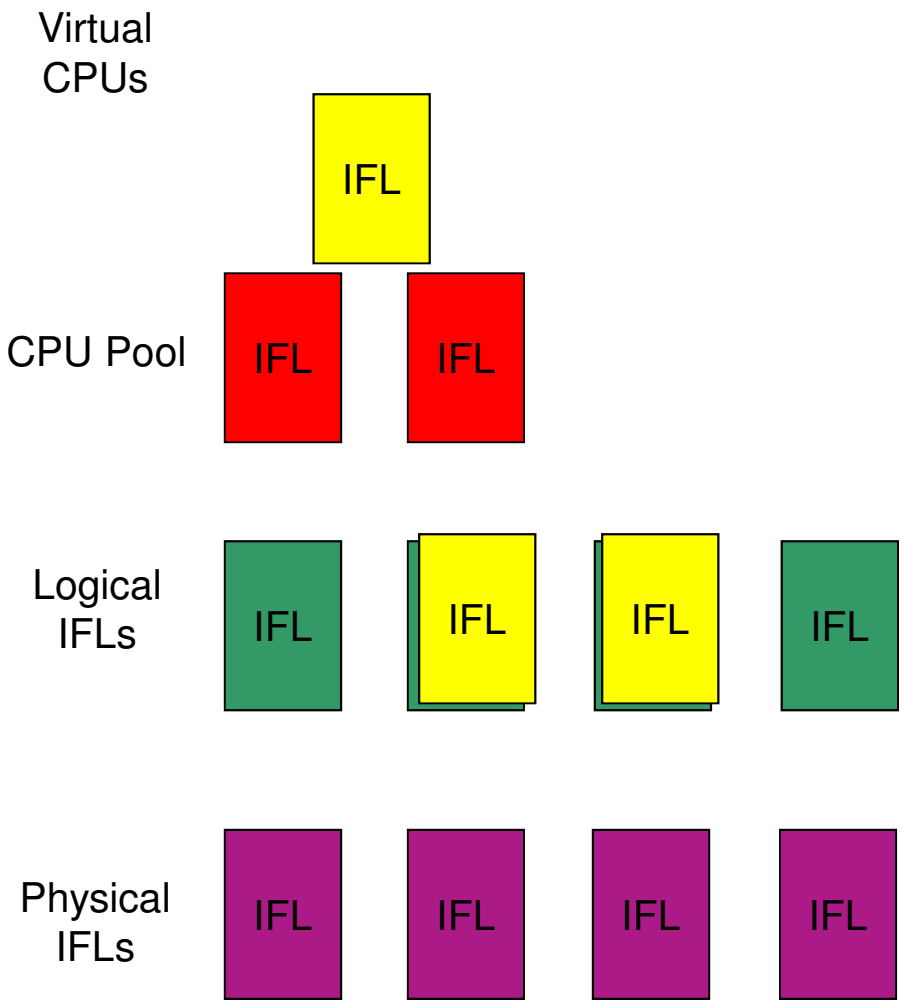
Pricing rule for products in z/VM guests: The lowest of the sum of the virtual engines available to guests running a product, the engine capacity of the (CAPACITY) CPU pool to which the guests are assigned, or the engine capacity of the z/VM LPAR from which the guests obtain their resources.

Linux Guest Software Pricing With CPU Pooling



Pricing rule for products in z/VM guests: The lowest of the sum of the virtual engines available to guests running a product, the engine capacity of the (CAPACITY) CPU pool to which the guests are assigned, or the engine capacity of the z/VM LPAR from which the guests obtain their resources.

Linux Guest Software Pricing With CPU Pooling



Pricing rule for products in z/VM guests: The lowest of the sum of the virtual engines available to guests running a product, the engine capacity of the (CAPACITY) CPU pool to which the guests are assigned, or the engine capacity of the z/VM LPAR from which the guests obtain their resources.

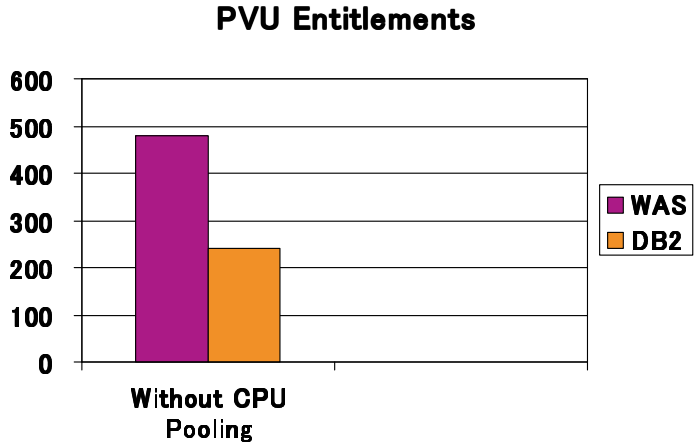
Maximum consumption: 2 IFLs

Use cases for CPU Pooling

- Department budgeting
  - Assign each department's guests to CPU pool with contracted capacity
- Grow workloads without affecting the budget
  - Add New Workload
  - Add Capacity
  - Combine LPARs
  - Handle fractional workload requirements
- Prevent resource over-consumption
  - Limit aggressive workloads

Add New Workload Without CPU Pooling

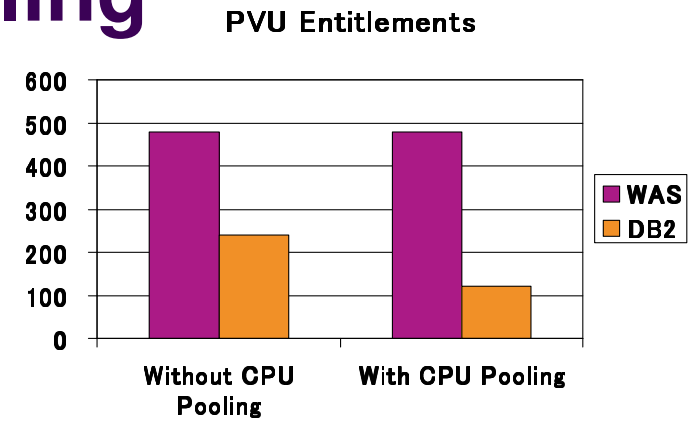
- 4 WAS production guests
  - Requires 4-engine WAS entitlement
- Add 2 DB2 production guests
  - Requires 2-engine DB2 entitlement

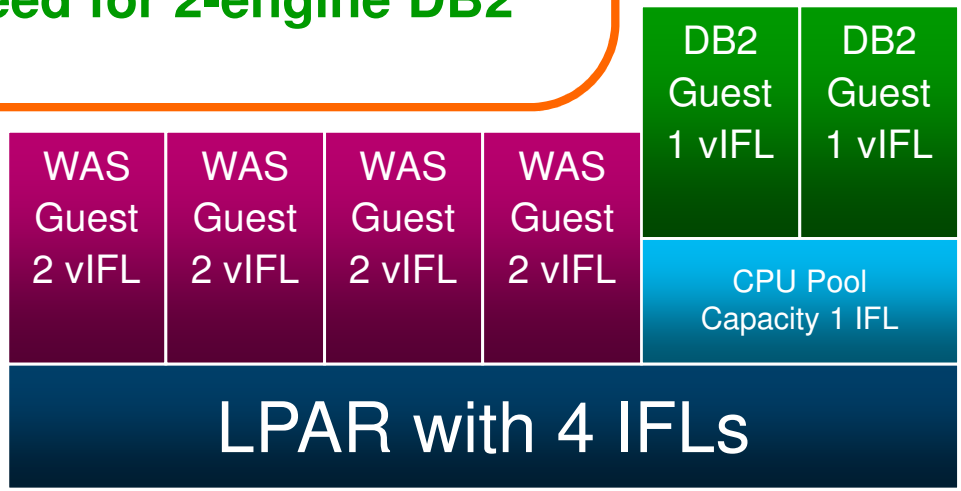


Note: All PVU Entitlement examples based on zEC12 (120 PVU per IFL) – will look proportionally the same on zBC12 (100 PVU per IFL)

Add New Workload With CPU Pooling

- 4 WAS production guests
  - Requires 4-engine WAS entitlement
- Create a 1-IFL pool
- Put the 2 DB2 production guests in pool
  - Requires 1-engine DB2 entitlement (avoiding the need for 2-engine DB2 entitlement)



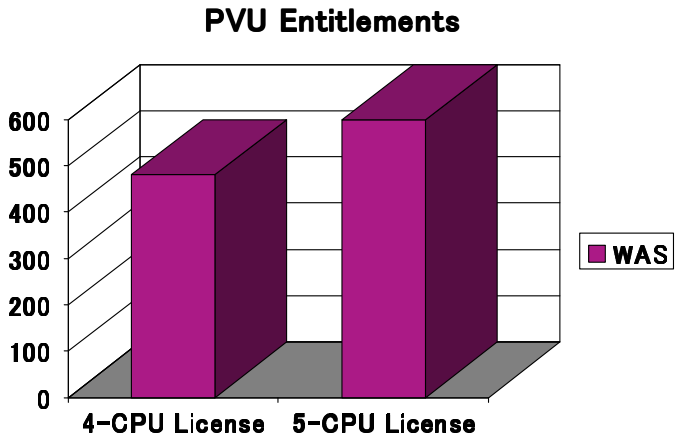


- Allows new workloads to be added cost effectively
- Encourages additional workload consolidation after initial success

Note: All PVU Entitlement examples based on zEC12 (120 PVU per IFL) – will look proportionally the same on zBC12 (100 PVU per IFL)

Add Capacity Without CPU Pooling

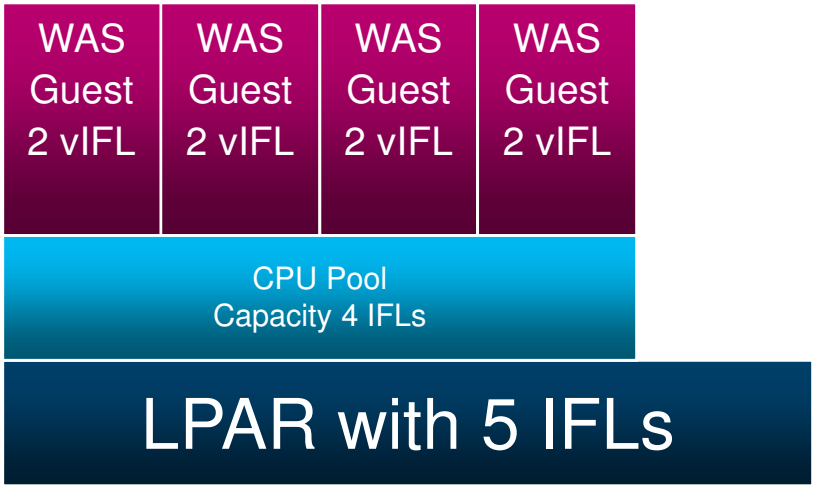
- 4 WAS production guests
  - Requires 4-engine WAS entitlement
- Add another IFL to the LPAR
  - Requires increase to 5-engine WAS entitlement

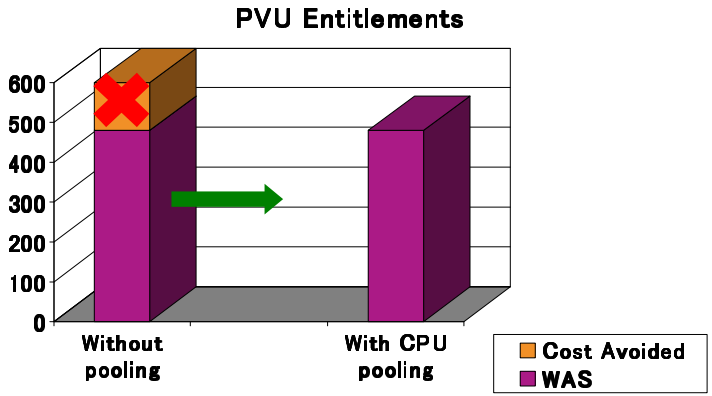


Note: All PVU Entitlement examples based on zEC12 (120 PVU per IFL) – will look proportionally the same on zBC12 (100 PVU per IFL)

Add Capacity With CPU Pooling

- LPAR with 4 IFLs
- Set up CPU Pooling for 4 IFLs
  - 4 WAS production guests require 4-engine WAS entitlement
- Add another IFL to the LPAR
- Avoids an incremental WAS entitlement license – allows capacity to be added without increasing software license charges
- Encourages adding capacity for other workloads (e.g., open source applications)





Note: All PVU Entitlement examples based on zEC12 (120 PVU per IFL) – will look proportionally the same on zBC12 (100 PVU per IFL)

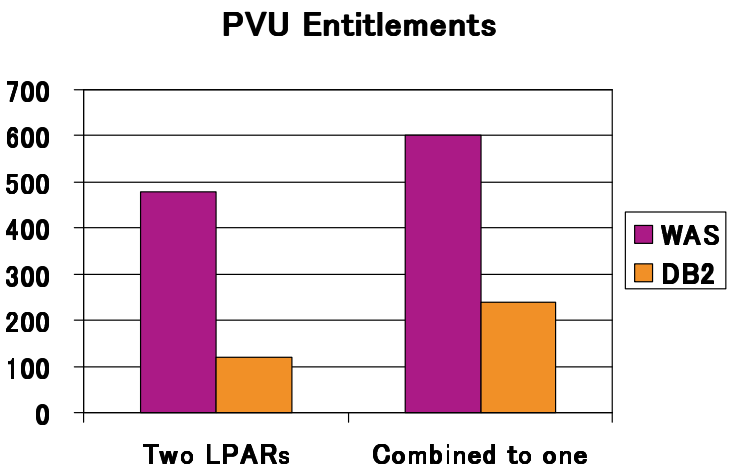
Combine LPARs Without CPU Pooling

- LPAR with 4 IFLs and 4 WAS production guests
  - Requires 4-engine WAS entitlement
- LPAR with 1 IFL and 2 DB2 production guests
  - Requires 1-engine DB2 entitlement

Note: All PVU Entitlement examples based on zEC12 (120 PVU per IFL) – will look proportionally the same on zBC12 (100 PVU per IFL)

Combine LPARs Without CPU Pooling

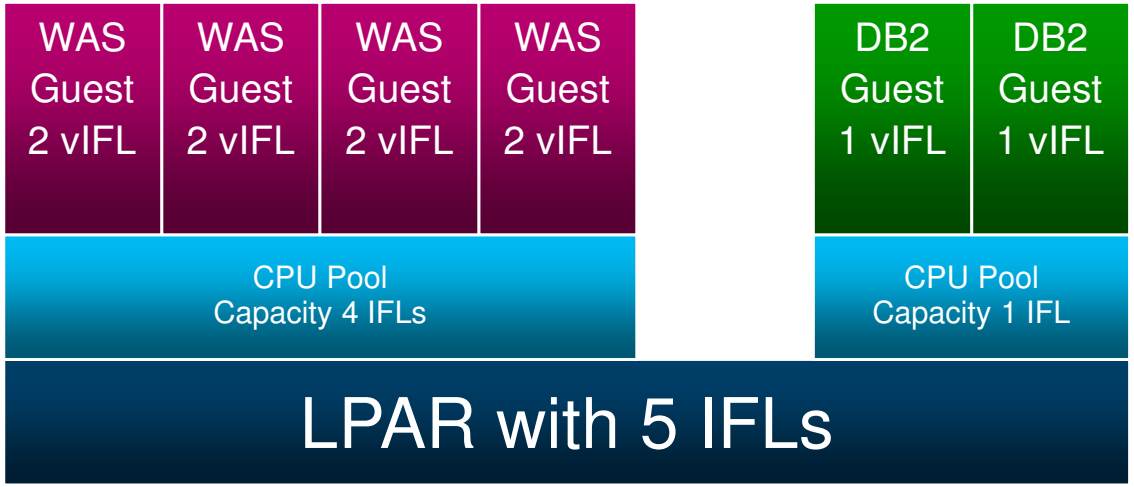
- LPAR with 4 IFLs and 4 WAS production guests
  - Requires 4-engine WAS entitlement
- LPAR with 1 IFL and 2 DB2 production guests
  - Requires 1-engine DB2 entitlement
- LPARs merge to one LPAR with 5 IFLs
  - Requires increase to 5-engine WAS entitlement
  - Requires increase to 2-engine DB2 entitlement

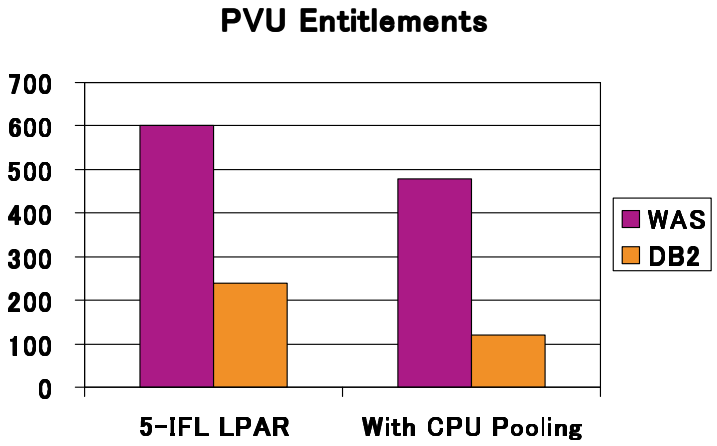


Note: All PVU Entitlement examples based on zEC12 (120 PVU per IFL) – will look proportionally the same on zBC12 (100 PVU per IFL)

Combine LPARs With CPU Pooling

- LPAR with 5 IFLs
- Create two Pools – one with 4 IFLs and one with 1 IFL
- Place the four WAS guests in the 4-IFL pool and the two DB2 guests in the 1-IFL pool
  - Requires 4-engine WAS entitlement
  - Requires 1-engine DB2 entitlement



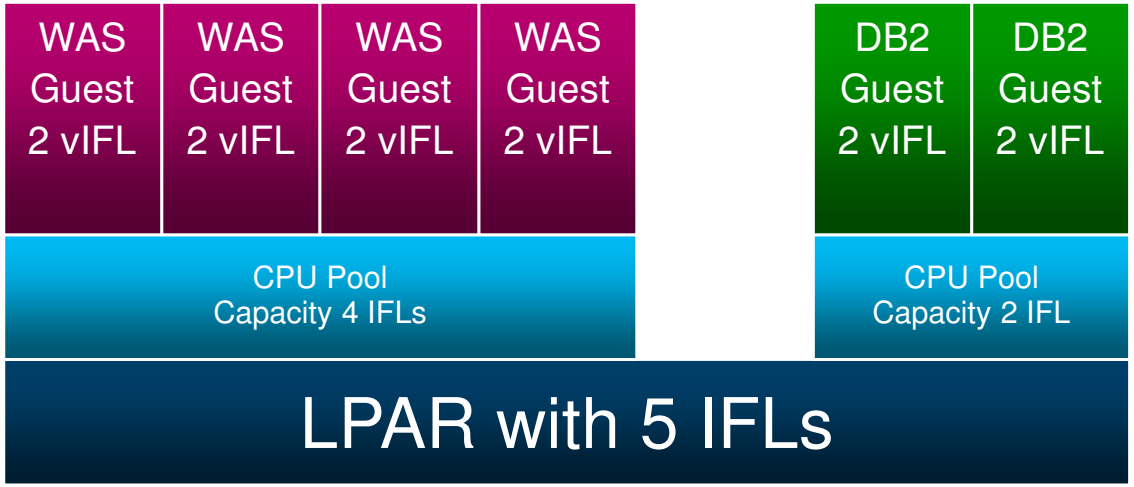


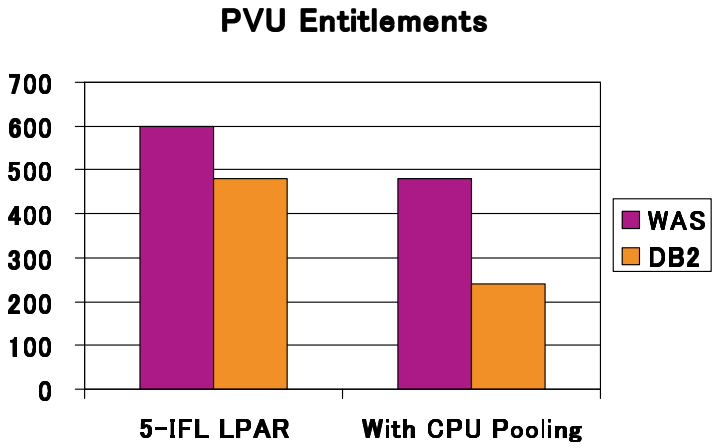
- Avoids increase in software license requirements (and costs)
- Reduces z/VM system management and maintenance workload
- Consolidates resources (memory, paging, network) for greater efficiency

Note: All PVU Entitlement examples based on zEC12 (120 PVU per IFL) – will look proportionally the same on zBC12 (100 PVU per IFL)

CPU Pools that Overcommit

- LPAR with 5 IFLs
- Create two Pools – one with 4 IFLs and one with 2 IFLs
- Place the four WAS guests in the 4-IFL pool and the two DB2 guests in the 2-IFL pool
  - Requires 4-engine WAS entitlement
  - Requires 2-engine DB2 entitlement



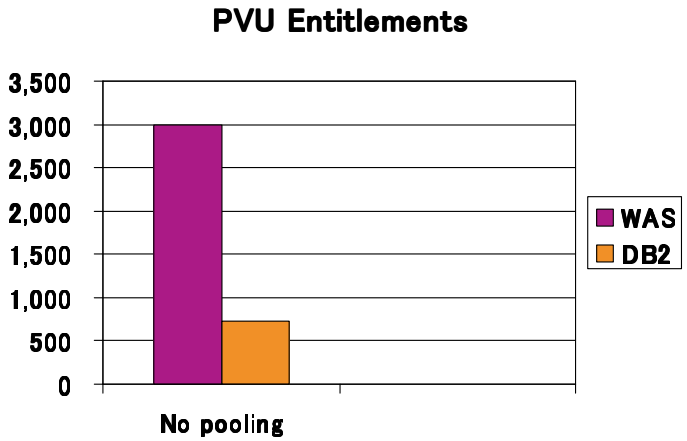


- Avoids increase in software license requirements (and costs)
- Reduces z/VM system management and maintenance workload

Note: All PVU Entitlement examples based on zEC12 (120 PVU per IFL) – will look proportionally the same on zBC12 (100 PVU per IFL)

Large system with virtual machines that require fractional IFL capacity

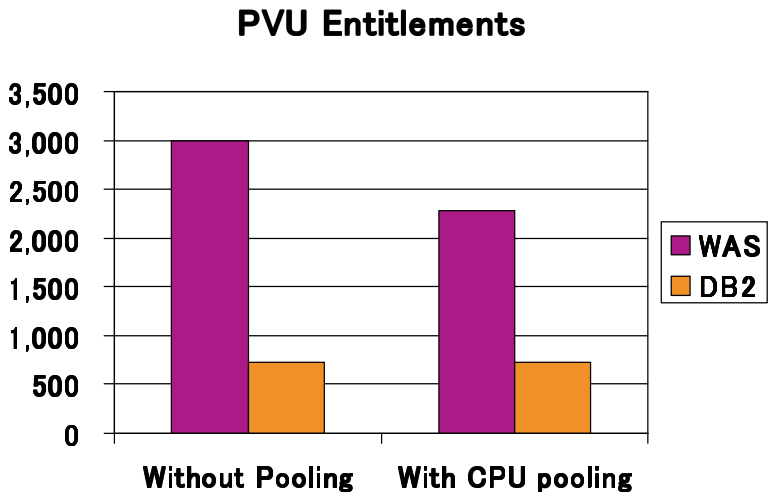
- LPAR with 25 IFLs
- 2 DB2 production guests
  - Requires 6-engine DB2 entitlement
- 3 WAS production guests and 12 small WAS test guests
  - Requires 25-engine WAS entitlement

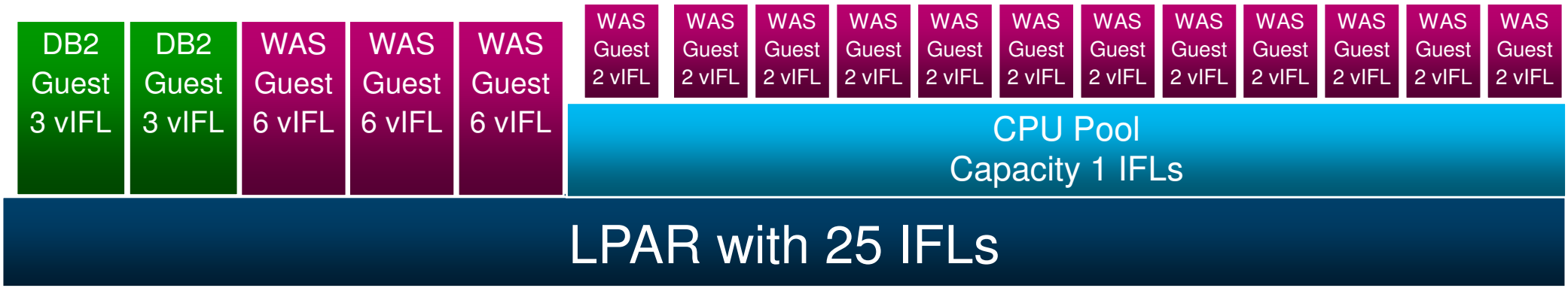


Note: All PVU Entitlement examples based on zEC12 (120 PVU per IFL) – will look proportionally the same on zBC12 (100 PVU per IFL)

Align fractional capacity virtual machines to small CPU pools

- LPAR with 25-IFLs
- Set up a 1-IFL pool
- 2 DB2 production guests
  - Requires 6-engine DB2 entitlement
- 3 WAS production guests and 1-IFL pool with 12 small WAS test guests
  - Requires 19-engine WAS entitlement

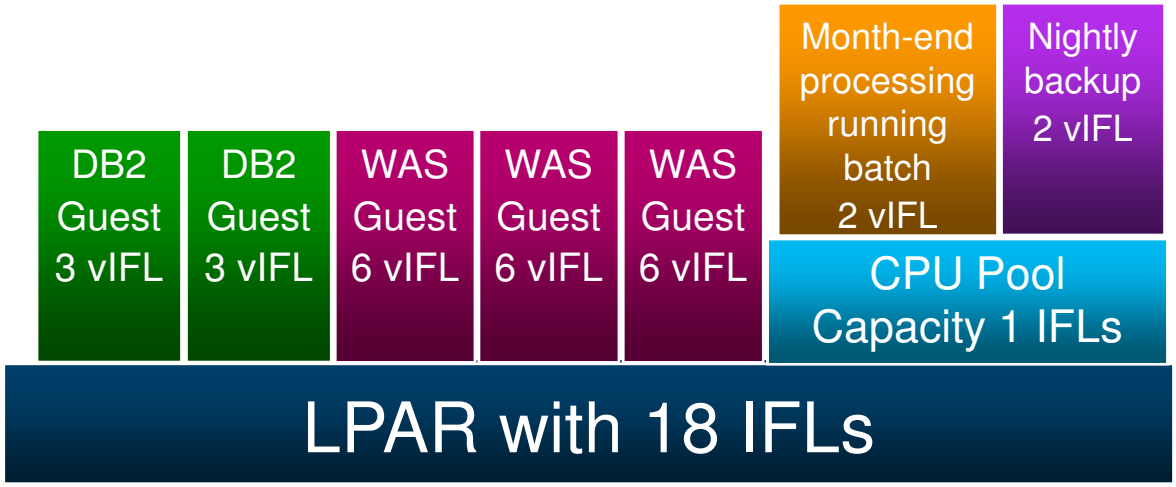




Note: All PVU Entitlement examples based on zEC12 (120 PVU per IFL) – will look proportionally the same on zBC12 (100 PVU per IFL)

Contain workloads that take too many resources

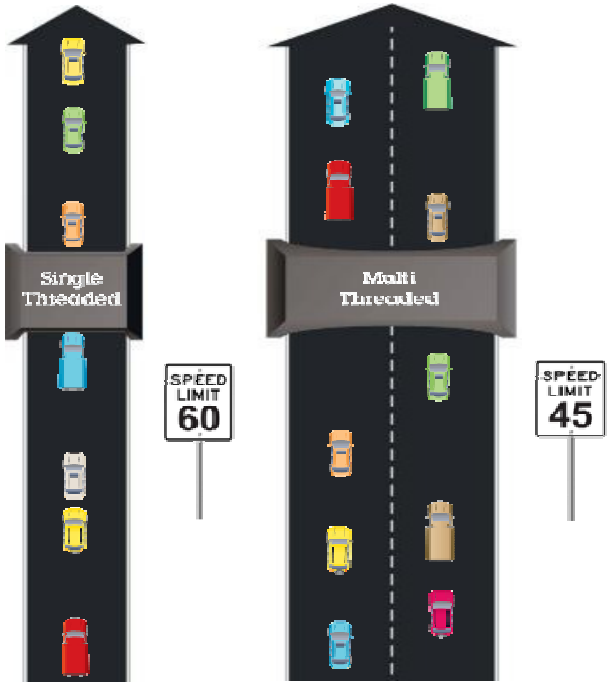
- LPAR with 18-IFLs
- 2 DB2 production guests and 3 WAS production guests are sharing the 18-IFLs
- Month-end processing or nightly backup uses any available capacity – could take from production guests
  - Set up a 1 IFL CPU pool for running these tasks



Note: All PVU Entitlement examples based on zEC12 (120 PVU per IFL) – will look proportionally the same on zBC12 (100 PVU per IFL)

Simultaneous Multithreading (SMT)

- Objective is to improve capacity, not performance
- Allows z/VM to dispatch work on up to two threads of a z13 IFL
- VM65586 for z/VM 6.3 only
  - PTFs available March 13, 2015
- At least z13 millicode bundle 11
- Transparent to virtual machine
  - Guest does not need to be SMT aware
  - SMT is not virtualized to the guest
- z13 SMT support limited to IFLs and zIIPs
  - z/VM support is only for IFLs
- SMT is disabled by default
  - Change requires System Configuration setting and re-IPL
  - Applies to entire z/VM partition
- Potential to increase overall capacity of system
  - Workload dependent



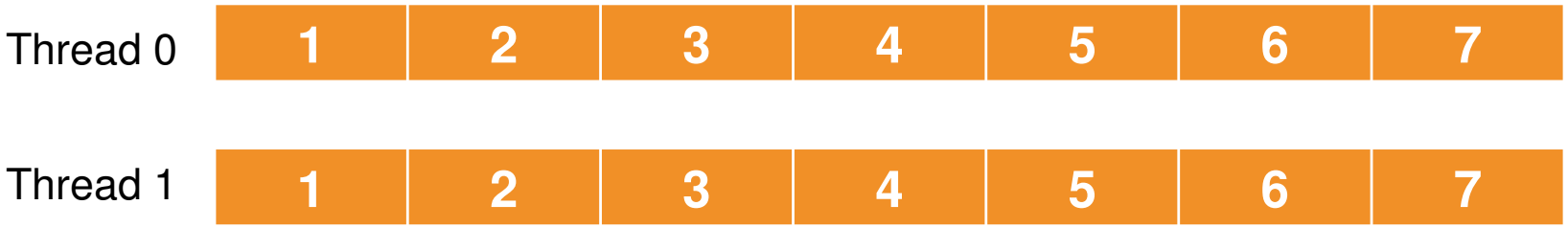
Which approach is designed for the higher volume of traffic? Which road is faster?

**Illustrative numbers only*

Additional Work Capacity

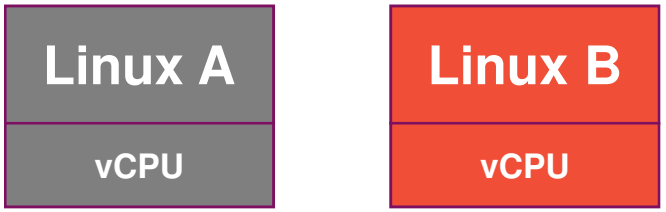
IFL (SMT disabled) – Time Slice Rate: 10

IFL (SMT enabled) – Time Slice Rate: 7



- **Numbers are for illustrative purposes only**
- **Without SMT: 10/second**
- **With SMT: 7/second but two threads yields capacity of 14/second**

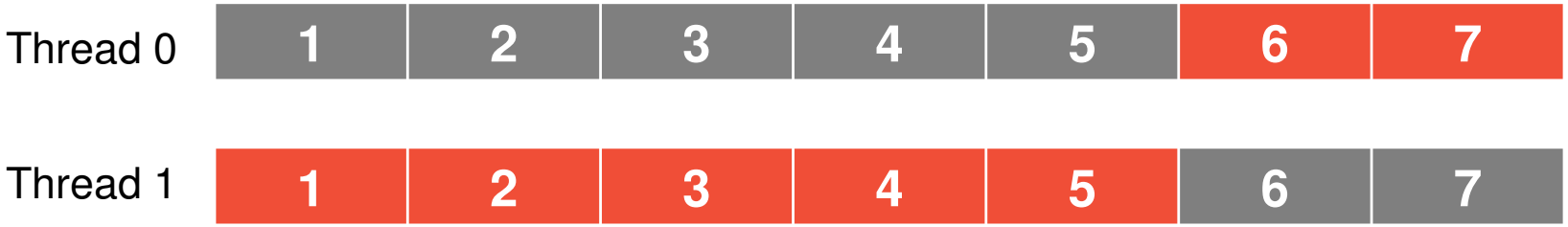
Interleaving Virtual CPUs of Guests



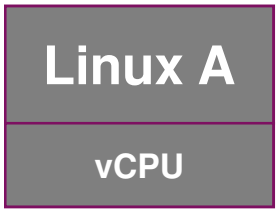
- In single core, we time slice access with each guest getting 5 time slices
- With SMT, each guest gets 7 time slices for total of 14

IFL (SMT disabled) – Time Slice Rate: 10

IFL (SMT enabled) – Time Slice Rate: 7



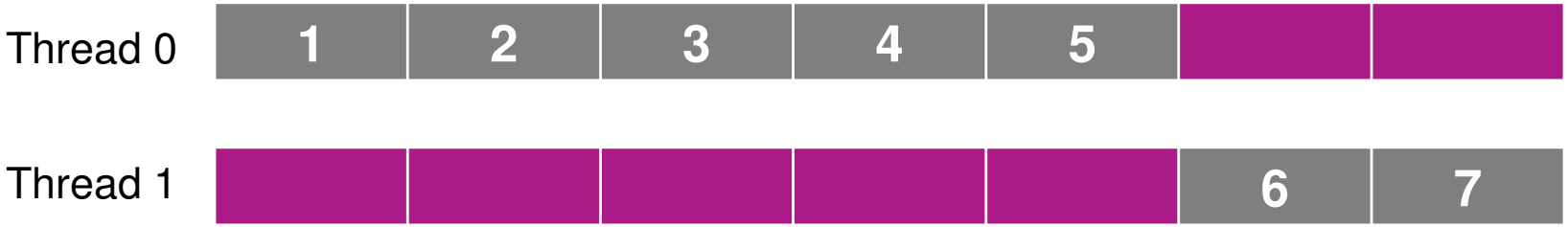
Potential Need to Increase Virtual CPUs



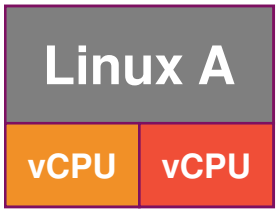
- Consider a guest that hits maximum of its virtual resources
- With a single core, it can receive 10 time slices, but only 7 with SMT, as there is only one virtual CPU to dispatch

IFL (SMT disabled) – Time Slice Rate: 10

IFL (SMT enabled) – Time Slice Rate: 7



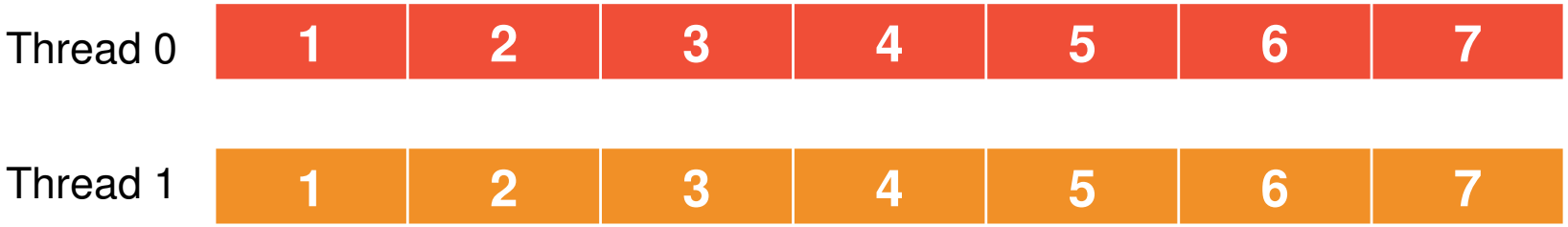
Potential Need to Increase Virtual CPUs …



- **Taking that guest and giving it a second virtual CPU allows additional work to be completed (if application can exploit multiple virtual CPUs)**

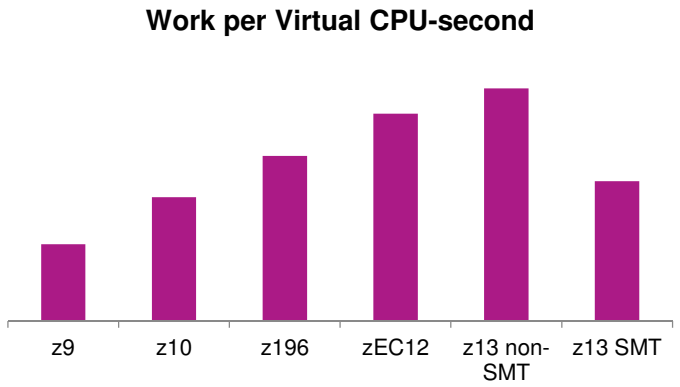
IFL (SMT disabled) – Time Slice Rate: 10

IFL (SMT enabled) – Time Slice Rate: 7



SMT - CPU Pooling Implications

- With SMT enabled
  - CAPACITY limit for CPU pools is defined as processing power of a number of IFL cores …. but limit enforcement is based on thread utilization (raw time)
  - In some cases, guests in a CPU pool will not be able to complete the same amount of work as without SMT with the same capacity limit
    - Capacity limits for CPU pools might need to be increased
    - More problematic when trying to match experience from zEC12 processor than from older, slower processors



Prorated Core Time

- Prorated core time divides core dispatch time proportionally among the threads dispatched in an interval
  - Full time charged while a vCPU runs alongside an idle thread
  - Half time charged while a vCPU is dispatched beside another active thread
- Therefore
  - CPU pool capacity consumed as if by cores
  - Suitable for core-based software licensing
- When SMT is enabled, prorated core time will be calculated for users who are
  - In a CPU pool limited by the CAPACITY or LIMITHARD option
  - Limited by the SET SHARE LIMITHARD command

  (currently raw time is used; raw time will continue to be used when SMT is disabled)
- QUERY CPUPOOL will report capacity in terms of cores of processing power instead of CPUs
- Prorated core time will be reported in monitor records and new Type F accounting record
- Watch for APAR VM65680

Summary

- CPU Pooling offers greater control over resource allocation
  - By workload
  - By department
  - By software product

- Together with ILMT 9.0.1, can limit software license costs, particularly where multiple software products run in the same z/VM system
  - Enables organic growth of individual workloads
  - Avoids paying for capacity not used by a software product
  - Broadens options for workload consolidation, lowering overhead and administrative costs

- New implications for capacity and licensing with IBM z13 and Simultaneous Multithreading
  - Watch for Prorated Core Time enhancement

More Information

- IBM z Systems Software Pricing
  - http://www-03.ibm.com/systems/z/resources/swprice/subcap/linux.html
- Processor Value Unit (PVU) Licensing for Distributed Software
  - http://www-01.ibm.com/software/passportadvantage/pvu_licensing_for_customers.html
- Passport Advantage Sub-Capacity FAQ:
  - http://www.ibm.com/software/passportadvantage/subcapfaqov.html
- Virtualization Capacity License Counting Rules
  - http://www.ibm.com/software/passportadvantage/Counting_Software_licenses_using_specific_virtualization_technologies.html
- ILMT 9.0.1 Blog on August Update with new CPU pooling support
  - http://ibm.biz/cpupoolilmt
- IBM Redpaper – Simplify Software Audits and Cut Costs by Using the IBM License Metric Tool (September 2014)
  - http://www.redbooks.ibm.com/abstracts/redp5107.html?Open
- ILMT Youtube page
  - https://www.youtube.com/user/IBMLicenseMetricTool

Thanks!

Romney White
IBM
z Systems Architecture and Technology
romneyw@us.ibm.com