LSF with OS Partitioning and Virtualization

Feb 2011

This document describes how LSF works in OS partitioning and virtualization environments. It will focus on Solaris containers and AIX partitions because these are the environments most commonly used by SAS Grid Manager customers who choose to use OS partitioning and virtualization.

LSF Version

LSF version covered in this report is LSF7.0. Most of the testing was done with LSF7.02, but the results are expected to be applicable to LSF7.05 and 7.06.

1. Summary

SOLARIS:

The only known configuration with which LSF works well (daemons and functionality) is in a non-global Solaris container that uses a dedicated, fixed number of CPUs.

AIX:

The known configuration with which LSF works well (daemons and functionality) is LPARs and DLPARs with dedicated processors. In more detail,

1. With LPARs, DLPARs and micro-partitions, all LSF daemons can run, and LSF is able to dispatch and run jobs on the partitions like on a physical host.
2. LSF cannot run on WPARs.
3. Specific to CPU utilization reporting,
    a. In micro-partitions with shared processors (capped or uncapped), LSF-reported CPU utilization is not consistent with *mpstat* output. So LSF CPU-based scheduling may not work well
    b. On LPARs and DLPARs with dedicated processors, LSF works well and reports CPU utilization correctly. LSF also reports the number of processors correctly after CPU allocation is changed dynamically on DLPARs.

2. Solaris Containers

This refers to the Solaris Containers introduced in Solaris 10.

2.1.Global zone

In a global zone of Solaris 10, LSF works well without any known issues.

2.2.Non-global zone

In non-global zone of Solaris 10, there are known limitations.

2.2.1. Privileges

The root user in non-global zones of Solaris containers does not have all the privileges that the root user in a global zone has. For LSF to work in a non-global zone, we must ensure all required privileges are configured for the root user. A typical issue is the root user lacks *proc_priocntl* privilege. Make sure this is assigned to the root user of the non-global zone, for example, through *zonecfg* command.

2.2.2. Resource Management

When all required privileges are available for the root user (as in 2.2.1), all LSF daemons can run and LSF will be able to dispatch and run jobs on a non-global zone like on a physical host.

However, the number of CPUs and CPU utilization detected by LSF may not be correct specific to a non-global zone, depending on what resource management mechanism is configured. This may interfere with LSF CPU-based scheduling.

The resource management features provided by Solaris 10 include a combination of Fair Share Scheduler, CPU caps, dedicated CPUs, and resource pools.

The only known configuration with which LSF works well in a non-global Solaris container is to use a dedicated fixed number of CPUs.

Dedicated CPUs

The *dedicated-cpu* resource specifies that a subset of the system's processors should be dedicated to a non-global zone while it is running. When the zone boots, the system will dynamically create a temporary pool for use while the zone is running. A fixed number of CPUs or a range, such as 2-4 CPUs, can be specified for the number of dedicated CPUs.

When dedicated CPUs are configured with a fixed number of CPUs (min=max), LSF works well and correctly reports the number of CPUs and CPU usage in the non-global container.

To be tested: If a range is specified for the number of CPUs (min != max), LSF does not report the number of CPUs and CPU usage accurately when CPUs are added or removed from the container.

Capped CPUs

The *capped-cpu* resource provides an absolute fine-grained limit (up to 2 decimal points) on the amount of CPU resources that can be consumed by a zone

CPU capping for Solaris containers, by design, will make all CPUs in the global zone visible to each of the individual containers. LSF detects the number of CPUs for the container to be the total number of CPUs on the physical host. The CPU utilization LSF reports is always consistent with *mpstat* output, which is the host/global-level metric for CPU-capped Solaris containers. There are no Solaris commands that show the CPU utilization specific to a single container when CPU capping is in effect.

To illustrate, suppose a physical host has 2 CPUs and there are two Solaris containers z1 and z2 with CPU capping of 0.8 each. If z1 has a job running using up all of the CPUs available for this container (0.8), and z2 does not have any jobs running, the CPU utilization seen from both z1 and z2 is host-level, 40%. This can be obtained from *mpstat* output on z1 and z2, and LSF reports the same number. In this case, everything else equal, LSF will have equal probability of dispatching new jobs to z1 and z2, even though z1 has no more capacity to run more jobs and z2 has the full capacity. This may be mitigated by the job slots configured for z1 and z2. For example, if both of them are manually configured with 1 job slot, since the job slot on z1 is used, and the one on z2 is available, LSF will dispatch the new job to z2. However, in general, CPU-based scheduling of LSF does not work in this environment. CPU capping for Solaris containers and LSF are essentially conflicting workload management tools and should not be used together.

Fair Share Scheduler

Fair Share Scheduler can be used to control the allocation of available CPU resources among zones, based on their importance. This importance is expressed by the number of shares of CPU resources that is assigned to each zone.

When Fair Share Scheduler is configured, the case is similar to capped CPUs for LSF. All CPUs are visible to the sharing zones and global-level CPU utilization is reported in the zones. This is another example of conflicting workload management tools and therefore CPU-based scheduling of LSF does not work in this environment.

Resource Pool

Resource pools can be configured to include a processor set with an optional scheduling class. Dynamic resource pools provide a mechanism for dynamically adjusting each pool's resource allocation in response to system events and application load changes. A zone can then be associated with a configured resource pool.

Though not tested, if a non-global zone is associated with a static resource pool where the processor set has a fixed number of CPUs, LSF may work well in this zone. In all other cases, CPU-based scheduling of LSF may not work well.

3. AIX Partitions

3.1.AIX Partitions Overview

There are several types of AIX partitions.

Logical partitions (LPAR): introduced with AIX5.1 and POWER4 technology. Multiple operating systems including AIX and Linux can run in separate partitions on the same system without interference. However, a reboot is required to move resources between LPARs.

Dynamic logical partitions (DLPAR): introduced in AIX5.2. More system flexibility is achieved by being able to move CPU, I/O adapters and memory dynamically without rebooting the LPARs.

Micropartitions: introduced in AIX5.3. Micro-Partitioning maps virtual processors to physical processors and the virtual processors are assigned to the partitions instead of the physical processors. With IBM eServer p5 servers, you can choose the following types of partitions:

- **Dedicated processor partition**: the entire processor is assigned to a particular logical partition.
- **Shared processor partition**: the physical processors are virtualized and then assigned to partitions. Sharing mode can be capped or uncapped.

Workload partitions (WPAR): introduced in AIX6.1. WPAR is a purely software partitioning solution that is provided by the operating system. WPAR provides a solution for partitioning one AIX operating instance in multiple environments, each environment being a WPAR. WPARs can be created within multiple AIX instances of the same physical server, whether they execute in dedicated LPARs or micropartitions (See Figure 1, copyright by IBM). A powerful feature of WPAR is *WPAR mobility* with checkpoint/restart.

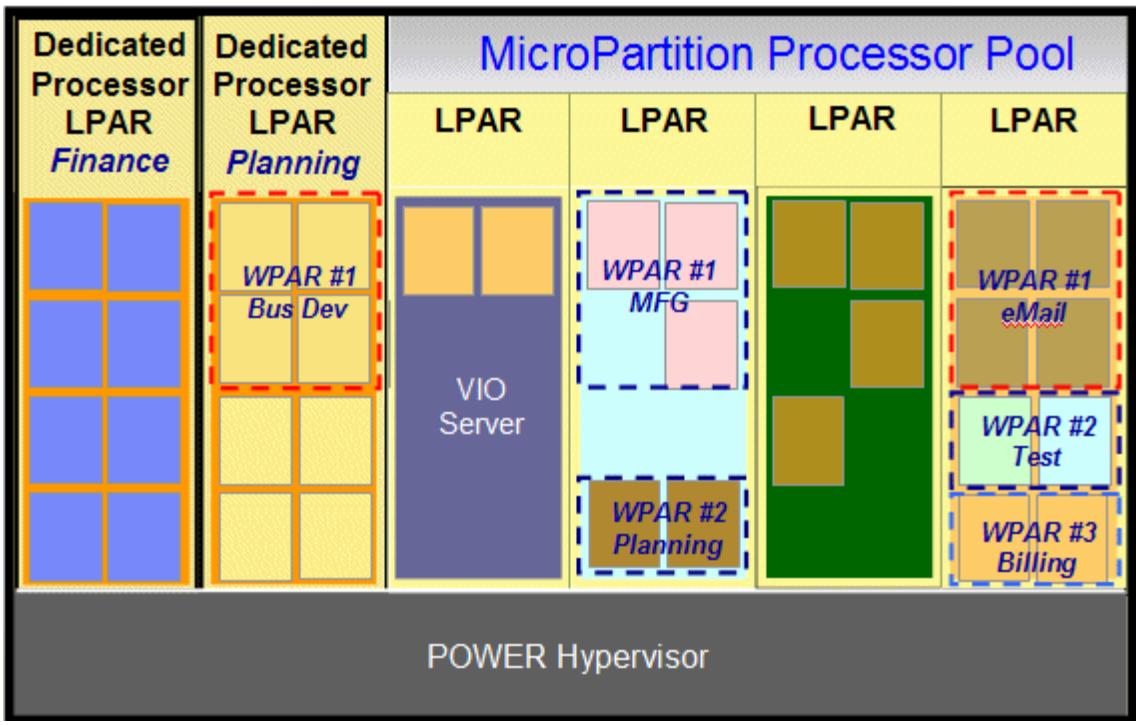

Figure 1. (Copyright by IBM)

Two types of WPAR are available: Application WPAR and System WPAR.

An application WPAR is a light environment suitable for execution of one or more processes. It is transient and has the same lifetime as the application.

A system WPAR is almost a full AIX environment. Each System WPAR has dedicated writable file systems, although it can share the global environment /usr and /opt file systems in read only mode. When a system WPAR is started, an init process is created for this WPAR, which in turns spawns other processes and daemons.

3.2.LSF on AIX Partitions

With LPARs, DLPARs and micropartitions, all LSF daemons can run, and LSF is able to dispatch and run jobs on the partitions like on a physical host.

LSF cannot run on WPARs.

3.2.1. LPARs with dedicated processors

On LPARs (whether on micropartitions or physical machine) with dedicated processors, LSF works well and is able to reports CPU utilization correctly.

3.2.2. DLPARs with dedicated processors

On DLPARs (whether on micropartitions or physical machine) with dedicated processors, LSF works well and is able to report CPU utilization correctly. LSF also reports number of processors correctly when CPU allocation is changed dynamically on DLPARs.

3.2.3. Micropartitions

In a micropartition that is configured with *shared* processors, LSF detects the number of CPUs to be the same as the configured virtual processors, and it reports CPU utilization independently for each partition. However, the reported CPU utilization is not consistent with *mpstat* output. So LSF CPU-based scheduling may not work well.

3.2.4. WPARs

Application WPARs are transient and not suitable for server applications like LSF daemons. Also, LSF cannot run on system WPARs because LSF needs access to devices like /dev/mem and /dev/kmem. These are typically removed on WPARs due to best practice recommendations from IBM.