IBM Spectrum Scale

on

Power Linux

Tuning Guide

Version Number: 7.2

Date: 04/23/2018

Authors:
Sven Oehme
Todd Tosseth
Daniel De Souza Casali

Table of Contents

1 Introduction

IBM Spectrum Scale is a unified file and object software-defined storage for high performance, large-scale workloads on-premises or in the cloud. Built upon IBM's award winning General Parallel Filesystem (GPFS), an ultra-scalable distributed filesystem used on many of the top 500 High Performance Computers in the World. Spectrum Scale includes the specialized protocols, services and performance required by Technical Computing, Big Data, HDFS and business critical applications.

IBM Spectrum Scale user interface and integrated information lifecycle tools simplify the management of petabytes of data and billions of files, so you can reduce the cost of storing ever-increasing amounts of data.

Benefits of IBM Spectrum Scale

- **Unified storage:** Support for diversified hardware and application portfolios, Spectrum Scale increases storage utilization and practically eliminates data silos for "files objects and HDFS".

- **Seamless scaling:** Data growth is manageable and cost effective when independently and seamlessly adding capacity, performance, additional file access protocols or processing capacity to your Spectrum Scale cluster.

- **Global collaboration:** Data anywhere access with a global namespace that spans storage types and geographic locations. Active File Management (AFM), coupled with advanced routing and caching accelerates applications across the data center or across the world.

- **Data aware intelligence:** Match data value to the right storage tier with Spectrum Scale's policy engine and software-defined storage. Eliminate filer hotspots with flash acceleration or transparently compress or migrate cold data to archival storage.

IBM Power Systems

IBM Power Systems are deployed in many of the largest clusters in the world. Configurable into highly scalable Linux clusters, Power Systems offer extreme performance for demanding workloads such as genomics, finance, computational chemistry, oil and gas exploration, and high performance data analytics.

Power Systems Scale-out servers are affordable, easy-to-deploy and energy efficient. Available with up to 24 cores, these 1 and 2 socket servers offer better economics and security for businesses that need smaller or scale-out deployment options for data-centric application.

Built with the first processor designed for big data workloads, the design of Power Systems combines the computing power, memory bandwidth and I/O in ways that are easier to consume and manage, building on strong resiliency, availability and security.

You can see that these features make IBM Power Systems a great match to Spectrum Scale workloads, this paper is intended to help you get the best performance out of this server.

2 Getting Your System Prepared

To get the best results make sure your system is correctly configured and has the correct firmware for the server, adapters and operating system version.

Operating system considerations

This paper focuses on the Red Hat Linux operating system version. Ensure you are running a supported kernel version by checking the Spectrum Scale FAQ. Register for Red Hat and use yum update service, or get the latest security packages from rhn.redhat.com.

Packages required

Make sure you have the executables and configuration files needed for the correct function of the system and to perform system tuning. The following packages need to be installed on all partitions that run Spectrum Scale.

- tuned-utils.noarch
- tuned.noarch
- numactl

Firmware considerations

For POWER8 systems, the minimum recommended server firmware is FW840_108 [FIX: Check FAQ for minimum, maybe pointer to FAQ question that has latest information]. Upgrade all the adapter firmware and drivers to the latest available levels prior to upgrading Spectrum Scale.

If you need help upgrading your firmware to a correct version, please follow the step-by-step guide to IBM Power Systems firmware update available at IBM Developer Works:

http://www.ibm.com/developerworks/aix/tutorials/au-power-systems-firmware-upgrade/

3 IBM Power Systems Tuning for Spectrum Scale

IBM Power Systems are a great option to drive performance for Spectrum Scale due to its high memory and bus bandwidth. This chapter describes how to tune all the components from the Logical Partition (LPAR) level up to Spectrum Scale in order to achieve the best performance.

LPAR hardware allocations for NUMA based POWER Servers

This paper focus is on POWER Servers Running PowerVM as the Hypervisor and being managed by a HMC. If you need more details on how IBM Power Systems work, please, consult IBM POWER documentation at:

http://www.ibm.com/support/knowledgecenter/POWER8/p8hdx/POWER8welcome.htm

If an IBM POWER server is running with more than one LPAR, then you need to validate the way memory and CPU are provisioned to each LPAR. It is a good idea to ensure the processes on the operating system are dispatched using the same NUMA node where the memory is located. This takes advantage of memory locality, and ensures the workload is equally distributed across NUMA nodes.

The goal is to use Dedicated Processors in the partition profile, which should align to better processor locality and being dispatched from the same processor module.

Memory should be divided equally among the partitions, leaving 5-10% for the server hypervisor. If the allocated memory for a given LPAR plus the amount required by the hypervisor is too much, it will need to allocate from multiple processor modules and performance will be impacted. You can check after bootup of the partition with the command 'numactl –H' to determine if you have all memory from only one partition assigned.

If your environment includes an HMC you can take advantage of the Dynamic Platform Optimizer (DPO), which can be run from the HMC command line. This can improve processor and memory affinity. Since this feature is dynamic it can be run while the partitions are active. If this is a new installation and you are configuring the system to run for the first time run DPO while the partitions are shut down so allocation can be done quickly without the need to migrate active pointers and data.

Example scenario for POWER server tuning

This scenario includes a Power8 server with 512GB of memory and 20 processors (across two processor modules), all divided between four LPARs which have some adapters virtualized by a Virtual I/O Server (VIOS) LPAR. If high availability is important you need at least two Virtual I/O Servers.

Attention!

Do not forget to leave enough CPU and memory capacity for the VIO Server or this could impact your environment if you are using Storage Pools, vdisks and/or Shared Ethernet Adapters (SEA). Do NOT use Shared Ethernet Adapters for handling Spectrum Scale Network I/O. Use dedicated adapters for this purpose instead.

The partition profiles for each LPAR should be set for Dedicated Processor mode, in our case with 5 processors for each LPAR.

For memory, the partition profiles for each LPAR should then be set with 100GB desired and 110GB maximum memory, the desired setting should leave the 10% reserved for the server hypervisor use. This can be refined to reduce the amount of unused memory.

Configuring your LPARs to run Spectrum Scale

NOTE: The steps discussed in this section presume the use of an HMC.

Since there is more than one LPAR per system make sure the best memory affinity is available to the partitions. If you have a single system with all processosrs and memory allocated to a single partition this step is not needed.

Shutdown the operating system to get the LPARs to the “Not Activated” state. Then change the partition profile for Spectrum Scale based on the “LPAR hardware allocations for NUMA based POWER Servers” section of this paper.

Push the memory and processor configuration you just created from the profile that is resident at the HMC to the hypervisor. To do that without the need to perform a full operating system boot, activate the profile you just changed, pointing the boot to SMS.

Shutdown the partitions again.

Run Dynamic Platform Optimizer (DPO) for the server after all the partitions on it are correctly configured. From the HMC command line issue this command:

```
optmem -m <Server_MTMS> -t affinity -o start
```

Monitor the progress of the DPO process from HMC command line with this command:

```
lsmemopt -m <Server_MTMS>
```

Once this command returns that the DPO process is finished then the partitions should be fully contained (processor and memory) to their own processor module. Activate the partitions using Current Configuration.

Attention!

Do not boot the partition using the profile, since that takes the configuration from the HMC again, in this case you will loose the optimization created by the Dynamic Platform Optimizer. Whenever you shutdown your partition remember to start it using “Current Configuration”. If you need to change the profile or DLPAR memory or processors you will need to run DPO again.

4 Operating System Tunining

Perform these tuning steps on each node of the Spectrum Scale cluster:

1. Install requisite software:

   ```
   yum -y install tuned-utils tuned numactl
   ```

2. Prepare tuning files:

   ```
   cp -av /usr/lib/tuned/throughput-performance /etc/tuned
   ```

   ```
   mv /etc/tuned/throughput-performance/ /etc/tuned/scale
   ```

3. Setup tuning configuration file:

   ```
   echo -e "\\n[script]\\nscript=script.sh\\n" >>
   /etc/tuned/scale/tuned.conf
   ```

4. Create tuning script (the following is to be copy/pasted to command line and executed):

   ```
   cat > /etc/tuned/scale/script.sh << EOF
   #!/bin/sh

   . /usr/lib/tuned/functions

   start() {
   /bin/cpupower idle-set -e 0
   /bin/cpupower idle-set -D 1
   /sbin/ppc64_cpu --smt=2
   }

   stop() {
   /sbin/ppc64_cpu --smt=off
   }

   process \$@
   EOF
   ```

5. Make the script file executable:

   ```
   chmod +x /etc/tuned/scale/script.sh
   ```

6. Start tuning service:

   ```
   systemctl enable tuned
   ```

7. List available tuning profiles, verify scale profile is now there:

   ```
   tuned-adm list
   ```

8. Switch tuning profile to scale:

```
tuned-adm profile scale
```

9. Reboot the node. Upon boot, the proper tuning settings should be applied. Verify with the following command:

   ```
   cpupower idle-info
   ```

   CEDE should be set to 'CEDE (disabled)' as Shown in the example bellow:

   ```
   [root@p8n06 ~]# cpupower idle-info
   CPUidle driver: pseries_idle
   CPUidle governor: menu

   Analyzing CPU 0:
   Number of idle states: 2
   Available idle states: snooze CEDE
   snooze:
   Flags/Description: snooze
   Latency: 0
   Usage: 14702332
   Duration: 1467490668
   CEDE (DISABLED) :
   Flags/Description: CEDE
   Latency: 10
   Usage: 3312
   Duration: 39018403
   ```

   Then check if the SMT is correctly set:

   ```
   ppc64-cpu --smt
   ```

   ```
   [root@p8n06 ~]# ppc64_cpu --smt
   SMT=2
   [root@p8n06 ~]#
   ```

   You should be able to see SMT set to 2.

5 Spectrum Scale Tuning

Enable NUMA memory interleave as the default, just run from one node:

- Change Spectrum Scale configuration

  ```
  mmchconfig numaMemoryInterleave=yes
  ```

- Validate it is set

  ```
  mmlsconfig numaMemoryInterleave
  ```

Spectrum Scale Performance Parameter Tuning:

The IBM Knowledge Center and FAQs have guidance on performance tuning; however, if you are running 4.2.0.3 or above, take advantage of the simplified performance tuning by setting the 'workerThreads' configuration parameter, which automatically configures other performance tuning parameters based on that setting. Default suggested is 512.

- Change command:

  ```
  mmchconfig workerThreads=512
  ```

- Verify:

  ```
  mmlsconfig workerThreads
  ```

Spectrum Scale Environment Variables:

This setting is specific to systems with Mellanox adapters, but will not hurt for systems without Mellanox adapters. These settings give hints to Mellanox code stack that the node runs Spectrum Scale and turns on related fixes.

- Export the variables in the current running session:

  ```
  mmchconfig envVar="MLX4_USE_MUTEX=1 MLX5_SHUT_UP_BF=1 MLX5_USE_MUTEX=1"
  ```

Filesystem Log size adjustment:

Make sure filesystem log size is 32M:

```
mmchfs <fs_name> -L 32M
```

You also should check that Relative atime Option is enabled on the filesystem, so not every file access causes a atime update, rather just once every 24 hours. To perform that change run :

Mmchfs <fs_name> -S relatime

6 Tuning Validation

Validate with numactl -H from the Linux node command line.

The goal is to have memory evenly spreaded across numa nodes and have Spectrum Scale correctly distribute memory allocation on all numa nodes.

Examples of numactl output:

Good Example: Large allocation of memory and cpu for the partition. Shows even distribution of memory and cpu across all NUMA nodes:

```
[root@p8n06 ~]# numactl -H
available: 4 nodes (0-3)
node 0 cpus: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
25 26 27 28 29 30 31 32 33 34 35 36 37 38 39
node 0 size: 62464 MB
node 0 free: 48928 MB
node 1 cpus: 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61
62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79
node 1 size: 62464 MB
node 1 free: 49168 MB
node 2 cpus: 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100
101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119
node 2 size: 59904 MB
node 2 free: 48890 MB
node 3 cpus: 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135
136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155
156 157 158 159
node 3 size: 62976 MB
node 3 free: 47819 MB
node distances:
node 0 1 2 3
0: 10 20 40 40
1: 20 10 40 40
2: 40 40 10 20
3: 40 40 20 10
[root@p8n06 ~]#
```

Bad Example: Partition with good cpu distribution across NUMA nodes, but workload not evenly distributed across numa nodes.

```
[root@p8n07 ~]# numactl -H
available: 4 nodes (0-3)
node 0 cpus: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
25 26 27 28 29 30 31 32 33 34 35 36 37 38 39
node 0 size: 62976 MB
node 0 free: 206 MB
node 1 cpus: 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61
62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79
node 1 size: 62720 MB
node 1 free: 58473 MB
node 2 cpus: 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100
101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119
node 2 size: 61952 MB
node 2 free: 55758 MB
node 3 cpus: 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135
136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155
156 157 158 159
node 3 size: 62720 MB
node 3 free: 54097 MB
node distances:
node 0 1 2 3
0: 10 20 40 40
1: 20 10 40 40
2: 40 40 10 20
3: 40 40 20 10
[root@p8n07 ~]#
```

Bad Example: Partition with cpu allocation and memory all from to the same NUMA node, few processors and memory on the second numa node.

```
[root@p8n08 ~]# numactl -H
available: 2 nodes (0-1)
node 0 cpus: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
51 52 53 54 55 56 57 58 59 60 61 62 63
node 0 size: 120064 MB
node 0 free: 69504 MB
node 1 cpus:
node 1 size: 6912 MB
node 1 free: 24 MB
node distances:
node 0 1
0: 10 10
1: 10 10
[root@p8n08 ~]#
```

Bad Example: Partition with opposite imbalanced cpu and memory allocation across NUMA nodes:

```
[root@p8n09 ~]# numactl -H
available: 2 nodes (0-1)
node 0 cpus: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 56 57 58
59 60 61 62 63 #<--- Lots of cpus
node 0 size: 6400 MB #<--- But has very few memory
node 0 free: 14 MB
node 1 cpus: 48 49 50 51 52 53 54 55 #<--- Very few cpus
node 1 size: 120576 MB #<--- But has most of the memory
node 1 free: 69757 MB
node distances:
node 0 1
0: 10 20
1: 20 10
[root@p8n09 ~]#
```