Suggested Test Approach for z/OS Capacity Provisioning

IBMCPM@de.ibm.com

Table of Contents

1 Motivation

After the initial configuration of Capacity Provisioning and related elements, such as z/OS Resource Measurement Facility (RMF) as the monitoring component, and CIM, it will usually be required to validate that all components were configured correctly. It will also be required to validate that the Capacity Provisioning policy matches the objectives.

Initial validation tests will frequently be performed in test environments under conditions that are not typical for the target production environments. For example, test systems may be not physically constrained, or the tests are performed in an LPAR that is defined with an extremely low weight. Also, testing will usually be performed within a small time window such that acceleration may be desired.

Through a combination of various processing parameters that are specified in the <HLQ>.<Domain>.PARM(PARM) member, it is possible to tweak the Capacity Provisioning Manager (CPM) to meet the test objectives.

If problems show up that cannot be diagnosed immediately, it is important to gather diagnostic data that allow the IBM service team to investigate the potential problem. Therefore, the following suggestions also contain recommended trace and log settings.

Should you encounter any problems implementing your Capacity Provisioning Manager setup or if you have further questions regarding an appropriate test setup, please contact IBM service or get in touch with the Capacity Provisioning development team sending an email to CPM@de.ibm.com.

2 Basic Functional Testing

These tests verify that the underlying componentry is correctly configured and allows for a successful end-to-end-operation. Basically they constitute an installation verification program (IVP) which would be hard to provide because multiple objects (hosts, workstation) are involved.

2.1 Validate CIM and Capacity Provider Setup

Objective: On the run-time system, i.e. the system(s) on which the CPM runs, verify that the CIM setup allows for communication between the CPM on the host with and either the CP Control Center (CPCC) or the z/OSMF Capacity Provisioning task. This path utilizes the CP CIM provider.

Processing Mode: MANUAL

Processing Parameters:

Trace options:

Test procedure: Choose one of the following procedures

(A) Using the CPCC:

Prepare a domain configuration and optionally a policy. Then

- Connect to the CPM of domain *dom*
- Install the domain configuration
- Install the policy

From the z/OS console activate domain configuration and policy:

```
'F CPOSERV,APPL=S D CFG=cfg'
'F CPOSERV,APPL=S D POL=pol'
```

(B) Using the z/OSMF CP task:

Prepare a domain configuration and optionally a policy. Then

- Install and activate the domain configuration to the CPM of domain *dom*

Install and activate the policy to the CPM of domain *dom*

Key results: For (A) on operator console:

```
CPO2015I Provisioning Manager successfully initialized.
CPO1041I Domain configuration cfg successfully activated
CPO1020I Policy successfully changed to pol
```

For (B) on z/OSMF:

IZUCP0725I Domain Configuration "*cfg*" was installed on the Provisioning Manager for domain "*dom*"

IZUCP0731I Domain Configuration "*cfg*" was activated on the Provisioning Manager for domain "*dom*"

IZUCP0797I Policy "*pol*" was installed on the Provisioning Manager for

domain “*dom*”
IZUCP0753I Policy “yyy” was activated on the Provisioning Manager for domain “*dom*”

2.2 Validate Processor Communication

Objective: Validate that the CPM can communicate with the Support Element or Hardware Management Console through z/OS BCPii.

Processing Mode: MANUAL

Processing Parameters:

Trace options:

```
'F CPOSERV,APPL=RESET TRACE'
'F CPOSERV,APPL=SET TRACE LEV=ALL'
```

For BCPii set CTRACE tracing in CTIHWI00 (located in SYS1.PARMLIB) to ALL.

Test procedure: Start CPM and set traces as listed above. Then restart the CPM.
Set the domain configuration via `'F CPOSERV,APPL=S D CFG=xxx'`.
Issue REPORT CONFIGURATION command `'F CPOSERV,APPL=R C'`.
Optionally, perform a CPC model change via the ACTIVATE RESOURCE command `'F CPOSERV,APPL=A R CPC=`*cpc*` MODEL=`*num*`'` or Defined Capacity change with ACTIVATE DEFINED CAPACITY `'F CPOSERV,APPL=A DC CPC=`*cpc*` LPAR=`*lpar*` MSU=`*num*`'`

Key results:

```
CPO2015I Provisioning Manager successfully initialized.
CPO1041I Domain configuration ... successfully activated
CPC ... with record * is enabled (default enabled)
 CPC is matched with serial 000510070B82 since 12/04/2009
13:56:59
 Hardware is of type 2097 with model E26
 Current model is 709 with 804 MSU, 4 zAAPs, and 4 zIIPs
 Active record ID is CR7LYKLY
 Hardware has 8 spare processors
 Activation limits are 4 zAAPs, and 4 zIIPs
 Active resources GP/zAAP/zIIP 0(0/0)/0/0
```

and no other report text referring to problems regarding authorization, expiration etc.

If a model upgrade was performed:

```
CPO1023I Temporary upgrade for CPC cpc to model num
successfully initiated
```

followed by

```
CPO3030I Command completed successfully for CPC cpc
```

If a defined capacity model upgrade was performed:

```
CPO3027I Defined capacity information for CPC ... is
available
...
```

```
CPO1289I Defined capacity limit for LPAR SYS4 on CPC EC12
increased to 55 MSU
```

Traces are written to trace directory

2.3 Validate CIM and Monitoring Provider Setup

Objective: Verify that all observed systems, i.e. the systems to be monitored by CPM, can be successfully monitored.

Processing Mode: ANALYSIS

Processing Parameters:

Trace options:

```
'F CPOSERV,APPL=RESET TRACE'
'F CPOSERV,APPL=SET TRACE LEV=ALL'
'F CPOSERV,APPL=ACTIVATE LOG'
```

Test procedure: Install and activate a policy that temporarily observes workloads on all defined systems. Ideally there should be active work in the defined service class periods. The provisioning conditions are irrelevant for this test.
Wait some minutes after activation of policy and issue commands:

```
'F CPOSERV,APPL=REPORT POLICY'
'F CPOSERV,APPL=REPORT CONFIGURATION'
'F CPOSERV,APPL=REPORT WORKLOAD TYPE=DETAILED'
```

Key results:

```
CPO3806I The system at address ... is ... in
sysplex ...
CPO3813I The system at address ... is running
on CPC ...
CPO3880I System ... in sysplex ... is now
monitored
```

Configuration report:

```
System ... in sysplex ... is enabled (default
enabled)
 Primary host address: ...
 Protocol: HTTP, port: 5988
 The system at primary host address is observed
  This system is available since 12/06/2009
12:18:43
  This system is running on the CPC ...
  WLM service definition: ..., active policy: ...
```

Workload report:

```
CPO1047I Workload report generated at 1...
Workload is analyzed for 3 system(s)
Workload for system ... of sysplex ... on CPC ...
 CPUMED.1 PL/PD/DL/DD/S 1.6 5 1.2 7 System
  PI from 12/06/2009 12:37 is 0.6
   Last limit crossing was 12/06/2009 12:37
```

Traces are written to trace directory.

Logs are written to log directory.

2.4 Validate CPM Restart Automation

Objective: Verify that the CPM will be restarted by ARM or other automation, when terminated unexpectedly.

Processing Mode:

Processing Parameters: In the PARM member specify:

```
# ARM definitions
ARM.Register=Yes
```

or have automation monitor the CPM AS

Trace options:

Test procedure: Start the CPM. After initialization, issue the `'F CPOSERV,APPL=STOP MANAGER MODE=FORCE'` command.

Key results:

```
CPO2020I Register with ARM using element type
SYSCPM and element name SYSCPO was successful
...
S CPOSERV,PMODE=*,POLICY=*
IXC812I JOBNAME CPOSERV, ELEMENT SYSCPO FAILED.
742
THE ELEMENT WAS RESTARTED WITH OVERRIDE START
TEXT.
IXC813I JOBNAME CPOSERV, ELEMENT SYSCPO 743
WAS RESTARTED WITH THE FOLLOWING START TEXT:
S CPOSERV,PMODE=*,POLICY=*
THE RESTART METHOD USED WAS DETERMINED BY THE
ACTIVE POLICY.
```

2.5 Validate WLM Setup for CPM

Objective: Verify that the CPM, the CIM server(s), the monitoring and networking infrastructure are set up to allow CPM monitoring even when capacity is constrained. In other words, all monitoring components need to run at an importance above the work to be monitored.

Processing Mode: ANALYSIS

Processing Parameters:

Trace options:

```
'F CPOSERV,APPL=RESET TRACE'
'F CPOSERV,APPL=SET TRACE LEV=ALL'
'F CPOSERV,APPL=ACTIVATE LOG'
```

Test procedure:	Start thrasher jobs on all monitored (test) systems, running at an importance level of typical application work. Increase the work until MVS busy and CPC (usually shared CP pool) utilization is close to 100%. Rerun the test #1.1.3.Validate CIM and Monitoring Provider Setup
Key results:	CPM runs normally and responds to commands. Workload report reports correctly about all observed systems. Only occasional messages (if any) `CPO3820W Long metrics retrieval interval for system at ....`

3 Testing the CPM policy and monitoring functions

Once the mechanics of systems monitoring and CPC communication are verified, further tests can be conducted to verify the CPM policies. In many cases such tests will need to be performed in a test environment within a given test window. This raises two problems:

1. In many cases the test environment will not be “really constrained”, but the constraint would be artificially introduced. For example, on a test system which might be constrained by a limited number of online logical processors, MVS busy could get to very high values, however, the CPC would **not** be busy. Normally, CPM would detect the fact that there is no “real” constraint and simply not perform or suggest a provisioning action.

2. In order to perform as many tests as possible, you may want to accelerate the test and minimize any delays.

3.1 Adapting to testing environment

In order to satisfy such testing requirements, as well as to allow tuning to a particular installation and workload, CPM supports several parameters to be specified in the <HLQ>.<Domain>.PARM(PARM) member. The specified values are processed at CPM initialization time, affect each managed system and server within the domain and remain effective until the CPM is shut down.

Table “Additional control parameters” in chapter “Setting up a Capacity Provisioning domain” of the Capacity Provisioning User's Guide summarizes some of the specifiable values. In a test environment consider changing the following parameters:

All these variables and the values are case-sensitive and must be entered exactly as shown. For releases prior to V2R1, no error messages are issued for incorrect variable names.

Planner.BlockingTime:
A minimum waiting period of 5 minutes between two temporary (physical) capacity changes is usually sufficient.

Planner.ProvisioningRejectTime/
Planner.DeprovisioningRejectTime:
Only applicable to CONFIRMATION mode. Specify a value of a few minutes to have CPM come back faster again with a new recommendation after the operator has rejected the last temporary (physical) capacity change proposal.

Planner.MinimumActivationTime:
Specify how many minutes CPM should leave provisioned temporary (physical) capacity active at least before the deprovisioning condition could deactivate it. 10 or 15 minutes might be a good value for accelerated tests.

Analyzer.RequestedShareLimit:
Minimum percentage of an LPARs weight in relation to the total weight of all LPARs to allow for a temporary (physical) provisioning action. If an LPAR being monitored is defined with a very low share (<5%), specify a value that is lower or equal to that share, e.g. for 1%

```
Analyzer.RequestedShareLimit = 1
```

A situation where a low LPAR weight prevents provisioning is indicated in the detailed workload report by the message

```
System CP LPAR-weight too low
```

Analyzer.Threshold.Decision001TotalSharedPhysicalUtilCp:
Setting this threshold to a lower value than the default 95 (equivalent to 95%) allows CPM adding temporary (physical) capacity if the shared CP pool of the CPC is not yet busy.
A situation where a low CPC busy value prevents provisioning is indicated in the detailed workload report by the message

```
CPC-wide CP-utilization too low
```

For zAAP and zIIP processors the respective parameters are

`Analyzer.Threshold.Decision001TotalSharedPhysicalUtilZaap` and
`Analyzer.Threshold.Decision001TotalSharedPhysicalUtilZiip`

For CPM on V2R1 you can also use the parameter synonyms

```
Analyzer.Threshold.TotalSharedPhysicalUtilCp
Analyzer.Threshold.TotalSharedPhysicalUtilZaap
Analyzer.Threshold.TotalSharedPhysicalUtilZiip
```

Analyzer.Threshold.Decision002MvsUtilCp:
The MVS busy value for the processor type that must be exceeded to allow for provisioning of temporary (physical) or defined or group capacity. Setting this threshold to a lower value than the default 95 allows CPM adding capacity if from an MVS perspective the processors are not busy. This number could we somewhat reduced (suggestion: 80). Reducing this value much further may not help because other checks, such as for CPU delays considered for physical (temporary) provisioning actions, are likely to fail such that no provisioning action would be performed.
A situation where a low MVS busy value prevents provisioning of physical (temporary) capacity is indicated in the detailed workload report by the messages
`System CP-utilization too low`
or
`System utilization of all processor types too low`

For zAAP and zIIP processors the respective parameters are
`Analyzer.Threshold.Decision002MvsUtilZaap` and
`Analyzer.Threshold.Decision002MvsUtilZiip`

For CPM on V2R1 you can also use the parameter synonyms
`Analyzer.Threshold.MvsUtilCp`
`Analyzer.Threshold.MvsUtilZaap`
`Analyzer.Threshold.MvsUtilZiip`

Analyzer.Threshold.Decision032LparSharedLogicalUtilCp:
The LPAR busy value for the processor type that must be exceeded to allow for recommendation of setting logical processors online. Setting this threshold to a lower value than the default 95, leads CPM to advise to set logical processors online even if the processors are not busy from LPAR perspective. In a first approach the value could be reduced to 80. Reducing this value much further may delay the provisioning of temporary (physical) capacity which could be postponed until logical demands are met.
A situation where a low LPAR busy value might prevent a recommendation for setting logical processors online is indicated in the workload report by the message
`Demand for additional logical CPs not recognized`

For zAAP and zIIP processors the respective parameters are
`Analyzer.Threshold.Decision032LparSharedLogicalUtilZaap` and
`Analyzer.Threshold.Decision032LparSharedLogicalUtilZiip`

For CPM on V2R1 you can also use the parameter synonyms
`Analyzer.Threshold.LparSharedLogicalUtilCp`
`Analyzer.Threshold.LparSharedLogicalUtilZaap`
`Analyzer.Threshold.LparSharedLogicalUtilZiip`

DefinedCapacity.LeadTime:
The default value of 5 allows for Defined Capacity or Group Capacity increases if a workload is suffering according to your workload criteria and the projected *TimeUntilCapping* value calculated by the Performance Monitor for the system/LPAR falls below 5 minutes. Change this to a higher value if you want CPM to recognize the need for a Defined Capacity or Group Capacity increase earlier.

DefinedCapacity.ProvisioningRejectTime/
DefinedCapacity.DeProvisioningRejectTime:
Only applicable to CONFIRMATION mode. Specify a value of a few minutes to have CPM come back faster again with a new recommendation after the operator has rejected the last defined capacity or group capacity change proposal.

Further parameters are available. It is usually advisable to contact IBM service or the Capacity Provisioning development team at CPM@de.ibm.com if you encounter any problems adapting CPM to a particular configuration.

3.2 Migrating to production

Once the tested CPM setup is being migrated to production mode, you should review which settings were modified for testing and determine whether they need to be reset to other values or be removed to use the defaults. It is usually recommended to not explicitly specify default values.

In addition, the log and trace settings should be re-evaluated:

- Consider leaving logs enabled. At least the error log should always be enabled.

- Traces *may* be left enabled. However, it is recommended to disable the very verbose CIMCLIENT trace:
  `'F CPOSERV,APPL=SET TRACE LEV=OFF COMP=CIMCLIENT'`

- The performance characteristics of the CPM may also be optimized The actual resource consumption (CPU, memory) depends on a number of variables, such as number of observed systems, size of the WLM service definition(s), number of monitored service class periods, monitoring cycle (e.g. DDS MINTIME).

It is important to specify a large enough Java heap size. The heap size is specified as an environment variable in the <hlq>.<domain>.PARM(ENV) member. The default heap size was increased to 384M via APAR OA31072 and requires a region size of 512M and should be adequate when monitoring a few (<6, likely <10) systems. When monitoring a two digit number of systems the heap size should be bumped to 768M, along with a region size of 1024M.