Systems and Technology Group

z/OS Capacity Provisioning Overview and Implementation

IBM System z10 Capacity Just-in-Time

Horst Sinram
z/OS WLM Development, Boeblingen, Germany
sinram@de.ibm.com

Charles E. Haight
z/OS System Verification Test, Poughkeepsie, N.Y
cehaight@us.ibm.com

Trademarks

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

APPN*
CICS*
DB2*
DB2 Connect
DirMaint
e-business logo*
ECKD
Enterprise Storage Server*
ESCON*
FICON*
GDPS*
Geographically Dispersed Parallel Sysplex

HiperSockets
HyperSwap
IBM*
IBM eServer
IBM e(logo)server*
IBM logo*
IMS
Language Environment*
MQSeries*
Multiprise*
NetView*
On demand business logo

OS/390*
Parallel Sysplex*
PR/SM
Processor Resource/Systems Manager
RACF*
Resource Link
RMF
S/390*
Sysplex Timer*
System z
System z9
System z10
TotalStorage*

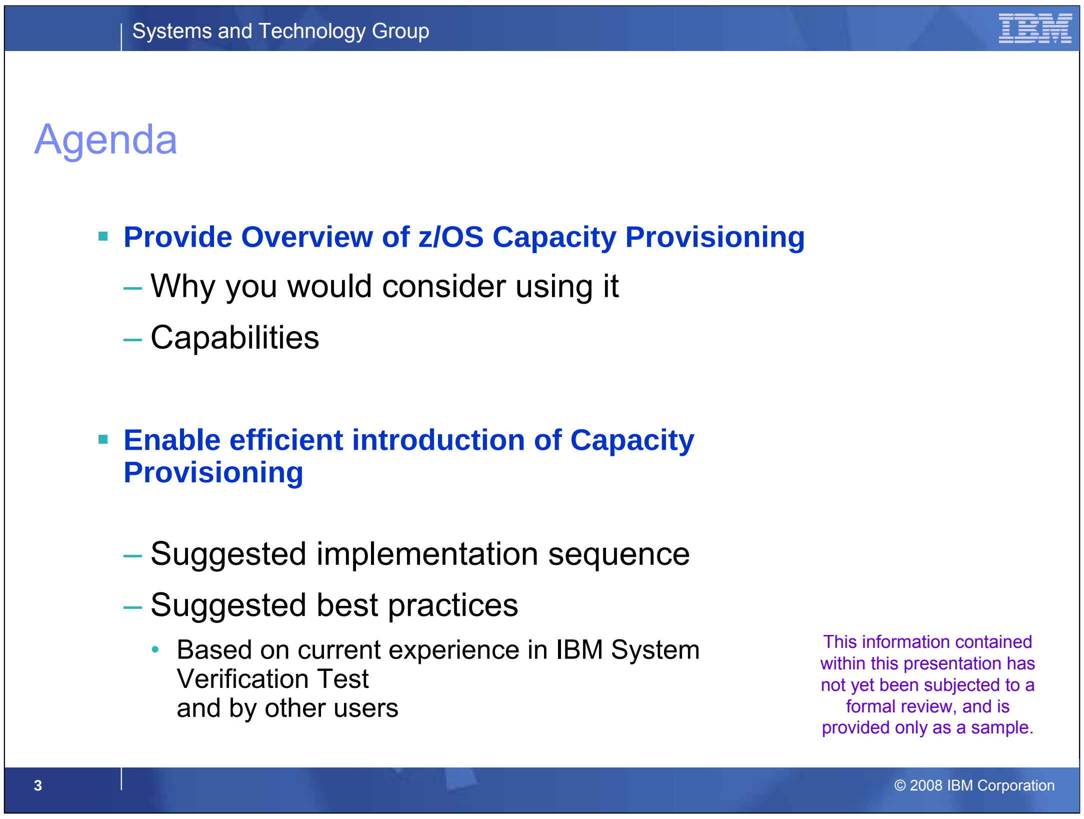
VM/ESA*
VSE/ESA
VTAM*
WebSphere*
z/Architecture
z/OS*
z/VM*
z/VSE
zSeries*
z10
z9

* Registered trademarks of IBM Corporation

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 in the United States, other countries, or both.
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.
Red Hat, the Red Hat "Shadow Man" logo, and all Red Hat-based trademarks and logos are trademarks or registered trademarks of Red Hat, Inc., in the United States and other countries.
SET and Secure Electronic Transaction are trademarks owned by SET Secure Electronic Transaction LLC.
* All other products may be trademarks or registered trademarks of their respective companies.

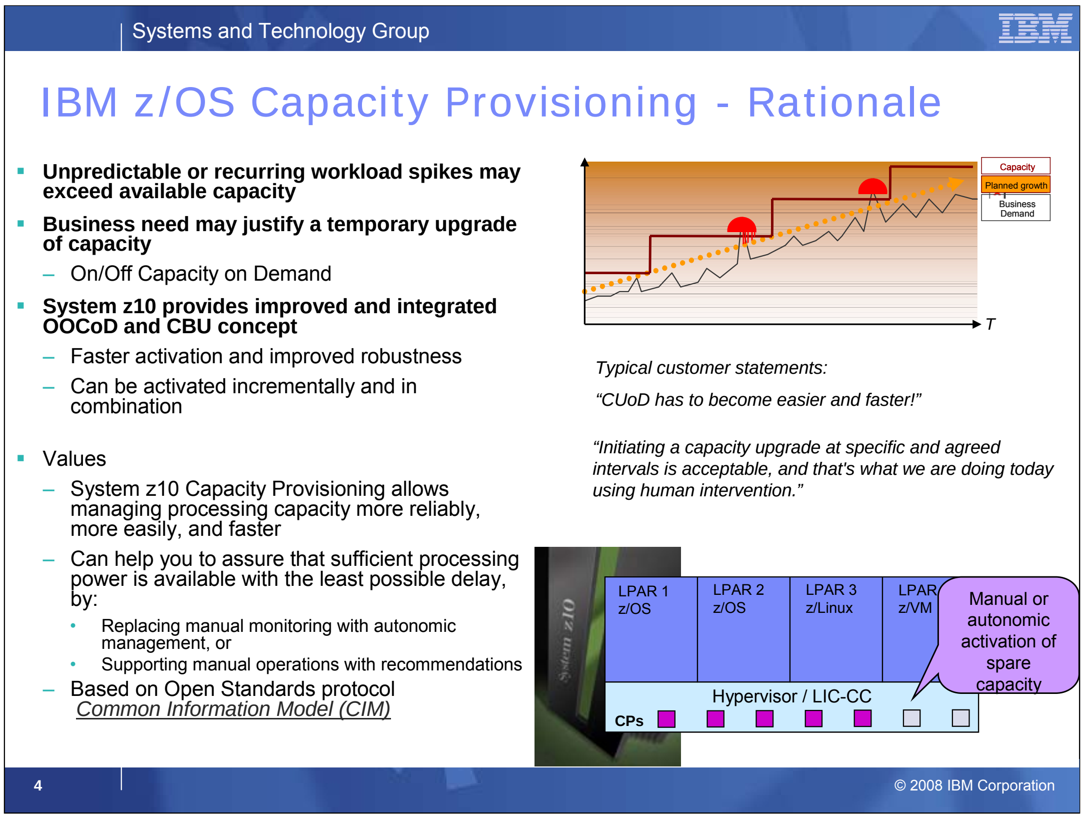
Notes:
Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's 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.



Agenda

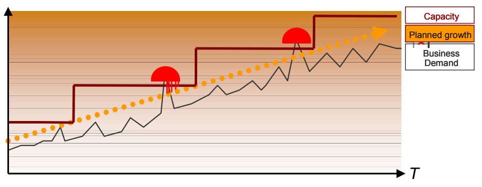
- **Provide Overview of z/OS Capacity Provisioning**
  - Why you would consider using it
  - Capabilities
- **Enable efficient introduction of Capacity Provisioning**
  - Suggested implementation sequence
  - Suggested best practices
    - Based on current experience in IBM System Verification Test and by other users

This information contained within this presentation has not yet been subjected to a formal review, and is provided only as a sample.



IBM z/OS Capacity Provisioning - Rationale

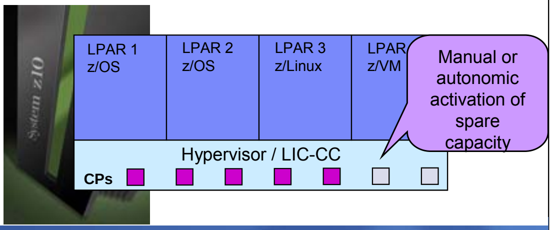
- **Unpredictable or recurring workload spikes may exceed available capacity**
- **Business need may justify a temporary upgrade of capacity**
  - On/Off Capacity on Demand
- **System z10 provides improved and integrated OOCoD and CBU concept**
  - Faster activation and improved robustness
  - Can be activated incrementally and in combination
- Values
  - System z10 Capacity Provisioning allows managing processing capacity more reliably, more easily, and faster
  - Can help you to assure that sufficient processing power is available with the least possible delay, by:
    - Replacing manual monitoring with autonomic management, or
    - Supporting manual operations with recommendations
  - Based on Open Standards protocol *Common Information Model (CIM)*



Typical customer statements:

"CUoD has to become easier and faster!"

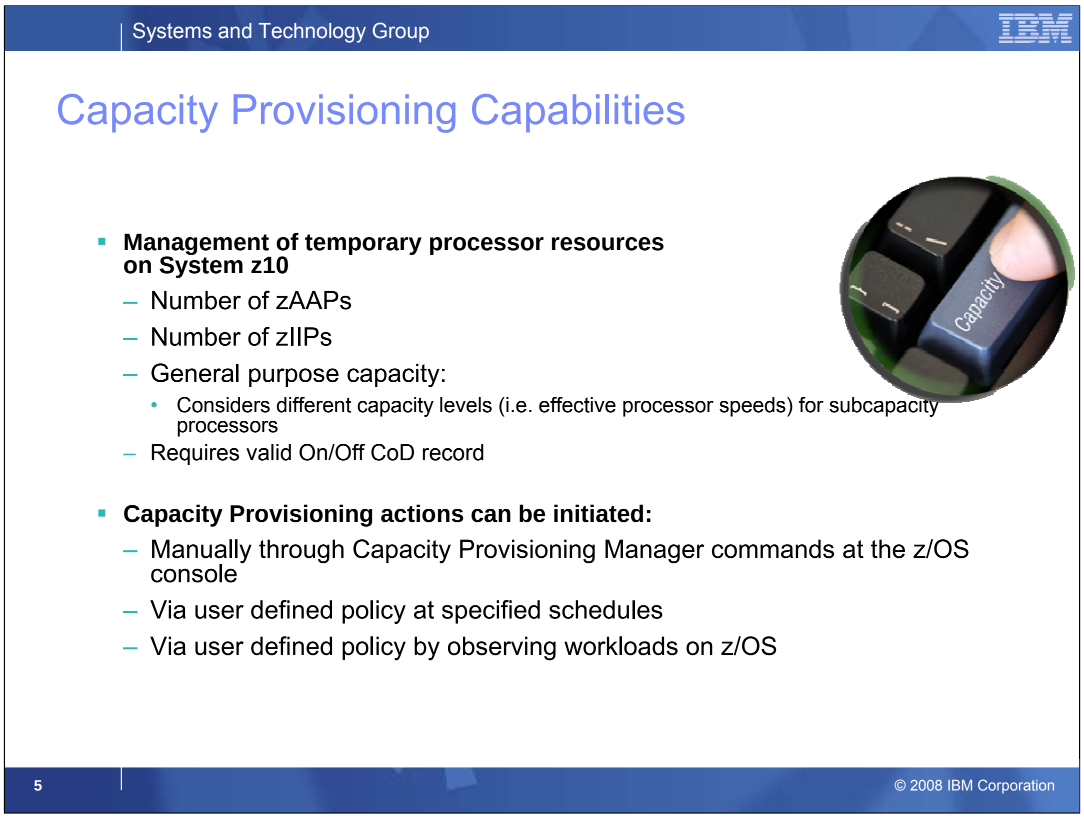
"Initiating a capacity upgrade at specific and agreed intervals is acceptable, and that's what we are doing today using human intervention."



Performance and capacity management on System z™ needs to ensure that the work is being processed according to the service level agreements that are in place. Guaranteeing service levels remains a relatively static task only as long as the workloads that need to be considered are sufficiently stable. However, in many environments workloads may fluctuate considerably over time. As the total workload or the mixture of workloads varies guaranteeing service levels may increasingly getting difficult. On z/OS, Workload Management (WLM) allows the incoming work to be classified with a performance goal and a priority that reflects the business priority of that work. WLM will try to accommodate the goals of all the work in the system.

However, even with an ideal workload management it may not be possible to achieve all specified goals when the total workload increases. In that case trade-offs need to be made. WLM decides which goals may be compromised first based on the assigned importance level. Discretionary, then low importance work will be displaced first. At some point, however, that may no longer be acceptable. In that case the processing capacity needs to be increased to accommodate the grown workload. The capacity change could be implemented via a permanent capacity increase or via a temporary capacity increase for seasonal or unpredictable peak periods. IBM System z provides the capability to quickly and nondisruptively activate additional processor capacity that is built directly into System z servers —IBM Capacity Upgrade on Demand (CUoD) for a permanent increase of processing capability, and IBM On/Off Capacity on Demand (On/Off CoD) for a temporary capacity increase that lets you revert to your previous processing level whenever you wish.

Capacity Provisioning is designed to simplify the management of temporary capacity. **The scope of z/OS Capacity Provisioning is to address capacity requirements for relatively short term workload fluctuations for which On/Off Capacity on Demand is applicable**. It is not a replacement for the Capacity Management process.



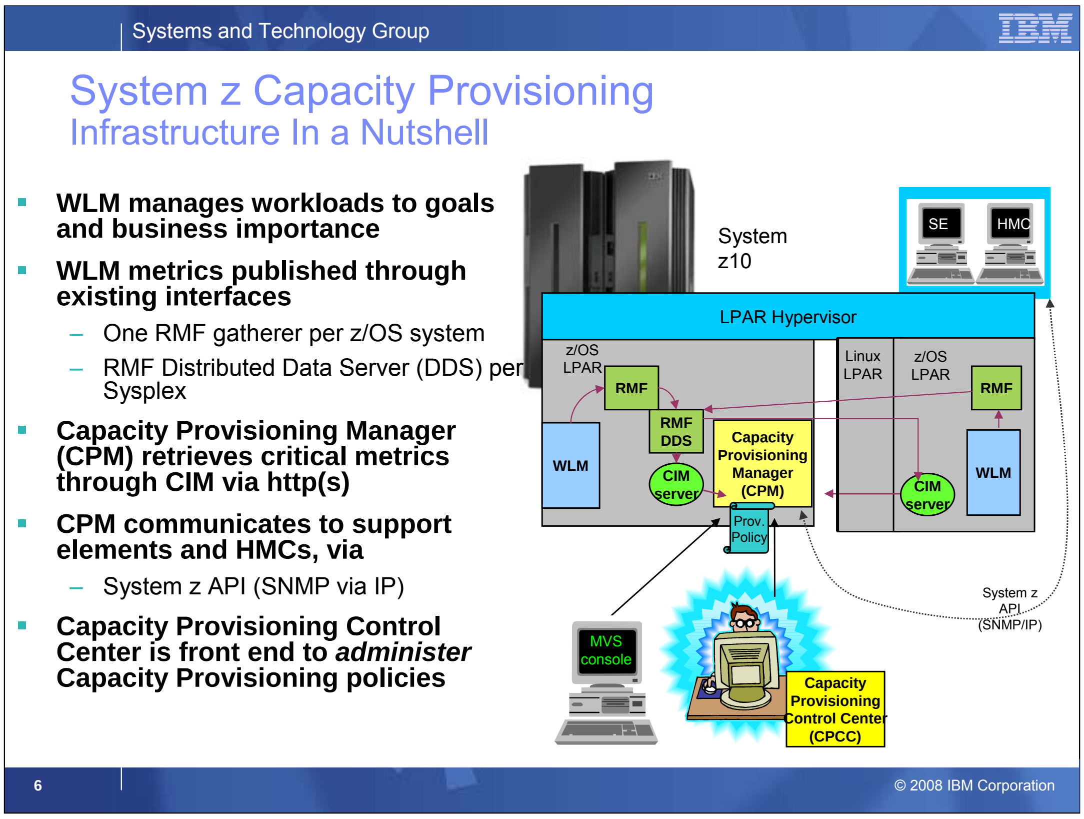
Capacity Provisioning Capabilities

- **Management of temporary processor resources on System z10**
  - Number of zAAPs
  - Number of zIIPs
  - General purpose capacity:
    - Considers different capacity levels (i.e. effective processor speeds) for subcapacity processors
  - Requires valid On/Off CoD record
- **Capacity Provisioning actions can be initiated:**
  - Manually through Capacity Provisioning Manager commands at the z/OS console
  - Via user defined policy at specified schedules
  - Via user defined policy by observing workloads on z/OS

Capacity Provisioning can help you to manage processor capacity on IBM System z10 when a suitable On/Off Capacity on Demand record is available. Capacity provisioning allows you to change the activation level of that On/Off CoD record with respect to general purpose capacity, and the number of zAAP or zIIP processors. For general purpose capacity on subcapacity models it differentiate between „speed“ demand for higher capacity levels, and “unqualified” demand that could be satisfied by a capacity level increase as well as by additional processors.

CPM differentiates between different types of Provisioning Requests

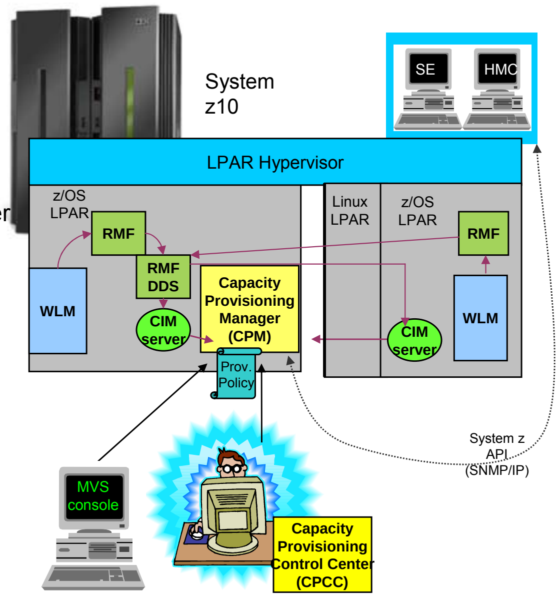
- •Manually through commands
  - •SE/HMC actions still possible, of course
- •Scheduled (time condition without workload condition)
- •Conditional (based on workload condition)



Systems and Technology Group

System z Capacity Provisioning
Infrastructure In a Nutshell

- **WLM manages workloads to goals and business importance**
- **WLM metrics published through existing interfaces**
  - One RMF gatherer per z/OS system
  - RMF Distributed Data Server (DDS) per Sysplex
- **Capacity Provisioning Manager (CPM) retrieves critical metrics through CIM via http(s)**
- **CPM communicates to support elements and HMCs, via**
  - System z API (SNMP via IP)
- **Capacity Provisioning Control Center is front end to *administer* Capacity Provisioning policies**



6

© 2008 IBM Corporation

On each z/OS system WLM manages the workload by goals and business importance

WLM metrics are available through existing interfaces and are reported through (e.g.) RMF Monitor III.

•One RMF gatherer per z/OS system

•Sysplex wide data aggregation & propagation occurs in the RMF distributed data server (DDS)

The RMF CIM providers (and associated CIM models) publish the RMF Monitor III performance data

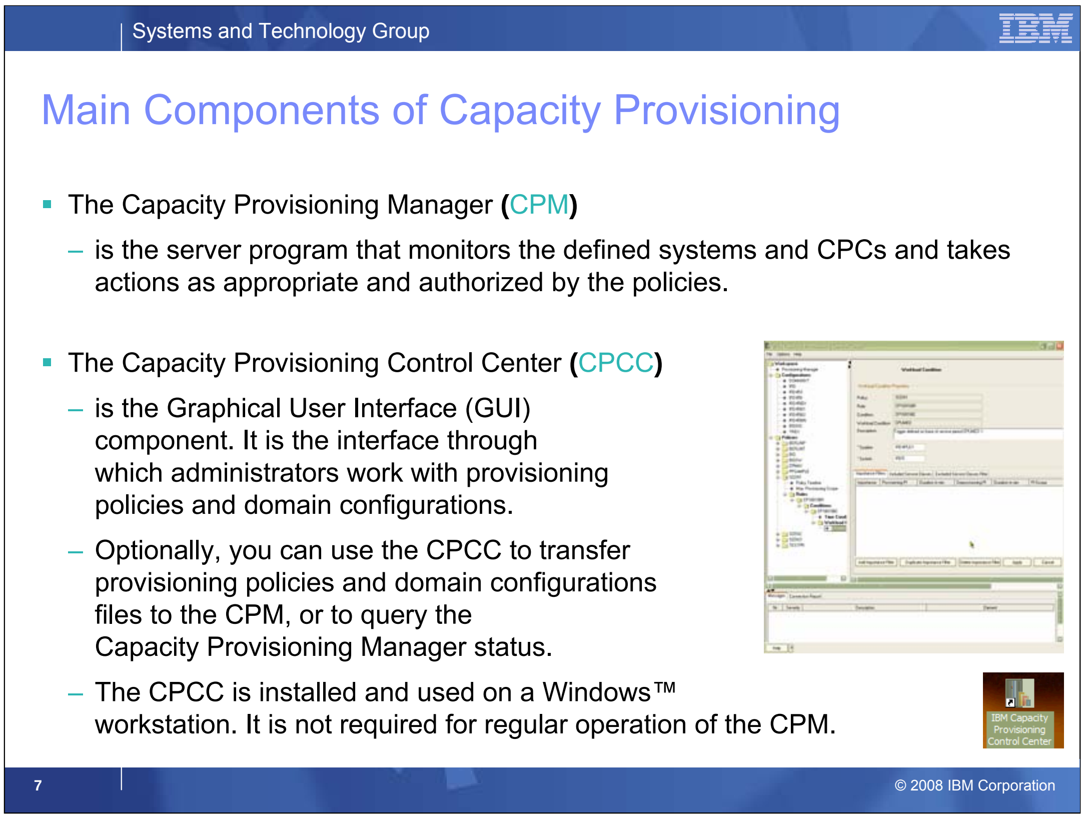
The Capacity Provisioning Manager (CPM) retrieves critical metrics from one or more z/OS systems through CIM via http or https.

The CPM communicates to (local or remote) support elements and HMCs, respectively, via

System z API (SNMP via IP)

The Capacity Provisioning Control Center is the user front end to *administer* Capacity Provisioning policies. It is installed on a Windows ™ workstation. It creates Capacity Provisioning policies in XML format.

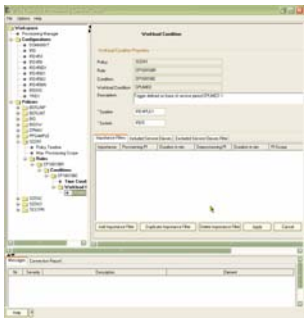
(It is not required for regular CPM operation but only for policy administration).



Systems and Technology Group

Main Components of Capacity Provisioning

- The Capacity Provisioning Manager (CPM)
  - is the server program that monitors the defined systems and CPCs and takes actions as appropriate and authorized by the policies.
- The Capacity Provisioning Control Center (CPCC)
  - is the Graphical User Interface (GUI) component. It is the interface through which administrators work with provisioning policies and domain configurations.
  - Optionally, you can use the CPCC to transfer provisioning policies and domain configurations files to the CPM, or to query the Capacity Provisioning Manager status.
  - The CPCC is installed and used on a Windows™ workstation. It is not required for regular operation of the CPM.



IBM Capacity Provisioning Control Center

7 © 2008 IBM Corporation

This chart summarizes again the two main components of Capacity Provisioning:

1. **The Capacity Provisioning Manager** (**CPM**) which is a started task on z/OS and performs all tasks as defined into its policies.
   It is started via START CPOSERV. When running, it accepts operator commands through the MODIFY interface.

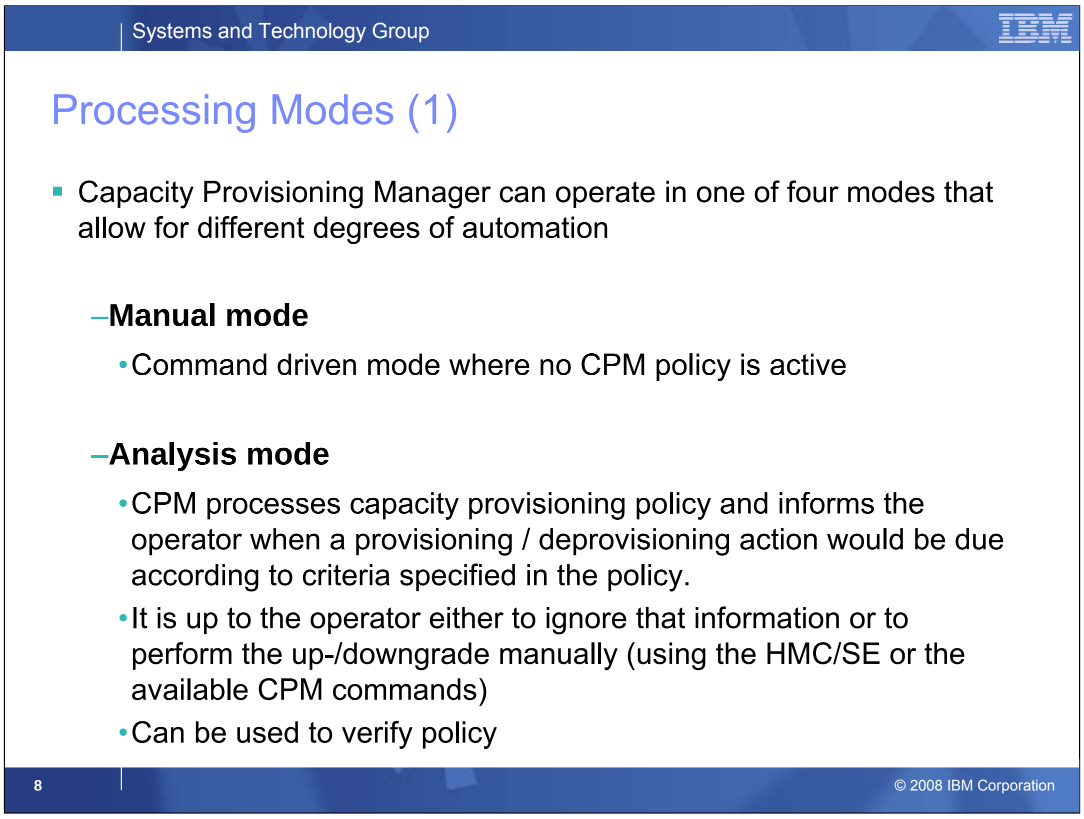
   E.g. F CPOSERV,APPL=<command>

2. **The Capacity Provisioning Control Center** (**CPCC**)

It is shipped with z/OS as /usr/lpp/cpo/pws/cpccsetup.exe. For installation it needs to be downloaded in binary format and executed which starts the install program.

After installation, the program is invoked via its icon, or through the Windows “start” menu.

The two types of policies that the Capacity Provisioning Manager observes are edited or viewed in the CPCC.


Systems and Technology Group

Processing Modes (1)

- Capacity Provisioning Manager can operate in one of four modes that allow for different degrees of automation
  - **Manual mode**
    - Command driven mode where no CPM policy is active
  - **Analysis mode**
    - CPM processes capacity provisioning policy and informs the operator when a provisioning / deprovisioning action would be due according to criteria specified in the policy.
    - It is up to the operator either to ignore that information or to perform the up-/downgrade manually (using the HMC/SE or the available CPM commands)
    - Can be used to verify policy

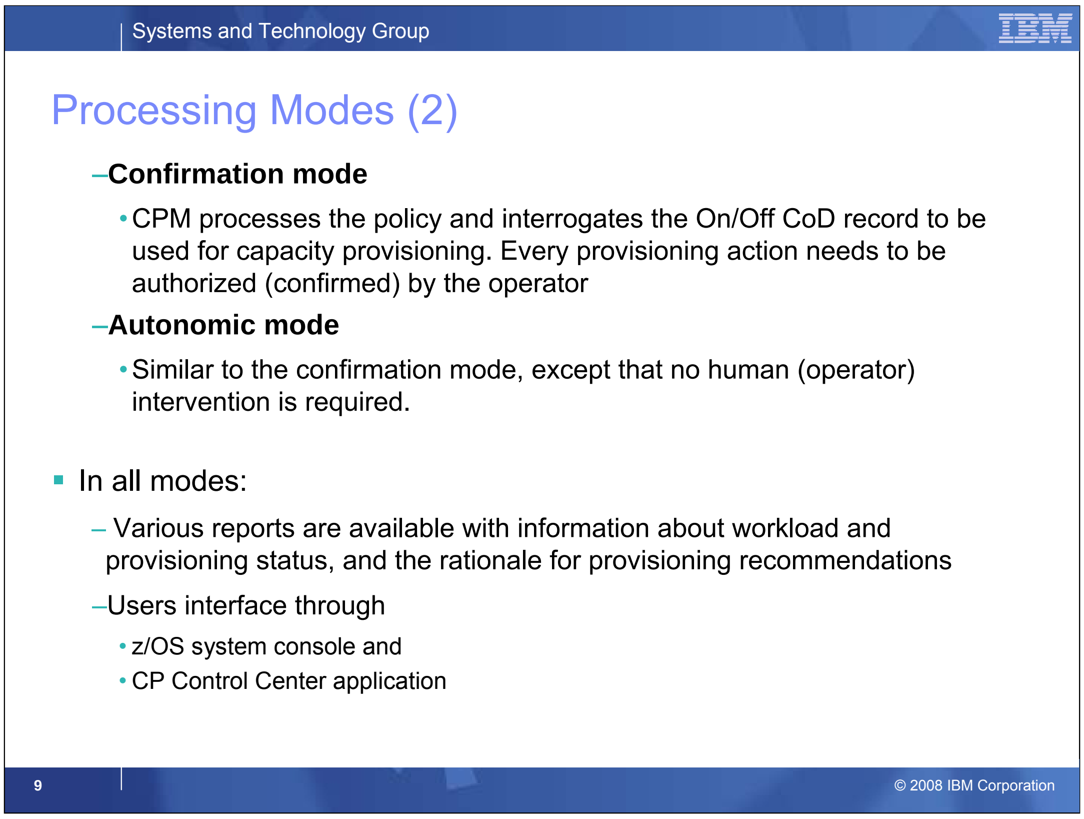
8 © 2008 IBM Corporation

The fully autonomic management of temporary capacity would usually only be implemented when an installation has verified that the CPM management is in line with the objectives.

Even then an installation may feel that it wants to control each single action suggested by the CPM.

In fact the CPM supports four different „processing modes“ that allow for different degrees of automation and serve different purposes:

1) The most basic mode is the ”Manual“ mode. In this mode the CPM is not policy-aware. However, it does listen to console commands and does accept and execute commands. Also, policies may be installed through the CPCC.
2) In the analysis mode the CPM does process a policy. In fact this mode is primarily intended for policy development. So the CPM processes the policy, observes workload and indicates when additional or less capacity of different types would be suggested. However, in this mode the CPM does not observe the hardware capabilities and does not interrogate the installed On/Off CoD records. Consequently the capacity suggestions are not matched with the On/Off CoD record.
   In addition, the “manual“ mode capabilities are available.



Processing Modes (2)

– **Confirmation mode**
  - CPM processes the policy and interrogates the On/Off CoD record to be used for capacity provisioning. Every provisioning action needs to be authorized (confirmed) by the operator

– **Autonomic mode**
  - Similar to the confirmation mode, except that no human (operator) intervention is required.

- In all modes:
  - – Various reports are available with information about workload and provisioning status, and the rationale for provisioning recommendations
  - –Users interface through
    - z/OS system console and
    - CP Control Center application

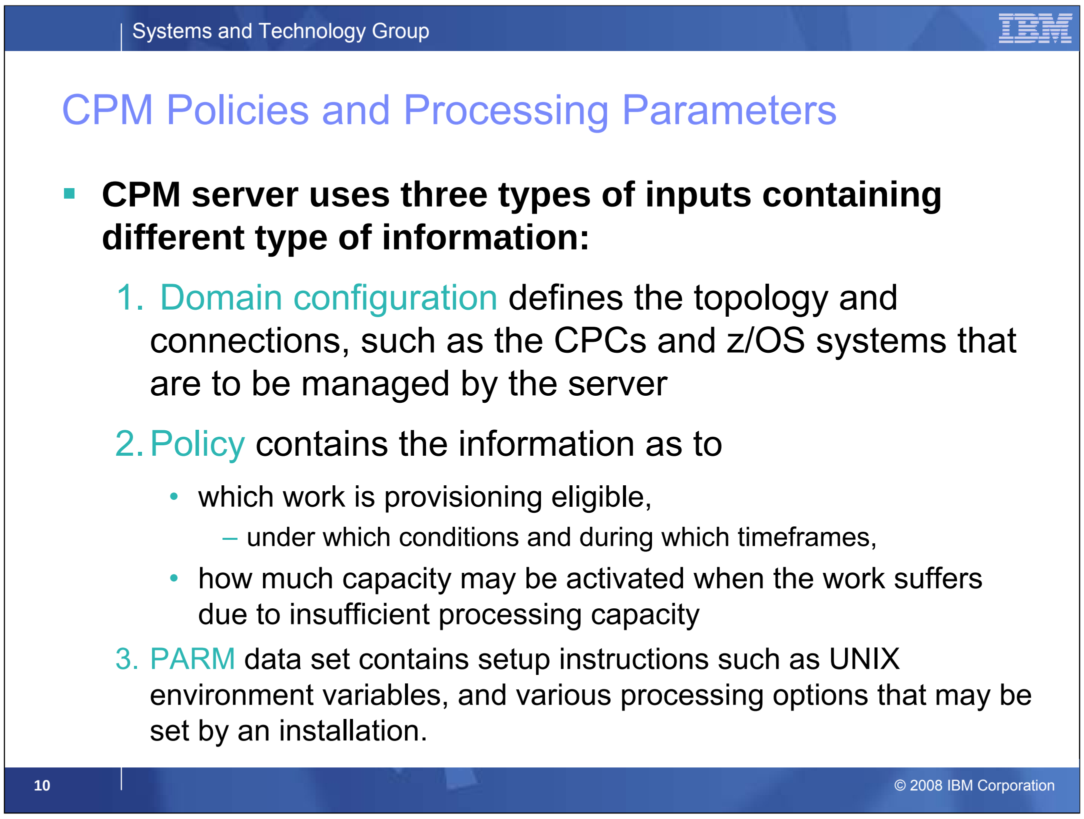
These two modes provide a similar, comprehensive functionality.

The CPM is fully aware of the active policy, interrogates the On/Off CoD record and listens to hardware capacity changes.

In „Confirmation“ mode, workloads are observed, and when a capacity change is warranted the CPM message (WTOR) is issued to the z/OS console. The reply can either be approved, in which case it is implemented right away, or rejected. Also, the reply may be just left unanswered. When the workload situation or policy changes, the CPM reply may be withdrawn or updated. Only approved changes will be implemented.

In autonomic mode, no operator intervention will be required and the capacity setting of the server would be updated automatically, as authorized by the policy and On/Off CoD record.

In all modes, various report commands can be issued to report on the hardware, system configuration and workloads as defined to, and observed by the CPM.



CPM Policies and Processing Parameters

- **CPM server uses three types of inputs containing different type of information:**
  1. Domain configuration defines the topology and connections, such as the CPCs and z/OS systems that are to be managed by the server
  2. Policy contains the information as to
     - which work is provisioning eligible,
       - under which conditions and during which timeframes,
     - how much capacity may be activated when the work suffers due to insufficient processing capacity
  3. PARM data set contains setup instructions such as UNIX environment variables, and various processing options that may be set by an installation.

The CPM processing is controlled by two definitions:

The **domain configuration** describes the domain or topology that the CPM controls, i.e.

- All the servers (CPCs) known to the CPM, and
- All the z/OS images observed by the CPM. Workloads can only be monitored on systems defined into the domain.

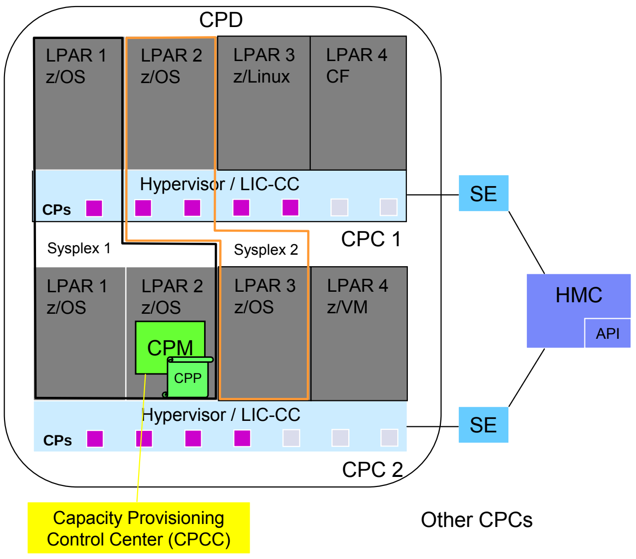
A domain configuration is required for all processing modes, including manual mode.

The **policy** describes the conditions and criteria governing the provisoning and deprovisioning of temporary capacity.

The **parm** member contains environmental setup and processing directives.

The domain configuration and policy are detailed on subsequent charts.

The Capacity Provisioning Domain



- Domain configuration defines the CPCs and z/OS systems that are controlled by an instance of the CPM
- One or more CPCs, sysplexes and z/OS systems can be defined into a domain
- Sysplexes and CPCs do not have to be completely contained in a domain but must not belong to more than one Capacity Provisioning domain
- One active Capacity Provisioning policy per domain
- Multiple Sysplexes and hence multiple WLM service definitions may be involved

Domain configuration defines the CPCs and z/OS systems that are controlled by an instance of the CPM.

Multiple CPCs, sysplexes and z/OS systems can be defined into a single domain.

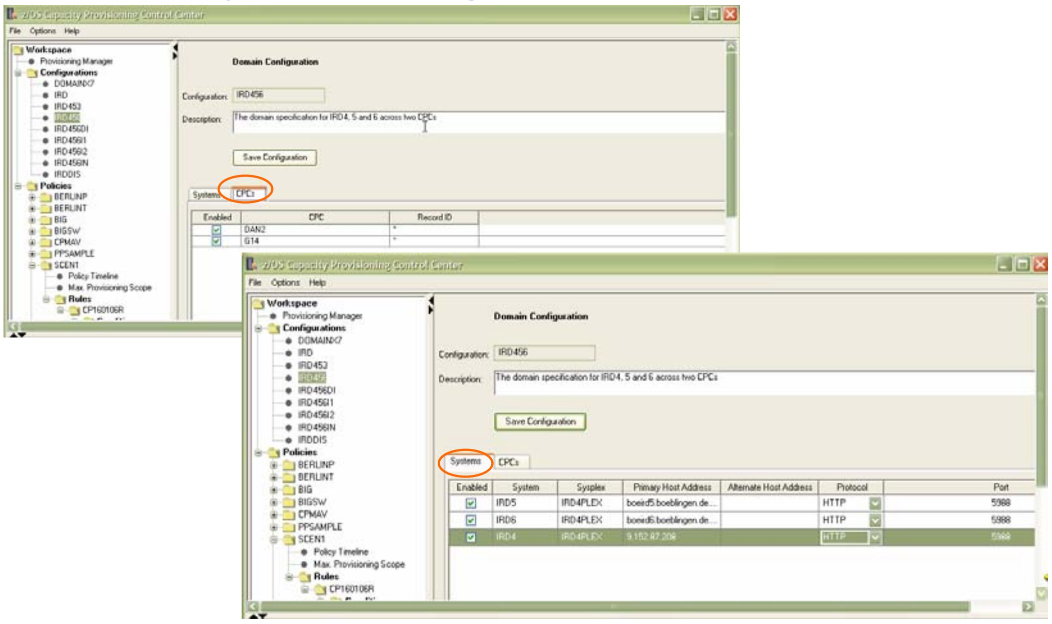
Sysplexes and CPCs do not have to be completely contained in a domain but must not belong to more than one CP domain. In other words, no CPC and no Sysplex must be defined into different domains (and therefore controlled by different CPMs).

At CPM run time, CPCs and systems can be dynamically enabled or disabled for CPM processing via commands.

At any point in time only one Capacity Provisioning policy can be active per domain (or no policy at all).

Because multiple Sysplexes may be defined into a domain the policy must allow for referencing multiple WLM service definitions.

Capacity Provisioning Domain in the CPCC



Domain Configuration

Configuration: IRD456

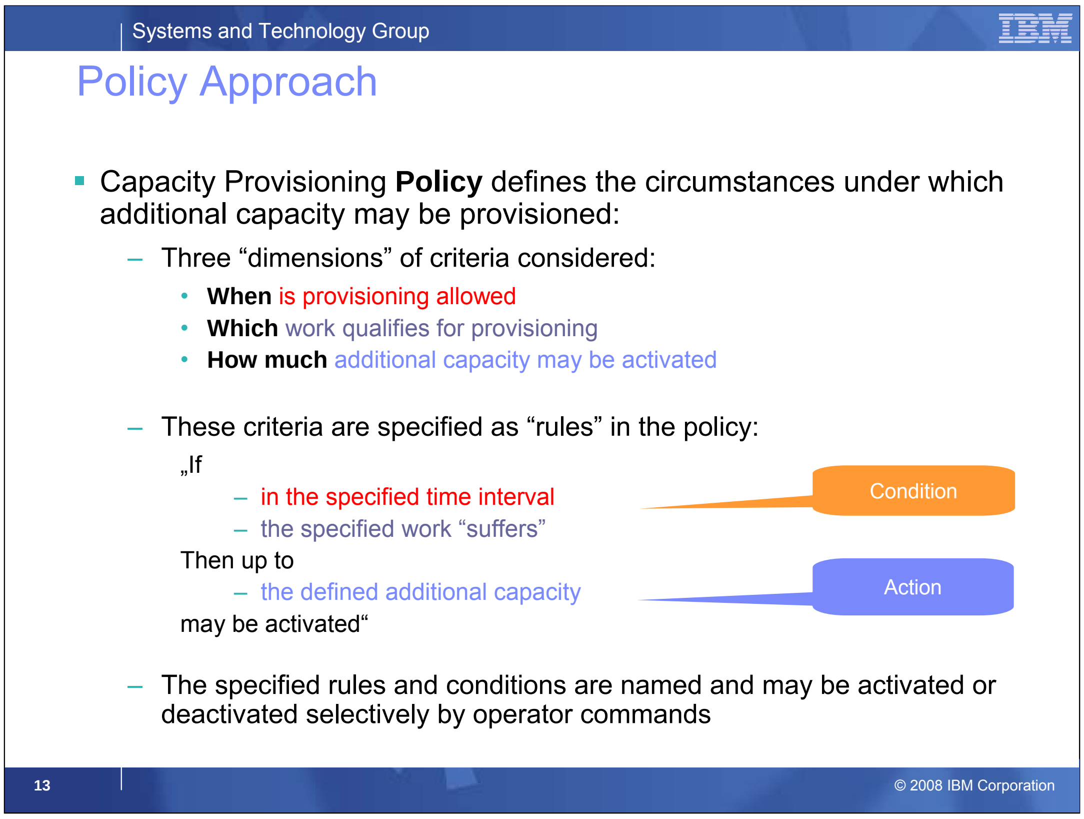
Description: The domain specification for IRD4, 5 and 6 across two CPCs

Enabled	CPC	Record ID
✓	DAN2	*
✓	G14	*

Enabled	System	Sysplex	Primary Host Address	Alternate Host Address	Protocol	Port
✓	IRD5	IRD4PLEX	boeird5.boeblingen.de...		HTTP	5988
✓	IRD6	IRD4PLEX	boeird6.boeblingen.de...		HTTP	5988
✓	IRD4	IRD4PLEX	9.152.87.208		HTTP	5988

On the “CPC” tab of the provisioning domain configuration you define the CPCs that are part of the domain. For each CPC a specific On/Off Capacity on Demand record can be identified that is to be used for activations. A wildcard specification “*” is possible but not recommended when multiple On/Off CoD records are installed at the same time.

On the “Systems” tab you define the systems to be observed (monitored). Each system is identified by the IP name or address and port number on which the CIM server is listening. By default it listens on the network ports 5988 (for plain HTTP) and 5989 (for secure HTTP) .



Policy Approach

- Capacity Provisioning **Policy** defines the circumstances under which additional capacity may be provisioned:
  - Three “dimensions” of criteria considered:
    - **When** is provisioning allowed
    - **Which** work qualifies for provisioning
    - **How much** additional capacity may be activated
  - These criteria are specified as “rules” in the policy:

    „If
      - in the specified time interval
      - the specified work “suffers”

    Then up to
      - the defined additional capacity

    may be activated“
  - The specified rules and conditions are named and may be activated or deactivated selectively by operator commands

Capacity Provisioning Policy defines the circumstances under which additional capacity may be provisioned. Three “dimensions” of criteria are considered:

1. When is provisioning allowed
2. Which work qualifies for provisioning
3. How much additional capacity may be activated

These criteria are specified as “rules” in the policy:

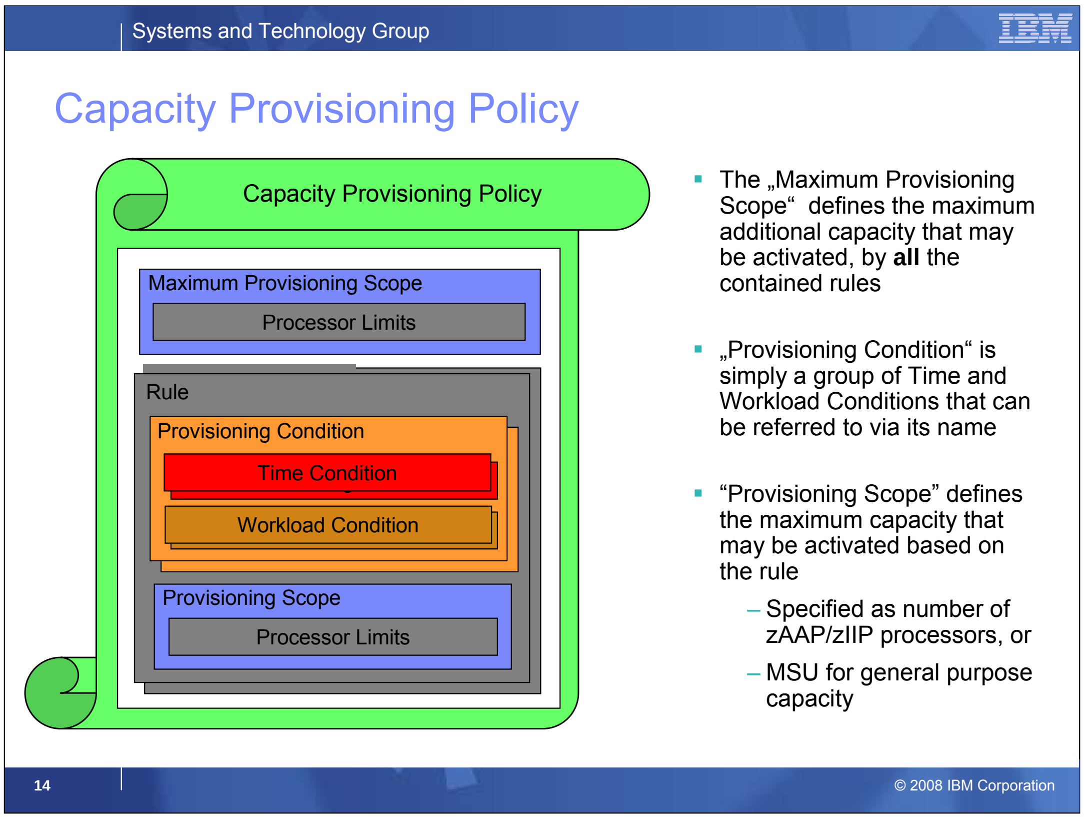
„If

- in the specified time interval
- the specified work “suffers”

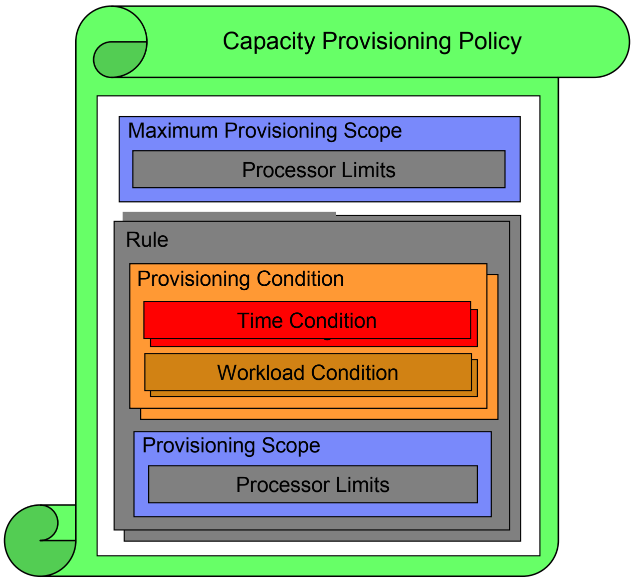
Then up to the defined additional capacity may be activated“

The specified rules and conditions are **named** constructs. At CPM run time, they can be dynamically enabled or disabled via commands without changing the installed policy.

.



Capacity Provisioning Policy



- The „Maximum Provisioning Scope“ defines the maximum additional capacity that may be activated, by **all** the contained rules
- „Provisioning Condition“ is simply a group of Time and Workload Conditions that can be referred to via its name
- “Provisioning Scope” defines the maximum capacity that may be activated based on the rule
  - Specified as number of zAAP/zIIP processors, or
  - MSU for general purpose capacity

Each Capacity Provisioning policy does usually consist of multiple parts that altogether allow for a comprehensive definition of the objectives.

The maximum provisioning scope specifies an upper limit to the capacity that may be activated by all contained rules at any point in time.

Then one or more rules can be defined into a policy.

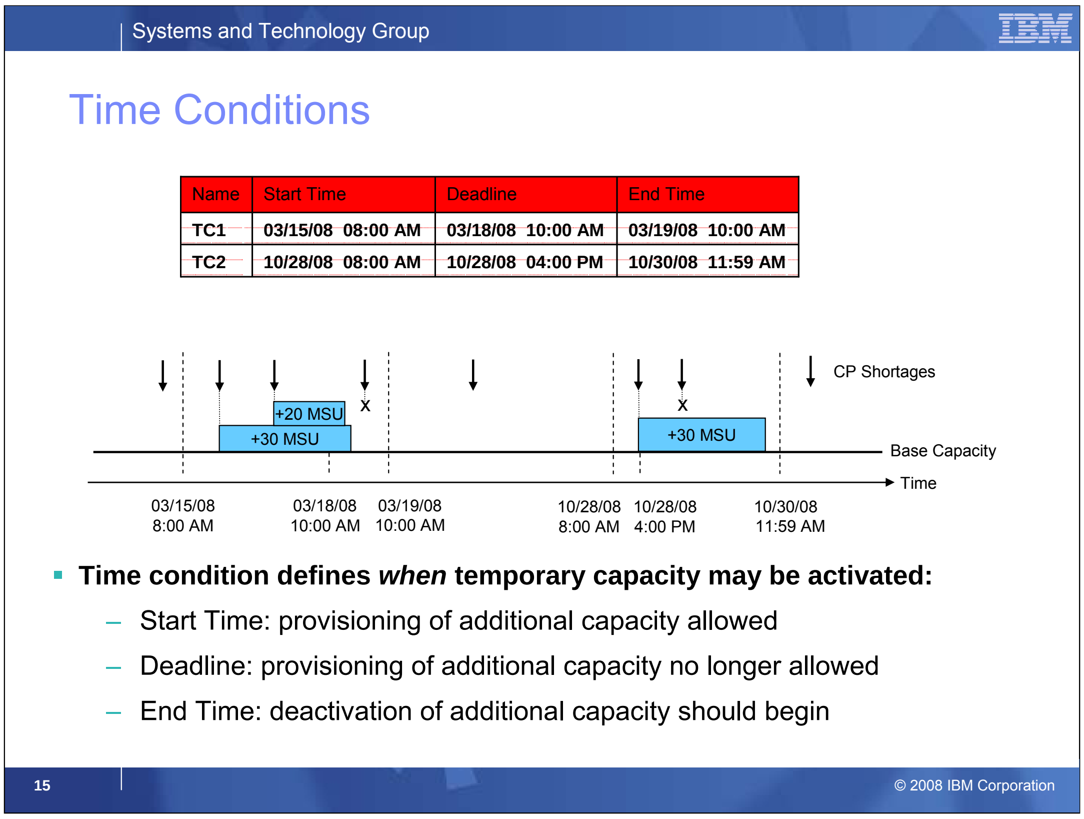
Each rule consists of a „provisioning condition“ and the provisioning scope. The latter defines how much capacity may be activated by that specific rule **at most**. The provisioning scope is expressed in number of zAAP and zIIP processors, and MSU for general purpose processor capacity.

The provisioning condition consists of a time condition and a workload condition:

- The time conditions specify the time interval during which capacity may be added or kept online.
- Optionally, a workload condition specifies for how long a workload must have „suffered“ before adding capacity should be considered.

If no workload condition is specified in a rule then the full capacity as specified in the processor limits will be activated and deactivated at the times defined into the time condition. This is an unconditional –scheduled- activation and deactivation.

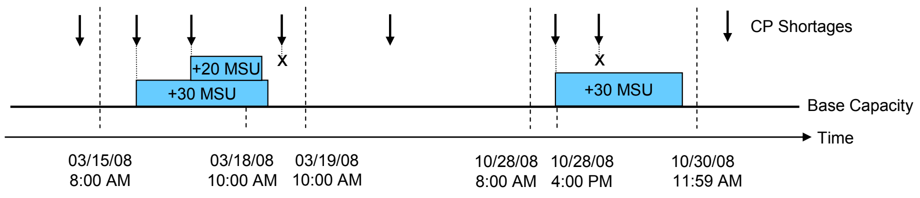
Note that the combination of scopes and maximum provisioning scope allows for a specification equivalent to e.g. „up to 3 processors for worklaod A, plus up to 2 processors for workload 2, but never more than 4 processors in total“.



Systems and Technology Group

Time Conditions

Name	Start Time	Deadline	End Time
TC1	03/15/08 08:00 AM	03/18/08 10:00 AM	03/19/08 10:00 AM
TC2	10/28/08 08:00 AM	10/28/08 04:00 PM	10/30/08 11:59 AM



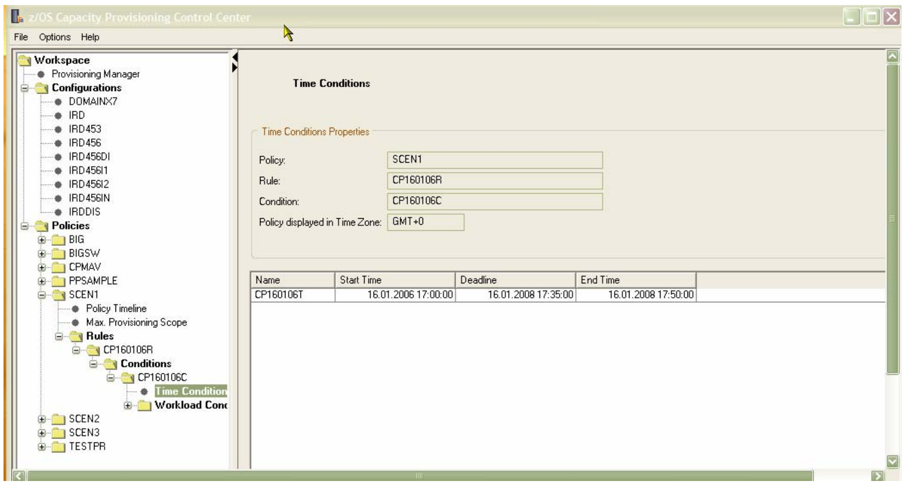
- **Time condition defines *when* temporary capacity may be activated:**
  - Start Time: provisioning of additional capacity allowed
  - Deadline: provisioning of additional capacity no longer allowed
  - End Time: deactivation of additional capacity should begin

15 © 2008 IBM Corporation

Time Conditions represent time periods during which additional capacity can be activated, up to the limit defined by the Provisioning Scope of the Rule.

The figure shows the definition of two Time Conditions and how CPM interprets them. On the left side of the figure, the effect of the first Time Condition is shown. Resource shortages are only considered between the specified start time and deadline. This means, additional resource shortages between the deadline and the end time are ignored. The boxes represent the capacity of additionally provisioned CP MSU capacity. On the right side of the figure, the effect of the second Time Condition is shown. Here, the period between start time and deadline is very small compared to the period between deadline and end time. This means, additionally provisioned capacity can remain active for a longer period, but it is not allowed to increase the provisioned capacity in that phase.

Time Condition in the CPCC



Time Conditions

Time Conditions Properties

Policy: SCEN1
Rule: CP160106R
Condition: CP160106C
Policy displayed in Time Zone: GMT+0

Name	Start Time	Deadline	End Time
CP160106T	16.01.2006 17:00:00	16.01.2008 17:35:00	16.01.2008 17:50:00

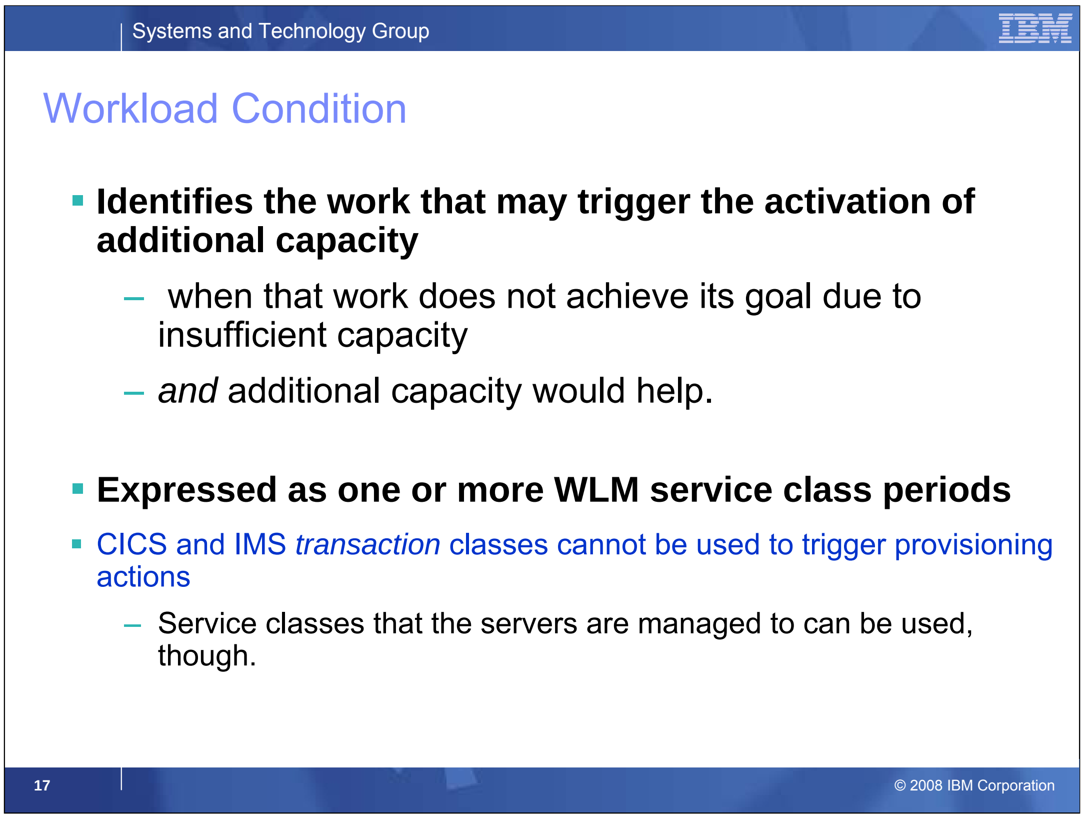
- Date and time values are entered and displayed using the localized settings (“regional options”) in effect for the workstation
- User can define which time zone to use
- Multiple time conditions may be defined for each rule

On the „Time Conditions“ panel the time conditions can be viewed or edited.

•The format of the date/time fields depend on the Windows „regional options“ that are in effect. The sample above show how dates and times would be displayed in central Europe. With, for example „English (United States)“ in effect the format could be mm/dd/yyyy hh:MM:SS AM/PM

•The Capacity Provisioning policy itself does always specify times in UTC (GMT). Also all server messages and reports refer to times using UTC.
However, to simplify editing of a policy, it is possible to choose the time zone in which a policy is viewed/editid in the CPCC. This time zone can be selected via the
Options→Preferences menu. The selected time zone is also displayed on the Time Conditions panel.

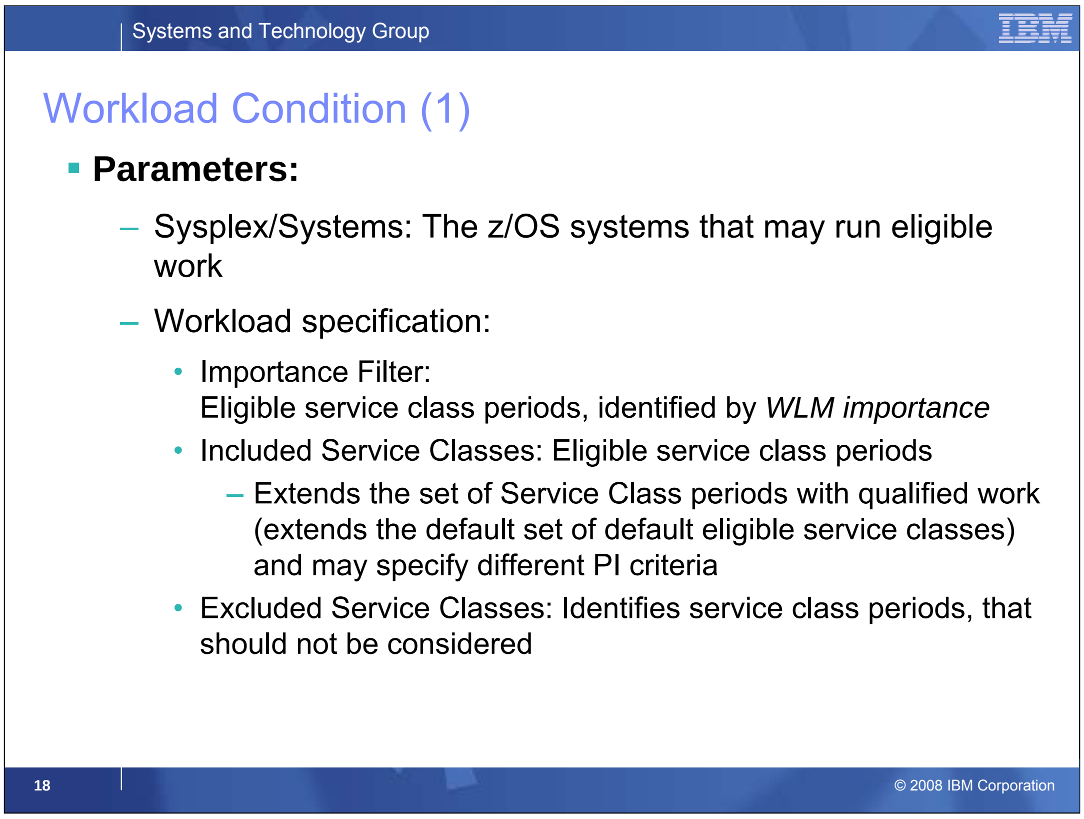
•If an invalid date/time is entered a pop-up lists the valid formats for the current regional options.



Workload Condition

- **Identifies the work that may trigger the activation of additional capacity**
  - when that work does not achieve its goal due to insufficient capacity
  - *and* additional capacity would help.
- **Expressed as one or more WLM service class periods**
- CICS and IMS *transaction* classes cannot be used to trigger provisioning actions
  - Service classes that the servers are managed to can be used, though.

A workload condition defines work that is eligible to cause activation of additional capacity, and the conditions under which that work can trigger activation. The specification of eligible work follows the workload model of the z/OS Workload Manager (WLM).



Workload Condition (1)

- **Parameters:**
  - Sysplex/Systems: The z/OS systems that may run eligible work
  - Workload specification:
    - Importance Filter:
      Eligible service class periods, identified by *WLM importance*
    - Included Service Classes: Eligible service class periods
      - Extends the set of Service Class periods with qualified work (extends the default set of default eligible service classes) and may specify different PI criteria
    - Excluded Service Classes: Identifies service class periods, that should not be considered

The first part of a workload condition specifies the Sysplexes and systems to be monitored for the specific condition.

To apply the condition to all sysplexes in the domain configuration insert an asterisk (*) here. Otherwise only workload on the named Sysplex will be considered to trigger provisioning for this workload condition.

Similarly, you can enter the name of a z/OS system in the **System** field. To apply the condition to all systems in the named Sysplex or the domain configuration insert an asterisk (*). Otherwise only workload on the named system will be considered to trigger provisioning for this workload condition.

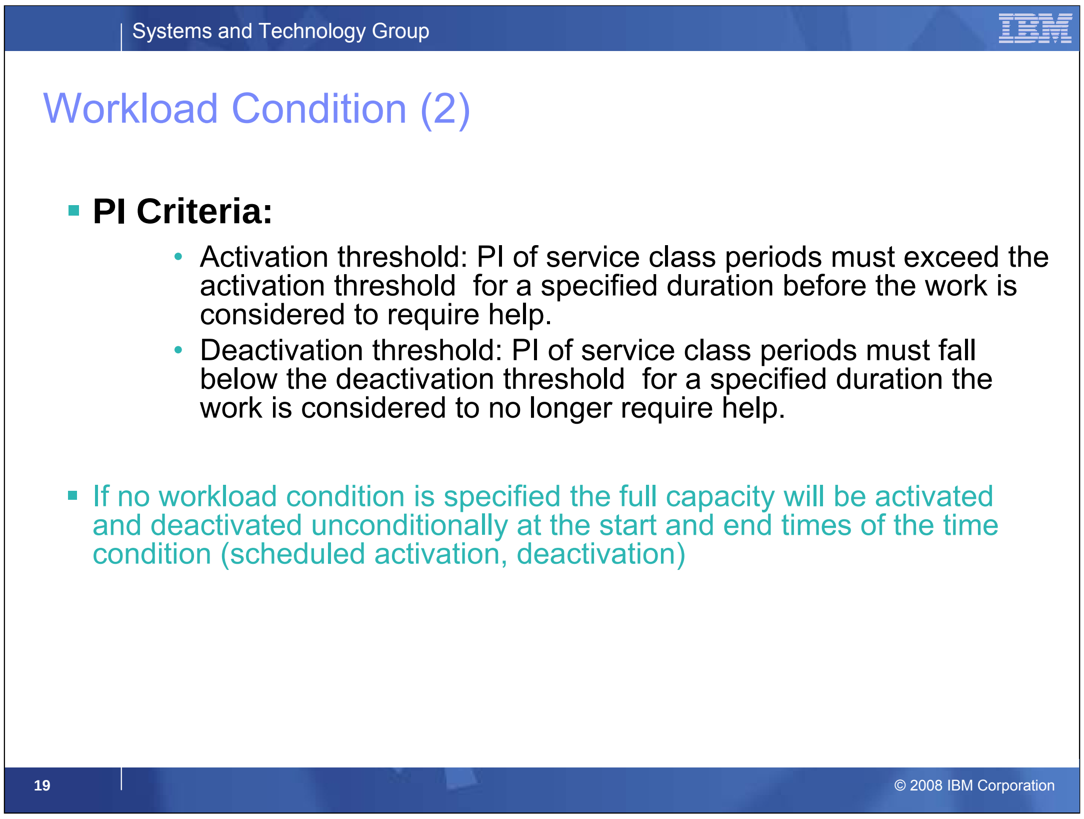
The workload to be observed can be defined by a combination of three tabs: Importance filters, includes service classes, and excluded service classes:

•**Importance Filters** allows you to specify service class periods to be monitored, based on their importance, and the performance index values and durations to trigger intervention.

•**Included Service Classes** allows you to specify service classes to be monitored in addition to any identified by importance filters, and the triggers for these.

•**Excluded Service Classes filter** allows you to specify service classes to be excluded from importance filters, or subsets

If specifications exist on multiple levels then the service class periods as derived from the importance filter are merged with the explicitly defined (included) service class period. Finally the excluded service class periods (if any) are removed from the previous set.



Workload Condition (2)

- **PI Criteria:**
  - Activation threshold: PI of service class periods must exceed the activation threshold for a specified duration before the work is considered to require help.
  - Deactivation threshold: PI of service class periods must fall below the deactivation threshold for a specified duration the work is considered to no longer require help.
- If no workload condition is specified the full capacity will be activated and deactivated unconditionally at the start and end times of the time condition (scheduled activation, deactivation)

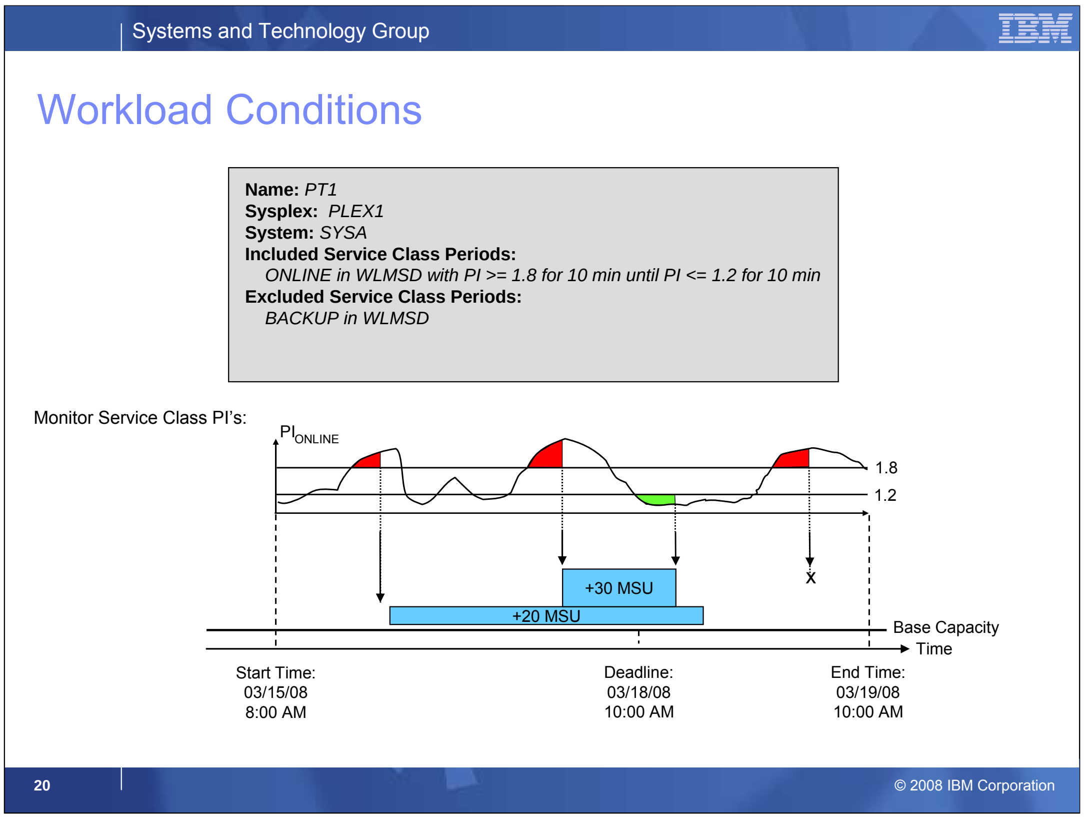
Provisioning PI: If the performance index of a service class period is equal or higher than the specified value the Provisioning Manager considers the service class period to be suffering.

Duration in min: This is the duration in minutes a service class period has to exceed the provisioning PI before the Provisioning Manager considers the service class period to be suffering.

Deprovisioning PI: If the performance index of a service class period is lower than the specified value it is not considered to be suffering. The deprovisioning PI must be at least 0.2 less than the provisioning PI limit, and must be not less than 1.1.

Duration in min : This is the duration in minutes the PI of the selected service classes must be lower than the specified deprovisioning PI for it to be considered no longer suffering. It must have a value between 5 and 1440 minutes.

PI-Scope: Indicates which PI the other criteria apply to. The possible values are **System (known as local PI in WLM)** and **Sysplex**. The default and recommended value for most situations is **System**.


Systems and Technology Group

Workload Conditions

Name: *PT1*
Sysplex: *PLEX1*
System: *SYSA*
Included Service Class Periods:
ONLINE in WLMSD with PI >= 1.8 for 10 min until PI <= 1.2 for 10 min
Excluded Service Class Periods:
BACKUP in WLMSD

Monitor Service Class PI's:

PI ONLINE
1.8
1.2
+30 MSU
+20 MSU
X
Base Capacity
Time
Start Time: 03/15/08 8:00 AM
Deadline: 03/18/08 10:00 AM
End Time: 03/19/08 10:00 AM

20 © 2008 IBM Corporation

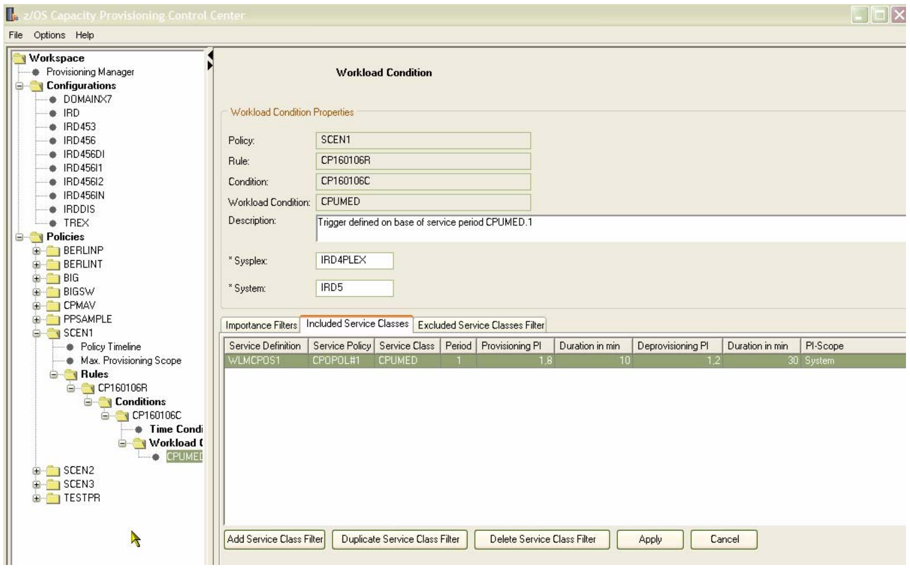
Workload Conditions identify the work that is eligible for provisioning and the conditions under which that work can trigger the activation of additional capacity. The specification of eligible work incorporates the workload model of the z/OS Workload Manager (WLM), i.e. it refers to service classes defined in the WLM service definition.

This chart assumes a Workload Condition that includes service class ONLINE defined into WLM Service Definition WLMSD. The provisioning PI equals 1.8, the Provisioning PI Duration is 10 min. The Deprovisioning PI equals 1.2, and Deprovisioning PI Duration is 10 min. If the PI of the service class period evolves within a defined Time Condition as shown, CPM would detect three instances of the provisioning PI criterion being fulfilled.

At the first two instants, CPM would activate additional capacity. The last instance does **not** lead to an additional activation because it is after the deadline. CPM would also detect an instance of the deprovisioning PI criterion being fulfilled. CPM decides here that service class ONLINE does no longer need help and deactivates the additional capacity.

Additional capacity would only be provisioned if demand for additional Capacity Provisioning/zAAp/zIIP is recognized. This analysis is based on (many) metrics on the CPC, system, and service class levels.

Workload Condition in the CPCC



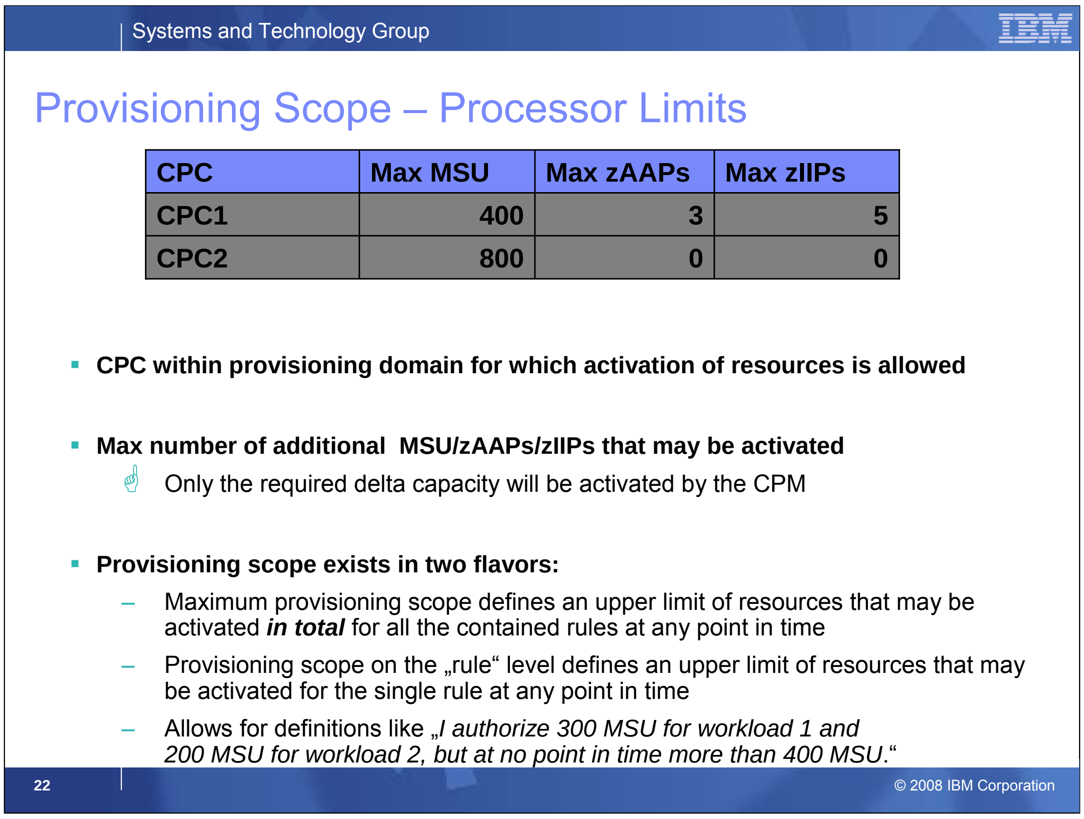
Workload Condition

Workload Condition Properties

Policy:	SCEN1
Rule:	CP160106R
Condition:	CP160106C
Workload Condition:	CPUMED
Description:	Trigger defined on base of service period CPUMED.1
* Sysplex:	IRD4PLEX
* System:	IRD5

Service Definition	Service Policy	Service Class	Period	Provisioning PI	Duration in min	Deprovisioning PI	Duration in min	PI-Scope
WLMCPOS1	CPOPOL#1	CPUMED	1	1,8	10	1,2	30	System

This sample picture shows the definition of a workload condition in the CPCC.



Provisioning Scope – Processor Limits

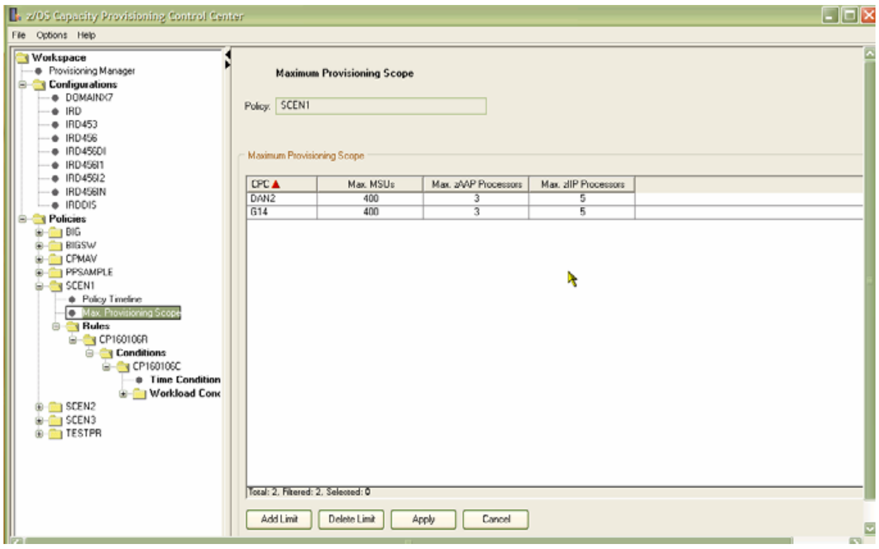
CPC	Max MSU	Max zAAPs	Max zIIPs
CPC1	400	3	5
CPC2	800	0	0

- **CPC within provisioning domain for which activation of resources is allowed**
- **Max number of additional MSU/zAAPs/zIIPs that may be activated**
  - Only the required delta capacity will be activated by the CPM
- **Provisioning scope exists in two flavors:**
  - Maximum provisioning scope defines an upper limit of resources that may be activated ***in total*** for all the contained rules at any point in time
  - Provisioning scope on the „rule“ level defines an upper limit of resources that may be activated for the single rule at any point in time
  - Allows for definitions like „*I authorize 300 MSU for workload 1 and 200 MSU for workload 2, but at no point in time more than 400 MSU*.“

A Provisioning Scope has CPC scope. It defines which capacity can additionally be activated on which CPC in the Provisioning Domain.

The table shows an example of a Provisioning Scope. Here, limits for two CPCs are defined. The first definition specifies that on CPC1 a maximum of 400 CP MSU, five additional zIIPs, and three additional zAAPs may be activated. The second definition specifies that on CPC2 800 MSU at a maximum, and no zIIPs or zAAPs may be activated.

Provisioning Scope Sample in the CPCC



Maximum Provisioning Scope

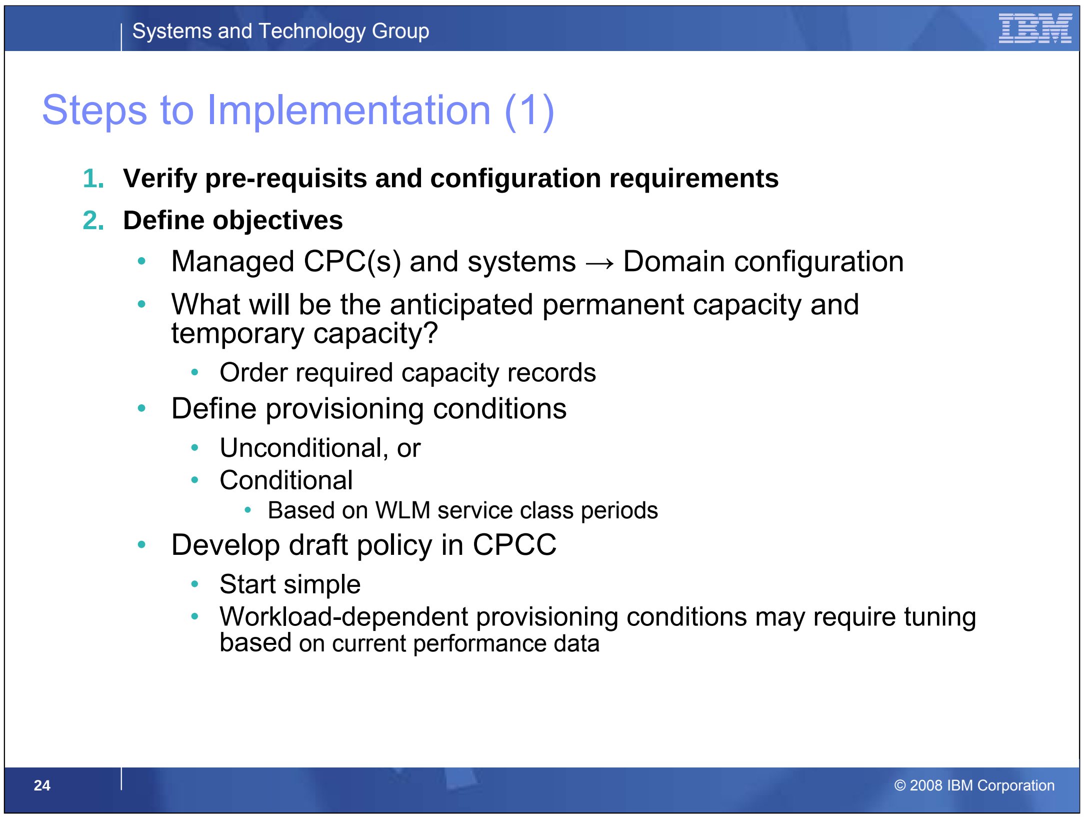
Policy: SCEN1

Maximum Provisioning Scope

CPC ▲	Max. MSUs	Max. zAAP Processors	Max. zIIP Processors
DAN2	400	3	5
G14	400	3	5

Total: 2, Filtered: 2, Selected: 0

This picture shows the Provisioning Scope panel in the CPCC.



Steps to Implementation (1)

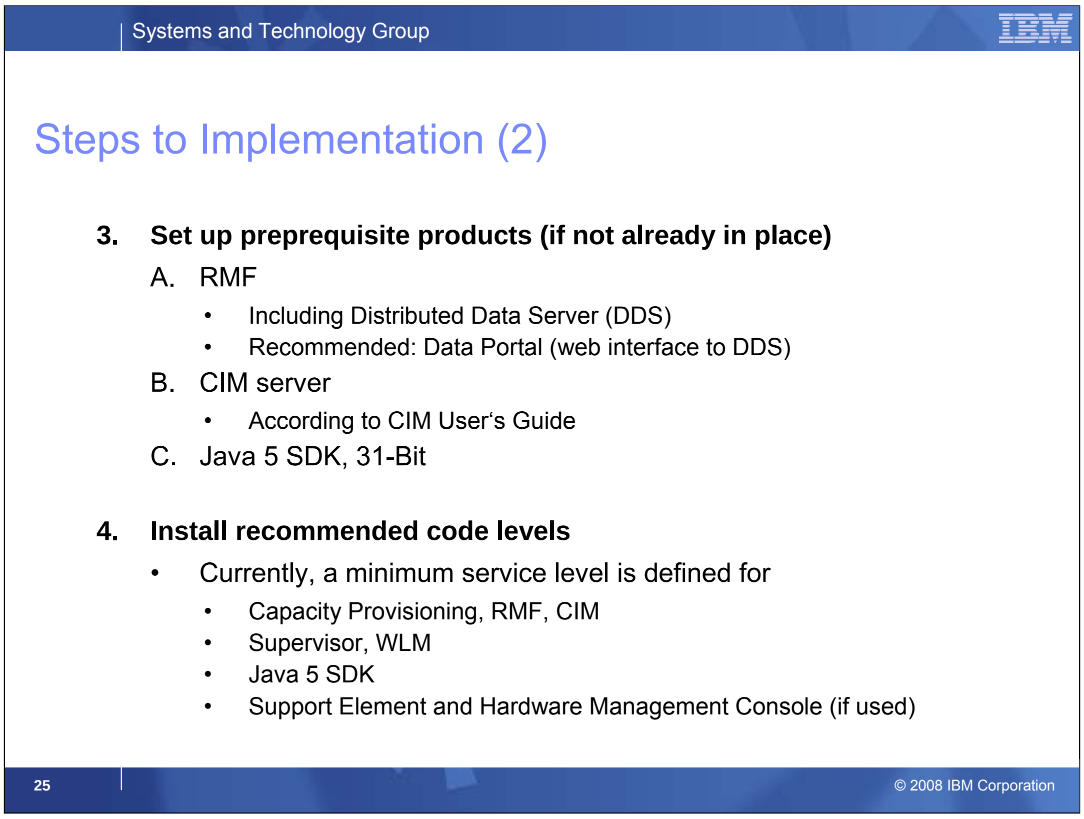
1. **Verify pre-requisits and configuration requirements**
2. **Define objectives**
   - Managed CPC(s) and systems → Domain configuration
   - What will be the anticipated permanent capacity and temporary capacity?
     - Order required capacity records
   - Define provisioning conditions
     - Unconditional, or
     - Conditional
       - Based on WLM service class periods
   - Develop draft policy in CPCC
     - Start simple
     - Workload-dependent provisioning conditions may require tuning based on current performance data

If not already done, review the list of configuration requirements and supported environments for CP. Especially requirements that currently exist on the number of logical processores (≥ highest number of physical processors) for each processor pool, and IP connectivity to Support Element or Hardware Management Console, must be met. Dedicated LPARs can only be managed with subcapacity processors.

Subsequently the topology (one or multiple CPCs, monitored systems, runtime systems) should be defined. This is the base for the next planning and customization steps, and will ultimately result in the CP domain configuration.

At this stage you should also develop what levels of permanent and temporary (OOCoD) capacity are likely to be used. If multiple processor types (CP, zAAP, zIIP) are to be managed consider each pool separately.

At this stage you should also examine RMF data (preferrably in conjunction with real time observation using RMF data portal) to determine which service class periods are good candidates to base provisioning action on. (As previously mentioned, you cannot use CICS and IMS transaction service classes).


Systems and Technology Group

Steps to Implementation (2)

3. **Set up preprequisite products (if not already in place)**
   A. RMF
      - Including Distributed Data Server (DDS)
      - Recommended: Data Portal (web interface to DDS)

   B. CIM server
      - According to CIM User‘s Guide

   C. Java 5 SDK, 31-Bit

4. **Install recommended code levels**
   - Currently, a minimum service level is defined for
     - Capacity Provisioning, RMF, CIM
     - Supervisor, WLM
     - Java 5 SDK
     - Support Element and Hardware Management Console (if used)

25 © 2008 IBM Corporation

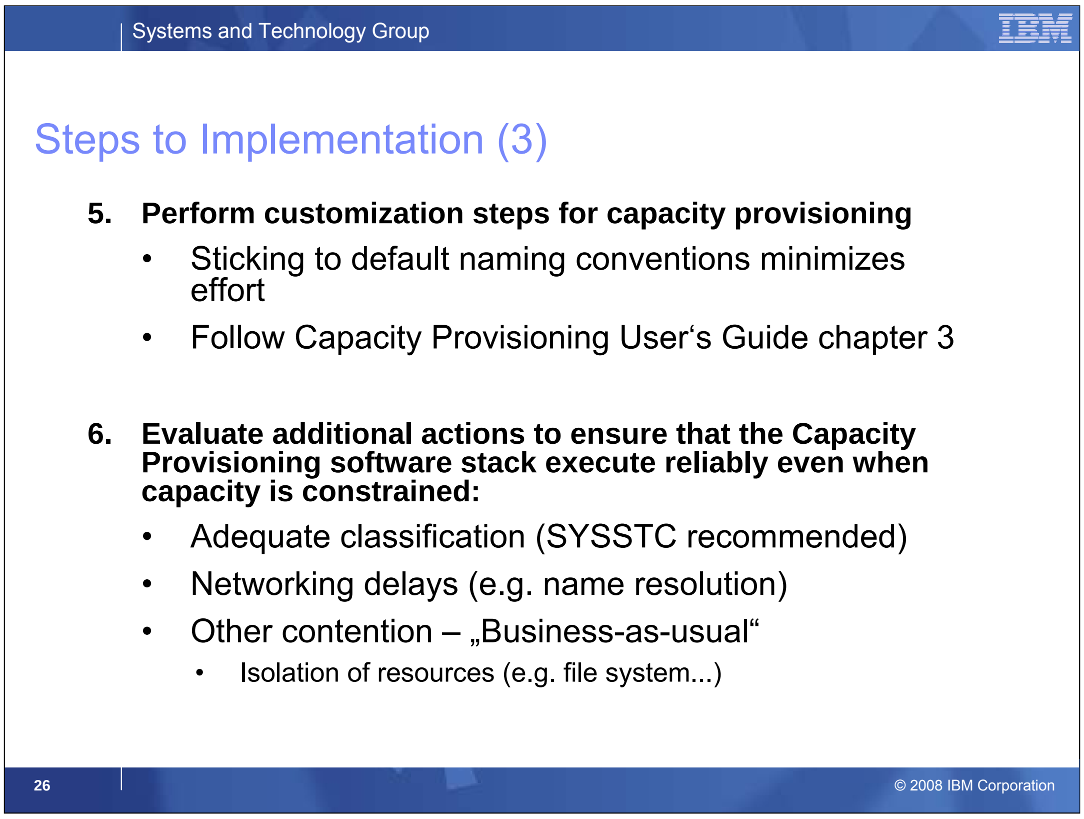
Before performing the required customization steps for Capacity Provisioning itself the pre-requisite z/OS elements need to be set up. Unless an installation would not plan to go beyond MANUAL mode, all required elements‘ customizations need to be performed. (For manual mode, the RMF and CIM elements are not required).

In most cases, RMF will already be set up and in use. Also the Distributed Data Server (DDS – GPMSERVE started task) needs to be set up.

While the RMF data portal is not required we recommend activating it. It‘s almost zero delta effort, and comes in very handy. The data portal presents the same real time data as CP uses for its management.

Then the CIM server needs to be set up. Because the CIM z/OS element was newly introduced with z/OS Release 7 it may not be set up already.

In addition, the Java 5 SDK (31-Bit), SR5a or later needs to be installed.



Steps to Implementation (3)

5. **Perform customization steps for capacity provisioning**
   - Sticking to default naming conventions minimizes effort
   - Follow Capacity Provisioning User‘s Guide chapter 3

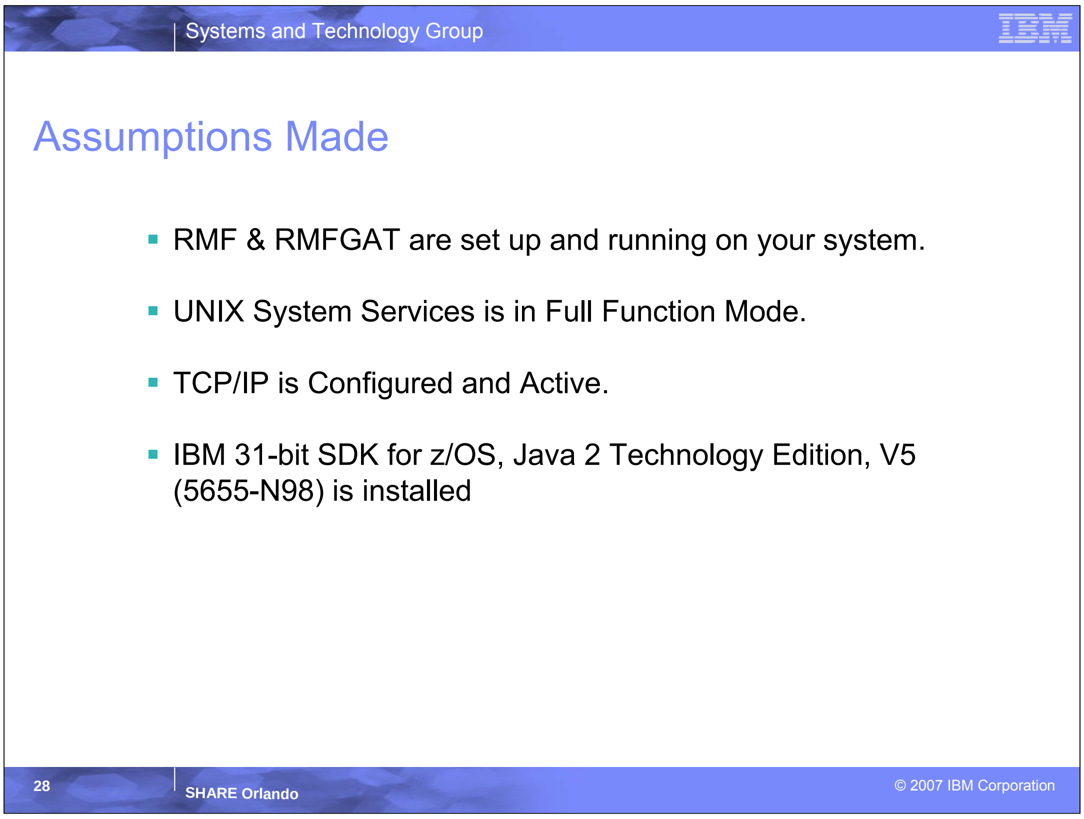
6. **Evaluate additional actions to ensure that the Capacity Provisioning software stack execute reliably even when capacity is constrained:**
   - Adequate classification (SYSSTC recommended)
   - Networking delays (e.g. name resolution)
   - Other contention – „Business-as-usual“
     - Isolation of resources (e.g. file system...)

Unless an installation does not plan to go beyond MANUAL mode operation the complete Capacity Provisioning customization needs to be performed. (For manual mode, the RMF and CIM related activities are not required).

At this time you should also pay attention to the question whether your system is set up such that all components required by the Capacity Provisioning solution can execute reliably even under high load, i.e. in a situation when you would want Capacity Provisioning to take an action.

Topics

- Assumptions Made
- Setting up the Distributed Data Server (DDS)
- Setting up the Common Information Model (CIM) Server
- Setting up the Capacity provisioning Manager Server (CPOSERV)
- Setting up the Hardware Connections


Systems and Technology Group

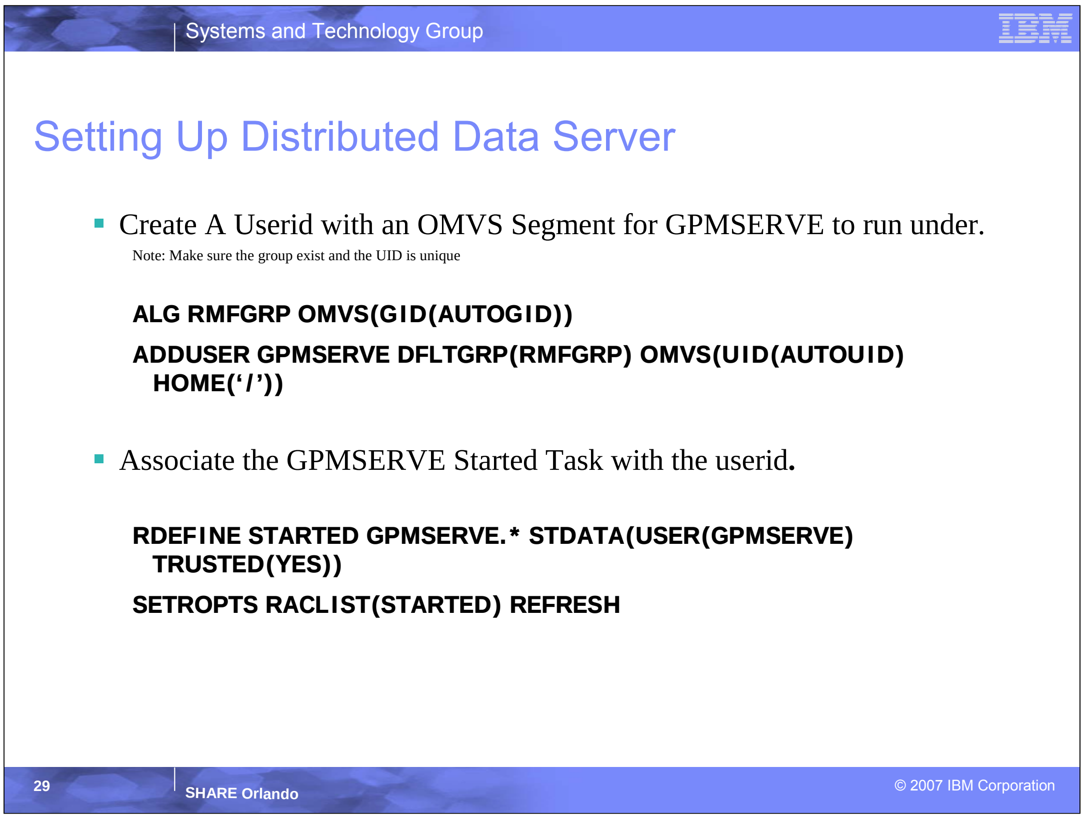
Assumptions Made

- RMF & RMFGAT are set up and running on your system.
- UNIX System Services is in Full Function Mode.
- TCP/IP is Configured and Active.
- IBM 31-bit SDK for z/OS, Java 2 Technology Edition, V5 (5655-N98) is installed

28 SHARE Orlando © 2007 IBM Corporation

In order to get this presentation done in the time allotted I made the following Assumption:

1. RMF and RMFGAT are set up and running on your system or some functional equivalent.
2. UNIX System Services Full function mode implies you have one or more BPXPRMxx members and have coded them in the IEASYSxx member so that Kernel services start up at IPL time. TCPIP does not work unless you are in Full Function Mode.
3. TCPIP needs to be configured and active. I suspect this is not a problem
4. The Capacity Provisioning manager (CPM) exploits 31 bit JAVA V5 SDK so it will need to be installed.



Setting Up Distributed Data Server

- Create A Userid with an OMVS Segment for GPMSERVE to run under.

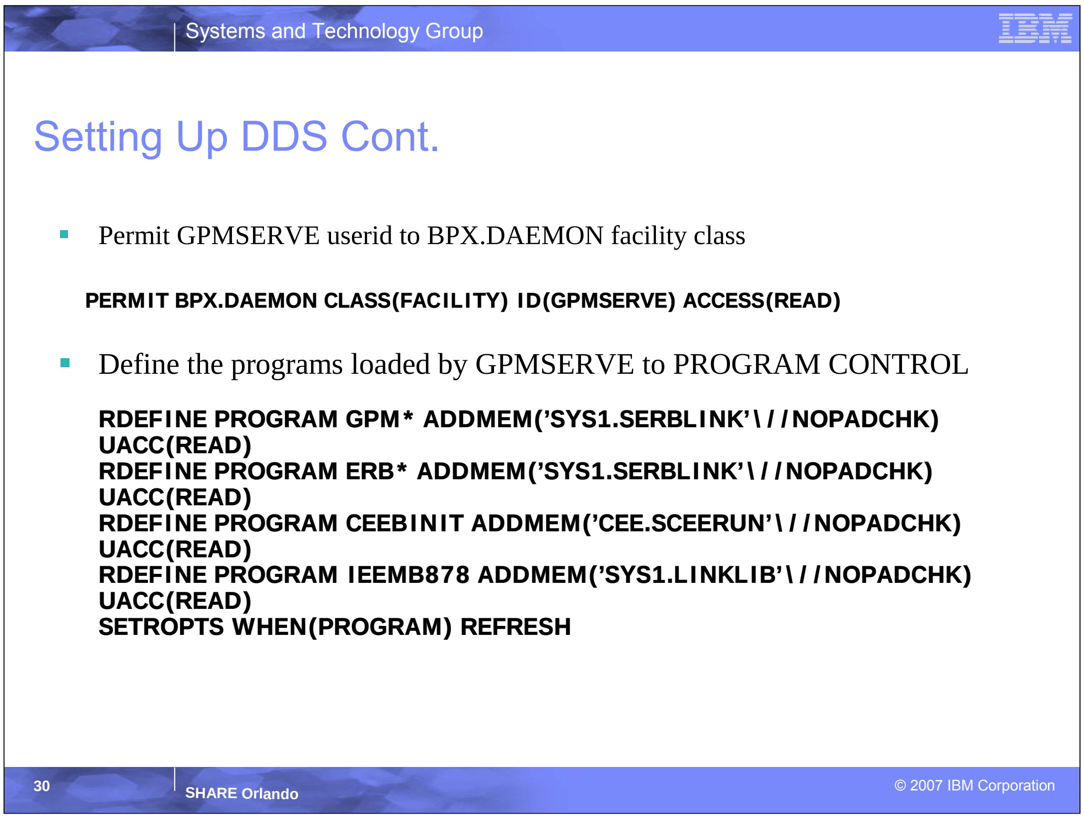
  Note: Make sure the group exist and the UID is unique

```
ALG RMFGRP OMVS(GID(AUTOGID))
ADDUSER GPMSERVE DFLTGRP(RMFGRP) OMVS(UID(AUTOUID)
  HOME('/'))
```

- Associate the GPMSERVE Started Task with the userid.

```
RDEFINE STARTED GPMSERVE.* STDATA(USER(GPMSERVE)
  TRUSTED(YES))
SETROPTS RACLIST(STARTED) REFRESH
```

1. The DDS, otherwise known as GPMSERVE, is run as a started task and requires a certain amount of security setup.
2. We must first define a userid with an OMVS segment for the started task to run under. For Simplicity that user should be named the same as the started task.
   1. To have an OMVS segment the id must have an OMVS GroupID (GID), an OMVS UserID (UID), and a home directory.
   2. We can use the AUTOGID and AUTOUID operand to generate a unique (Not previously used) GID and UID.
3. The above commands alter an existing RACF group and give it an OMVS GID, then creates the user GPMSERVE that is a member of that group and has a unique UID.
4. Next thing we do is associate the started task with the new user so that it will always run under that ID. See the above command.



Setting Up DDS Cont.

- Permit GPMSERVE userid to BPX.DAEMON facility class

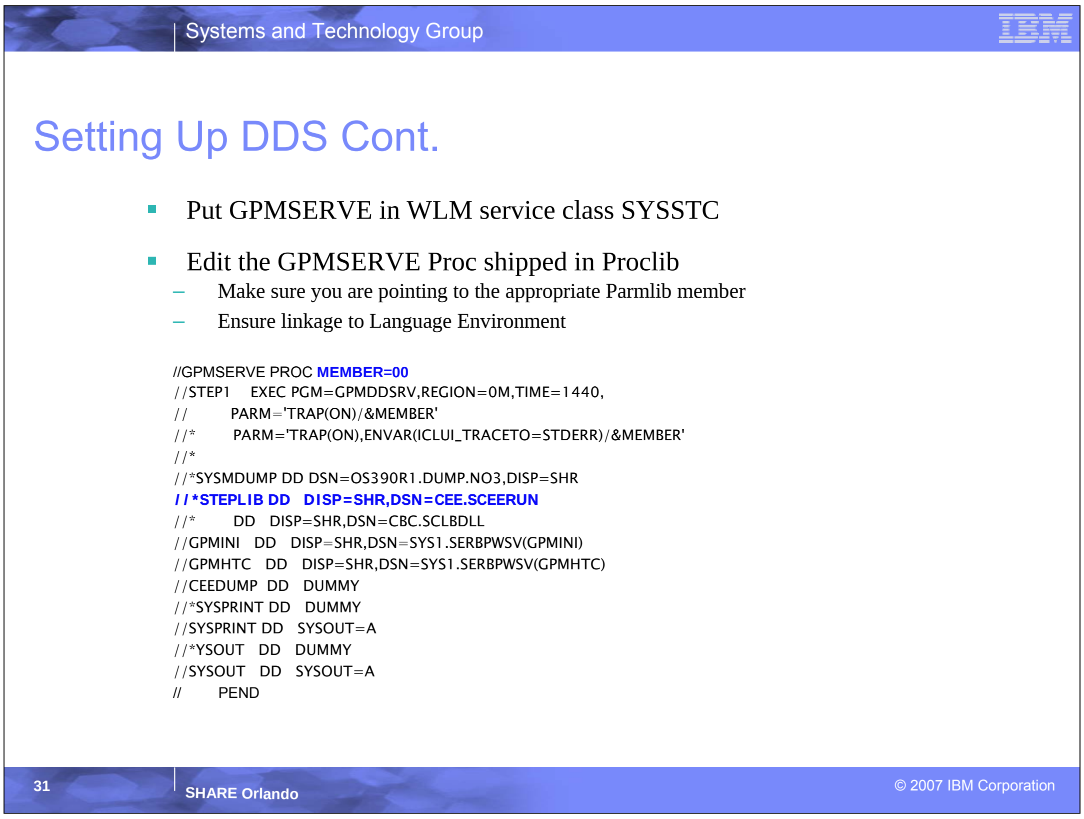
```
PERMIT BPX.DAEMON CLASS(FACILITY) ID(GPMSERVE) ACCESS(READ)
```

- Define the programs loaded by GPMSERVE to PROGRAM CONTROL

```
RDEFINE PROGRAM GPM* ADDMEM('SYS1.SERBLINK'\//NOPADCHK)
UACC(READ)
RDEFINE PROGRAM ERB* ADDMEM('SYS1.SERBLINK'\//NOPADCHK)
UACC(READ)
RDEFINE PROGRAM CEEBINIT ADDMEM('CEE.SCEERUN'\//NOPADCHK)
UACC(READ)
RDEFINE PROGRAM IEEMB878 ADDMEM('SYS1.LINKLIB'\//NOPADCHK)
UACC(READ)
SETROPTS WHEN(PROGRAM) REFRESH
```

1. Permitting GPMSEREVE to BPX.DAEMON provides a more granular level of security.
2. If the BPX.DAEMON resource in the FACILITY class is defined, your system has z/OS UNIX security. Your system can exercise more control over your superusers. BPX.DAEMON serves two functions in the z/OS UNIX environment:
   1. Only superusers permitted to this profile have the daemon authority to change MVS identities via z/OS UNIX services without knowing the target user ID's password. Without it all superusers can.
   2. z/OS UNIX will verify that the address space has not loaded any executables that are uncontrolled before it allows any of the following services that are controlled by z/OS UNIX to succeed: - seteuid - setuid - setreuid - pthread_security_np() - auth_check_resource_np() - _login() - _spawn() with user ID change - _password() Daemon authority is required only when a program does a setuid(), seteuid(),
3. If you have not defined BPX.DAEMON or if you want more information about it see:

   z/OS UNIX System Services Planning (GA22-7800)
4. All programs loaded in to an address space that require daemon authority must be defined to RACF program control. The above is a list of those programs and the commands to do so.

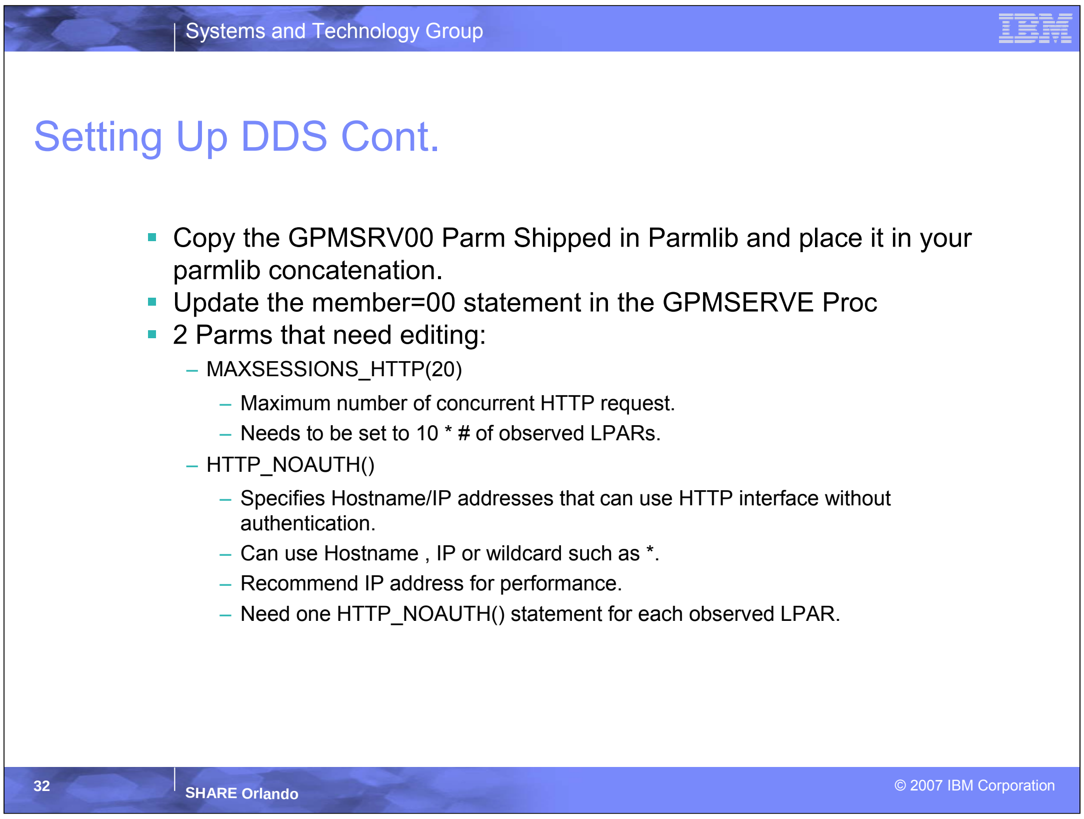


Setting Up DDS Cont.

- Put GPMSERVE in WLM service class SYSSTC
- Edit the GPMSERVE Proc shipped in Proclib
  - Make sure you are pointing to the appropriate Parmlib member
  - Ensure linkage to Language Environment

```
//GPMSERVE PROC MEMBER=00
//STEP1    EXEC PGM=GPMDDSRV,REGION=0M,TIME=1440,
//         PARM='TRAP(ON)/&MEMBER'
//*        PARM='TRAP(ON),ENVAR(ICLUI_TRACETO=STDERR)/&MEMBER'
//*
//*SYSMDUMP DD DSN=OS390R1.DUMP.NO3,DISP=SHR
//*STEPLIB DD   DISP=SHR,DSN=CEE.SCEERUN
//*        DD   DISP=SHR,DSN=CBC.SCLBDLL
//GPMINI   DD   DISP=SHR,DSN=SYS1.SERBPWSV(GPMINI)
//GPMHTC   DD   DISP=SHR,DSN=SYS1.SERBPWSV(GPMHTC)
//CEEDUMP  DD   DUMMY
//*SYSPRINT DD  DUMMY
//SYSPRINT DD   SYSOUT=A
//*YSOUT   DD   DUMMY
//SYSOUT   DD   SYSOUT=A
//       PEND
```

1. Since CPOSERV relies on metric data retrieved from GPMSERVE to make provisioning decisions, and we would need this data at a point when we suspect important work is suffering, we need to ensure GPMSERVE gets the best performance possible . We put it in SYSSTC to get it above the service class of the work we are monitoring.
2. GPMSERVE ships in proclib. There are only two things you may need to alter in the proc:
   1. You may need to change the **member=xx** to point to a different parm member if you choose to make and alter a copy of the original (More on this in the next slide).
   2. You may need to uncomment the steplib to LE. I assume SYS1.SCEERUN might already be in linklist, but if it is not and you do not want to put it there, you can uncomment the steplib in the GPMSERVE proc.

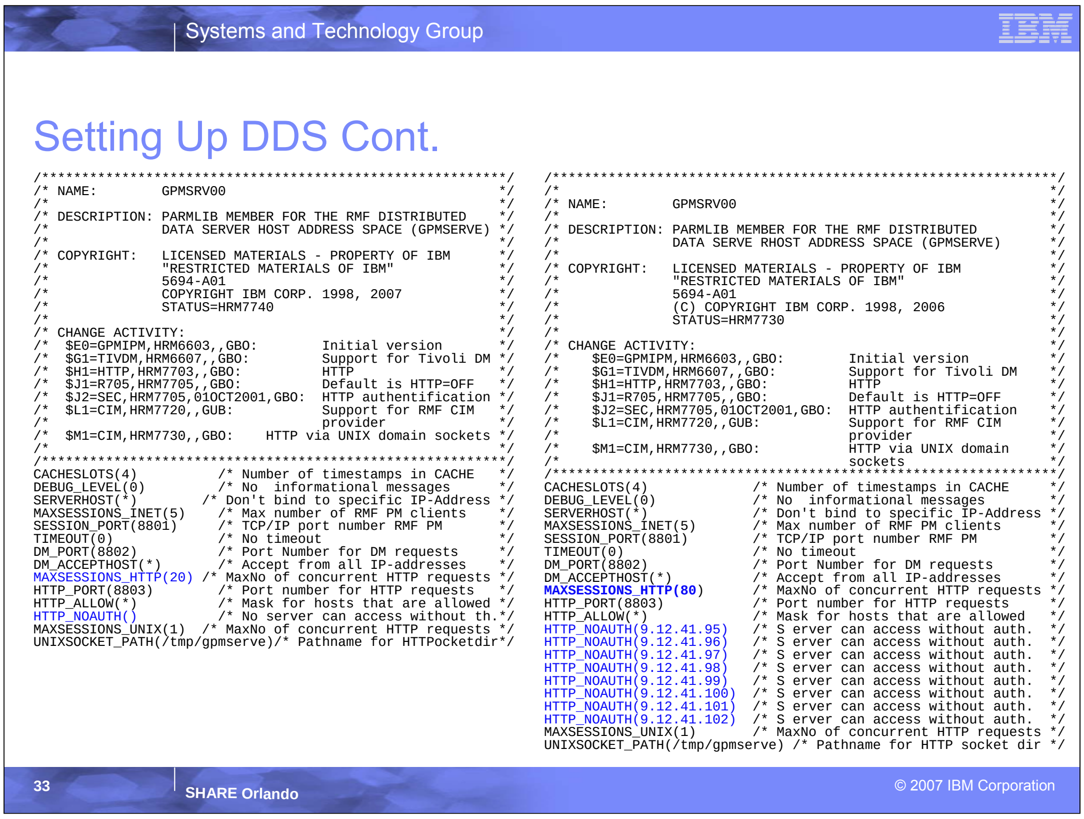


Setting Up DDS Cont.

- Copy the GPMSRV00 Parm Shipped in Parmlib and place it in your parmlib concatenation.
- Update the member=00 statement in the GPMSERVE Proc
- 2 Parms that need editing:
  - MAXSESSIONS_HTTP(20)
    - Maximum number of concurrent HTTP request.
    - Needs to be set to 10 * # of observed LPARs.
  - HTTP_NOAUTH()
    - Specifies Hostname/IP addresses that can use HTTP interface without authentication.
    - Can use Hostname , IP or wildcard such as *.
    - Recommend IP address for performance.
    - Need one HTTP_NOAUTH() statement for each observed LPAR.

- The parm member GPMSRV00 ships in parmlib and will work out of the box if you were just installing the DDS to view performance metrics via a web interface. Since we are using it as the source for performance metrics for Capacity provisioning we will need to make a couple changes.
  1. To preserve the original GPMSRV00 I made a copy, that I called GPMSRVS1, and placed it in my parmlib concatenation.
  2. I edited my SYS1.PROCLIB(GPMSERVE) proc changing the member=00 to member=s1 so that I am pointing at the copy when I start GPMSERVE. (Note change in example on last slide)

- Now I need to edit 2 parms in this member.
  1. MAXSESSIONS_HTTP(20)
     - This is the number of concurrent HTTP request that can be made to GPMSERVE. The CIM Server on each LPAR you plan to observe, for Capacity Provisioning purposes, will be averaging 6 concurrent request. By rounding this up to an even 10 you allow for the possibility that at some point something other then Capacity Provisioning may need to make a http request of GPMSERVE. So for example in my environment I am observing 8 LPARS so I set MAXSESSIONS_HTTP(80).
  2. HTTP_NOAUTH()
     - This specifies the host names/IP addresses that can use the HTTP interface without authentication. In our case the CIM server on each observed LPAR will be using this interface to collect metric data to be passed back to the CPM.
     - As mentioned this parm will take the Host name, IP address or a wildcard such as a *. It is easiest to just code a * , but as usual it is recommended that you grant the least amount of access possible. For this reason, and to decrease overhead, I recommend that you code IP addresses on these parms. What this means is that you will need one HTTP_NOAUTH() statement for each observed LPAR in your domain. See next slide for an example of how this is coded.

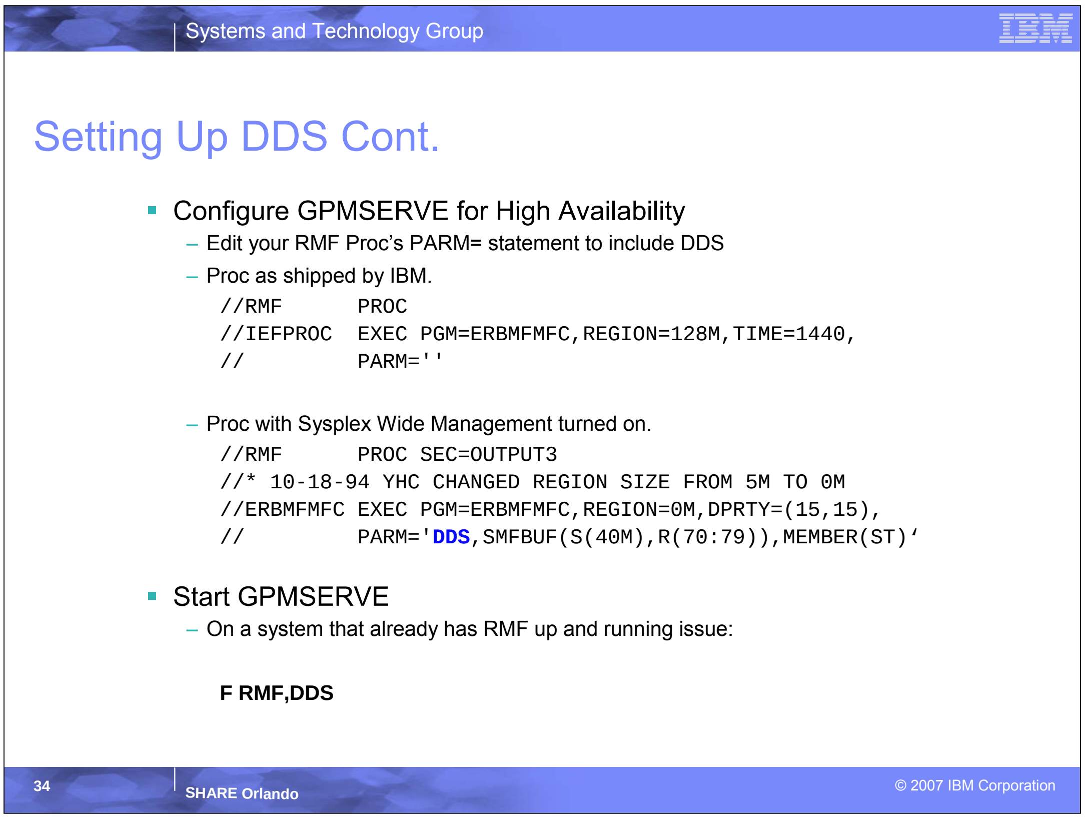


Setting Up DDS Cont.

```
/***********************************************************/
/* NAME:        GPMSRV00                                   */
/*                                                         */
/* DESCRIPTION: PARMLIB MEMBER FOR THE RMF DISTRIBUTED     */
/*              DATA SERVER HOST ADDRESS SPACE (GPMSERVE) */
/*                                                         */
/* COPYRIGHT:   LICENSED MATERIALS - PROPERTY OF IBM       */
/*              "RESTRICTED MATERIALS OF IBM"              */
/*              5694-A01                                   */
/*              COPYRIGHT IBM CORP. 1998, 2007             */
/*              STATUS=HRM7740                             */
/*                                                         */
/* CHANGE ACTIVITY:                                        */
/*  $E0=GPMIPM,HRM6603,,GBO:        Initial version        */
/*  $G1=TIVDM,HRM6607,,GBO:         Support for Tivoli DM */
/*  $H1=HTTP,HRM7703,,GBO:          HTTP                   */
/*  $J1=R705,HRM7705,,GBO:          Default is HTTP=OFF    */
/*  $J2=SEC,HRM7705,01OCT2001,GBO:  HTTP authentification */
/*  $L1=CIM,HRM7720,,GUB:           Support for RMF CIM    */
/*                                  provider               */
/*  $M1=CIM,HRM7730,,GBO:    HTTP via UNIX domain sockets */
/*                                                         */
/***********************************************************/
CACHESLOTS(4)          /* Number of timestamps in CACHE   */
DEBUG_LEVEL(0)         /* No  informational messages      */
SERVERHOST(*)        /* Don't bind to specific IP-Address */
MAXSESSIONS_INET(5)    /* Max number of RMF PM clients    */
SESSION_PORT(8801)     /* TCP/IP port number RMF PM       */
TIMEOUT(0)             /* No timeout                      */
DM_PORT(8802)          /* Port Number for DM requests     */
DM_ACCEPTHOST(*)       /* Accept from all IP-addresses    */
MAXSESSIONS_HTTP(20) /* MaxNo of concurrent HTTP requests */
HTTP_PORT(8803)        /* Port number for HTTP requests   */
HTTP_ALLOW(*)          /* Mask for hosts that are allowed */
HTTP_NOAUTH()          /* No server can access without th.*/
MAXSESSIONS_UNIX(1)  /* MaxNo of concurrent HTTP requests */
UNIXSOCKET_PATH(/tmp/gpmserve)/* Pathname for HTTPocketdir*/
```

```
/**************************************************************/
/*                                                            */
/* NAME:        GPMSRV00                                      */
/*                                                            */
/* DESCRIPTION: PARMLIB MEMBER FOR THE RMF DISTRIBUTED        */
/*              DATA SERVE RHOST ADDRESS SPACE (GPMSERVE)     */
/*                                                            */
/* COPYRIGHT:   LICENSED MATERIALS - PROPERTY OF IBM          */
/*              "RESTRICTED MATERIALS OF IBM"                 */
/*              5694-A01                                      */
/*              (C) COPYRIGHT IBM CORP. 1998, 2006            */
/*              STATUS=HRM7730                                */
/*                                                            */
/* CHANGE ACTIVITY:                                           */
/*    $E0=GPMIPM,HRM6603,,GBO:        Initial version         */
/*    $G1=TIVDM,HRM6607,,GBO:         Support for Tivoli DM   */
/*    $H1=HTTP,HRM7703,,GBO:          HTTP                    */
/*    $J1=R705,HRM7705,,GBO:          Default is HTTP=OFF     */
/*    $J2=SEC,HRM7705,01OCT2001,GBO:  HTTP authentification   */
/*    $L1=CIM,HRM7720,,GUB:           Support for RMF CIM     */
/*                                    provider                */
/*    $M1=CIM,HRM7730,,GBO:           HTTP via UNIX domain    */
/*                                    sockets                 */
/**************************************************************/
CACHESLOTS(4)             /* Number of timestamps in CACHE    */
DEBUG_LEVEL(0)            /* No  informational messages       */
SERVERHOST(*)             /* Don't bind to specific IP-Address */
MAXSESSIONS_INET(5)       /* Max number of RMF PM clients     */
SESSION_PORT(8801)        /* TCP/IP port number RMF PM        */
TIMEOUT(0)                /* No timeout                       */
DM_PORT(8802)             /* Port Number for DM requests      */
DM_ACCEPTHOST(*)          /* Accept from all IP-addresses     */
MAXSESSIONS_HTTP(80)      /* MaxNo of concurrent HTTP requests */
HTTP_PORT(8803)           /* Port number for HTTP requests    */
HTTP_ALLOW(*)             /* Mask for hosts that are allowed  */
HTTP_NOAUTH(9.12.41.95)   /* S erver can access without auth. */
HTTP_NOAUTH(9.12.41.96)   /* S erver can access without auth. */
HTTP_NOAUTH(9.12.41.97)   /* S erver can access without auth. */
HTTP_NOAUTH(9.12.41.98)   /* S erver can access without auth. */
HTTP_NOAUTH(9.12.41.99)   /* S erver can access without auth. */
HTTP_NOAUTH(9.12.41.100)  /* S erver can access without auth. */
HTTP_NOAUTH(9.12.41.101)  /* S erver can access without auth. */
HTTP_NOAUTH(9.12.41.102)  /* S erver can access without auth. */
MAXSESSIONS_UNIX(1)       /* MaxNo of concurrent HTTP requests */
UNIXSOCKET_PATH(/tmp/gpmserve) /* Pathname for HTTP socket dir */
```

•Print this slide full size



Setting Up DDS Cont.

- Configure GPMSERVE for High Availability
  - Edit your RMF Proc's PARM= statement to include DDS
  - Proc as shipped by IBM.

```
//RMF       PROC
//IEFPROC   EXEC PGM=ERBMFMFC,REGION=128M,TIME=1440,
//          PARM=''
```

  - Proc with Sysplex Wide Management turned on.

```
//RMF       PROC SEC=OUTPUT3
//* 10-18-94 YHC CHANGED REGION SIZE FROM 5M TO 0M
//ERBMFMFC EXEC PGM=ERBMFMFC,REGION=0M,DPRTY=(15,15),
//          PARM='DDS,SMFBUF(S(40M),R(70:79)),MEMBER(ST)'
```

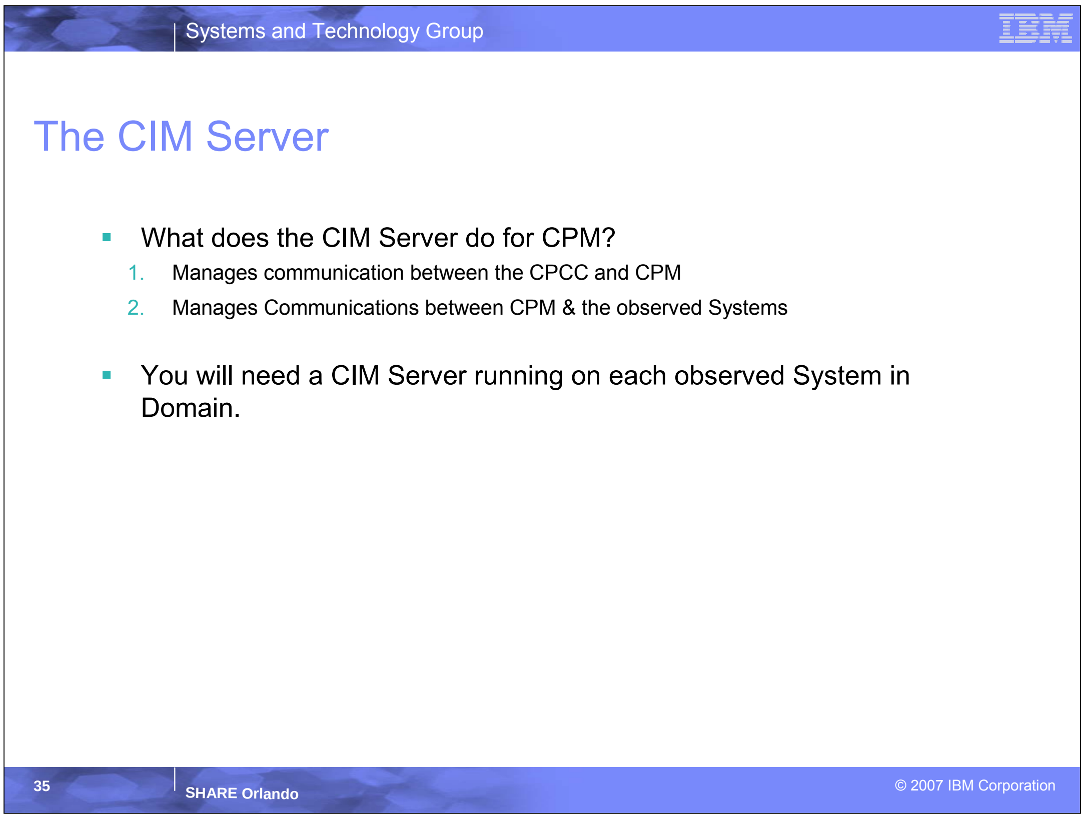
- Start GPMSERVE
  - On a system that already has RMF up and running issue:

F RMF,DDS

•In zOS Release 1.8 RMF introduced Sysplex Wide management of the DDS. When you use the sysplex wide DDS management, DDS is automatically started on the best suited system in the sysplex. This system is the one with the highest RMF release in the sysplex and where both, RMF and RMFGAT are active. If there are multiple systems with the highest RMF release, the system where RMF was started first is taken. What is important to you about all this is that Sysplex wide management monitors the DDS and if it goes down for any reason it gets restarted. No other automation product required.

•Caution – The only down side to all of this is that if the LPAR that the DDS is running on goes down then the DDS is moved to another suitable LPAR. This sounds good except that all the CIM servers will still be looking for it in the old location. See the backup data for a more in depth solution to this problem that makes the location of the DDS irrelevent.

•To start GPMSERVE on a system that already has RMF up and running just issue the **F RMF,DDS** console command. If you want to stop Sysplex wide management of the DDS you can issue the **F RMF,NODDS.** If you want to just start GPMSERVE manually just use the **S GPMSERVE** console command.



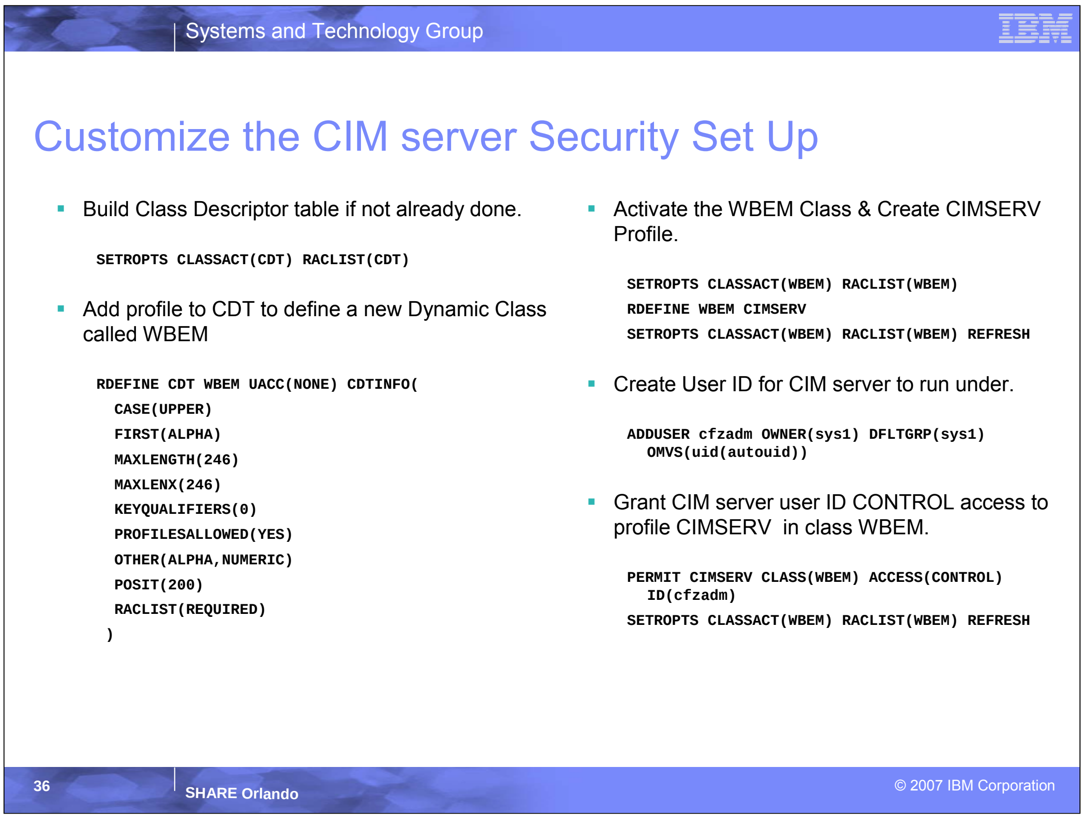
The CIM Server

- What does the CIM Server do for CPM?
  1. Manages communication between the CPCC and CPM
  2. Manages Communications between CPM & the observed Systems
- You will need a CIM Server running on each observed System in Domain.

- What is the CIMserver?

  The CIM server manages communication between clients and providers. The CIM server also provides several management functions, including security, and a set of commands that provide configuration and management functions to administrators. The CIM server implementation on z/OS is based on the **OpenPegasus CIM server 2.6.1** from **The Open Group**. See the ***OpenPegasus web site*** for more information (Copyright 2005, 2007. The Open Group. All Rights Reserved).

- In a Capacity provisioning Domain the CIM server is used for 2 tasks.
  1. The first is to communicate, using the http protocol, between the CPCC and the CPM. This is used tp provide CPM status and to upload new policies to the host.
  2. The second is to communicate, using the http protocol, between the CPM and each observed system. The CIM server is used to transmit the metric data gathered by GPMSERVE about an observed system back to the CPM server.
- We need one CIM Server on each observed System in the Domain.



Customize the CIM server Security Set Up

- Build Class Descriptor table if not already done.

```
SETROPTS CLASSACT(CDT) RACLIST(CDT)
```

- Add profile to CDT to define a new Dynamic Class called WBEM

```
RDEFINE CDT WBEM UACC(NONE) CDTINFO(
  CASE(UPPER)
  FIRST(ALPHA)
  MAXLENGTH(246)
  MAXLENX(246)
  KEYQUALIFIERS(0)
  PROFILESALLOWED(YES)
  OTHER(ALPHA,NUMERIC)
  POSIT(200)
  RACLIST(REQUIRED)
 )
```

- Activate the WBEM Class & Create CIMSERV Profile.

```
SETROPTS CLASSACT(WBEM) RACLIST(WBEM)
RDEFINE WBEM CIMSERV
SETROPTS CLASSACT(WBEM) RACLIST(WBEM) REFRESH
```

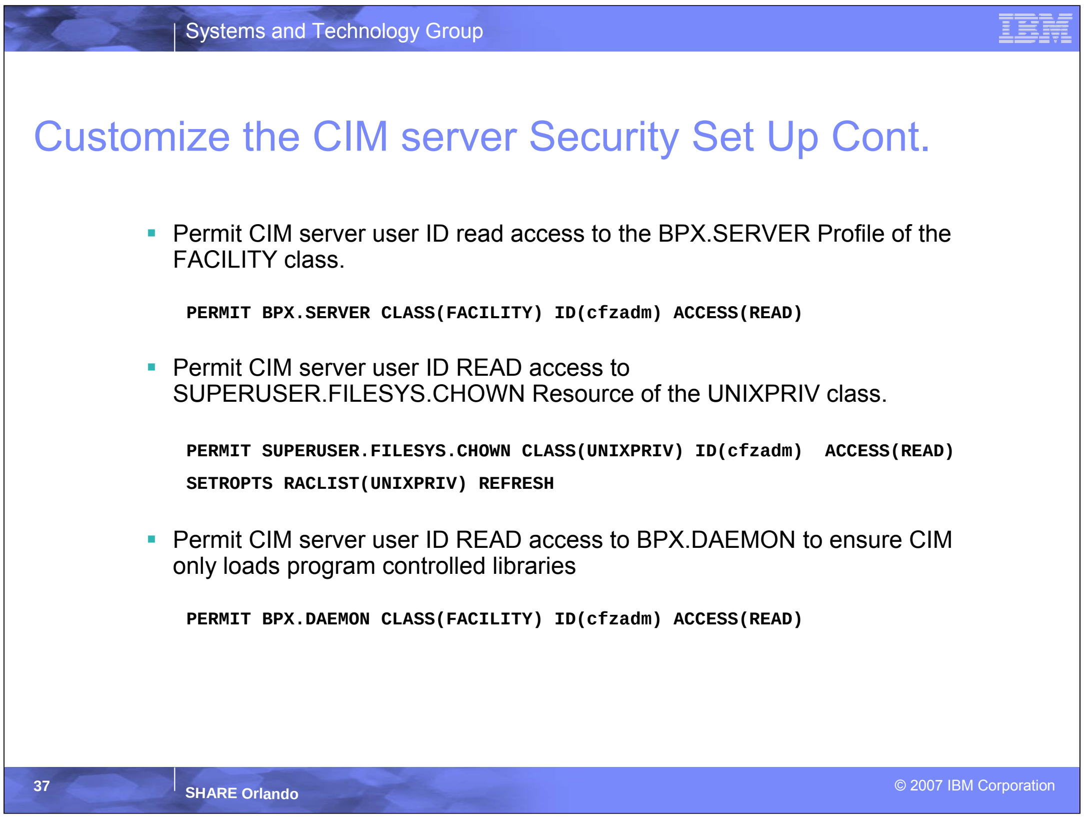
- Create User ID for CIM server to run under.

```
ADDUSER cfzadm OWNER(sys1) DFLTGRP(sys1)
  OMVS(uid(autouid))
```

- Grant CIM server user ID CONTROL access to profile CIMSERV in class WBEM.

```
PERMIT CIMSERV CLASS(WBEM) ACCESS(CONTROL)
  ID(cfzadm)
SETROPTS CLASSACT(WBEM) RACLIST(WBEM) REFRESH
```

- First off there is a lot of security set up for CIM Server and I have encapsulated my setup In a job that I will provide for you so that you can customize it if you wish.
- Cimserver security is broken down in to 2 parts. Authentication and Authorization.
  1. Authentication: Managed through the CIMSERV profile of the WBEM RACF Class, which implies that every CIM user must have a zOS user ID .
  2. Authorization is on a per resource basis and is granted based on the requesting user IDs authority. This is done via UNIX System services Thread level Security.
- The first thing you will need to do is to create a dynamic class for the WBEM class and define the CIMSERV profile. This is a 4 step process.
  1. Build the Class Descriptor Table (CDT). Note: This may already be done in your environment.
  2. Add profile to the CDT to define a new Dynamic Class called WBEM.
  3. Activate the dynamic class.
  4. Create the CIMSERV profile.
- Define the user that CIMSERVER will run under and permit him to the CIMSERV profile of the WBEM class. This is much like we did above for GPMSERVE. Note that I used pegadm as my user ID and SYS1 as my group ID and a UID of 990. The group ID and UID will need to be customized for your environment.



Customize the CIM server Security Set Up Cont.

- Permit CIM server user ID read access to the BPX.SERVER Profile of the FACILITY class.

```
PERMIT BPX.SERVER CLASS(FACILITY) ID(cfzadm) ACCESS(READ)
```

- Permit CIM server user ID READ access to SUPERUSER.FILESYS.CHOWN Resource of the UNIXPRIV class.

```
PERMIT SUPERUSER.FILESYS.CHOWN CLASS(UNIXPRIV) ID(cfzadm)  ACCESS(READ)
SETROPTS RACLIST(UNIXPRIV) REFRESH
```

- Permit CIM server user ID READ access to BPX.DAEMON to ensure CIM only loads program controlled libraries

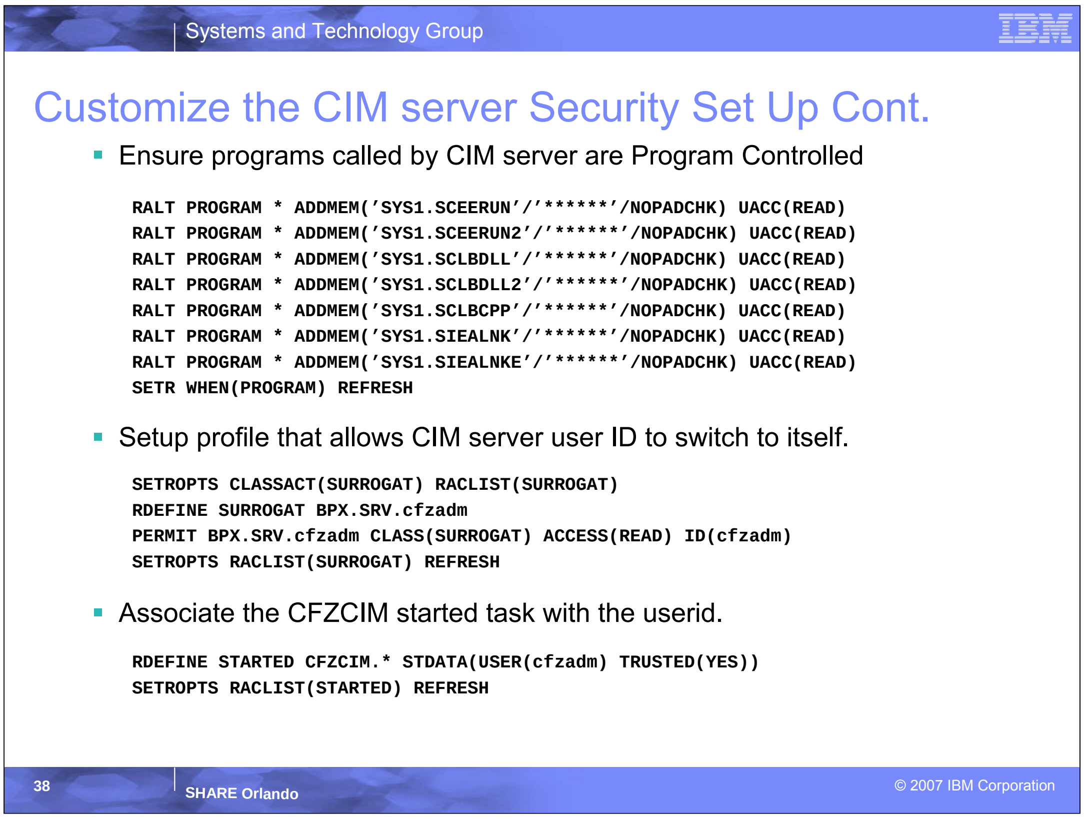
```
PERMIT BPX.DAEMON CLASS(FACILITY) ID(cfzadm) ACCESS(READ)
```

•Let me first say that the rest of these steps are necessary to follow the Enhanced Security Model. Many of these steps could be deleted by simply providing UID 0 to user ID that CIM server runs under. This presenter recommends you take the additional time to have the enhanced security set up.

•So now we are working on establishing the resource authorization model for the CIM server. As mentioned above we are using a model were access is granted per resource and is based on the requesting users access authority. In order to grant the narrowest scope of authority necessary, we will first grant the CIM server user ID read access to BPX.SERVER. By granting read access, the CIM server and the client must both be authorized to access the resource.

•By permitting the CIM user ID to the UNIXPRIV profile SUPERUSER.FILESYS.CHOWN it allows it to have superuser authority to change the ownership of files.

•See notes under GPMSERVE for a discussion of BPX.DAEMON.



Customize the CIM server Security Set Up Cont.

- Ensure programs called by CIM server are Program Controlled

```
RALT PROGRAM * ADDMEM('SYS1.SCEERUN'/'******'/NOPADCHK) UACC(READ)
RALT PROGRAM * ADDMEM('SYS1.SCEERUN2'/'******'/NOPADCHK) UACC(READ)
RALT PROGRAM * ADDMEM('SYS1.SCLBDLL'/'******'/NOPADCHK) UACC(READ)
RALT PROGRAM * ADDMEM('SYS1.SCLBDLL2'/'******'/NOPADCHK) UACC(READ)
RALT PROGRAM * ADDMEM('SYS1.SCLBCPP'/'******'/NOPADCHK) UACC(READ)
RALT PROGRAM * ADDMEM('SYS1.SIEALNK'/'******'/NOPADCHK) UACC(READ)
RALT PROGRAM * ADDMEM('SYS1.SIEALNKE'/'******'/NOPADCHK) UACC(READ)
SETR WHEN(PROGRAM) REFRESH
```

- Setup profile that allows CIM server user ID to switch to itself.

```
SETROPTS CLASSACT(SURROGAT) RACLIST(SURROGAT)
RDEFINE SURROGAT BPX.SRV.cfzadm
PERMIT BPX.SRV.cfzadm CLASS(SURROGAT) ACCESS(READ) ID(cfzadm)
SETROPTS RACLIST(SURROGAT) REFRESH
```

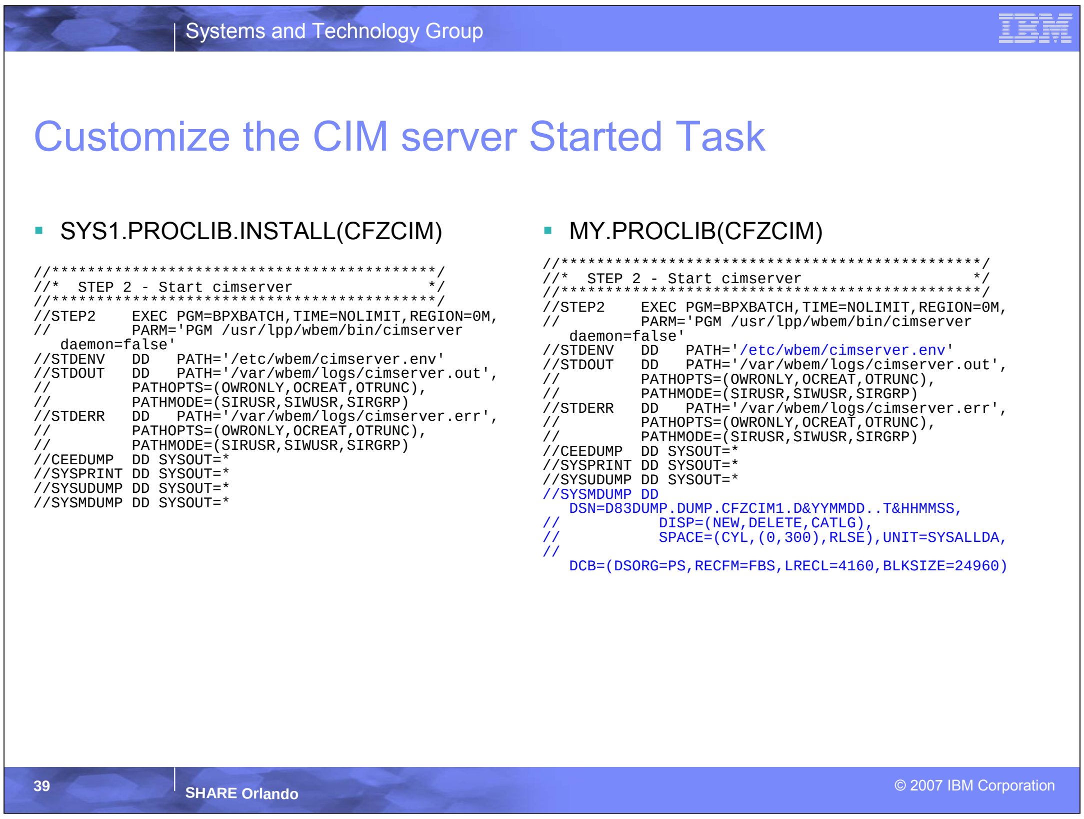
- Associate the CFZCIM started task with the userid.

```
RDEFINE STARTED CFZCIM.* STDATA(USER(cfzadm) TRUSTED(YES))
SETROPTS RACLIST(STARTED) REFRESH
```

•As noted during GPMSERVE setup, all programs loaded by CIM server must be program controlled. We will need to ensure that they are by issuing an RALTER command to mark them program controlled.

•Since the CIM server uses services that can be run from either the server or the clients security context the CIM server user ID must be able to switch to himself or to the clients ID. In order to do this the BPX.SRV profile of the SURROGAT class must be defined and the CIM server user ID must be permitted READ access to it. Later, as clients are defined you will also need to permit them to the BPX.SRV profile.

•Lastly just like with GPMSERVE we need to Associate the CFZCIM started task with the CIM server user ID.



Customize the CIM server Started Task

- SYS1.PROCLIB.INSTALL(CFZCIM)

```
//********************************************/
//*  STEP 2 - Start cimserver               */
//********************************************/
//STEP2    EXEC PGM=BPXBATCH,TIME=NOLIMIT,REGION=0M,
//         PARM='PGM /usr/lpp/wbem/bin/cimserver
   daemon=false'
//STDENV   DD   PATH='/etc/wbem/cimserver.env'
//STDOUT   DD   PATH='/var/wbem/logs/cimserver.out',
//         PATHOPTS=(OWRONLY,OCREAT,OTRUNC),
//         PATHMODE=(SIRUSR,SIWUSR,SIRGRP)
//STDERR   DD   PATH='/var/wbem/logs/cimserver.err',
//         PATHOPTS=(OWRONLY,OCREAT,OTRUNC),
//         PATHMODE=(SIRUSR,SIWUSR,SIRGRP)
//CEEDUMP  DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSMDUMP DD SYSOUT=*
```

- MY.PROCLIB(CFZCIM)

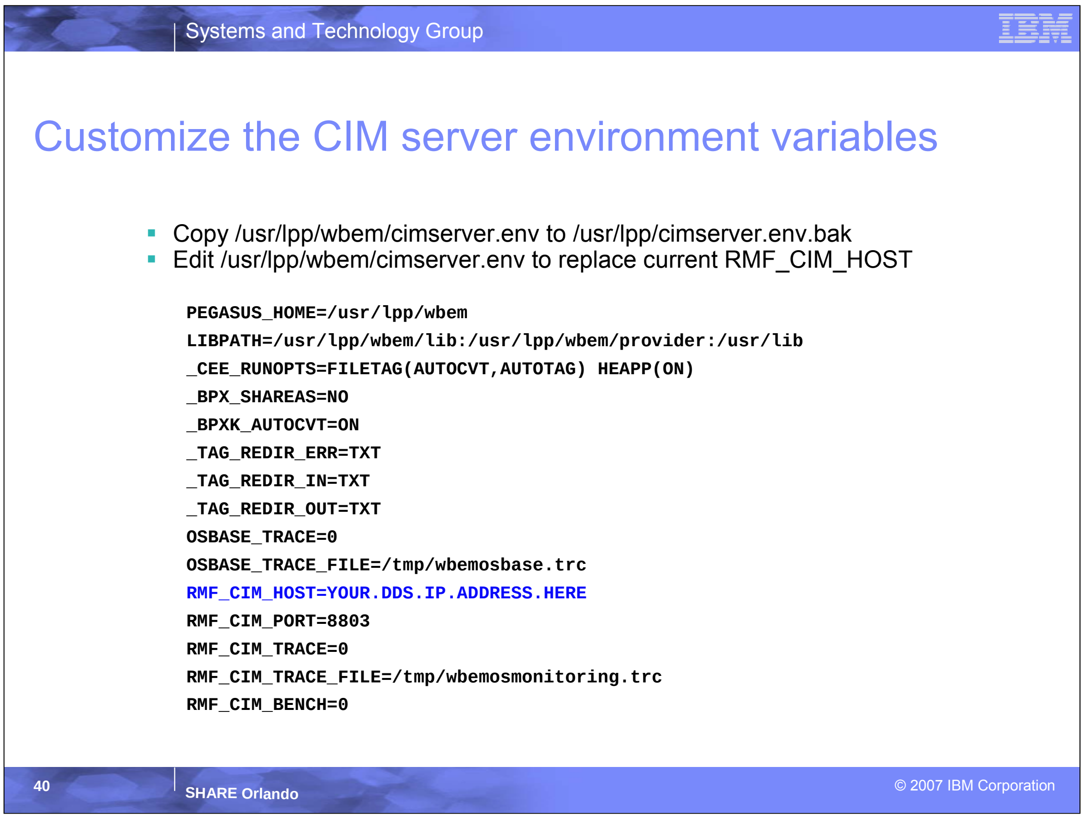
```
//**********************************************/
//*  STEP 2 - Start cimserver                 */
//**********************************************/
//STEP2    EXEC PGM=BPXBATCH,TIME=NOLIMIT,REGION=0M,
//         PARM='PGM /usr/lpp/wbem/bin/cimserver
   daemon=false'
//STDENV   DD   PATH='/etc/wbem/cimserver.env'
//STDOUT   DD   PATH='/var/wbem/logs/cimserver.out',
//         PATHOPTS=(OWRONLY,OCREAT,OTRUNC),
//         PATHMODE=(SIRUSR,SIWUSR,SIRGRP)
//STDERR   DD   PATH='/var/wbem/logs/cimserver.err',
//         PATHOPTS=(OWRONLY,OCREAT,OTRUNC),
//         PATHMODE=(SIRUSR,SIWUSR,SIRGRP)
//CEEDUMP  DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSMDUMP DD
   DSN=D83DUMP.DUMP.CFZCIM1.D&YYMMDD..T&HHMMSS,
//           DISP=(NEW,DELETE,CATLG),
//           SPACE=(CYL,(0,300),RLSE),UNIT=SYSALLDA,
//
   DCB=(DSORG=PS,RECFM=FBS,LRECL=4160,BLKSIZE=24960)
```

•We need to copy the CFZCIM proc from the install proclib in to your working proclib.

•This proc could be used as is, but if for some reason the CM server takes a dump that dump will end up in the CIM server users home directory. For that reason we customized the SYSMDUMP statement to point the dump to a dump dataset of our choosing.

•The other point worth mentioning here is that when cimserver is started using the started task, the environment variables are set using a file called cimserver.env that lives in /etc/wbem/ on each of your LPARs. We will look at that file more on the next slide.

•So of course to start the CIM server all you need to do is type S CFZCIM and to stop it P CFZCIM.



Customize the CIM server environment variables

- Copy /usr/lpp/wbem/cimserver.env to /usr/lpp/cimserver.env.bak
- Edit /usr/lpp/wbem/cimserver.env to replace current RMF_CIM_HOST

```
PEGASUS_HOME=/usr/lpp/wbem
LIBPATH=/usr/lpp/wbem/lib:/usr/lpp/wbem/provider:/usr/lib
_CEE_RUNOPTS=FILETAG(AUTOCVT,AUTOTAG) HEAPP(ON)
_BPX_SHAREAS=NO
_BPXK_AUTOCVT=ON
_TAG_REDIR_ERR=TXT
_TAG_REDIR_IN=TXT
_TAG_REDIR_OUT=TXT
OSBASE_TRACE=0
OSBASE_TRACE_FILE=/tmp/wbemosbase.trc
RMF_CIM_HOST=YOUR.DDS.IP.ADDRESS.HERE
RMF_CIM_PORT=8803
RMF_CIM_TRACE=0
RMF_CIM_TRACE_FILE=/tmp/wbemosmonitoring.trc
RMF_CIM_BENCH=0
```

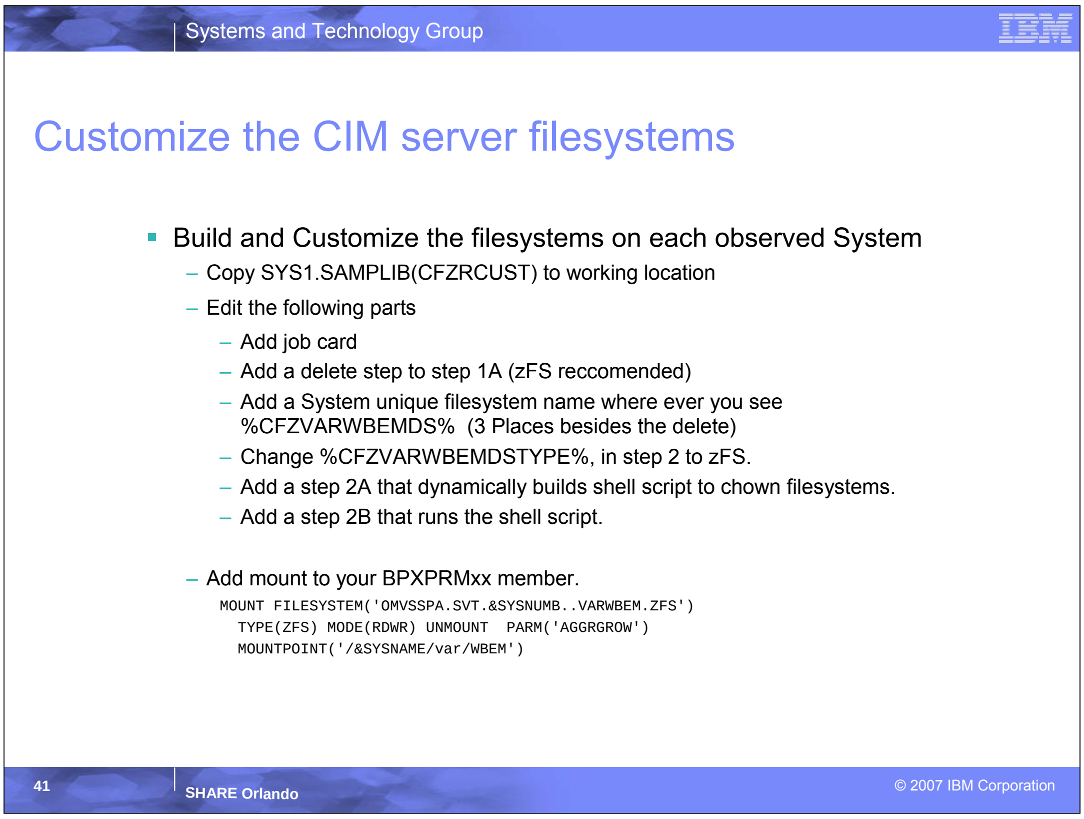
•As I mentioned before, the environment variable used for running the CIM server are set from a file called /etc/wbem/cimserver.env.

•This file is established during the filesystem customization step that will follow in the next slides. The original of this file is shipped in /usr/lpp/wbem/cimserver.env.

•We only need to make one change to the .env file in order to make it work for Capacity Provisioning. This change is to the RMF_CIM_HOST variable.

•RMF_CIM_HOST defines the target ip address or hostname for the RMF DDS. In our case we will use the IP address since that will give us the best possible performance.

•We could do the filesystem customization and then edit cimserver.env on each LPAR, but it makes more sense to go to /usr/lpp/wbem/cimserver.env make a quick copy to preserve the original and then edit this copy. This way the change only needs to be made once and is propagated to all the LPARs during the customization step.



Customize the CIM server filesystems

- Build and Customize the filesystems on each observed System
  - Copy SYS1.SAMPLIB(CFZRCUST) to working location
  - Edit the following parts
    - Add job card
    - Add a delete step to step 1A (zFS reccomended)
    - Add a System unique filesystem name where ever you see %CFZVARWBEMDS% (3 Places besides the delete)
    - Change %CFZVARWBEMDSTYPE%, in step 2 to zFS.
    - Add a step 2A that dynamically builds shell script to chown filesystems.
    - Add a step 2B that runs the shell script.
  - Add mount to your BPXPRMxx member.

```
MOUNT FILESYSTEM('OMVSSPA.SVT.&SYSNUMB..VARWBEM.ZFS')
  TYPE(ZFS) MODE(RDWR) UNMOUNT  PARM('AGGRGROW')
  MOUNTPOINT('/&SYSNAME/var/WBEM')
```

•Edit & Run CFZRCUST (this needs to be done once for each system in the plex).

•Filesystem will be mounted on /&sysname/var/wbem

•These mount points are in the shipped versions of the filesystem

•Contains directories that hold:

•Logs-cimserver.err and cimserver.out

•Repository-contains provider class repository and instance information

•The contents of these directories is variable and potentially unique and would not necessarily fit in your /&sysname//var/wbem directory.

•We recommend that you add a delete of the unique filesystem name at the top of step 1A simply because the job will fail if for some reason you need to rerun it after the allocate step has completed.

•We also added a reset of the return code to 0. We did this because the delete is of course going to fail the first time the job is run. (See Next Slide for example)

•Since IBM designated zFS as the strategic filesystem in zOS 1.7 we recommend using zFS for your /var/wbem filesystem.

•If you use a middle level qualifier in your file system name, that relates to a system specific symbol, it will make mounting these filesystems much easier. In my example on the next slide you will see that the 3rd qualifier is an S1. We created a Symbol on our system &SYSNUMB that resolves to S1 on the first LPAR S2 on the second and so on.

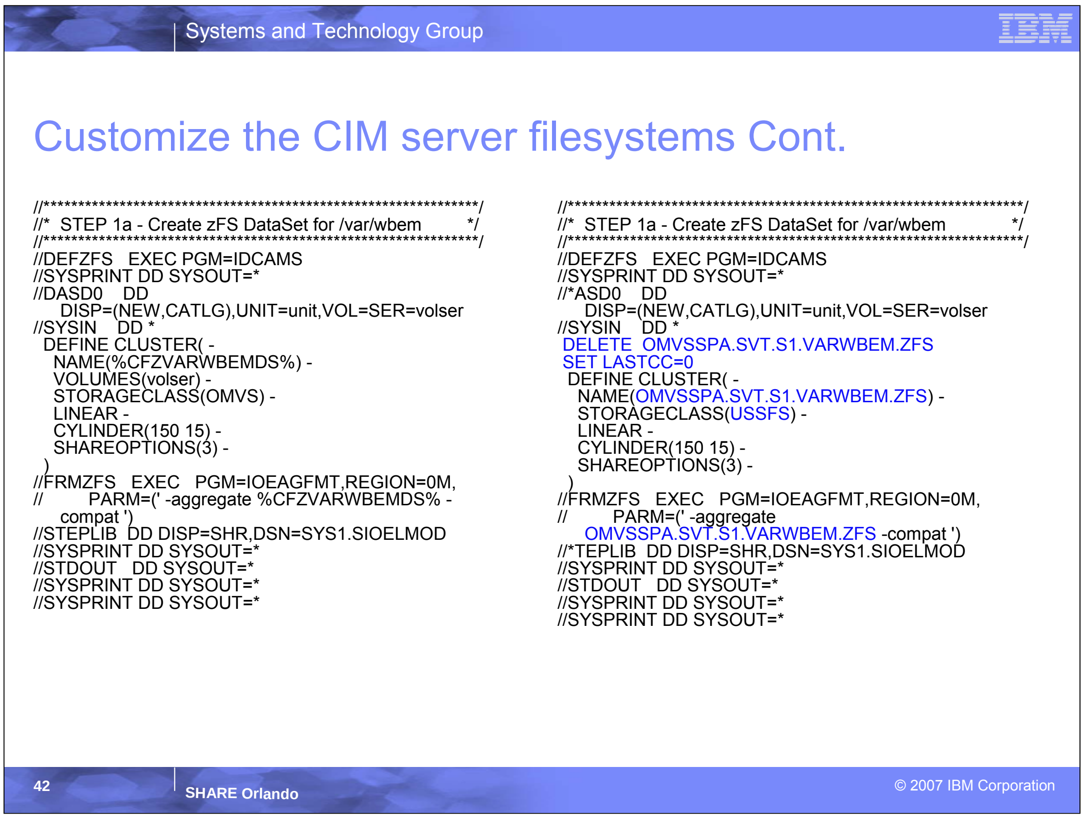
•As I mentioned during security setup, we need to change the ownership of all the CIM files to the CIM user ID. In order to do this I have added a new step 2A to the CFZRCUST job that dynamically builds a shell script in /tmp. As you can see from the example on the next page, this has the user ID and group ID that I declared earlier. You will need to adjust this for your environment.

•A step 2B was added to then execute this new script.

•If you have used the system symbol I mentioned above you will only need to add a single mount statement to your BPXPRMxx member in order to mount the filesystems across your sysplex.

•Note that the filesystem needs to be mounted Read/Write so that log entries and provider instance updates can be written out.

•We added a zFS AGGRGROW command to the mount statement. This will allow zFS to grow the filesystem, as long as there is space on the device, to reduce the danger of filling the filesystem.



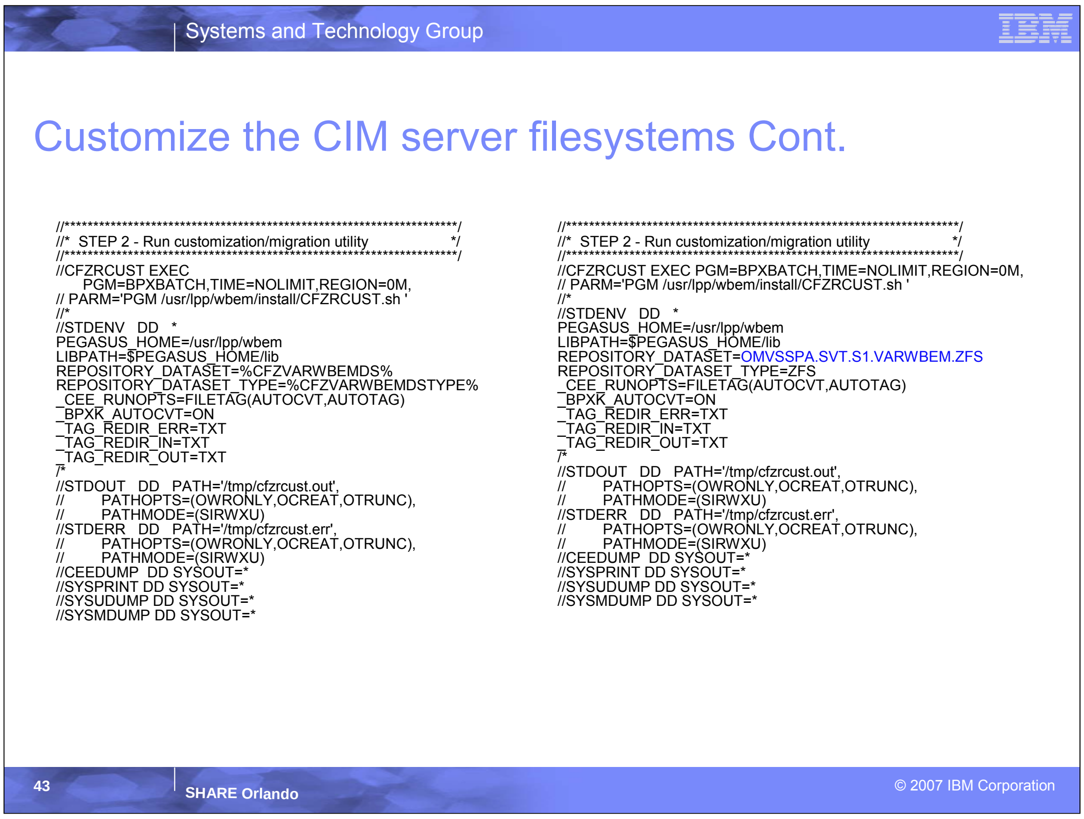
Customize the CIM server filesystems Cont.

```
//**************************************************************/
//*  STEP 1a - Create zFS DataSet for /var/wbem         */
//**************************************************************/
//DEFZFS   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//DASD0    DD
     DISP=(NEW,CATLG),UNIT=unit,VOL=SER=volser
//SYSIN    DD *
  DEFINE CLUSTER( -
    NAME(%CFZVARWBEMDS%) -
    VOLUMES(volser) -
    STORAGECLASS(OMVS) -
    LINEAR -
    CYLINDER(150 15) -
    SHAREOPTIONS(3) -
  )
//FRMZFS   EXEC   PGM=IOEAGFMT,REGION=0M,
//        PARM=(' -aggregate %CFZVARWBEMDS% -
     compat ')
//STEPLIB  DD DISP=SHR,DSN=SYS1.SIOELMOD
//SYSPRINT DD SYSOUT=*
//STDOUT   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
```

```
//**************************************************************/
//*  STEP 1a - Create zFS DataSet for /var/wbem             */
//**************************************************************/
//DEFZFS   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//*ASD0    DD
     DISP=(NEW,CATLG),UNIT=unit,VOL=SER=volser
//SYSIN    DD *
 DELETE  OMVSSPA.SVT.S1.VARWBEM.ZFS
 SET LASTCC=0
  DEFINE CLUSTER( -
    NAME(OMVSSPA.SVT.S1.VARWBEM.ZFS) -
    STORAGECLASS(USSFS) -
    LINEAR -
    CYLINDER(150 15) -
    SHAREOPTIONS(3) -
  )
//FRMZFS   EXEC   PGM=IOEAGFMT,REGION=0M,
//        PARM=(' -aggregate
     OMVSSPA.SVT.S1.VARWBEM.ZFS -compat ')
//*TEPLIB  DD DISP=SHR,DSN=SYS1.SIOELMOD
//SYSPRINT DD SYSOUT=*
//STDOUT   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
```

•The Blue Highlights are what have been changed.

•Remember that you will need to make a unique filesystem name for each LPAR in your Plex

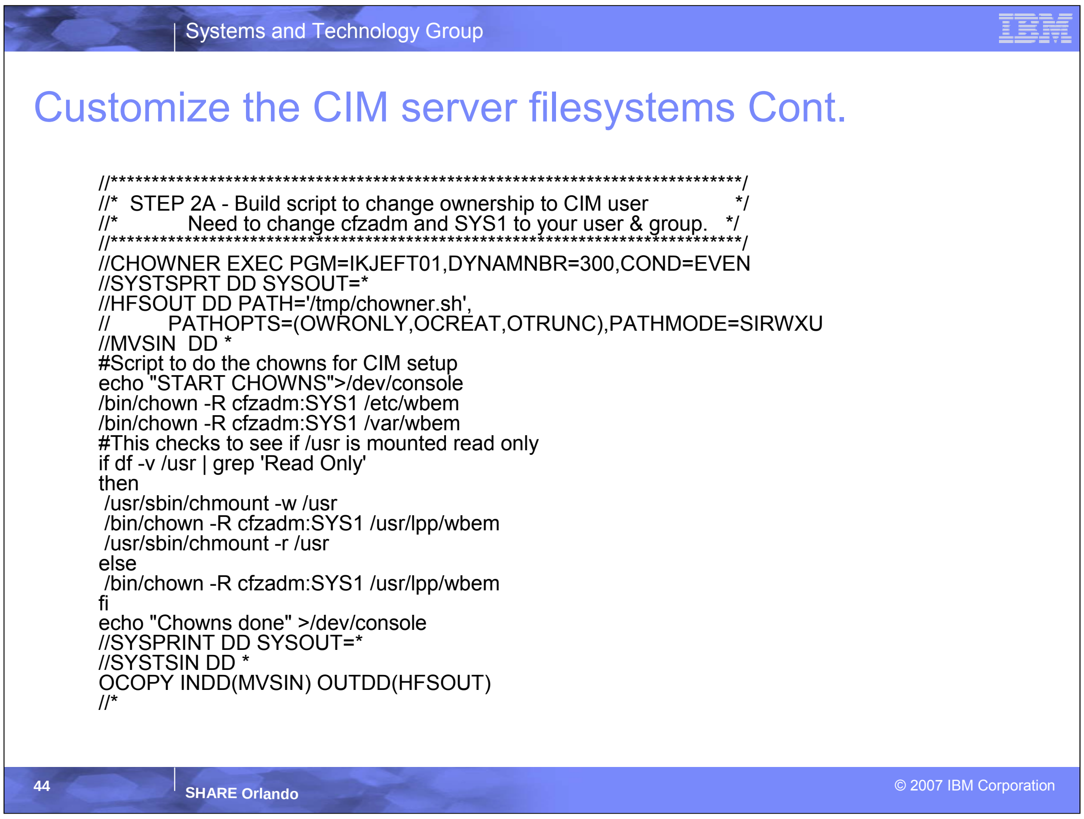


Customize the CIM server filesystems Cont.

```
//*******************************************************************/
//*  STEP 2 - Run customization/migration utility                   */
//*******************************************************************/
//CFZRCUST EXEC
      PGM=BPXBATCH,TIME=NOLIMIT,REGION=0M,
// PARM='PGM /usr/lpp/wbem/install/CFZRCUST.sh '
//*
//STDENV   DD   *
PEGASUS_HOME=/usr/lpp/wbem
LIBPATH=$PEGASUS_HOME/lib
REPOSITORY_DATASET=%CFZVARWBEMDS%
REPOSITORY_DATASET_TYPE=%CFZVARWBEMDSTYPE%
_CEE_RUNOPTS=FILETAG(AUTOCVT,AUTOTAG)
_BPXK_AUTOCVT=ON
_TAG_REDIR_ERR=TXT
_TAG_REDIR_IN=TXT
_TAG_REDIR_OUT=TXT
/*
//STDOUT   DD   PATH='/tmp/cfzrcust.out',
//         PATHOPTS=(OWRONLY,OCREAT,OTRUNC),
//         PATHMODE=(SIRWXU)
//STDERR   DD   PATH='/tmp/cfzrcust.err',
//         PATHOPTS=(OWRONLY,OCREAT,OTRUNC),
//         PATHMODE=(SIRWXU)
//CEEDUMP  DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSMDUMP DD SYSOUT=*
```

```
//*******************************************************************/
//*  STEP 2 - Run customization/migration utility                   */
//*******************************************************************/
//CFZRCUST EXEC PGM=BPXBATCH,TIME=NOLIMIT,REGION=0M,
// PARM='PGM /usr/lpp/wbem/install/CFZRCUST.sh '
//*
//STDENV   DD   *
PEGASUS_HOME=/usr/lpp/wbem
LIBPATH=$PEGASUS_HOME/lib
REPOSITORY_DATASET=OMVSSPA.SVT.S1.VARWBEM.ZFS
REPOSITORY_DATASET_TYPE=ZFS
_CEE_RUNOPTS=FILETAG(AUTOCVT,AUTOTAG)
_BPXK_AUTOCVT=ON
_TAG_REDIR_ERR=TXT
_TAG_REDIR_IN=TXT
_TAG_REDIR_OUT=TXT
/*
//STDOUT   DD   PATH='/tmp/cfzrcust.out',
//         PATHOPTS=(OWRONLY,OCREAT,OTRUNC),
//         PATHMODE=(SIRWXU)
//STDERR   DD   PATH='/tmp/cfzrcust.err',
//         PATHOPTS=(OWRONLY,OCREAT,OTRUNC),
//         PATHMODE=(SIRWXU)
//CEEDUMP  DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSMDUMP DD SYSOUT=*
```

•Again this just shows what needs to be changed.



Systems and Technology Group

Customize the CIM server filesystems Cont.

```
//*******************************************************************************/
//*  STEP 2A - Build script to change ownership to CIM user              */
//*           Need to change cfzadm and SYS1 to your user & group.   */
//*******************************************************************************/
//CHOWNER EXEC PGM=IKJEFT01,DYNAMNBR=300,COND=EVEN
//SYSTSPRT DD SYSOUT=*
//HFSOUT DD PATH='/tmp/chowner.sh',
//         PATHOPTS=(OWRONLY,OCREAT,OTRUNC),PATHMODE=SIRWXU
//MVSIN  DD *
#Script to do the chowns for CIM setup
echo "START CHOWNS">/dev/console
/bin/chown -R cfzadm:SYS1 /etc/wbem
/bin/chown -R cfzadm:SYS1 /var/wbem
#This checks to see if /usr is mounted read only
if df -v /usr | grep 'Read Only'
then
 /usr/sbin/chmount -w /usr
 /bin/chown -R cfzadm:SYS1 /usr/lpp/wbem
 /usr/sbin/chmount -r /usr
else
 /bin/chown -R cfzadm:SYS1 /usr/lpp/wbem
fi
echo "Chowns done" >/dev/console
//SYSPRINT DD SYSOUT=*
//SYSTSIN DD *
OCOPY INDD(MVSIN) OUTDD(HFSOUT)
//*
```

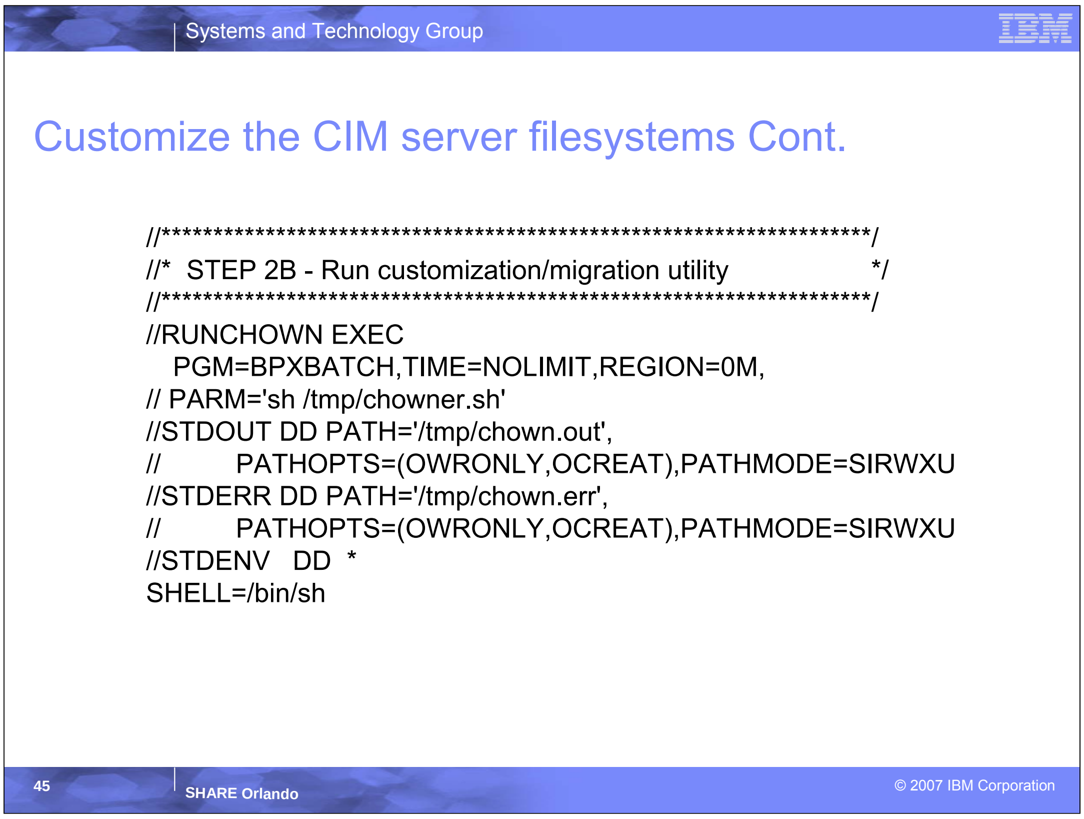
44 SHARE Orlando © 2007 IBM Corporation

•So this step is going to create a file named chowner.sh in the /tmp directory and then it is going to copy the script defined on the MVSIN DD to that file.

•This script uses cfzcim as the CIMserver user ID and SYS1 as the group ID. It will need to be modified if you use anything else.

•It first changes the file ownership on all the files that were copied to your local /etc and then your new local filesystem mounted on /var/wbem.

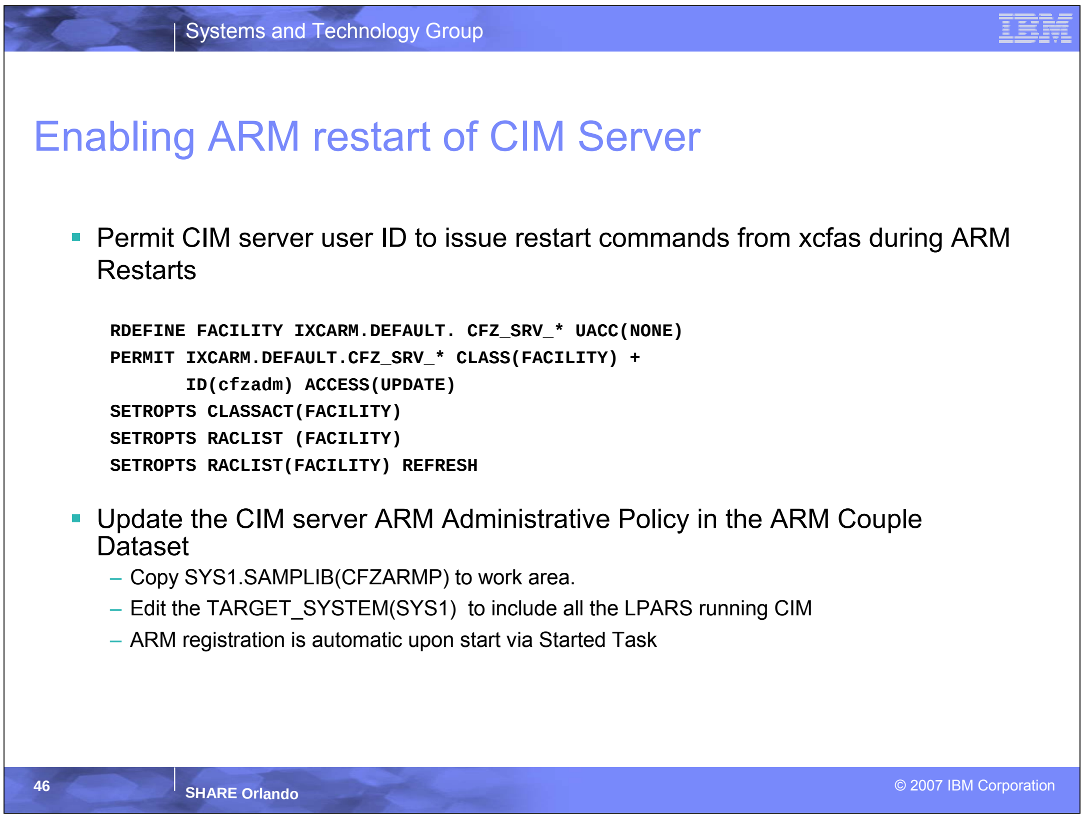
•The next part checks to see if the filesystem mounted on /usr is mounted READ Only. If it is it will switch it to READ/WRITE and do the change owner then put it back to READ. If it isn't it will just do the change owner.



Customize the CIM server filesystems Cont.

```
//*********************************************************************/
//*  STEP 2B - Run customization/migration utility                    */
//*********************************************************************/
//RUNCHOWN EXEC
   PGM=BPXBATCH,TIME=NOLIMIT,REGION=0M,
// PARM='sh /tmp/chowner.sh'
//STDOUT DD PATH='/tmp/chown.out',
//         PATHOPTS=(OWRONLY,OCREAT),PATHMODE=SIRWXU
//STDERR DD PATH='/tmp/chown.err',
//         PATHOPTS=(OWRONLY,OCREAT),PATHMODE=SIRWXU
//STDENV   DD  *
SHELL=/bin/sh
```

•This step simply executes the shell script that was placed in /tmp in the last step. We could have written other steps to clean up the script when done, but didn't. Also you can see that we wrote standard out and error to /tmp also. If something goes wrong you will need to go to /tmp to view the output.



Enabling ARM restart of CIM Server

- Permit CIM server user ID to issue restart commands from xcfas during ARM Restarts

```
RDEFINE FACILITY IXCARM.DEFAULT. CFZ_SRV_* UACC(NONE)
PERMIT IXCARM.DEFAULT.CFZ_SRV_* CLASS(FACILITY) +
       ID(cfzadm) ACCESS(UPDATE)
SETROPTS CLASSACT(FACILITY)
SETROPTS RACLIST (FACILITY)
SETROPTS RACLIST(FACILITY) REFRESH
```

- Update the CIM server ARM Administrative Policy in the ARM Couple Dataset
  - Copy SYS1.SAMPLIB(CFZARMP) to work area.
  - Edit the TARGET_SYSTEM(SYS1) to include all the LPARS running CIM
  - ARM registration is automatic upon start via Started Task

•Since Capacity Provisioning is used to improve the systems ability to meet its goals, it is important that it be set up to maintain High availability. To that end we will now do the necessary configuration to allow ARM restarts of the CIM Server, thus ensuring continuous communication of metric data between the DDS and CPM.

•First off we will need to do yet a little more security setup. Since the CIM server does not run authorized, the CIM userid needs UPDATE access to the FACILITY class IXCARM.DEFAULT.CFZ_SRV_*. This allows the CIM start commands to be issued out of the XCF address space.

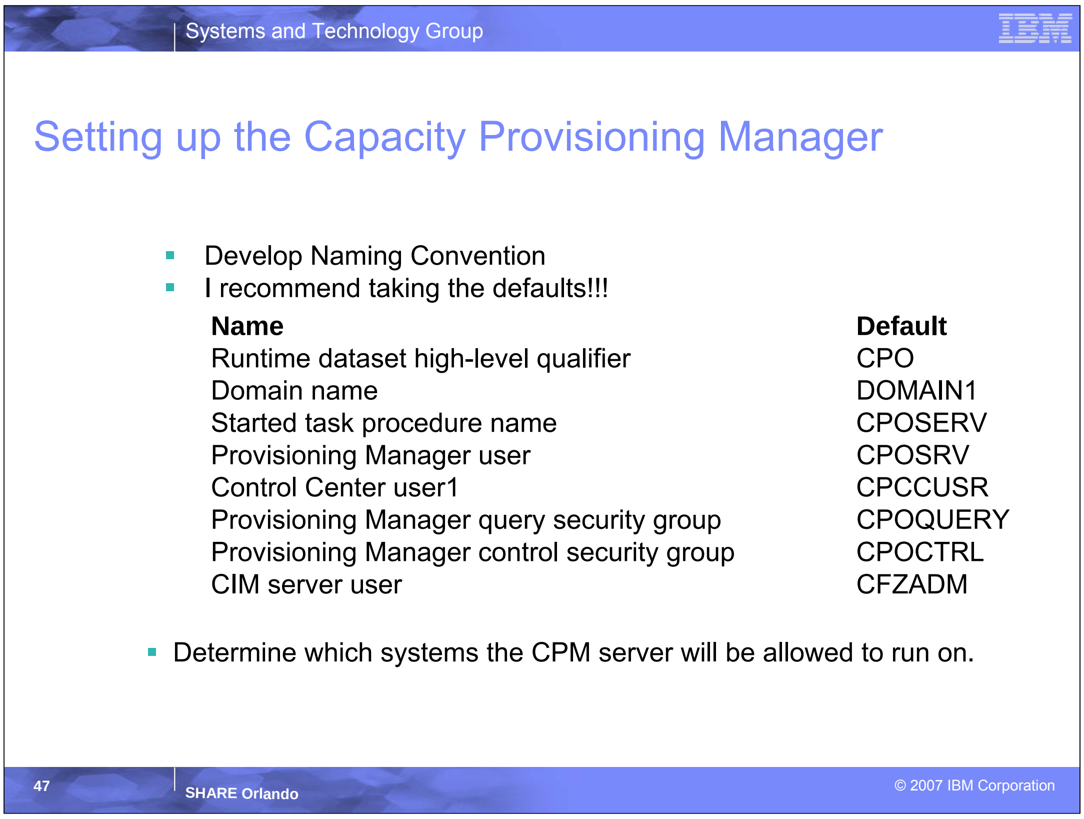
•I am assuming that you have ARM active in your environment. If you do not then refer to *z/OS MVS Setting Up a Sysplex* for information on setting it up.

•We need to add/update the CIM ARM policy in order for ARM to take action in the case of a CIM outage. To do this we will need to copy SYS1.SAMPLIB(CFZARMP) to our work area and edit it.

•Besides the jobcard there is really only one change that needs to be made. You will need to add each system in your sysplex, that will be running CIM, to the TARGET_SYSTEM(SYS1) statement.

•Just a reminder. CIM can be started from the UNIX Shell or started task, but ARM restart can only occur when the started task is used. This is one of the reasons for the recommendation that you use the started task.

•At this point the CIM servers should be ready to start on each of the observed systems.



Setting up the Capacity Provisioning Manager

- Develop Naming Convention
- I recommend taking the defaults!!!

Name	**Default**
Runtime dataset high-level qualifier	CPO
Domain name	DOMAIN1
Started task procedure name	CPOSERV
Provisioning Manager user	CPOSRV
Control Center user1	CPCCUSR
Provisioning Manager query security group	CPOQUERY
Provisioning Manager control security group	CPOCTRL
CIM server user	CFZADM

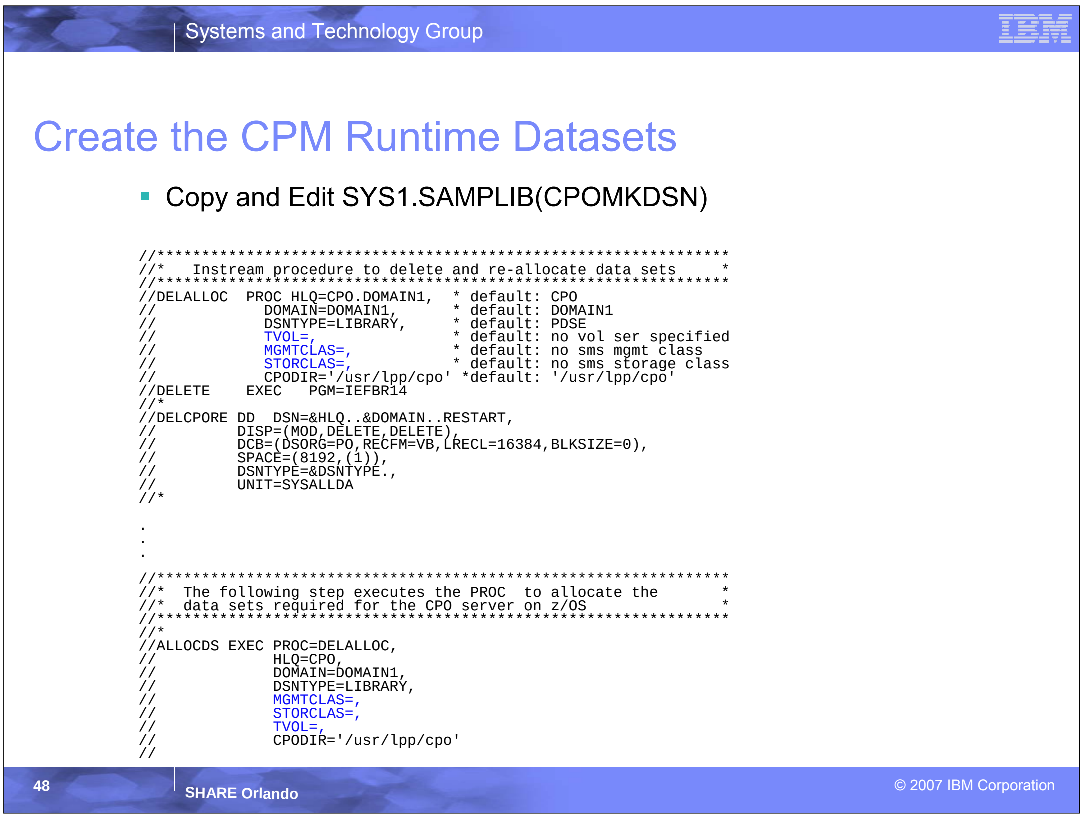
- Determine which systems the CPM server will be allowed to run on.

•Before we get started doing the actual setup we need to first develop our naming convention. These are the names used for things such as the control datasets, user IDs, Started tasks, etc.

•I recommend you take the defaults to start with. The primary reason for this is that it will be much easier to follow in the manual and my examples, and it will be much easier for us to help you if you have any trouble.

•We need to decide on a system that will be our primary location for the CPM server and Alternative locations where CPM can run if for some reason the primary becomes unavailable.

•What this means is that both locations must have access to the CPM datasets and the path structure to the prerequisite software must be the same. This should not be a problem if you are putting the primary and alternate on the same sysplex, but will need to be verified. On my system I made it possible to run the CPM server on any LPAR in the plex.



Create the CPM Runtime Datasets

- Copy and Edit SYS1.SAMPLIB(CPOMKDSN)

```
//******************************************************************
//*   Instream procedure to delete and re-allocate data sets      *
//******************************************************************
//DELALLOC  PROC HLQ=CPO.DOMAIN1,  * default: CPO
//             DOMAIN=DOMAIN1,     * default: DOMAIN1
//             DSNTYPE=LIBRARY,    * default: PDSE
//             TVOL=,              * default: no vol ser specified
//             MGMTCLAS=,          * default: no sms mgmt class
//             STORCLAS=,          * default: no sms storage class
//             CPODIR='/usr/lpp/cpo' *default: '/usr/lpp/cpo'
//DELETE    EXEC   PGM=IEFBR14
//*
//DELCPORE DD  DSN=&HLQ..&DOMAIN..RESTART,
//          DISP=(MOD,DELETE,DELETE),
//          DCB=(DSORG=PO,RECFM=VB,LRECL=16384,BLKSIZE=0),
//          SPACE=(8192,(1)),
//          DSNTYPE=&DSNTYPE.,
//          UNIT=SYSALLDA
//*

.
.
.

//******************************************************************
//*  The following step executes the PROC  to allocate the        *
//*  data sets required for the CPO server on z/OS                *
//******************************************************************
//*
//ALLOCDS EXEC PROC=DELALLOC,
//              HLQ=CPO,
//              DOMAIN=DOMAIN1,
//              DSNTYPE=LIBRARY,
//              MGMTCLAS=,
//              STORCLAS=,
//              TVOL=,
//              CPODIR='/usr/lpp/cpo'
//
```

- We start the actual setup by creating the CPM runtime datasets. There are 4 datasets required and the are used for the following:
  1. Provisioning Manager Parameters – used to hold environment variables and configuration data.
  2. Restart Dataset – Maintains information necessary in the event that CPM needs to be restarted on the primary or one of the alternate systems.
  3. Policy Repository – Holds CPM Policies
  4. Domain Configuration Repository - Holds Domain Configurations

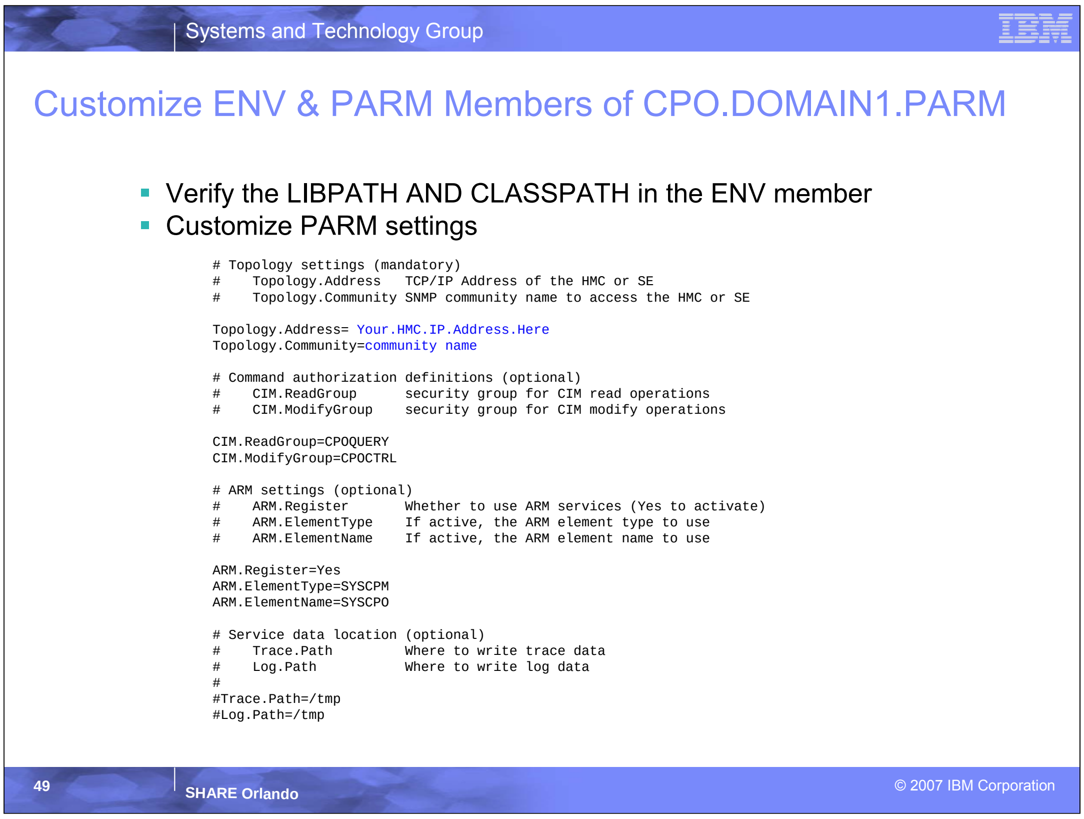
- There is a job SYS1.SAMPLIB(CPOMKDSN) that will create these datasets. If you choose to take the default naming convention of CPO for the HLQ and DOMAIN1 for your first Domain, and as the second level qualifier, then you only need to provide a job card and Identify where you want the filesystems allocated.
- Remember that you need to update this job in two places with the dataset location information. Under the DEALLOC and ALLOCDS steps.
- If the defaults are taken you will end up with the following Datasets:

  CPO.DOMAIN1.DOMCFG

  CPO.DOMAIN1.PARM

  CPO.DOMAIN1.POLICIES

  CPO.DOMAIN1.RESTART
- You will also find that the CPO.DOMAIN1.PARM has been populated with a base PARM and ENV member.



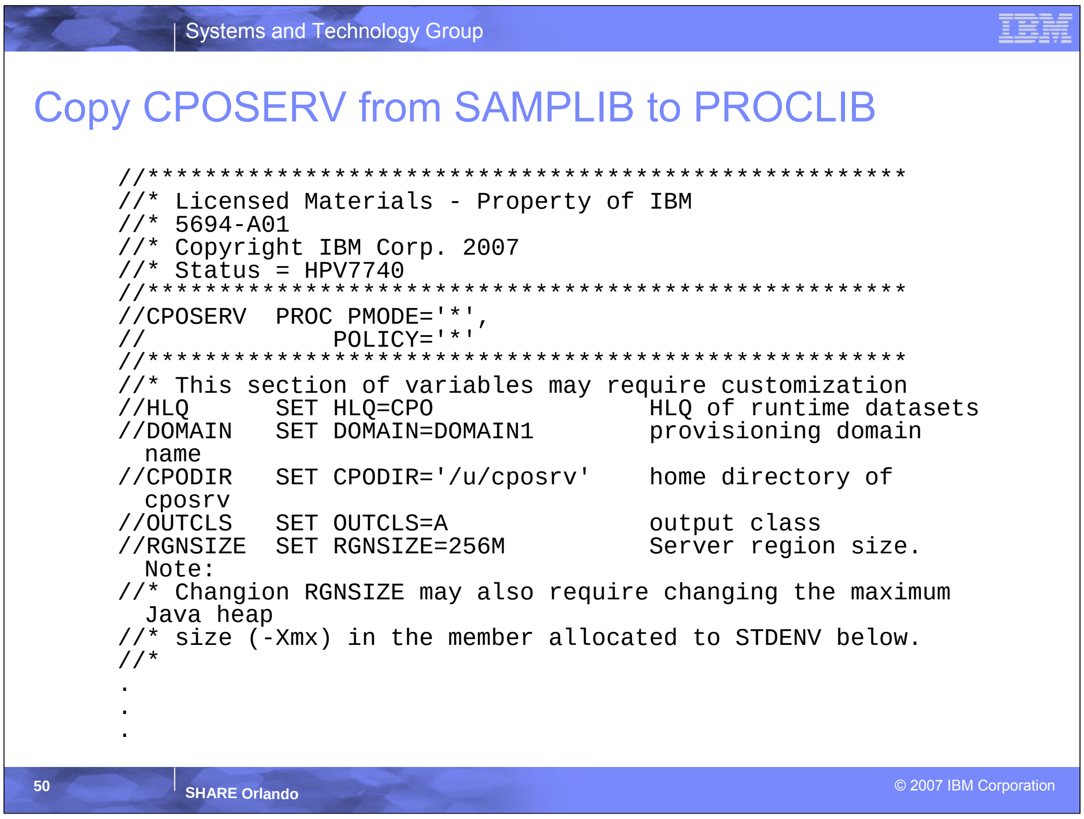
Customize ENV & PARM Members of CPO.DOMAIN1.PARM

- Verify the LIBPATH AND CLASSPATH in the ENV member
- Customize PARM settings

```
# Topology settings (mandatory)
#    Topology.Address   TCP/IP Address of the HMC or SE
#    Topology.Community SNMP community name to access the HMC or SE

Topology.Address= Your.HMC.IP.Address.Here
Topology.Community=community name

# Command authorization definitions (optional)
#    CIM.ReadGroup      security group for CIM read operations
#    CIM.ModifyGroup    security group for CIM modify operations

CIM.ReadGroup=CPOQUERY
CIM.ModifyGroup=CPOCTRL

# ARM settings (optional)
#    ARM.Register       Whether to use ARM services (Yes to activate)
#    ARM.ElementType    If active, the ARM element type to use
#    ARM.ElementName    If active, the ARM element name to use

ARM.Register=Yes
ARM.ElementType=SYSCPM
ARM.ElementName=SYSCPO

# Service data location (optional)
#    Trace.Path         Where to write trace data
#    Log.Path           Where to write log data
#
#Trace.Path=/tmp
#Log.Path=/tmp
```

- If you have used default locations for your CPM Installation directories, JAVA SDK 5.0 (31bit), and SAF Jar file location you should not need to make any changes to the CPO.DOMAIN1.PARM(ENV) member. You will want to just verify that the LIBPATH and CLASSPATH members do match your installations file structure.
- There are four parameter sections in the PARM Member that need to be considered.
  1. Topology Settings – Which consist of the IP address of the HMC that will be used for Hardware Communications and the SNMP Community name. The SNMP community essentially defines a group of devices and management stations that will communicate with each other. You can make this any name you like as long as what you put in this member matches what you define on the HMC (More on the HMC definition later).
  2. Command Authorization definitions – These settings allow Communication between the CPCC user and the CPM. We will need to uncomment this.
  3. ARM Settings – These settings allow for ARM management of the CPM. We will want to change ARM.REGISTER to yes and uncomment `ARM.Register=Yes,ARM.ElementType=SYSCPM, ARM.ElementName=SYSCPO`
  4. Trace and Log data settings – This is used ONLY if you want to move these from their default location. If you would like to move them then you will need to make sure the new locations exist and allow write access. The Default location is /tmp.

- For simplicity and performance I once again that I took the defaults for all settings where there existed a default and we are using IP addresses rather then hostnames.

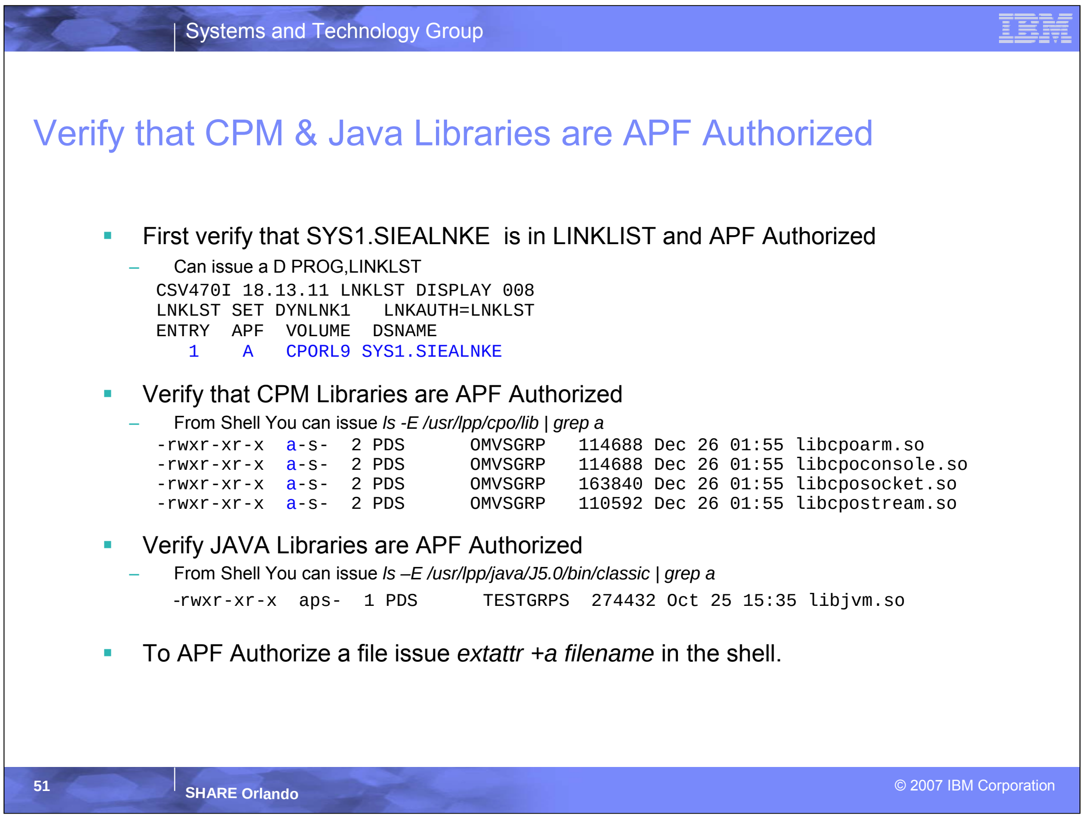


Copy CPOSERV from SAMPLIB to PROCLIB

```
//*****************************************************
//* Licensed Materials - Property of IBM
//* 5694-A01
//* Copyright IBM Corp. 2007
//* Status = HPV7740
//*****************************************************
//CPOSERV  PROC PMODE='*',
//             POLICY='*'
//*****************************************************
//* This section of variables may require customization
//HLQ      SET HLQ=CPO               HLQ of runtime datasets
//DOMAIN   SET DOMAIN=DOMAIN1        provisioning domain
  name
//CPODIR   SET CPODIR='/u/cposrv'    home directory of
  cposrv
//OUTCLS   SET OUTCLS=A              output class
//RGNSIZE  SET RGNSIZE=256M          Server region size.
  Note:
//* Changion RGNSIZE may also require changing the maximum
  Java heap
//* size (-Xmx) in the member allocated to STDENV below.
//*
.
.
.
```

•You will need to copy the Started Task from SYS1.SAMPLIB(CPOSERV) to you proclib.

•If you have taken the defaults you will not need to make any changes to the proc. If you didn't take the defaults for HLQ, Domain or Home directory for the CPOSERV user ID you will be able to make these changes in the variable section of the proc.



Verify that CPM & Java Libraries are APF Authorized

- First verify that SYS1.SIEALNKE is in LINKLIST and APF Authorized
  - Can issue a D PROG,LINKLST

```
CSV470I 18.13.11 LNKLST DISPLAY 008
LNKLST SET DYNLNK1   LNKAUTH=LNKLST
ENTRY  APF  VOLUME  DSNAME
   1    A   CPORL9 SYS1.SIEALNKE
```

- Verify that CPM Libraries are APF Authorized
  - From Shell You can issue *ls -E /usr/lpp/cpo/lib | grep a*

```
-rwxr-xr-x  a-s-  2 PDS      OMVSGRP   114688 Dec 26 01:55 libcpoarm.so
-rwxr-xr-x  a-s-  2 PDS      OMVSGRP   114688 Dec 26 01:55 libcpoconsole.so
-rwxr-xr-x  a-s-  2 PDS      OMVSGRP   163840 Dec 26 01:55 libcposocket.so
-rwxr-xr-x  a-s-  2 PDS      OMVSGRP   110592 Dec 26 01:55 libcpostream.so
```

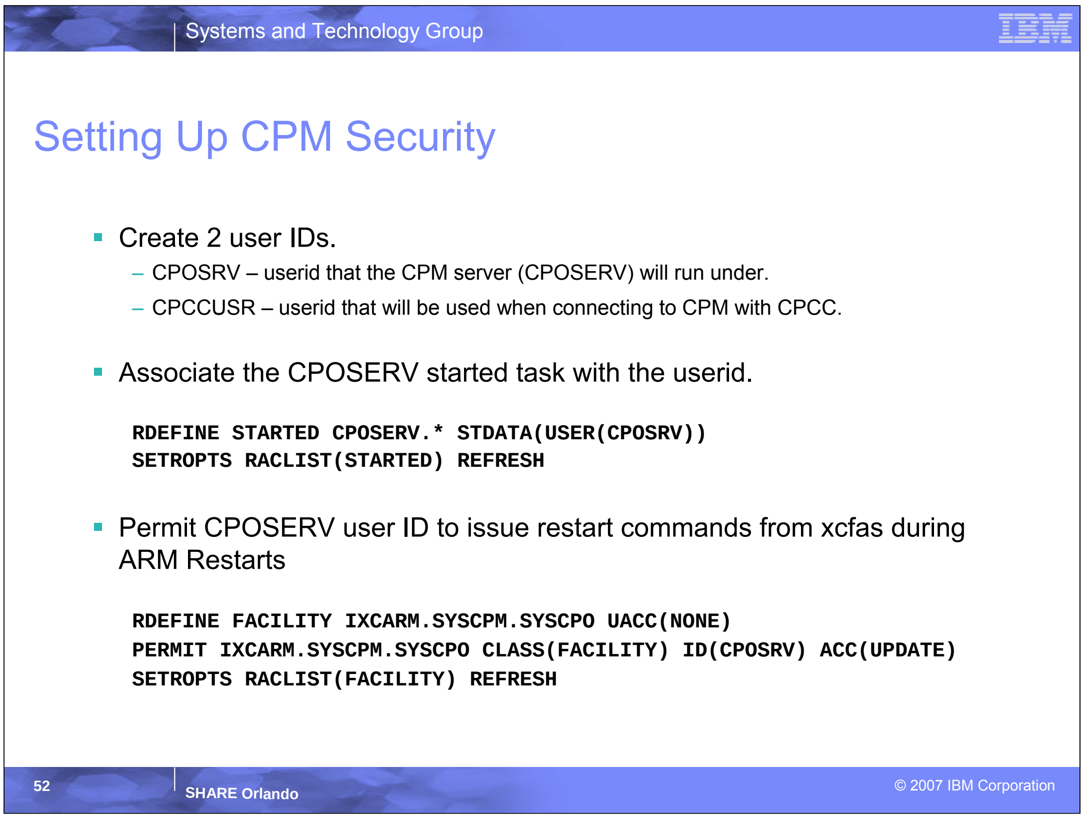
- Verify JAVA Libraries are APF Authorized
  - From Shell You can issue *ls –E /usr/lpp/java/J5.0/bin/classic | grep a*

```
-rwxr-xr-x  aps-  1 PDS      TESTGRPS  274432 Oct 25 15:35 libjvm.so
```

- To APF Authorize a file issue *extattr +a filename* in the shell.

•If for some reason the CPM libraries or the JAVA libraries were copied from one place to another they may have lost their APF Authorization. You should verify that they are still APF authorized and if not issue a extattr +a command.

•It is also worth mentioning again that you need the 31 bit JAVA SDK 5.0 not the 64 bit. The 64 bit version is incompatible with CPM.



Setting Up CPM Security

- Create 2 user IDs.
  - CPOSRV – userid that the CPM server (CPOSERV) will run under.
  - CPCCUSR – userid that will be used when connecting to CPM with CPCC.
- Associate the CPOSERV started task with the userid.

```
RDEFINE STARTED CPOSERV.* STDATA(USER(CPOSRV))
SETROPTS RACLIST(STARTED) REFRESH
```

- Permit CPOSERV user ID to issue restart commands from xcfas during ARM Restarts

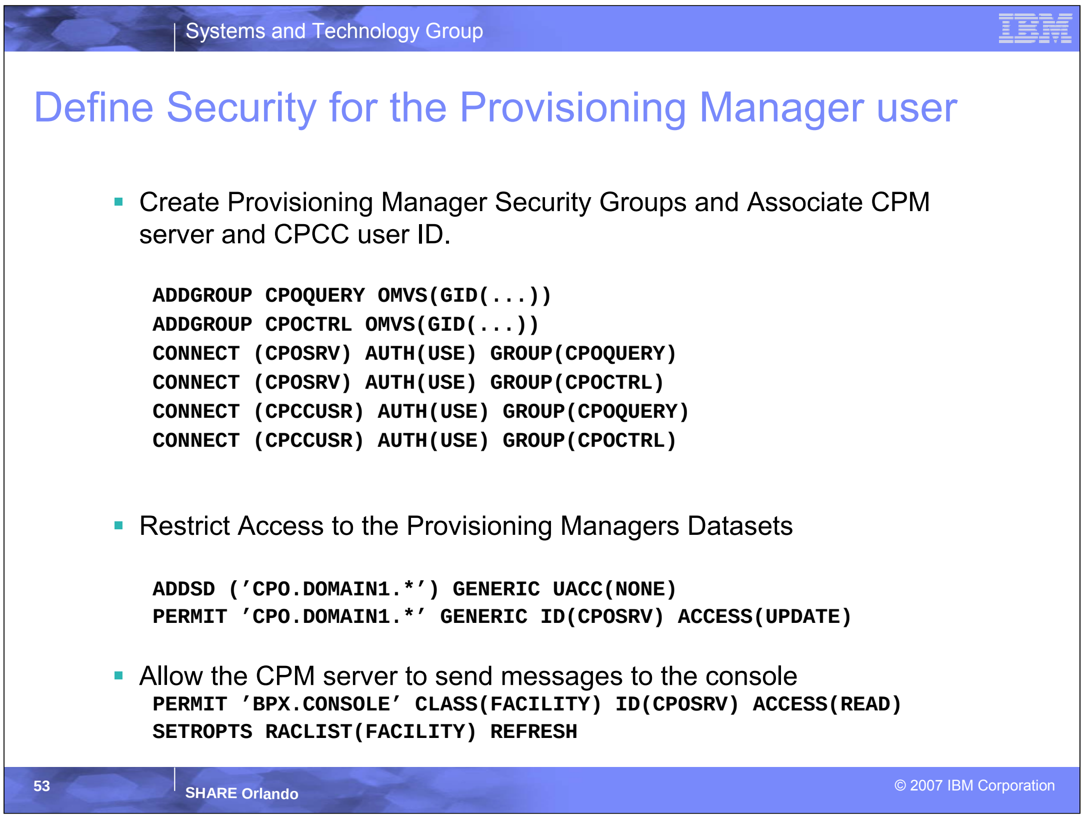
```
RDEFINE FACILITY IXCARM.SYSCPM.SYSCPO UACC(NONE)
PERMIT IXCARM.SYSCPM.SYSCPO CLASS(FACILITY) ID(CPOSRV) ACC(UPDATE)
SETROPTS RACLIST(FACILITY) REFRESH
```

•I created 2 user IDs that match those being used in the documentation. CPOSRV is the userid that the CPM server (CPOSERV) will run under and CPCCUSR is the userid is the userid that we will use to connect from the CPCC on your desktop to the Capacity Provisioning Manager on the host system.

•As with the DDS and the CIM server we need to associate the CPOSERV started task with a the userid that it will run under. In this case CPOSRV. We will be using the same RDEFINE STARTED command as listed above.

•As mentioned before we need to configure CPM for High availability by making it ARM restartable. For this to happen we need the CPOSERV user ID to be able to issue restart commands from the xcf address space during restart.

•You may notice that we are defining the facility class **IXCARM.SYSCPM.SYSCPO.** This corresponds to the ARM Element type and Element name we defined in the CPO.DOMAIN1.PARM(PARM) member. If you used some other names you will want to do this define with those values.



Define Security for the Provisioning Manager user

- Create Provisioning Manager Security Groups and Associate CPM server and CPCC user ID.

```
ADDGROUP CPOQUERY OMVS(GID(...))
ADDGROUP CPOCTRL OMVS(GID(...))
CONNECT (CPOSRV) AUTH(USE) GROUP(CPOQUERY)
CONNECT (CPOSRV) AUTH(USE) GROUP(CPOCTRL)
CONNECT (CPCCUSR) AUTH(USE) GROUP(CPOQUERY)
CONNECT (CPCCUSR) AUTH(USE) GROUP(CPOCTRL)
```

- Restrict Access to the Provisioning Managers Datasets

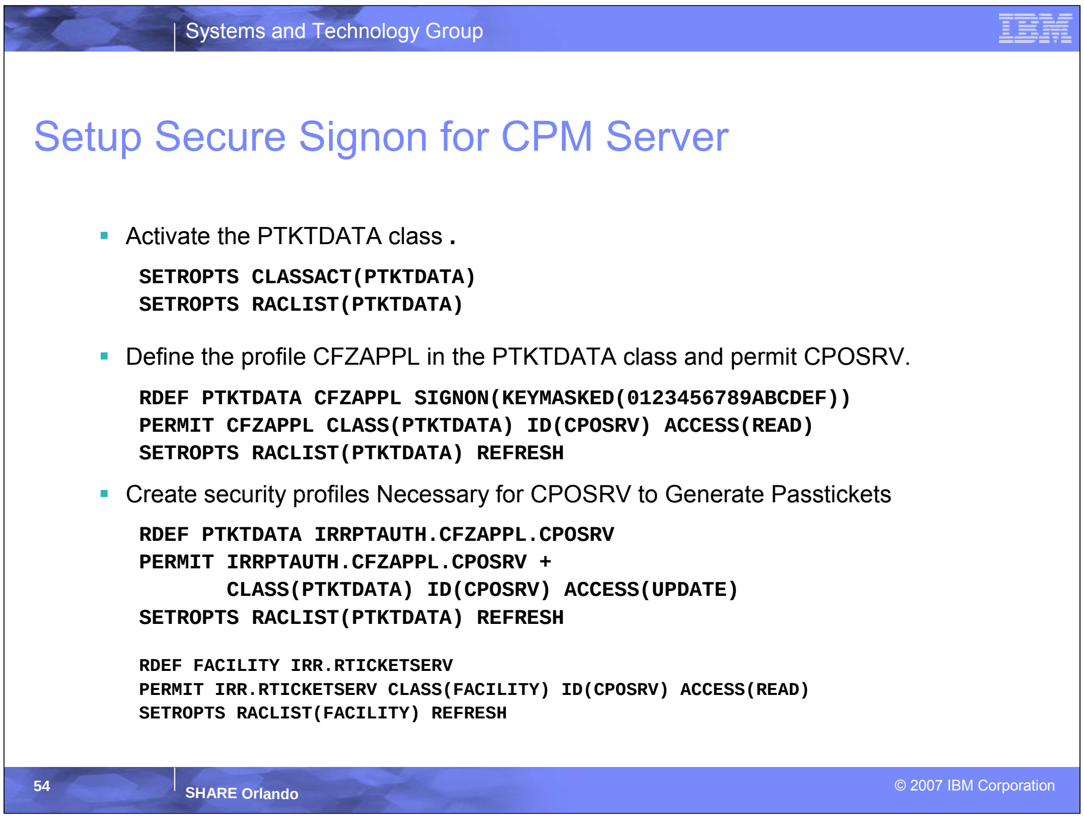
```
ADDSD ('CPO.DOMAIN1.*') GENERIC UACC(NONE)
PERMIT 'CPO.DOMAIN1.*' GENERIC ID(CPOSRV) ACCESS(UPDATE)
```

- Allow the CPM server to send messages to the console

```
PERMIT 'BPX.CONSOLE' CLASS(FACILITY) ID(CPOSRV) ACCESS(READ)
SETROPTS RACLIST(FACILITY) REFRESH
```

•We need to create two security groups CPOQUERY & CPOCTRL that will control Read and Update operations to CPM resources. You will notice that these group names correspond to the group names defined in CPO.DOMAIN1.PARM(PARM). You need to update CPO.DOMAIN1.PARM(PARM) if you choose to use different GROUP names.

•Besides granting CPOSRV (The CPM server user ID) read and update access authority we also want to grant the CPCC user read and update access. The update access is necessary if you want the CPCCUSR to be able to upload CPM Policies and Domain Configurations.



Setup Secure Signon for CPM Server

- Activate the PTKTDATA class .

```
SETROPTS CLASSACT(PTKTDATA)
SETROPTS RACLIST(PTKTDATA)
```

- Define the profile CFZAPPL in the PTKTDATA class and permit CPOSRV.

```
RDEF PTKTDATA CFZAPPL SIGNON(KEYMASKED(0123456789ABCDEF))
PERMIT CFZAPPL CLASS(PTKTDATA) ID(CPOSRV) ACCESS(READ)
SETROPTS RACLIST(PTKTDATA) REFRESH
```

- Create security profiles Necessary for CPOSRV to Generate Passtickets

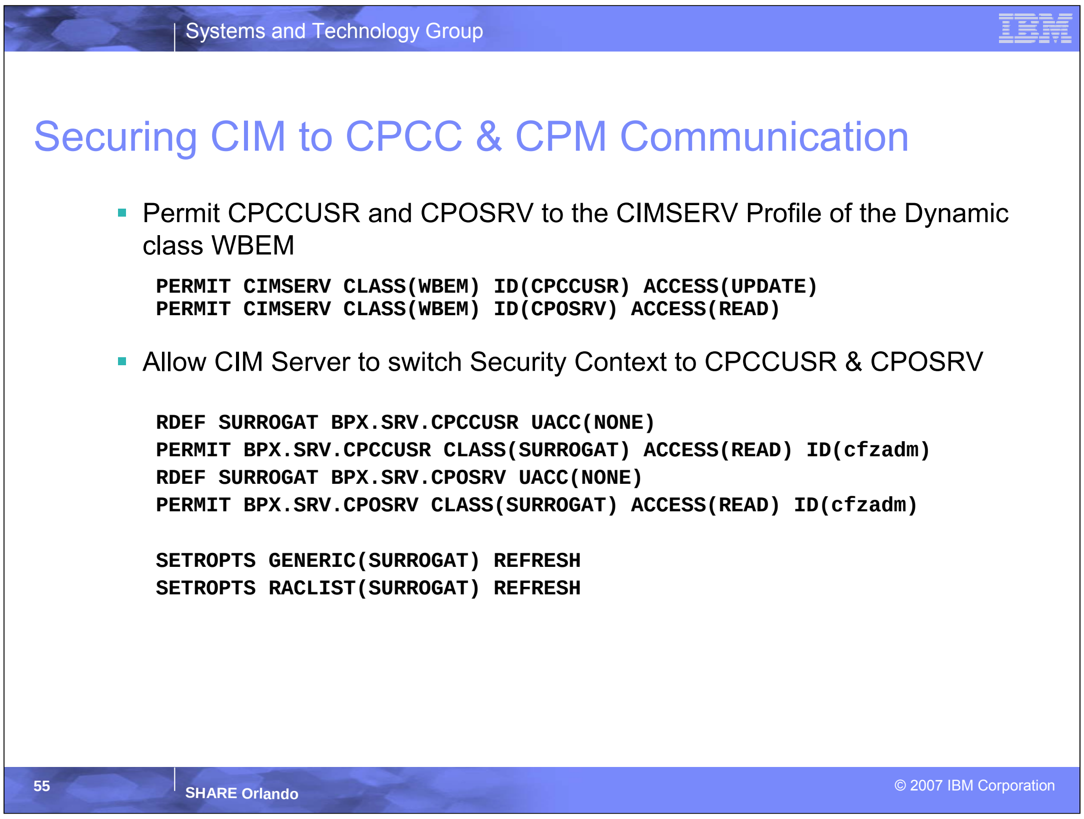
```
RDEF PTKTDATA IRRPTAUTH.CFZAPPL.CPOSRV
PERMIT IRRPTAUTH.CFZAPPL.CPOSRV +
       CLASS(PTKTDATA) ID(CPOSRV) ACCESS(UPDATE)
SETROPTS RACLIST(PTKTDATA) REFRESH

RDEF FACILITY IRR.RTICKETSERV
PERMIT IRR.RTICKETSERV CLASS(FACILITY) ID(CPOSRV) ACCESS(READ)
SETROPTS RACLIST(FACILITY) REFRESH
```

•Communication between the runtime system and the observed system is secured using PassTicket Authentication.

•First you will need to activate the PTKTDATA class if it isn’t already.

•You will then need to define the CFZAPPL profile to the PTKTDATA class. In this example we used the SIGNON operand to define a 64 bit KEYMASKED value, but if you have a a cryptographic product installed on your system you could use a KEYENCRYPTED value instead.

•The provisioning manager user ID will need to be able to generate PassTickets so you will also need to define a security profile **IRRPTAUTH.CFZAPPL.CPOSRV** that allows CPOSRV UPDATE access and profile **IRR.RTICKETSERV** that allows CPOSRV READ access



Securing CIM to CPCC & CPM Communication

- Permit CPCCUSR and CPOSRV to the CIMSERV Profile of the Dynamic class WBEM

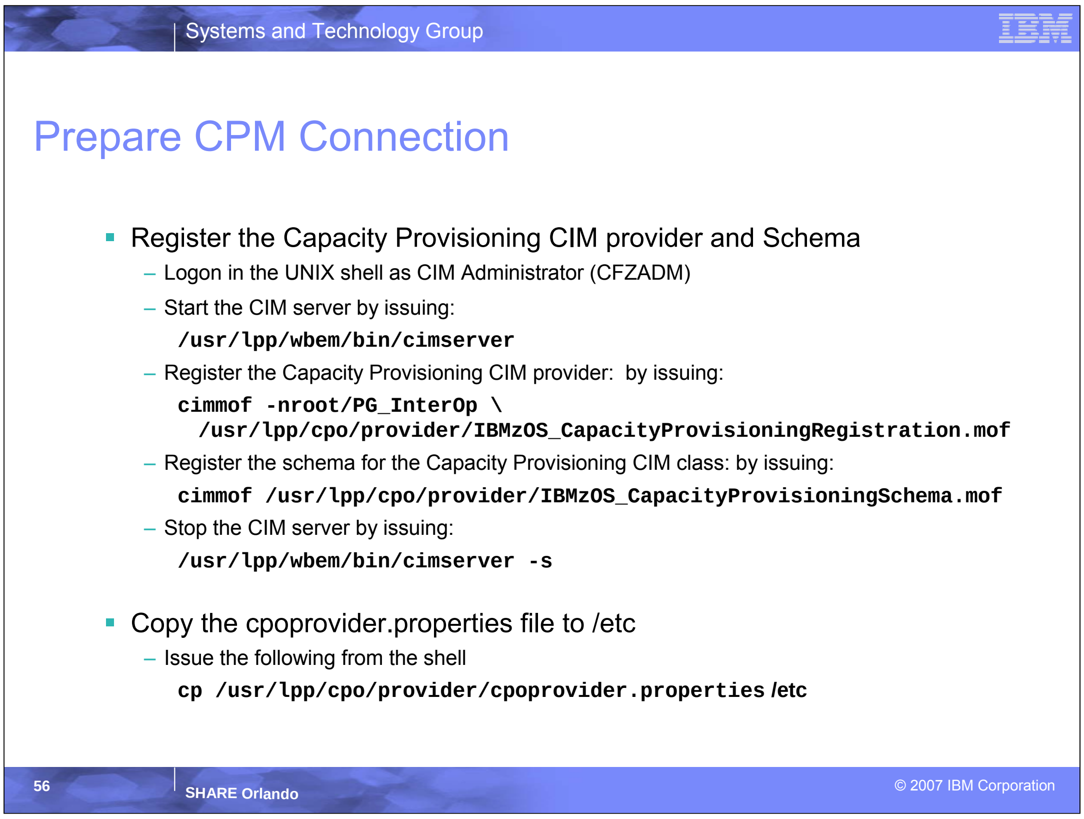
```
PERMIT CIMSERV CLASS(WBEM) ID(CPCCUSR) ACCESS(UPDATE)
PERMIT CIMSERV CLASS(WBEM) ID(CPOSRV) ACCESS(READ)
```

- Allow CIM Server to switch Security Context to CPCCUSR & CPOSRV

```
RDEF SURROGAT BPX.SRV.CPCCUSR UACC(NONE)
PERMIT BPX.SRV.CPCCUSR CLASS(SURROGAT) ACCESS(READ) ID(cfzadm)
RDEF SURROGAT BPX.SRV.CPOSRV UACC(NONE)
PERMIT BPX.SRV.CPOSRV CLASS(SURROGAT) ACCESS(READ) ID(cfzadm)

SETROPTS GENERIC(SURROGAT) REFRESH
SETROPTS RACLIST(SURROGAT) REFRESH
```

•Recall that we use the CIM server for communication between the runtime system and the observed systems and for communication between the CPCC and the runtime system. When we set up the CIM server we defined the dynamic class WBEM and created the profile CIMSERV to secure the CIM server. User CPCCUSR will need UPDATE access while user CPOSRV will only need READ.

•You may also recall that the CIM server uses services that can run under the clients security context. In order for CIM server to switch identities to CPCCUSR or CPOSRV we need to define the profiles BPX.SRV.CPCCUSR and BPX.SRV.CPOSRV of the SURROGAT class. We then need to permit CPCCUSR and CPOSRV READ access to the respective profile.



Prepare CPM Connection

- Register the Capacity Provisioning CIM provider and Schema
  - Logon in the UNIX shell as CIM Administrator (CFZADM)
  - Start the CIM server by issuing:

    ```
    /usr/lpp/wbem/bin/cimserver
    ```
  - Register the Capacity Provisioning CIM provider: by issuing:

    ```
    cimmof -nroot/PG_InterOp \
      /usr/lpp/cpo/provider/IBMzOS_CapacityProvisioningRegistration.mof
    ```
  - Register the schema for the Capacity Provisioning CIM class: by issuing:

    ```
    cimmof /usr/lpp/cpo/provider/IBMzOS_CapacityProvisioningSchema.mof
    ```
  - Stop the CIM server by issuing:

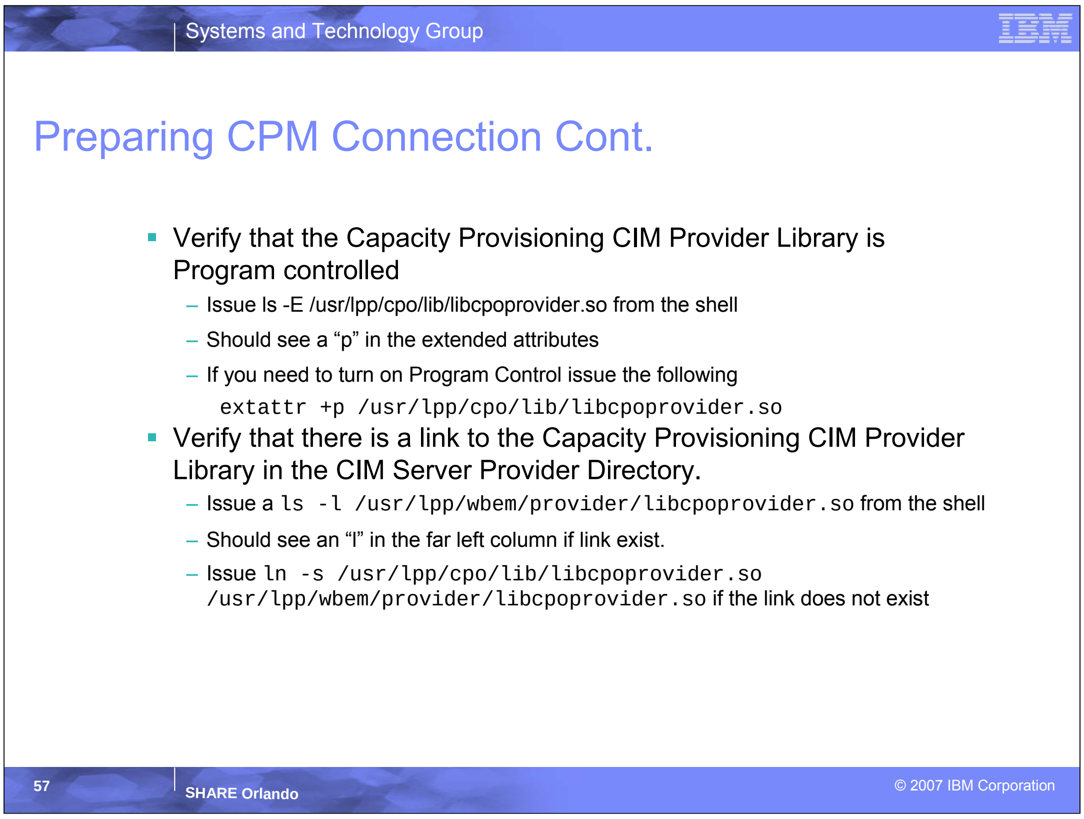
    ```
    /usr/lpp/wbem/bin/cimserver -s
    ```
- Copy the cpoprovider.properties file to /etc
  - Issue the following from the shell

    ```
    cp /usr/lpp/cpo/provider/cpoprovider.properties /etc
    ```

•This part is a number of small steps that need to be done in the shell. I created a job that dynamically builds a shell script to do the steps and then executes the script. It is pretty convenient, but it lacks a lot of the checking an “official” piece of code would have. If you used a Domain name different then the default of DOMAIN1, you will need to edit this file and make that change before copying it.

•You need to register the Capacity Provisioning CIM Provider and schema. This provider provides the interfaces to the CIM server that the CPCC will use for communication. It is important to note that you cannot do the registration unless the CIM server is running. Registration needs to take place on each observed LPAR.

•The cpoprovider.properties file is currently a list of all domains being managed by the Capacity Provisioning Manager. Since this file needs to be copied to /etc and /etc is local to each LPAR in the plex you will need to copy this file on each observed system.



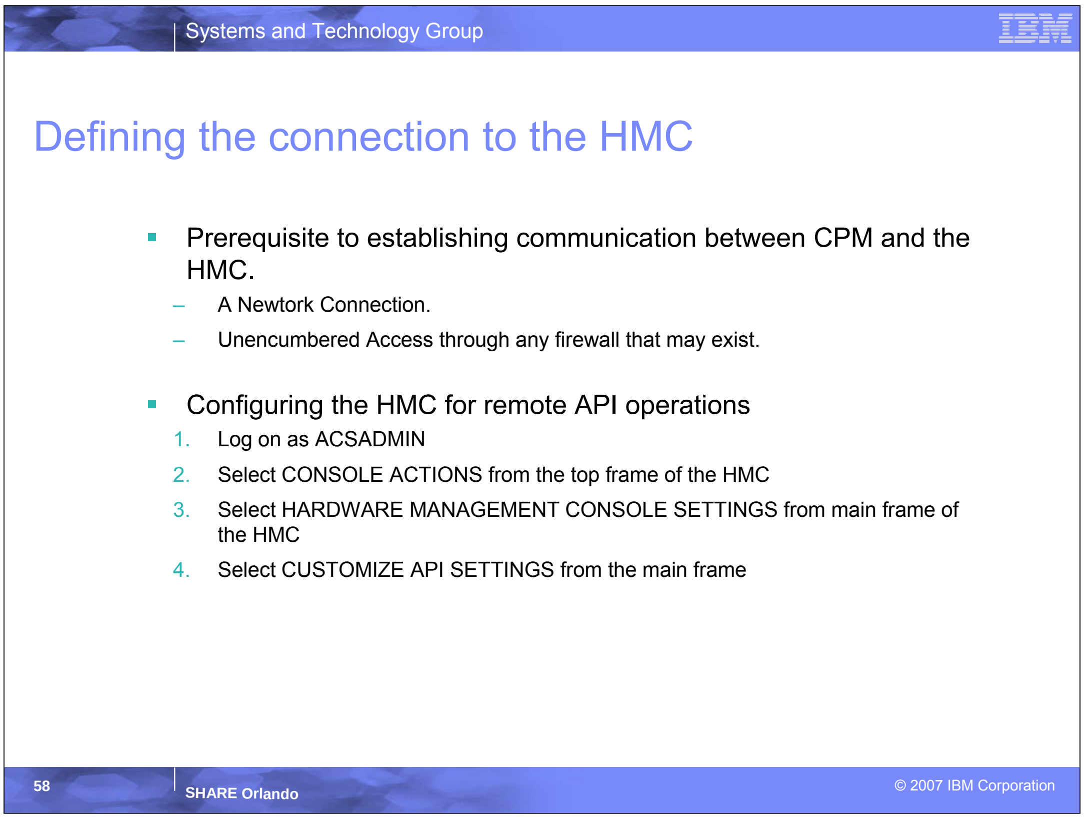
Preparing CPM Connection Cont.

- Verify that the Capacity Provisioning CIM Provider Library is Program controlled
  - Issue ls -E /usr/lpp/cpo/lib/libcpoprovider.so from the shell
  - Should see a “p” in the extended attributes
  - If you need to turn on Program Control issue the following

    `extattr +p /usr/lpp/cpo/lib/libcpoprovider.so`
- Verify that there is a link to the Capacity Provisioning CIM Provider Library in the CIM Server Provider Directory.
  - Issue a `ls -l /usr/lpp/wbem/provider/libcpoprovider.so` from the shell
  - Should see an “l” in the far left column if link exist.
  - Issue `ln -s /usr/lpp/cpo/lib/libcpoprovider.so /usr/lpp/wbem/provider/libcpoprovider.so` if the link does not exist

•The Capacity Provisioning CIM Provider library is should be program controlled upon installation, but if it was manually copied from place to place it may have not carried the attribute. By issuing the above command you will see the extended attributes. The second of these attributes should be a “p” if program control is set.

•Just Like the program control settings, the Symlink from the CIM provider directory to the Capacity Provisioning Provider library can get broken if parts are manually copied from place ot place. Use the commands indicated inthis slide to verify and correct if this happens in your environment.

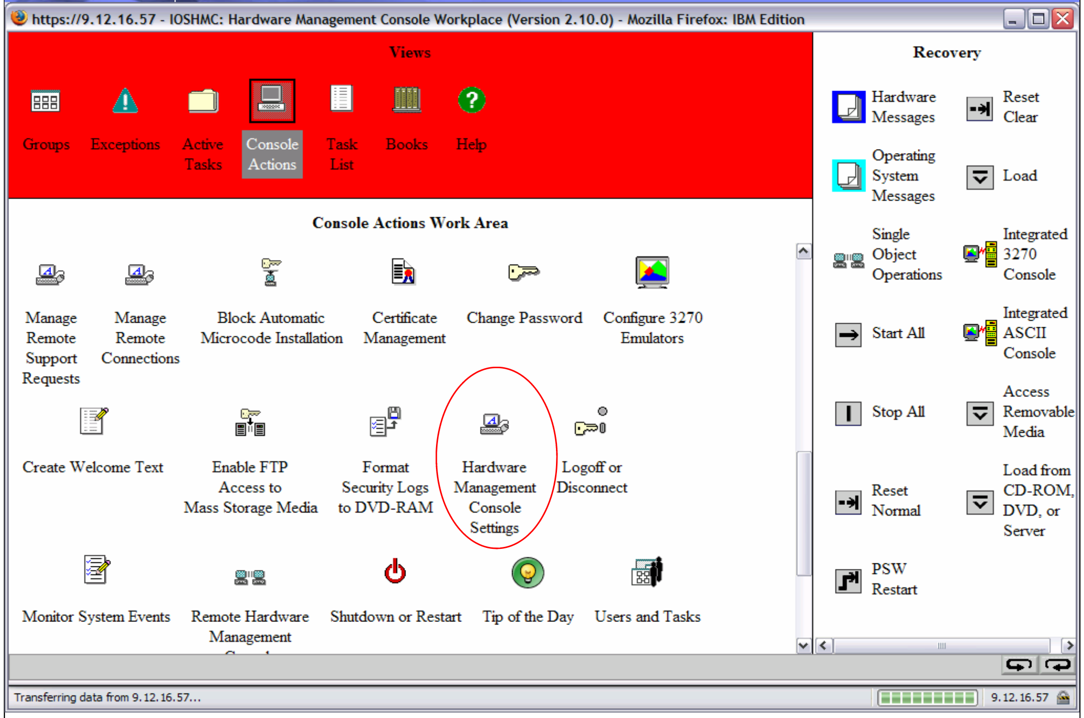


Defining the connection to the HMC

- Prerequisite to establishing communication between CPM and the HMC.
  - A Newtork Connection.
  - Unencumbered Access through any firewall that may exist.
- Configuring the HMC for remote API operations
  1. Log on as ACSADMIN
  2. Select CONSOLE ACTIONS from the top frame of the HMC
  3. Select HARDWARE MANAGEMENT CONSOLE SETTINGS from main frame of the HMC
  4. Select CUSTOMIZE API SETTINGS from the main frame

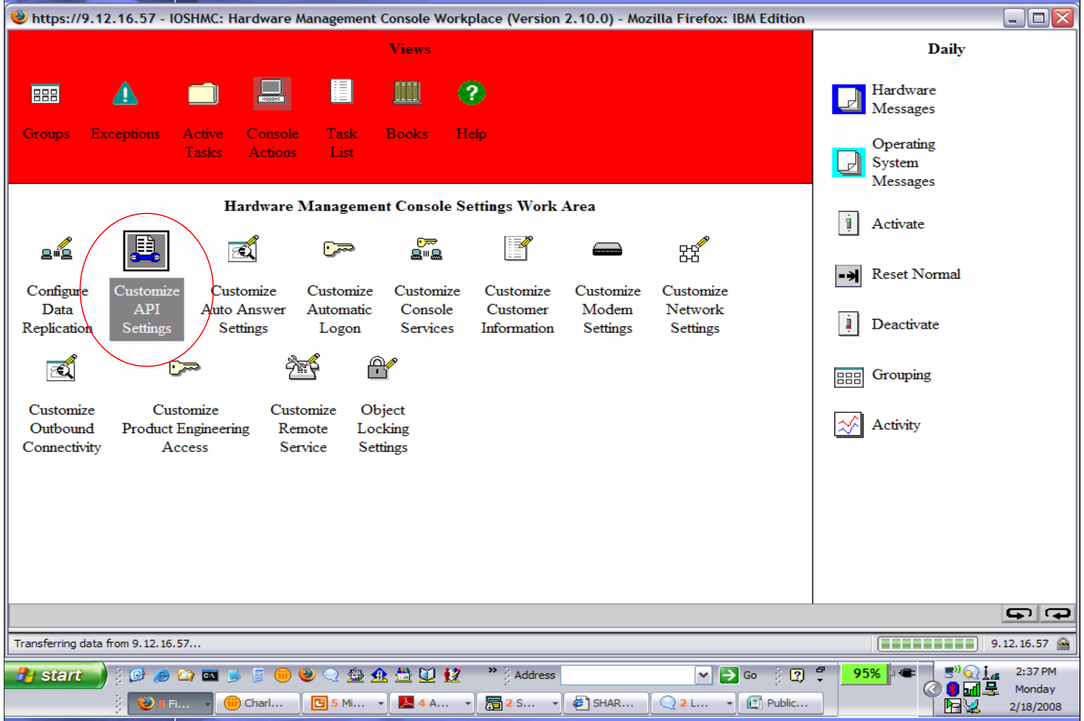
•In order for Capacity Provisioning to effect hardware changes it will need to have a communication link to the hardware. That communication takes place between the runtime system and the HMC or SE using a network connection and SNMP.

•The above steps setup that communication from the hardware perspective.



https://9.12.16.57 - IOSHMC: Hardware Management Console Workplace (Version 2.10.0) - Mozilla Firefox: IBM Edition

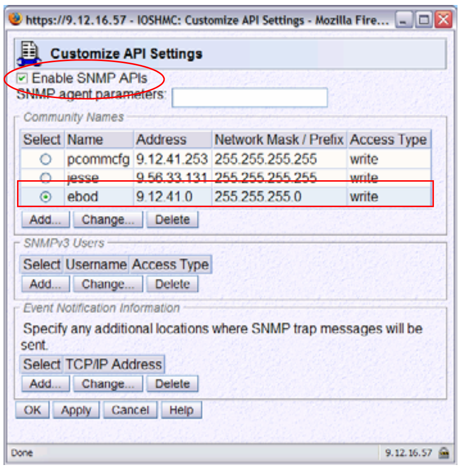
Transferring data from 9.12.16.57...



https://9.12.16.57 - IOSHMC: Hardware Management Console Workplace (Version 2.10.0) - Mozilla Firefox: IBM Edition

Views
Groups
Exceptions
Active Tasks
Console Actions
Task List
Books
Help
Hardware Management Console Settings Work Area
Configure Data Replication
Customize API Settings
Customize Auto Answer Settings
Customize Automatic Logon
Customize Console Services
Customize Customer Information
Customize Modem Settings
Customize Network Settings
Customize Outbound Connectivity
Customize Product Engineering Access
Customize Remote Service
Object Locking Settings
Daily
Hardware Messages
Operating System Messages
Activate
Reset Normal
Deactivate
Grouping
Activity
Transferring data from 9.12.16.57...
9.12.16.57
start
Address
Go
95%
2:37 PM
Monday
2/18/2008

CUSTOMIZE API SETTINGS


https://9.12.16.57 - IOSHMC: Customize API Settings - Mozilla Fire...
Customize API Settings

☑ Enable SNMP APIs

SNMP agent parameters:

Community Names

Select	Name	Address	Network Mask / Prefix	Access Type
○	pcommcfg	9.12.41.253	255.255.255.255	write
○	jesse	9.56.33.131	255.255.255.255	write
●	ebod	9.12.41.0	255.255.255.0	write

Select	Username	Access Type

Specify any additional locations where SNMP trap messages will be sent.

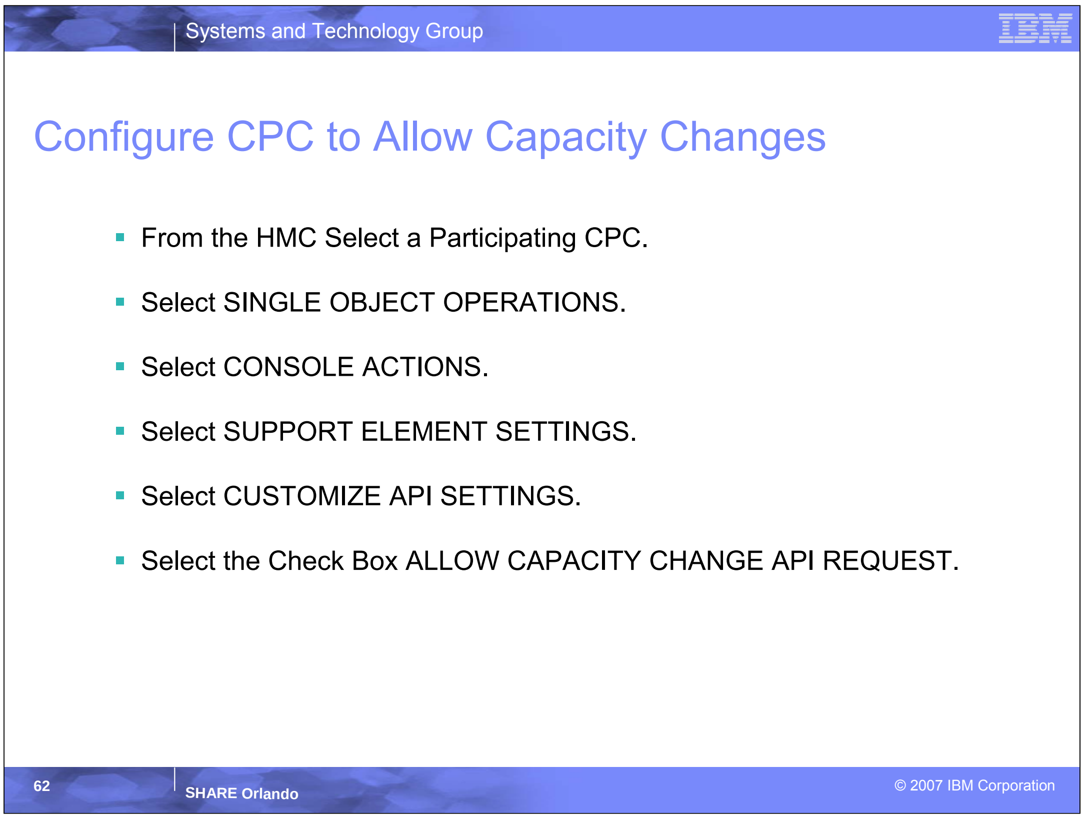
Select	TCP/IP Address

•On this panel you will need to click the Enable SNMP APIs check box in the upper left hand corner.

•You will also need to select ADD under Community Names and add the community name that you defined previously in you CPO.DOMAIN1.PARM(PARM) member.

•In my example case I used a Community Name of ebod and I set it up so that all of my lpars with a 9.12.41.xx could access the hardware and thus act as the runtime system.

•CPM requires WRITE Access when doing this setup.



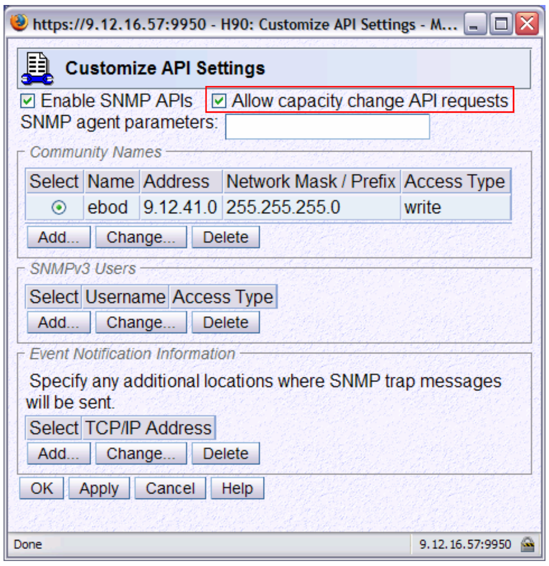
Configure CPC to Allow Capacity Changes

- From the HMC Select a Participating CPC.
- Select SINGLE OBJECT OPERATIONS.
- Select CONSOLE ACTIONS.
- Select SUPPORT ELEMENT SETTINGS.
- Select CUSTOMIZE API SETTINGS.
- Select the Check Box ALLOW CAPACITY CHANGE API REQUEST.

•Once you have established communications between the HMC and the runtime system you will still need to enable the system or systems to allow Capacity Changes.

•Follow the above steps for <u>each</u> CPC in your domain.

•After the Single Object step the next 3 steps are similar to those you did for setting up communications between the HMC and the runtime system.



https://9.12.16.57:9950 - H90: Customize API Settings - M...

Customize API Settings

☑ Enable SNMP APIs ☑ Allow capacity change API requests

SNMP agent parameters:

Community Names

Select	Name	Address	Network Mask / Prefix	Access Type
●	ebod	9.12.41.0	255.255.255.0	write

SNMPv3 Users

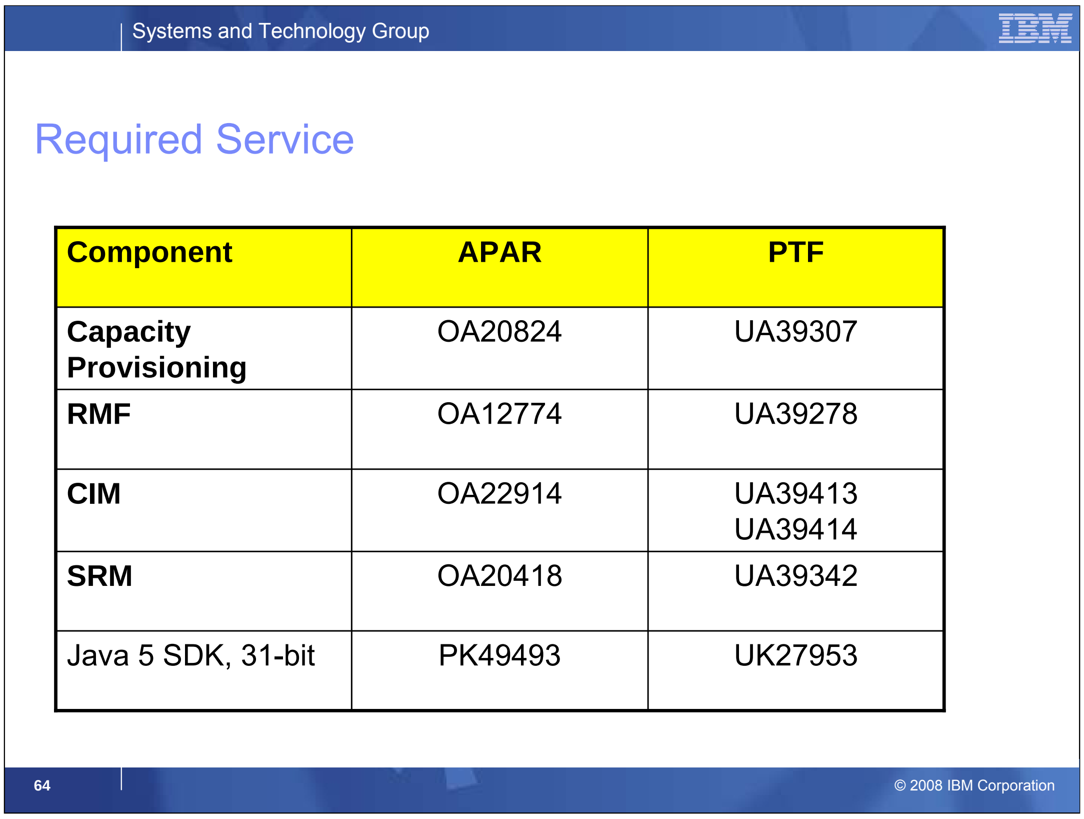
Select	Username	Access Type

Event Notification Information

Specify any additional locations where SNMP trap messages will be sent.

Select	TCP/IP Address

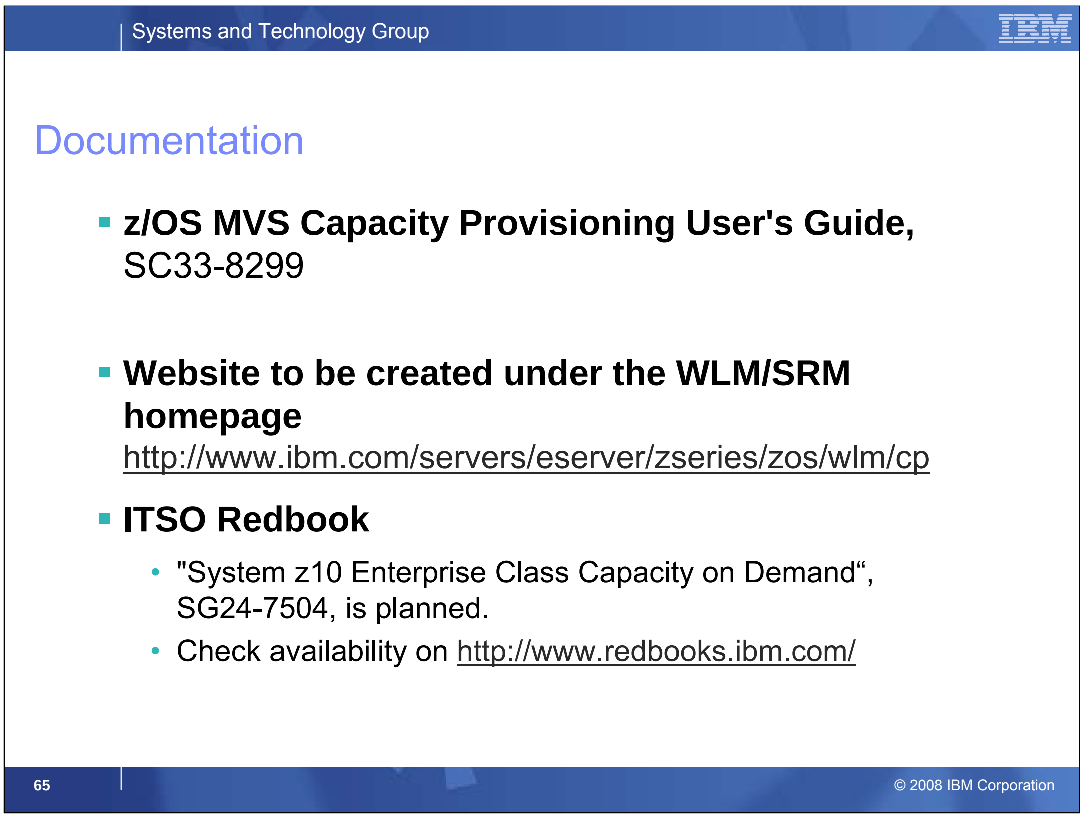
•You will notice that I have also enabled SNMP communication directly to the SE. If you are running Capacity Provisioning in a Domain with a single CPC you can enable communication directly through the CPC.



Required Service

Component	APAR	PTF
Capacity Provisioning	OA20824	UA39307
RMF	OA12774	UA39278
CIM	OA22914	UA39413 UA39414
SRM	OA20418	UA39342
Java 5 SDK, 31-bit	PK49493	UK27953

These PTFs will be required for Capacity Provisioning on IBM z10 EC servers. OA20824 is the APAR that enables the Capacity Provisioning function on z/OS Release 9.



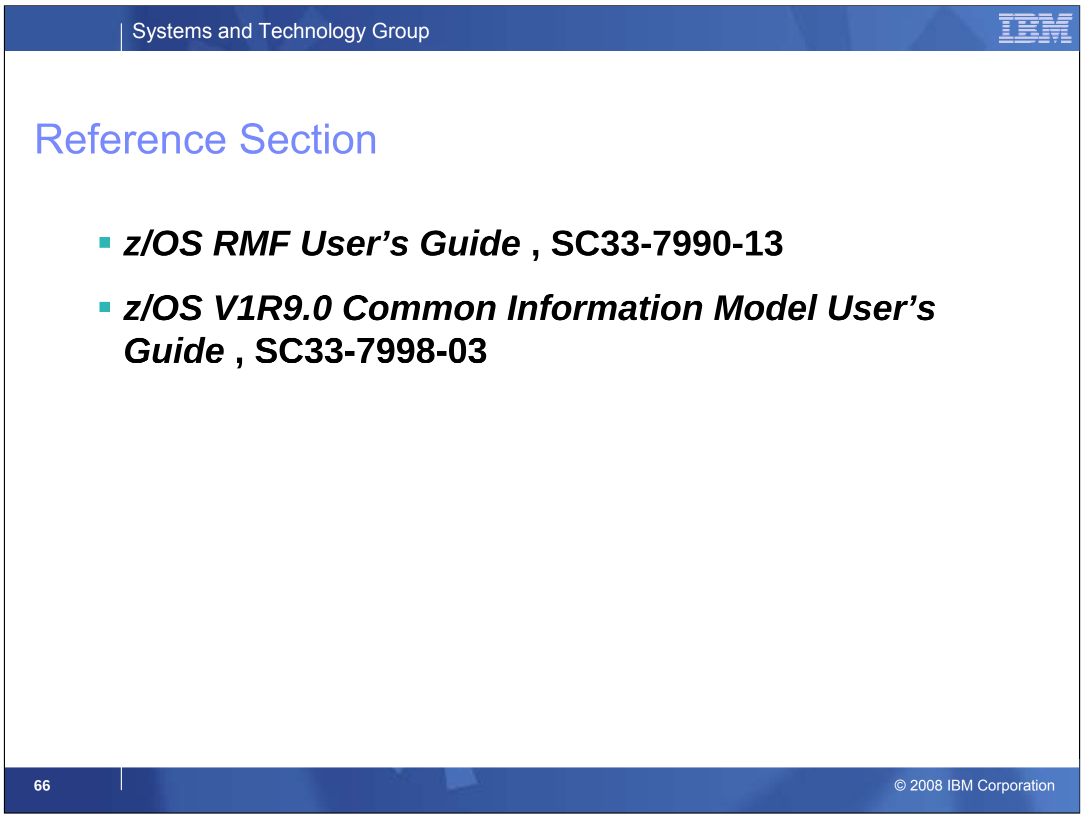
Documentation

- **z/OS MVS Capacity Provisioning User's Guide,** SC33-8299
- **Website to be created under the WLM/SRM homepage** http://www.ibm.com/servers/eserver/zseries/zos/wlm/cp
- **ITSO Redbook**
  - "System z10 Enterprise Class Capacity on Demand“, SG24-7504, is planned.
  - Check availability on http://www.redbooks.ibm.com/

z/OS MVS Capacity Provisioning User's Guide, SC33-8299, is a new publication in the z/OS Release 9 library.

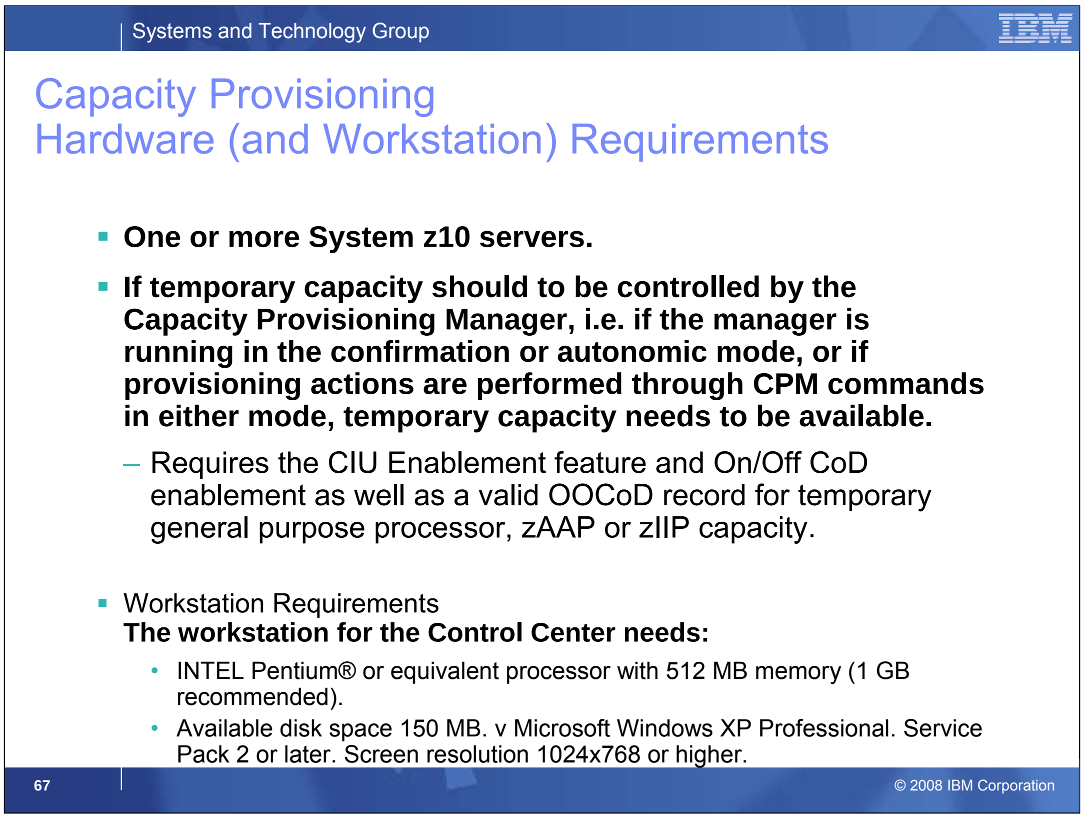
The ITSO is developing a redbook ITSO Redbook "System z10 Enterprise Class Capacity on Demand“, SG24-7504.

More current information will be hosted under the WLM homepage
Website to be created under the WLM/SRM homepage
http://www.ibm.com/servers/eserver/zseries/zos/wlm



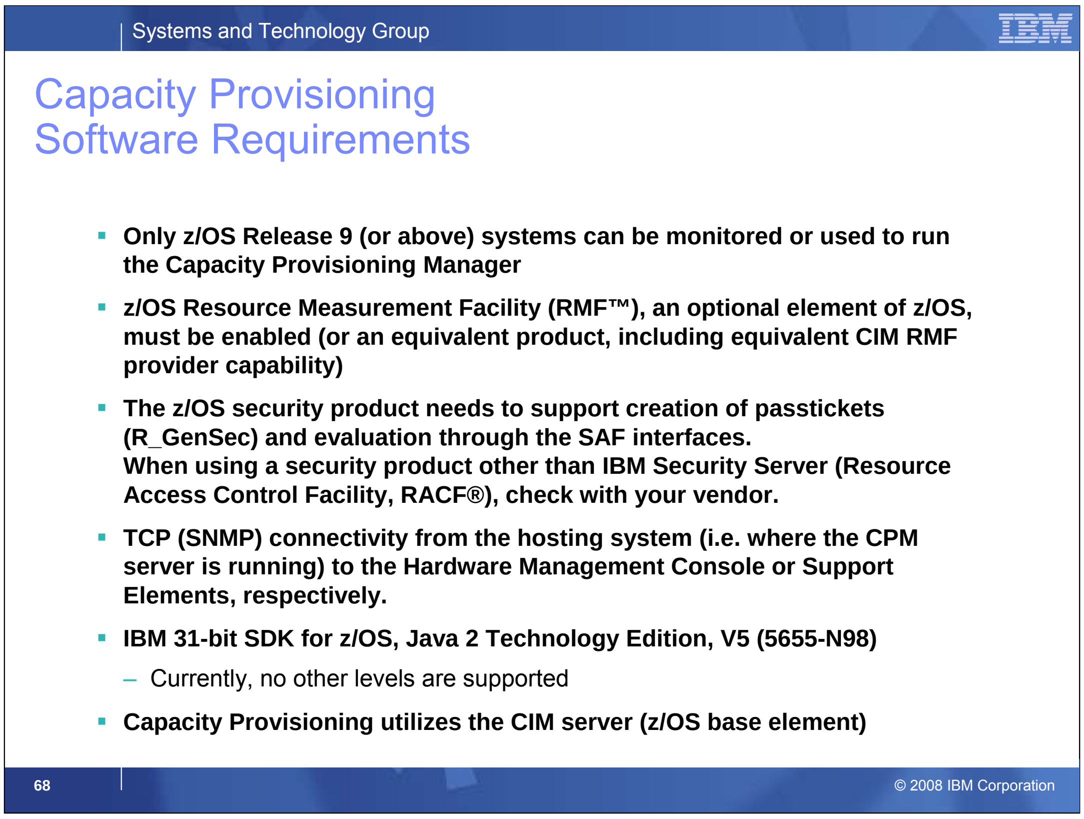
Reference Section

- ***z/OS RMF User's Guide*** **, SC33-7990-13**
- ***z/OS V1R9.0 Common Information Model User's Guide*** **, SC33-7998-03**



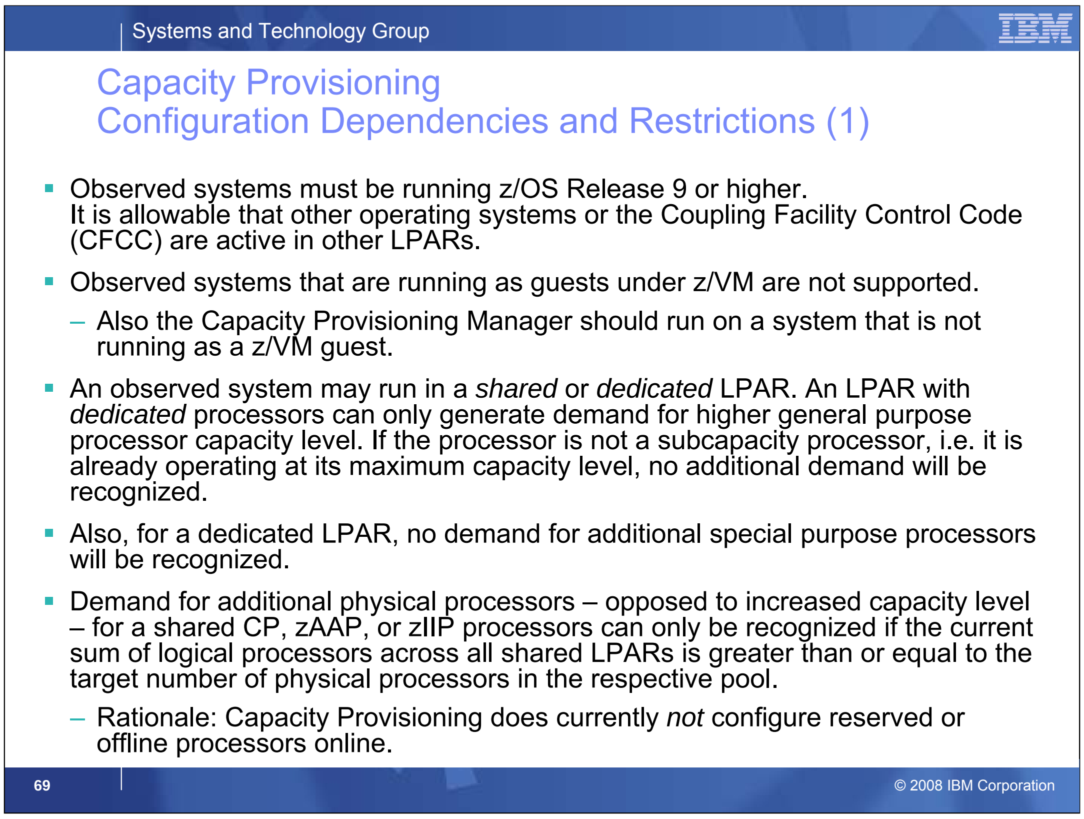
Capacity Provisioning Hardware (and Workstation) Requirements

- **One or more System z10 servers.**
- **If temporary capacity should to be controlled by the Capacity Provisioning Manager, i.e. if the manager is running in the confirmation or autonomic mode, or if provisioning actions are performed through CPM commands in either mode, temporary capacity needs to be available.**
  - Requires the CIU Enablement feature and On/Off CoD enablement as well as a valid OOCoD record for temporary general purpose processor, zAAP or zIIP capacity.
- Workstation Requirements
  The workstation for the Control Center needs:
  - INTEL Pentium® or equivalent processor with 512 MB memory (1 GB recommended).
  - Available disk space 150 MB. v Microsoft Windows XP Professional. Service Pack 2 or later. Screen resolution 1024x768 or higher.



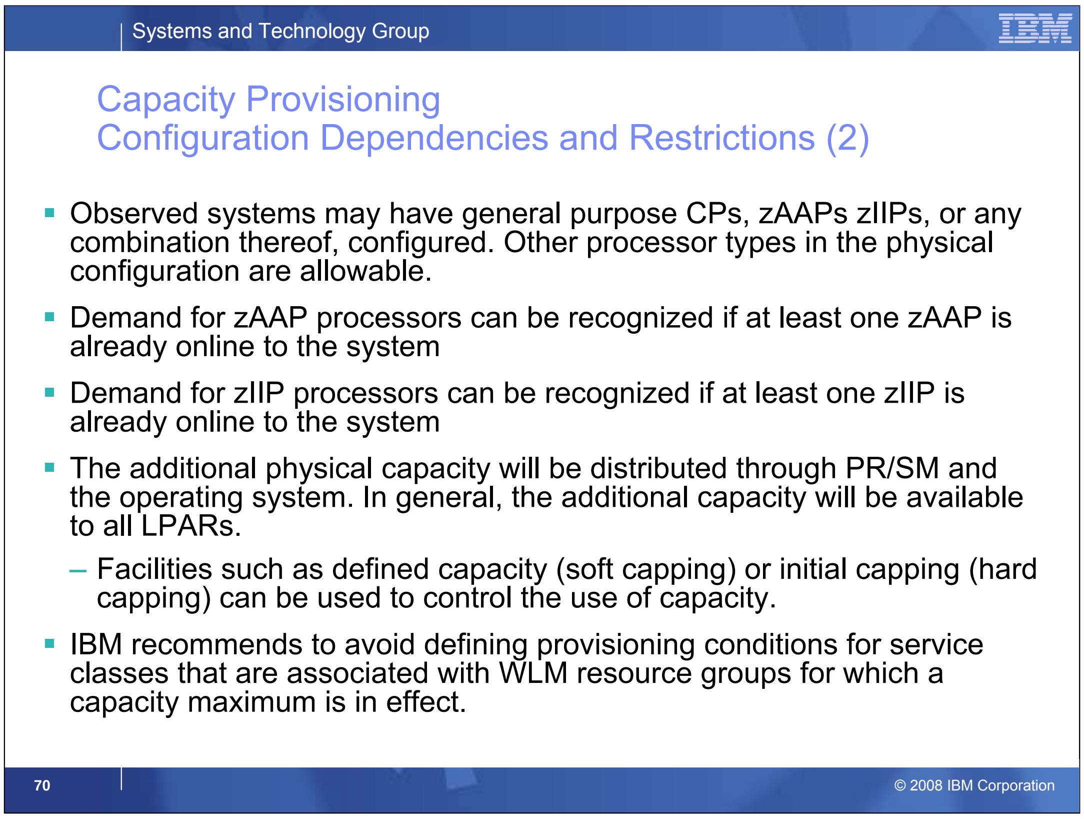
Capacity Provisioning Software Requirements

- **Only z/OS Release 9 (or above) systems can be monitored or used to run the Capacity Provisioning Manager**
- **z/OS Resource Measurement Facility (RMF™), an optional element of z/OS, must be enabled (or an equivalent product, including equivalent CIM RMF provider capability)**
- **The z/OS security product needs to support creation of passtickets (R_GenSec) and evaluation through the SAF interfaces.**
  When using a security product other than IBM Security Server (Resource Access Control Facility, RACF®), check with your vendor.
- **TCP (SNMP) connectivity from the hosting system (i.e. where the CPM server is running) to the Hardware Management Console or Support Elements, respectively.**
- **IBM 31-bit SDK for z/OS, Java 2 Technology Edition, V5 (5655-N98)**
  - Currently, no other levels are supported
- **Capacity Provisioning utilizes the CIM server (z/OS base element)**



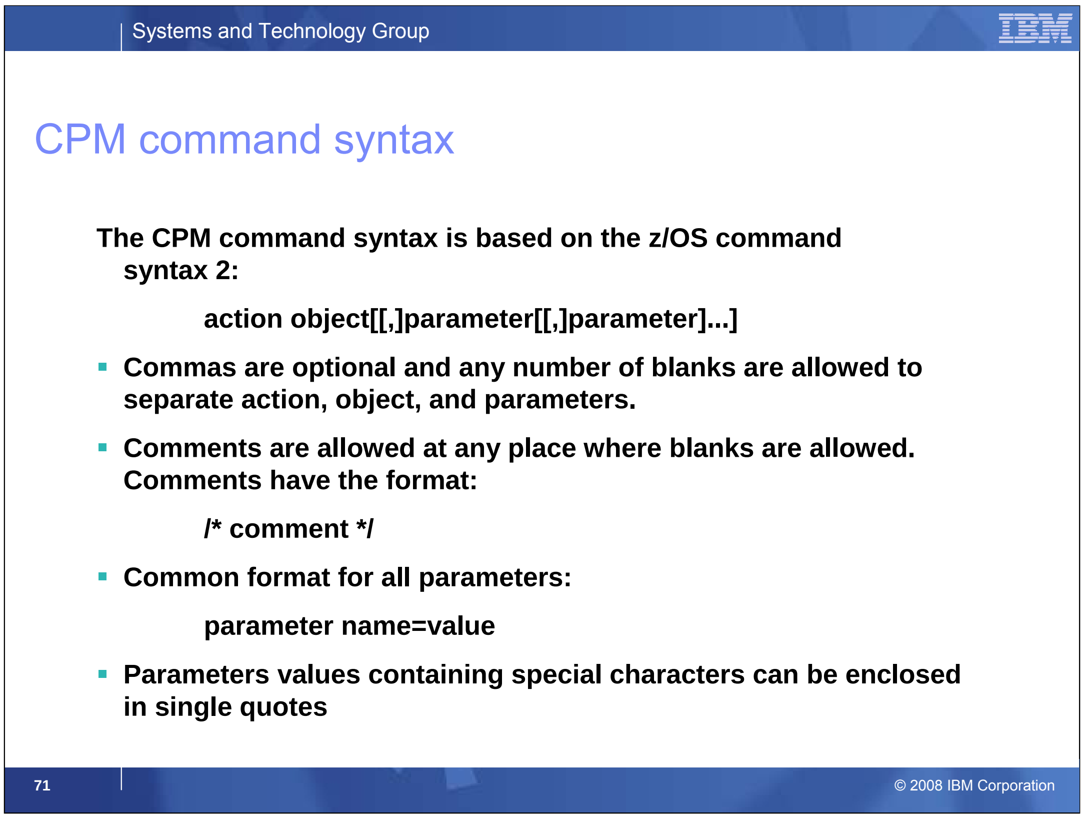
Capacity Provisioning
Configuration Dependencies and Restrictions (1)

- Observed systems must be running z/OS Release 9 or higher.
  It is allowable that other operating systems or the Coupling Facility Control Code (CFCC) are active in other LPARs.
- Observed systems that are running as guests under z/VM are not supported.
  - Also the Capacity Provisioning Manager should run on a system that is not running as a z/VM guest.
- An observed system may run in a *shared* or *dedicated* LPAR. An LPAR with *dedicated* processors can only generate demand for higher general purpose processor capacity level. If the processor is not a subcapacity processor, i.e. it is already operating at its maximum capacity level, no additional demand will be recognized.
- Also, for a dedicated LPAR, no demand for additional special purpose processors will be recognized.
- Demand for additional physical processors – opposed to increased capacity level – for a shared CP, zAAP, or zIIP processors can only be recognized if the current sum of logical processors across all shared LPARs is greater than or equal to the target number of physical processors in the respective pool.
  - Rationale: Capacity Provisioning does currently *not* configure reserved or offline processors online.



Capacity Provisioning
Configuration Dependencies and Restrictions (2)

- Observed systems may have general purpose CPs, zAAPs zIIPs, or any combination thereof, configured. Other processor types in the physical configuration are allowable.
- Demand for zAAP processors can be recognized if at least one zAAP is already online to the system
- Demand for zIIP processors can be recognized if at least one zIIP is already online to the system
- The additional physical capacity will be distributed through PR/SM and the operating system. In general, the additional capacity will be available to all LPARs.
  - Facilities such as defined capacity (soft capping) or initial capping (hard capping) can be used to control the use of capacity.
- IBM recommends to avoid defining provisioning conditions for service classes that are associated with WLM resource groups for which a capacity maximum is in effect.



CPM command syntax

The CPM command syntax is based on the z/OS command syntax 2:

```
action object[[,]parameter[[,]parameter]...]
```

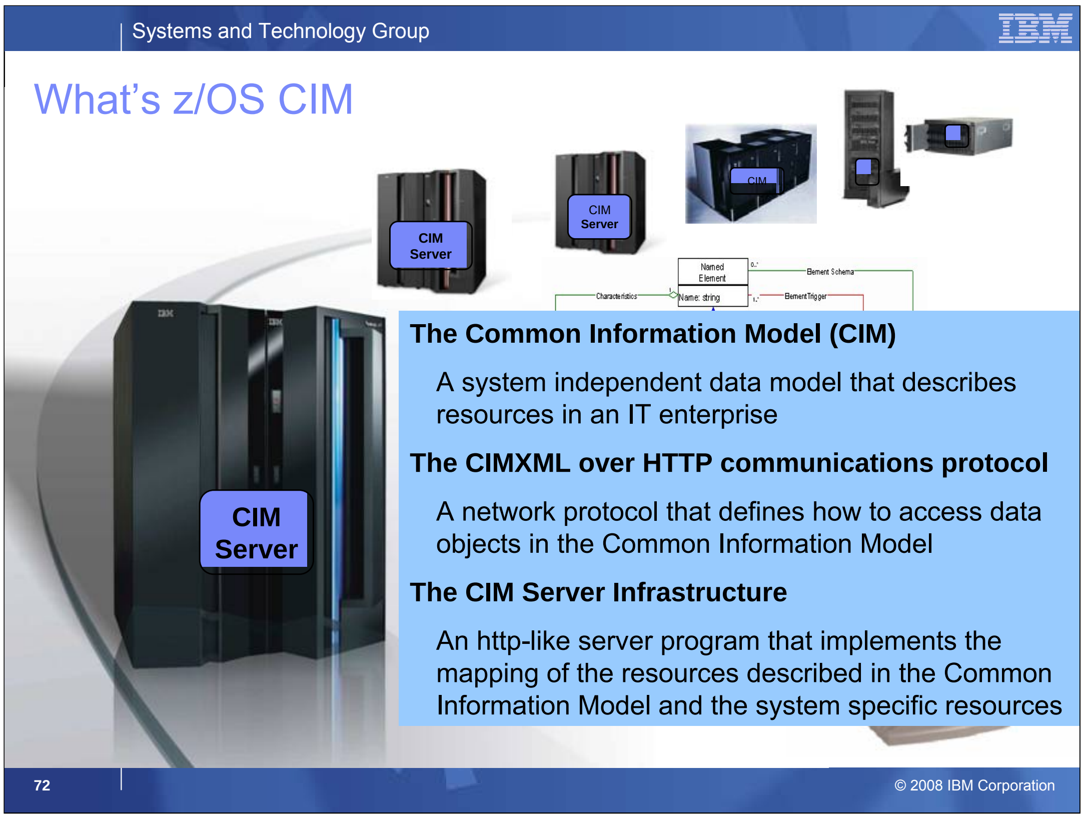
- Commas are optional and any number of blanks are allowed to separate action, object, and parameters.
- Comments are allowed at any place where blanks are allowed. Comments have the format:

```
/* comment */
```

- Common format for all parameters:

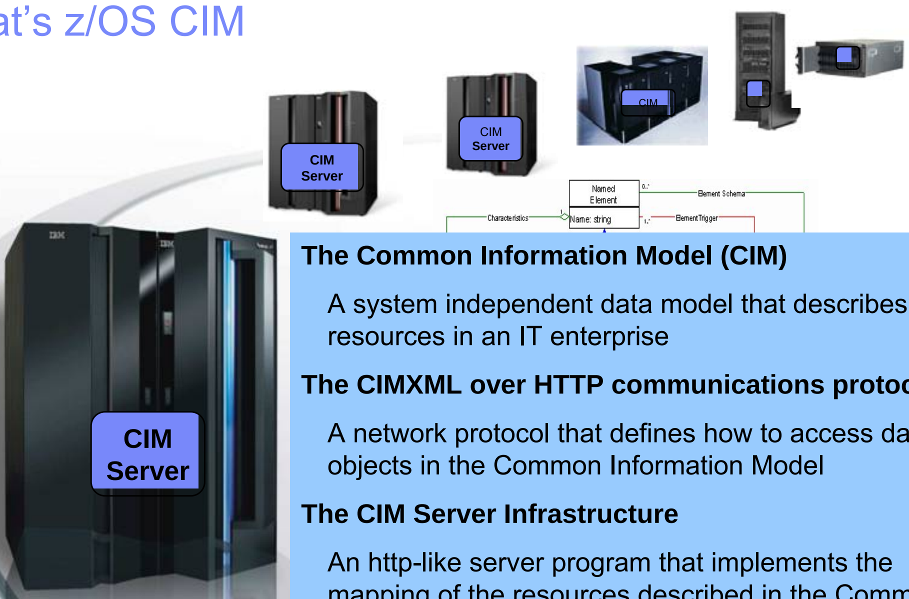
```
parameter name=value
```

- Parameters values containing special characters can be enclosed in single quotes



Systems and Technology Group

What's z/OS CIM



The Common Information Model (CIM)

A system independent data model that describes resources in an IT enterprise

The CIMXML over HTTP communications protocol

A network protocol that defines how to access data objects in the Common Information Model

The CIM Server Infrastructure

An http-like server program that implements the mapping of the resources described in the Common Information Model and the system specific resources

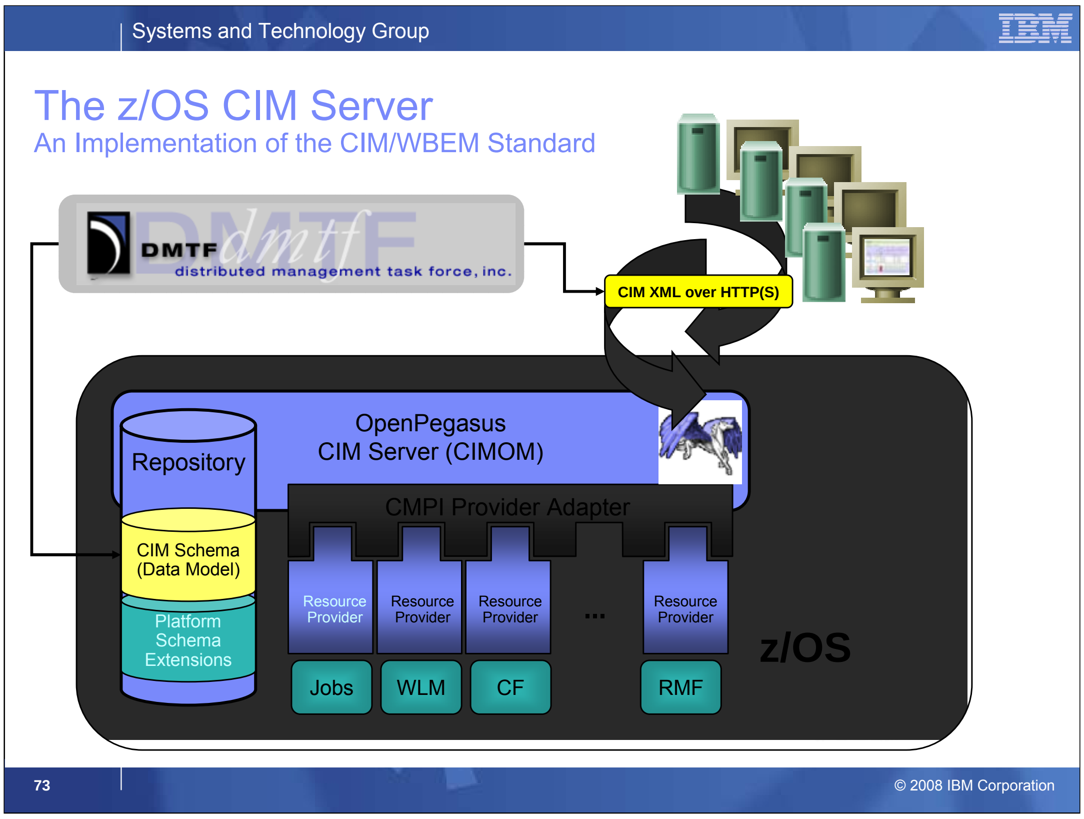
72

© 2008 IBM Corporation

For mastering the challenge to manage complex IT environments, that consist of software and hardware from many different vendors, each using its own set of technologies, a standard way is needed to describe the resources to be managed and to define operations that can be executed on these resources.

By implementing the Common Information Model (CIM), the z/OS Operating System on the mainframe supports an open standards based interface for management over the network. This enables z/OS for a new generation of management applications that provide modern, web based, user interfaces. Management applications that are specialized for the aspects of managing mainframes, as well as management applications that provide inventory and status information for distributed enterprise environments and business applications.

The development of such new management applications is accellerated by eliminating the need to talk multiple different protocols for different systems or even having to develop specific agents that run on the managed systems and facilitate remote access. Instead the management applications talk to the CIM Server that comes as part of z/OS as well as of most other popular operating systems, storage or network devices. For talking to a CIM Server it uses the Web Based Enterprise Management (WBEM) network protocol and the data model defined by the CIM Schema, which is the core of the Common Information Model.


Systems and Technology Group

The z/OS CIM Server

An Implementation of the CIM/WBEM Standard

73

© 2008 IBM Corporation

The OpenPegasus CIM Server, also referred to as CIM Object Manager (CIMOM), is an implementation of the WBEM standards provided by DMTF. It is developed as an OpenSource project that runs on a large number of Operating Systems, including IBM's z/OS.

It comes with an XML based data repository where it stores the CIM Schema classes and all platform specific extensions of the Schema.

While the CIM Server has knowledge only about the CIM Schema Classes and extensions, that task for connecting these classes to actual IT resources is delegated to so called 'Dynamic Resource Providers' which can be registered for the CIM classes known by the CIM Server. When the CIM Server receives a request (CIM Operation) directed against a certain CIM class it looks up the list of applicable providers for that class and all of its subclasses and then forwards the request to each of the providers. A provider is a program library that implements a standard interface through which it plugs into the CIM Server on the one hand, and which implements the access to a resource that maps to a CIM class describing this resource.

The interface used for the providers on z/OS is called the Common Manageability Programming Interface (CMPI) and is also based on a standard publish by TheOpenGroup. It guarantees binary compatibility between the providers and the CIM Server and thus enables providers to be supplied from different vendors, independent of the CIM Server.

The CIM Server at its front end works like an HTTP server. By default it listens on the network ports 5988 (for plain HTTP) and 5989 (for secure HTTP) to receive requests from management applications over the CIM XML over HTTP protocol.

Outline of *z/OS MVS Capacity Provisioning User's Guide* (SC33-8299)