Goal-oriented dynamic performance management of KVM for IBM z Systems virtual server CPU resources

Yüksel Günal
ygu@us.ibm.com

Anthony Giorgio
agiorgio@us.ibm.com

IBM Systems
Poughkeepsie, NY, USA

October 18th, 2016

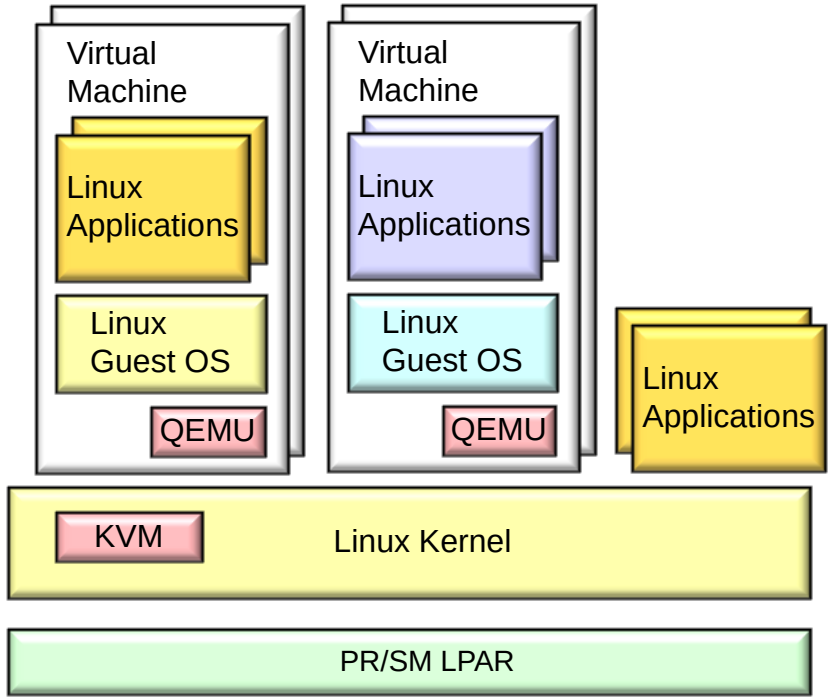
Outline

- Introduction to KVM for IBM z Systems
- Objectives of IBM z Systems Hypervisor Performance Manager (zHPM)
- zHPM Functional Capabilities
  - Concept of a workload resource group
  - Concept of a performance policy
  - Concept of a service class
  - Monitoring workloads
  - Basics of CPU management
  - CPU-critical support
  - How do cgroups CPU shares work?
  - RESTful APIs to interact with zHPM
- Usage Case 1
- Usage Case 2
- Demo
- References
- Q & A

KVM for IBM z Systems

- Kernel-Based Virtual Machine
- An open source hypervisor implemented in the Linux kernel
  - Linux provides the base capabilities
  - KVM turns Linux into a hypervisor
  - QEMU provides I/O device virtualization and emulation

- KVM for IBM z Systems runs in its own LPAR



What problem is zHPM trying to solve?

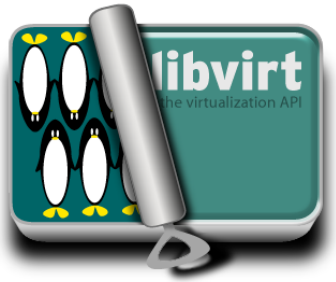
zHPM protects important workloads while maintaining high utilization of KVM for IBM z Systems CPU resources.

- Users define performance goals for line of business applications, and zHPM works to ensure those goals are met.
- zHPM simplifies performance management by automatically monitoring applications and shifting system resources around as necessary.
- This real-time resource rebalancing works to protect the most important work from delays due to resource contention.

What does zHPM bring to the table?

- It introduces the concept of a workload resource group
- It supports performance policy based resource monitoring
- It introduces the concept of a goal-oriented performance policy.
  - This assigns business importance levels and performance objectives to virtual servers
- It support goal-oriented CPU resource management
  - Dynamic management of cgroups CPU shares based on performance objectives and business importance levels

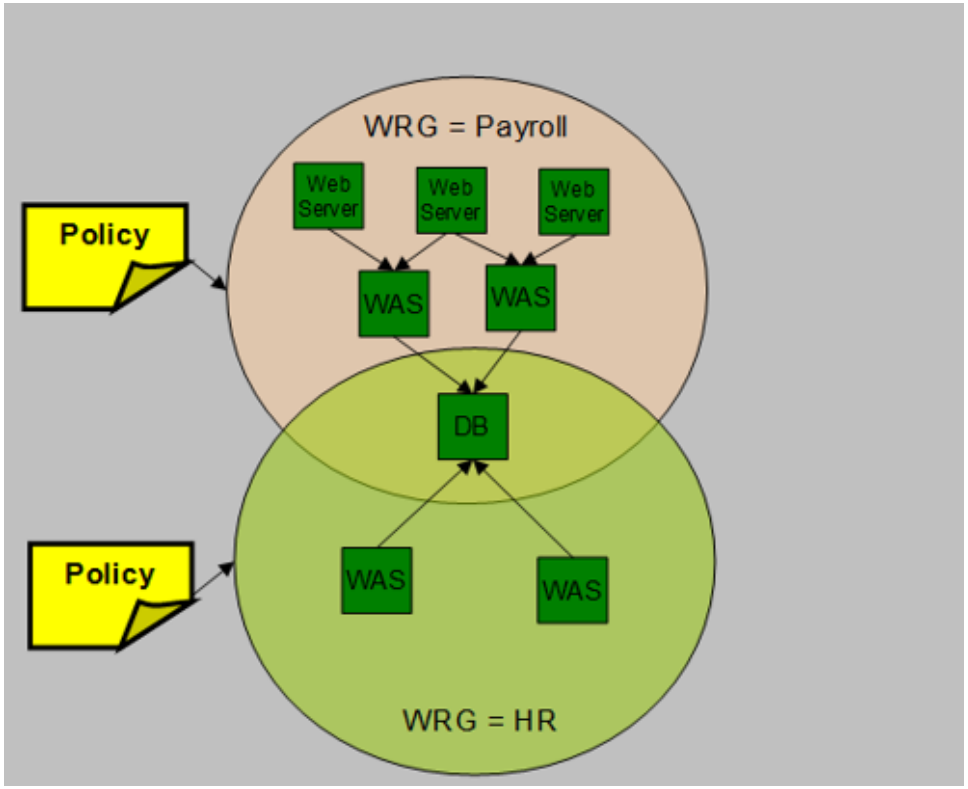
What is zHPM?



- z Systems Hypervisor Performance Manager
- A multi-threaded Java program with some native libraries
- Interacts with libvirt (virtualization library) and the Linux kernel
- Supports policy-based goal-oriented monitoring and management of CPU resources
- Shipped as part of KVM for IBM z Systems
  - Optionally enabled (does not run by default)
- Scope of management is single KVM for IBM z Systems instance
- zHPM will have no knowledge outside of its KVM instance
- Controlled through RESTful Web Services APIs and CLI
  - APIs
    - Point of integration with higher-level virtualization management solutions
    - Support for scripting
    - Fully documented external interface
  - CLIs provide support for local administration

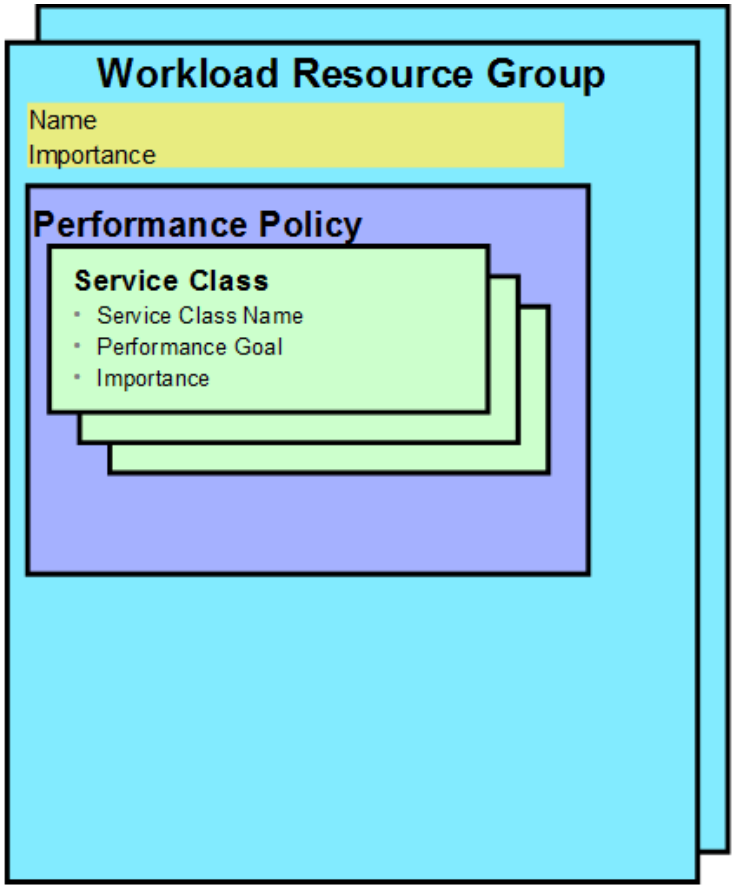
Workload Resource Group

- A **workload resource group** (WRG) is a grouping mechanism and “management view” of virtual servers supporting a business application
- Provides the context within which associated platform resources are presented, monitored, reported, and managed
- In the figure we show a payroll workload and an HR workload
- Each WRG is assigned a single Performance Policy



Performance Policies

- A **performance policy** defines performance objectives for virtual servers in a WRG
  - Conceptually similar to simplified z/OS WLM policy
  - Provides basis for monitoring and management of platform resources used by virtual servers in a WRG
- Policy structure
  - A **service class** defines a performance goal and importance and a set of classification rules
  - Currently supports velocity goals
  - A policy contains a set of service classes
  - Virtual Servers within the workload are assigned to a service class



Service Classes

- Services classes are used to:
  - Assign performance objectives to guests with a WRG
    - Velocity goal and relative importance within WRG
  - Differentiate guests within the WRG. For example:
    - One service class for the WRG's web servers
    - One service class for the WRG's app servers
    - One service class for the WRG's database servers
- Guests are assigned to service classes based on a set of classification rules
  - These rules are defined by the user

Workload Resource Group Importance

- Defines how important it is to achieve the goals in a workload resource group's performance policy
  - Same basic definition as in z/OS WLM
  - Defined as *Highest, High, Medium, Low, Lowest*

- WRG importance controls the order zHPM looks for resources when helping a guest which is not achieving their performance goal
- zHPM will steal resources from the following tiers, in order:
  1. Guests achieving their performance goal**,** if the action will not cause donor guests to miss their goals
  2. Guests not achieving performance goals**,** in **less important** WRGs, starting from least important
  3. Guests in **equally important** WRGs**,** if the action will equalize goal achievement

Velocity Goal

- **Velocity** is the measure of impact on the guest in a service class from contention with other guests for resources
- Same basic definition as z/OS WLM
- The higher a service class velocity, the less delays the guests in the service class are experiencing from resource contention
- Internally calculated as:
  - (CPU used * 100) / ((CPU used) + (CPU delay time))
- Specified in performance policy in named ranges:
  - *Fastest, Fast, Moderate, Slow, Slowest*
- A CPU delay time of 0 means a velocity of 100
- A CPU used time of 0 means a velocity of 0

Monitoring and Reporting

- Reporting shows usage of hypervisor resources in a WRG context
- Performance goal vs actual performance reporting
  - Easily identify WRGs not achieving performance goals
- Drill down from overall WRG “health” view to contributions of individual virtual servers
  - Quickly isolate virtual servers contributing to performance issues
  - From Workload Resource Group view monitor how each service class is performing
  - For each service class monitor the associated virtual servers

Basics of CPU Management

- Objective function to optimize
  - **Performance Index** (PI) is calculated based on the *performance objective* defined in a performance policy and *actual performance*
    - PI = (velocity goal) / (observed velocity)
    - If PI <= 1, performance goals are met
    - If PI > 1, performance goals are missed
    - Function of CPU utilization and CPU delay.
    - Greater velocity == better performance.
    - Performance goals defined in terms of velocity goal in performance policies.
- Management knob is **cgroups CPU shares**, a relative resource allocation control
- Higher importance workloads that miss performance goals are helped by moving CPU shares from virtual servers in lower or same importance workloads with better performance
  - In the case of same importance, the pain is spread evenly
  - CPU shares moved are calculated based on PI improvement projections

How do cgroups CPU Shares work?

- Relative resource allocation knob

- Consider a KVM hypervisor with 4 CPUs and 2 virtual servers, VS_A and VS_B
  - Assume VS_A has 1024 CPU shares and VS_B has 4096 CPU shares
    - VS_A = (4 * 1024 / (1024 + 4096)) = 0.8 CPUs
    - VS_B = (4 * 4096 / (1024 + 4096)) = 3.2 CPUs

- Note that the calculations above assume the virtual servers have enough virtual CPUs assigned and enabled to support the CPU time that their CPU shares entitle them to.
- For instance, VS_B should be defined with at least 4 virtual CPUs to be able to consume 3.2 CPUs.
  -
- The Linux scheduler will enforce these CPU limits for us. Note that if VS_B does not use its entire CPU allotment, the scheduler is free to let VS_A utilize the extra cycles.

Support for CPU Critical Workloads

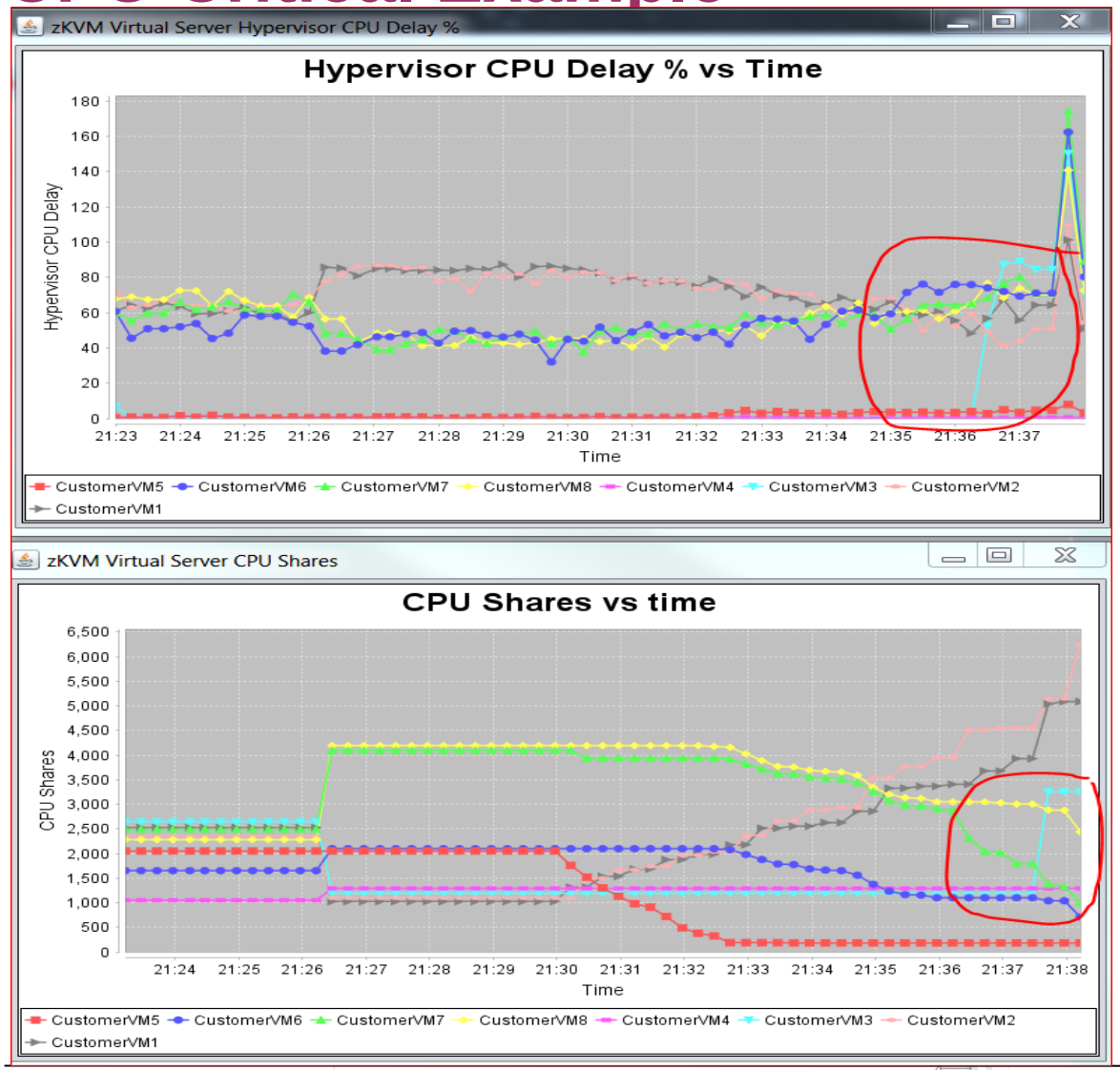
- **CPU critical** is a service class attribute that specifies whether CPU allocation of virtual servers in the service class should be managed aggressively.
  - CPU-critical attribute does not override business importance.

- Virtual Servers associated with a service class designated as CPU Critical will be monitored for sharp changes in workloads.

- If a CPU-critical virtual server suddenly experiences *sharp changes* in hypervisor CPU delays, its CPU shares will be adjusted by a large amount rather than rely on CPU management to do the job gradually.

- Do not overuse this feature, because setting it impacts zHPM's ability to efficiently balance available CPU resources across the zHPM instance.

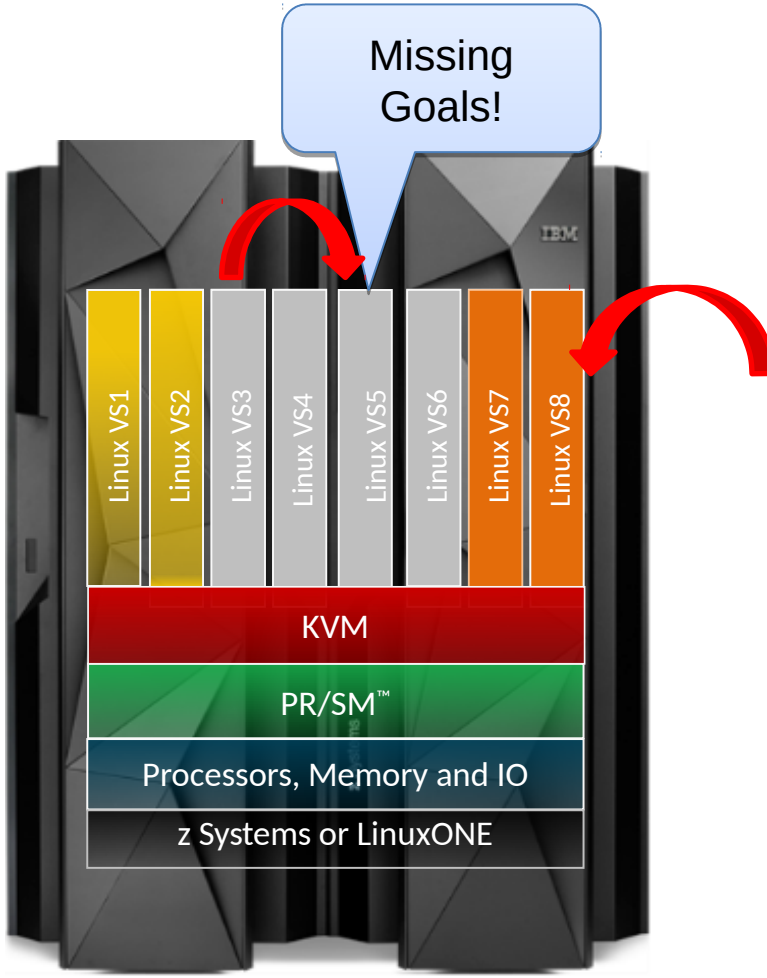
CPU Critical Example



- Note in the upper graph on the left the sharp spike in CPU delay experienced by virtual server **CustomerVM3** (light blue)
- Note in the lower plot how zHPM takes an immediate action to remedy the performance issue with this virtual server by sharply increasing its CPU shares.

Managing Resources

- Manage CPU resources across virtual servers (VS) on a hypervisor to achieve performance goals
  - Detect that a VS in a WRG is not achieving its goals
  - Determine that the VS performance can be improved with additional resources
  - Project impact of reallocating resources on all affected VS
  - If good trade-off based on policy, redistribute processor resources
  - Current support for CPU management, potential to extend to other resources



RESTful APIs to interact with zHPM

- Web services APIs for workload resource group (WRG) operations
  - Add virtual server to a WRG
  - Remove a virtual server from a WRG
  - Create WRG
  - List WRGs
  - Get WRG properties
  - Delete WRG
  - List performance policy for WRGs
  - List WRG detail for all WRGs
  - Update performance policy for a WRG

- Virtual server operations
  - List virtual servers known to zHPM
  - Get virtual server properties
  - Update virtual server properties

RESTful APIs to interact with zHPM (cont.)

- Metrics
  - Get raw metrics for the hypervisor
  - Get calculated metrics for the hypervisor
  - Get raw and calculated metrics for workload resource groups
  - Get velocity goal range for mappings
  - Get virtual server raw and calculated metrics for a workload resource group

- APIs for Virtual server CPU management
  - Get current CPU management setting
  - Enable / Disable CPU management
  - Report dynamic resource adjustments

- APIs for zHPM diagnostics
  - Generate diagnostic dump
  - List current trace settings
  - Update current trace settings

Use Case 1: Hypothetical Scenario

In this basic scenario, the hypothetical customer has a set of 6 virtual servers running development, test, and production workloads.

- There are 6 virtual servers (*VS1-VS6*) in a KVM for IBM z Systems™ hypervisor, and all are running WebSphere® Application Server (WAS).
  - Two of these virtual servers, ***VS1 and VS2***, are running WAS for a development team.
  - Two of these virtual servers, ***VS3 and VS4***, are running WAS for a test team.
  - Two of these virtual servers, ***VS5 and VS6***, are running WAS in production.

Use Case 1: Identify Workload Resource Groups

- ***WasDev*** WRG for development virtual servers:
  - is associated with virtual servers ***VS1 and VS2***
  - has business importance level of **low**
  - has a user-defined service class called *WasDevServiceClass* with a business importance of **medium** and a performance goal of **moderate**.

- ***WasTest*** WRG for test virtual servers:
  - is associated with virtual servers ***VS3 and VS4***
  - has business importance level of **medium**
  - has a user-defined service class called *WasTestServiceClass* with a business importance of **medium** and a performance goal of **moderate**.

- ***WasProd*** WRG for production virtual servers:
  - is associated with virtual servers ***VS5 and VS6***
  - has business importance level of **highest**
  - has a user-defined service class called *WasProdServiceClass* with a business importance of **medium** and a performance goal of **fast**.

Use Case 1: Policies

- Sample Policy File for ***WasProd*** workload resource group, WasProd.pp

```
{
    "performance-policy": {
        "perf-policy-info": {
            "name": "Production_Policy",
            "description": "WAS Production Performance Policy",
            "business-importance": "medium"
        },
        "service-classes": [{
            "name": "WasProdServiceClass",
            "description": "WAS Production Service class",
            "business-importance": "highest",
            "velocity-goal": "fast",
            "cpu-critical": true,
            "virtual-server-name-filters": [“.*”]
        }]
}
```

- Additional Documentation

Use Case 1: Setup Workloads and Policies

- Run the following commands to set up the workloads and policies. (The policy files are expected to be in the directory where the commands are run.)

- Set up the development WRG with its policy:

```
# zhpm wrg-create --wrg-name WasDev --description 'WAS
Development Sandbox' --perf-policy-file=./WasDev.pp
```

- Set up the test WRG with its policy:

```
# zhpm wrg-create --wrg-name WasTest --description 'WAS
Test Sandbox' --perf policy-file=./WasTest.pp
```

- Set up the production WRG with its policy:

```
# zhpm wrg-create --wrg-name WasProd --description 'WAS
Production Servers' --perf-policy-file=./WasProd.pp
```

Use Case 1: Associate Virtual Servers with Workloads

- Associate the virtual servers with the corresponding WRGs as follows:
  - Associate virtual servers running development work with ***WasDev*** WRG:

```
# zhpm vs-wrg-add --wrg-name WasDev --vs-name VS1
# zhpm vs-wrg-add --wrg-name WasDev --vs-name VS2
```

  - Associate virtual servers running test work with ***WasTest*** WRG:

```
# zhpm vs-wrg-add --wrg-name WasTest --vs-name VS3
# zhpm vs-wrg-add --wrg-name WasTest --vs-name VS4
```

  - Associate virtual servers running development work with ***WasProd*** WRG:

```
# zhpm vs-wrg-add --wrg-name WasProd --vs-name VS5
# zhpm vs-wrg-add --wrg-name WasProd --vs-name VS6
```

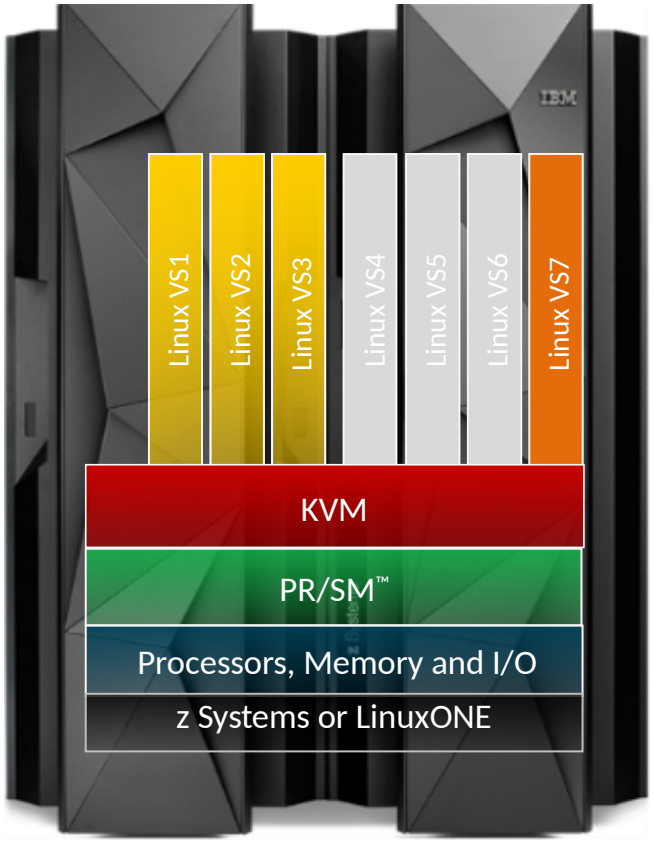
- Enable CPU management. Then zHPM will optimize CPU resource utilization based on business importance and performance goals.

```
# zhpm config --cpu-mgmt on
```

Use Case 2

- This scenario assumes that the hypothetical customer is running two sets of three-tiered Web applications and some CPU intensive batch workloads.
- The customer has a set of virtual servers running development, test, and production workloads.

- There are seven virtual servers (*VS1-VS7*) in a KVM for IBM z Systems™ hypervisor.
- ***VS1, VS2, and VS3*** are supporting three-tier Web transactions by running IHS, WAS, and DB2®, respectively. These transactions are considered to be *very important* to the business.
- ***VS4, VS5, and VS6*** are supporting three-tier Web transactions by running IHS, WAS, and DB2, respectively. These transactions are considered to be *less important* to the business as compared to the transactions running in VS1, VS2, and VS3.
- ***VS7*** runs nightly backup processes.



- Additional Documentation

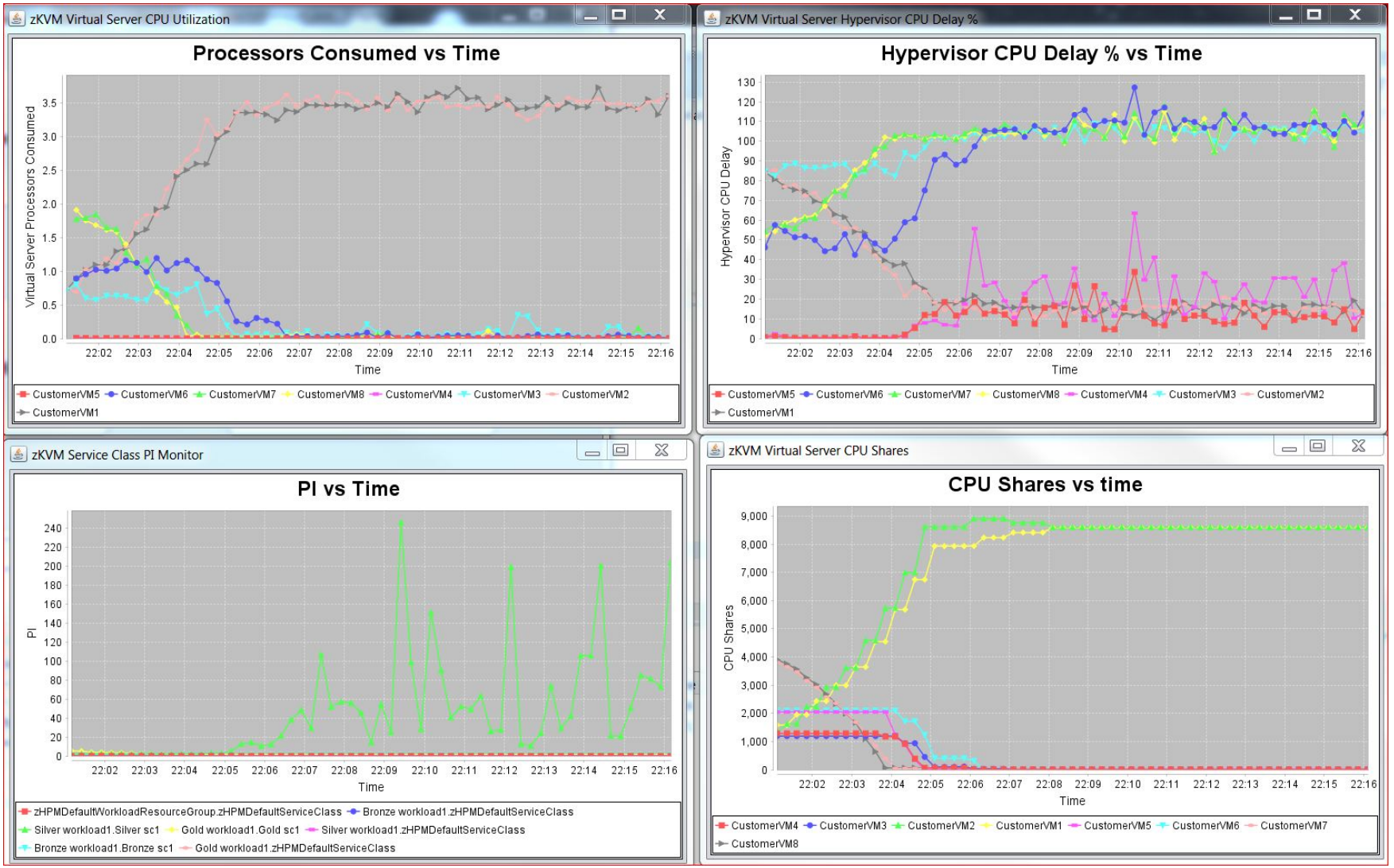
Demo Setup

- Hypervisor has 8 CPUs
- 8 Virtual Servers in the hypervisor, all running CPU-intensive work, like Monte-Carlo simulations
- 3 Workloads
  - ***Gold workload*** has the *highest* business importance and has a service class with a *fastest* performance objective
  - ***Silver workload*** has *medium* level business importance and has a service class with a *moderate* performance objective
  - ***Bronze workload*** is the least important workload and has a service class with a *slowest* performance objective
- Virtual Servers *CustomerVM1* and *CustomerVM2* are part of the ***Gold workload***
- Virtual Servers *CustomerVM7*, *CustomerVM8* are in the ***Bronze workload***
- The rest of the virtual servers are in the ***Silver workload***

Expected Result

- As work increases on important virtual servers, performance objectives start to be missed.
- zHPM takes CPU management actions to help the important work .
  - The **least** important workloads will be impacted **first**.

Expected Run Captured On Real-Time Monitors



Start Demo Run

- Start the demo
  - Turn CPU management on

Summary

- zHPM brings workload-aware virtualization management to KVM for IBM z Systems
  - Concepts derived from z/OS goal-oriented management approach
    - Concept of a workload resource group
    - Policy-based goal-oriented monitoring and management of CPU resources
    - Goals are defined in terms of “velocity”
- zHPM enables exploitation with higher level virtualization management solutions
  - REST APIs allow programmatic interaction with higher level virtualization management tools.
  - CLI allows local administration
- zHPM will evolve capabilities over time
  - New resource management capabilities in the future

Wrangling resources

Dynamic Resource Management with KVM for IBM z Systems

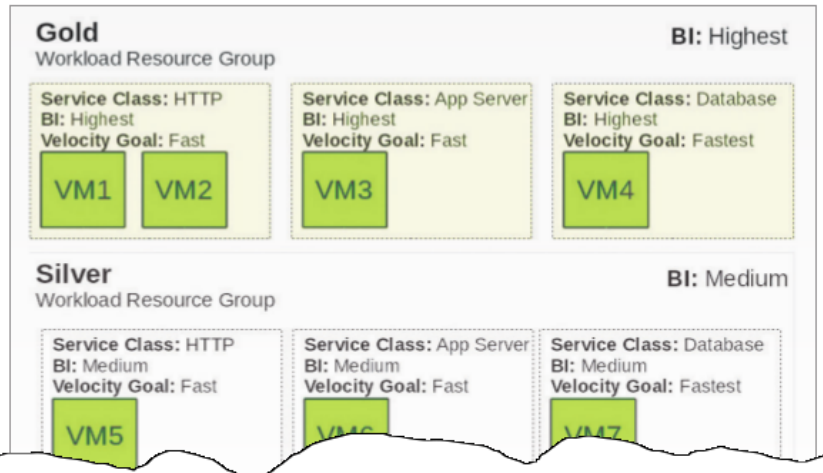
HOT TOPICS
A z/OS NEWSLETTER

BY ANTHONY GIORGIO AND JUSTIN MCCOY

Hypervisor Performance Manager on KVM for IBM z Systems (zHPM) software achieves your performance objectives by transferring CPU resources between workload resource groups.

Protecting business objectives

Consider the following scenario: a business has a hypervisor with a single workload. This is a high-importance workload that drives the hypervisor at a constant CPU utilization rate of 25%. A second, lower importance workload is then started, which consumes as much CPU as it can. The result is that the first workload is given only half the CPU resources it was previously [illegible] causing significant CPU delay.



- zHPM Article in z/OS Hot Topics Magazine

Additional Resources

- Portal: KVM for IBM z Systems
- Product documentation
  - KVM for IBM z Systems: Planning and Installation Guide SC27-8236-00
  - KVM for IBM z Systems: Administration Guide SC27-8237-00
  - Linux on z Systems: Virtual Server Management SC34-2752
  - Linux on z Systems: Virtual Server Quick Start SC34-2753
  - Linux on z Systems: Device Drivers, Features, and Commands for Linux as a KVM Guest SC34-2754
  - Linux on z Systems: Installing SUSE Linux Enterprise Server 12 as a KVM Guest SC34-2755
- Redbook: Getting Started with KVM for IBM z Systems
- Hypervisor Performance Manager on IBM Knowledge Center
- Article on IBM z Systems Development Blog

Questions?

Trademarks & Legal

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

DB2*	ECKD	IBM*	LinuxONE	PR/SM	z13	z Systems
DB2 Connect	FICON*	ibm.com	LinuxONE Emperor	Storwize*	zEnterprise*	z/VSE*
DS8000*	FlashSystem	IBM (logo)*	LinuxONE Rockhopper	XIV*	z/OS*	z/VM*

* 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 .
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.