



**Red Hat Reference Architecture Series**

# Optimizing RHEL for SAS 9.2

**Author: Barry Marson – Principal Software Engineer**

**Version 1.0**

**January 2011**





## Optimizing RHEL for SAS 9.2

1801 Varsity Drive™  
Raleigh NC 27606-2072 USA  
Phone: +1 919 754 3700  
Phone: 888 733 4281  
Fax: +1 919 754 3701  
PO Box 13588  
Research Triangle Park NC 27709 USA

Linux is a registered trademark of Linus Torvalds. Red Hat, Red Hat Enterprise Linux and the Red Hat "Shadowman" logo are registered trademarks of Red Hat, Inc. in the United States and other countries.

Microsoft and Windows are U.S. registered trademarks of Microsoft Corporation.

UNIX is a registered trademark of The Open Group.

Intel, the Intel logo, Xeon and Itanium are registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

All other trademarks referenced herein are the property of their respective owners.

© 2011 by Red Hat, Inc. This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, V1.0 or later (the latest version is presently available at <http://www.opencontent.org/openpub/>).

The information contained herein is subject to change without notice. Red Hat, Inc. shall not be liable for technical or editorial errors or omissions contained herein.

Distribution of modified versions of this document is prohibited without the explicit permission of Red Hat Inc.

Distribution of this work or derivative of this work in any standard (paper) book form for commercial purposes is prohibited unless prior permission is obtained from Red Hat Inc.



# Table of Contents

1 Executive Summary.....	5
1.1 Recommended Best Practices – Bare Metal.....	5
1.2 Recommended Best Practices - Virtualization.....	6
2 Background.....	7
3 Software Stack.....	8
SAS Software:.....	8
Operating System:.....	8
3.1 SAS.....	8
3.2 Red Hat Enterprise Linux.....	9
3.2.1 Red Hat Enterprise Linux 5.5.....	9
3.2.2 Red Hat Enterprise Linux 6.0.....	9
4 Reference Architecture Configuration.....	12
4.1 Server Configuration.....	12
4.2 Software Configuration.....	12
4.2.1 Operating System.....	12
4.2.2 Storage and Volume Management.....	12
4.3 File Systems.....	13
4.3.1 Supporting Scripts.....	13
4.2.5 SAS Workload Test Execution.....	14
4.2.6 SAS Data.....	15
5 Bare Metal Performance Testing.....	16
5.1 File systems.....	17
5.2 I/O Barriers.....	17
5.3 Transparent Huge Pages.....	18
5.4 Best Practices.....	18
6 Virtualization Testing.....	19
6.1 Host Configuration.....	19
6.2 KVM Guest Configuration.....	19
6.2.1 RHEL 5.5 guest configuration.....	20
6.2.2 RHEL 6.0 guest configuration.....	20
6.3 Support Script Changes.....	20



6.4 Virtualization Results.....	20
6.5 Host and Guest by Version.....	21
6.6 File systems.....	22
6.7 I/O Barriers.....	22
6.8 Transparent Huge Pages in the Host.....	22
6.9 Best Practices.....	23



# 1 Executive Summary

Today, companies are increasingly utilizing analytics to discover new revenue and cost-saving opportunities. Many business professionals turn to SAS, a leader in business analytics software and service, to help them improve performance and make better decisions faster. Analytics are also being employed in risk management, fraud detection, life sciences, sports, and many more emerging markets. However, to maximize the value to the business, analytics solutions need to be deployed quickly and cost-effectively, while also providing the ability to readily scale without degrading performance. Of course, in today's demanding environments, where budgets are still shrinking and mandates to reduce carbon footprints are growing, the solution must deliver excellent hardware utilization, power efficiency, and ROI.

To help solve some of these challenges, Red Hat and SAS have collaborated to recommend the best practices for configuring SAS 9.2 running Red Hat Enterprise Linux 5 and 6. **Areas researched include I/O subsystem and file system selection, and kernel tuning, both in a bare metal and virtualized (KVM) environment.** The results will help you deploy faster, reduce risk, lower cost, and plan for future upgrades.

## 1.1 Recommended Best Practices – Bare Metal

The configuration producing the best results in the RHEL 5.5 environment was clearly xfs with I/O barriers disabled. Ext4 is still in Technical Preview in RHEL 5.5. The system should be tuned via **ktune** (RHEL 5).

For RHEL 6.0, using either xfs or ext4, with I/O barriers and Transparent Huge Pages both disabled is best. Xfs performs better than ext4 and additionally used 20% less system CPU resources. If there are more compute based SAS jobs running, ext4's extra CPU requirements could reduce performance.

The system should be tuned via **tuned [enterprise-storage]** (RHEL 6). Since Transparent Huge Pages is enabled by tuned, we recommend creating a run level init script to disable it after the tuned service has started.

The LUNs and logical volumes which the SAS file systems reside on should be tuned for increased **read ahead** support. Even though RHEL6 tuned elevates these values, they are still not typically large enough for SAS. The best way to tune them is with the **blockdev** command. This is not persistent between boots so we recommend creating a run level init script to disable it after the tuned service has started.



## 1.2 Recommended Best Practices - Virtualization

To obtain the best performance in a virtualized environment, we recommend using RHEL 6 in both the host and guest. SAS performed significantly better this way.

### Host

- The system should be tuned via **tuned [enterprise-storage]** (RHEL 6).
- **Transparent Huge Pages (THP) enabled** (default) in the RHEL 6 host improves performance as well. SAS wall clock and especially system time is dramatically reduced.

### Guest

- The system should be tuned via **tuned [enterprise-storage]** (RHEL 6).
- xfs is the ideal file system for either RHEL 5 or RHEL 6. Ext4 can be used with a RHEL 6 guest.
- **I/O barriers** should be disabled.
- **Transparent Huge Pages** should be disabled. Since this is enabled by tuned, we recommend creating a run level init script to disable it after the tuned service has started.
- **Read ahead** tuning for all LUNs and logical volumes which the SAS file systems directly mount need to be elevated. Even though RHEL6 **tuned** elevates these values, they are typically not large enough for SAS heavy I/O workloads. The best way to tune them is with the **blockdev** command. This is not persistent between boots so we recommend creating a run level init script to disable it after the tuned service has started.



## 2 Background

According to the Gartner EXP CIO report *Leading in Times of Transition: The 2010 CIO Agenda*, the top three CIO business priorities are business process improvement, reducing enterprise costs, and increasing the use of information/analytics<sup>1</sup>. Subscription-based offerings from SAS and Red Hat can help deliver the technology you need to improve business processes and maintain a competitive edge, at prices you can afford. SAS is a leader in business analytics, helping companies unearth insights buried in information. And Red Hat's leading open-source operating system enables enterprise-class performance, flexibility, and cost savings over Windows and UNIX RISC platforms. The analytical and budgetary needs of companies around the world are bringing Red Hat and SAS into close strategic alignment.

“SAS and Red Hat are long-time technology partners, and we look forward to expanding our strategic partnership,” said Keith Collins, SAS vice president and chief technology officer. “Together, we are increasing customer value with the combination of our best-in-class products and the open source, enterprise-ready Red Hat Enterprise Linux platform.”

### **Game changing performance and scalability**

The ability to take advantage of performance enhancements in the latest processors and the ability to tune I/O and virtual memory, makes Red Hat Enterprise Linux an ideal platform for SAS Business Analytics. Industry benchmarks reflect the scalability and performance of Red Hat Enterprise Linux in both scale-up (vertical) and scale-out (grid) models. And the wonderful thing about tuning for SAS is that it's so very easy to do to get market leading performance results.



## 3 Software Stack

This testing was performed with the following software components:

SAS Software:

- Foundation SAS 9.2, SAS/STAT®

Operating System:

- Red Hat Enterprise Linux 5.5 64-bit
- Red Hat Enterprise Linux 6.0 64-bit

### 3.1 SAS

SAS 9.2 provides the core components of the SAS Business Analytics Framework and significant performance improvements over SAS 9.1 on Linux. SAS 9.2 helps users gain insights that are often hidden in data, so they can reach evidence-based decisions with confidence. SAS 9.2 supports the entire analysis process — from data access to the point of decision — however varied or complex. A wide range of data integration techniques empowers users to collect, classify, process, analyze, and interpret data to reveal new insights. SAS 9.2 advances the capabilities of SAS analytical products, including forecasting, data mining, optimization, and model management. SAS Analytics provide rapid answers to key business questions, allowing decision makers to react more quickly to fast changing conditions.

SAS 9.2 is available as 64-bit enabled applications supporting 64-bit extended architectures. This enables you to scale up or consolidate multiple SAS instances within one affordable, powerful, commodity system.

### 3.2 Red Hat Enterprise Linux

#### 3.2.1 Red Hat Enterprise Linux 5.5

Red Hat Enterprise Linux performance features include:

- **SMP performance and scalability.** Multi-process or threaded applications can be optimally scheduled in large SMP systems. A vast virtual address space enables SAS Business Analytics to effectively use more memory to work on larger data sets. Enhancements enable applications to effectively use more processors.



- **Intelligent performance.** Efficient use of software threads, support for hyper- threading technology, and the ability to change the clock speed on a running processor increase performance.
- **Automated energy efficiency.** Power technologies from Intel and AMD and optimizations in the operating system lower power consumption during off-peak times. Consuming less power means lower cooling requirements, which contributes to further savings and greener data centers.
- **Tuning** for optimum I/O throughput. I/O performance can be optimized on a per- device basis. Support for 10 gigabit Ethernet, iSCSI, and Fibre Channel over Ethernet allow the latest storage technologies to be used. MPIO allows multiple connections from servers to storage to increase availability and throughput.

### 3.2.2 Red Hat Enterprise Linux 6.0

Red Hat Enterprise Linux performance features include:

- **CPU Scheduler CFS and Ticketed Spinlocks:** The RHEL 6 scheduler uses ticketed spinlocks for scalability on x86\_64 large SMP system and to ensure Completely Fair Scheduling (CFS) as well as avoiding process/NUMA node starvation.
- **VM Scalability: Split-LRU and Transparent Hugepages (THP):** RHEL 6 is NUMA aware and will place processes and their associated memory on a NUMA node to ensure lowest memory latency, best response time and therefore the highest possible throughput on the HP DL980 G7 server.

Additionally, RHEL 6 implements a new split LRU VM algorithm that separates the Linux page cache from anonymous memory and locking reduction done by developers from HP and Red Hat.

RHEL 6 also implements Transparent Hugepages (THP) to dynamically allocate x86\_64 2MB pages when available compared to the base page size for x86\_64 which architecturally is 4KB.

- **Disk I/O: BDI Flush, MPIO:** RHEL 6 replaced pdflush for processing buffered writeback, opting to flush threads using Backing Device Information (BDI) allowing for linear scalability as LUNs are presented to the OS.

RHEL 6 continues to implement Linux native multipath (MPIO) for high availability.

- **File System Scalability: EXT3 / EXT4 / XFS:** RHEL 6 will support standard EXT3 file systems and either EXT4 or XFS as enhancements in scalability for large file system volumes. EXT4 and XFS have improved



logging and recovery for files greater than 1 TB which can be an order of magnitude faster than EXT3.

This reference architecture focuses on EXT3 to compare against a RHEL 5.5 based system with EXT3. The file system is tuned to enhance performance using the I/O **elevator=deadline** scheduler and disabling the files system I/O barriers which are not needed for enterprise storage. Although not used to produce the results presented in this document, this tuning may be accomplished using the tuned-adm infrastructure described in the following section.

- **RHEL 6 tuned-adm Infrastructure**

RHEL 5 introduced the utility ktune to adjust common system control (sysctl) parameters in RHEL 5 for optimizing CPU, memory, network and I/O for throughput or latency. In RHEL 6, Red Hat extended the utility to include:

- tuned-adm list
- available profiles:
  - default
  - latency-performance
  - enterprise-storage
  - throughput-performance
  - laptop-ac-powersave
  - laptop-battery-powersave
- optimizations in for latency / throughput and enterprise-storage including:
  - adjusting the I/O elevator=deadline (versus CFQ default)
  - altering the powersave mode from OnDemand to Performance
  - setting the VM reclaim parameters for dirty\_ratio back to the RHEL 5 value of 40 (RHEL 6 adjusted default to 20)
- additional optimization throughput and enterprise-storage also adjusts:
  - block device and LVM read ahead values increased by a factor of 4



- scheduler tunable quantum back to RHEL 5 default of 10 milliseconds (RHEL 6)
- default quantum is 4 milliseconds
- additional optimization for enterprise-storage includes remounting the file system using “-o barrier=0” (assumes enterprise storage). Future updates to RHEL 6 may do this automatically. See /proc/mount to view the barrier settings on the server.



# 4 Reference Architecture Configuration

## 4.1 Server Configuration

### Server Configuration:

- Two Socket Intel Nehalem Server (4 cores per socket) – 8 total cores/16 threads
- Intel Hyper-threading Technology
- Intel Turbo Boost Technology on
- 48 GB RAM
- 2 x 4 G-bit fiber connections
- 1 Storage array presenting 8 x 200 GB LUNS.

## 4.2 Software Configuration

### 4.2.1 Operating System

The test system was staged with dual boot systems disks to house RHEL 5.5 and RHEL 6.0. The specific tuning infrastructure was enabled for each version of RHEL. For RHEL 5.5, **ktune** was used. For RHEL6, **tuned** was used with the **enterprise-storage** profile.

### 4.2.2 Storage and Volume Management

Multipath device mapper support was enabled to easily manage the 4 fiber channel paths to each LUN. The “round robin” policy was used and all paths were active/active.

LVM was used to manage and carve up the 8 LUNs. Five LVM striped logical volumes were created for each LUN. See Table 1.



```
[root@sastest input]# lvs -o+stripes,stripesize
LV          VG      Attr      LSize   #Str    Stripe
lARCHIVE    sas_vg  -wi-ao    300.00g 8       64.00k
lASUITE     sas_vg  -wi-ao    50.00g  8       64.00k
lASUITEcdata sas_vg  -wi-ao    100.00g 8       64.00k
lASUITEinput sas_vg  -wi-ao    250.00g 8       64.00k
lASUITEoutput sas_vg  -wi-ao    200.00g 8       64.00k
lASUITEsaswork sas_vg  -wi-ao    150.00g 8       64.00k
```

**TABLE 1: Logical Volume Sizes including Striping Configuration.**

Separating the workload’s primary directories into their own logical volume and file system provided us with a means to better understand resource utilization.

The lArchive volume was created specifically for storing results and keeping archives of the input data and test kits.

## 4.3 File Systems

The following file systems were tested

- **ext3** - legacy file system and default for RHEL 5.5
- **ext4** - technical preview for RHEL 5.5 and default for RHEL 6.
- **xfs** - Available for RHEL 5.5 with the Scalable file system license and fully available with RHEL 6.
- **gfs2** - Tested in **lock\_no**lock mode outside of a cluster. While this mode is not officially supported, testing exercises the gfs2 file system code, and provides a base line for understanding how this SAS workload will run in a gfs2 cluster in the future.

### 4.3.1 Supporting Scripts

A series of scripts were created to automatically setup and and launch the SAS work load varying the following subsystems/parameters:

- rebuilding all of the test file systems with default parameters (ext3, ext4, xfs, gfs2)
- mounting the file systems with the appropriate barriers settings (enabled or disabled)
- Elevate read ahead values for each LUN/logical volume by using the blockdev command. For example:



```
# blockdev --setra 4096 /dev/mapper/sas_vg-lASUITEsaswork
```

- enabling/disabling Transparent Huge Pages (THP).

Once the specific file system and system parameters were configured:

- the workload scripts and input data were copied into place.
- vmstat and iostat data was collected for each run.
- The page buffer cache was flushed before each run to always have a cold cache at startup. This was accomplished with:

```
# sync  
# echo 3 > /proc/sys/vm/drop_caches
```

- The workload was then launched.
- Upon completion, the output logs and results were archived.

## 4.2.5 SAS Workload Test Execution

The test scenario used was the Mixed 8 core workload (mix of CPU and I/O intensive jobs). The specifics of the workload are:

- 34 jobs launched during the scenario
- PROCs: GLM, LOGISTIC, RISK, REG, MEANS, SORT, FREQ, SUMMARY, SQL
- Mixed shorter and longer running jobs
- Goal of scenario is to leverage a mix of CPU and I/O resources
- This version was designed to run on an 8-core system

Each test scenario consists of a set of SAS jobs run in a multi-user fashion to simulate a typical SAS batch, SAS® Enterprise Guide® user environment. All SAS jobs are a combination of computational- and I/O-intensive SAS procedures. Each scenario launches jobs simultaneously at a set interval to help simulate a multi-user environment where users come and go from the system. The test is designed to run in a period of 30 to 60 minutes.



## 4.2.6 SAS Data

Characteristics of the data for this scenario:

- SAS data sets and text files.
- Row counts up to 90 million.
- Variable counts up to 297.
- 1.1 GB total input/output data.
- File sizes ranging from several kilobytes to 45 GB in size.

Data volumes were designed to be larger than the hardware cache in order to place realistic stress on the hardware and operating system file cache.

**Note: The SAS benchmarking scenarios are designed to replicate a typical Foundation SAS customer's resource use. However, particular customer applications can vary greatly depending on tasks, PROCs used, data volumes and other customer requirements.**

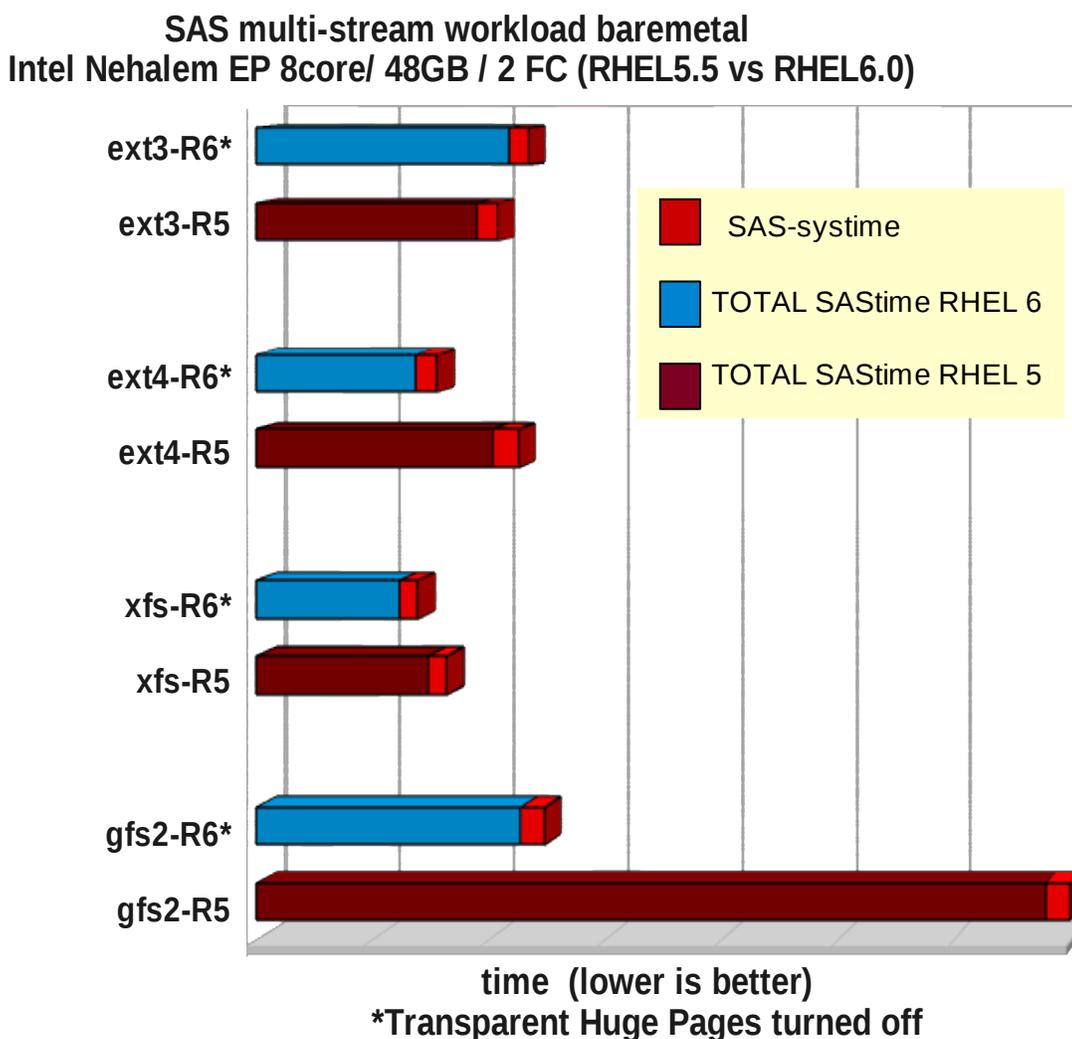


## 5 Bare Metal Performance Testing

Below is a graph (Figure 1) of the configuration which produced the best results. This configuration is:

- tuned with **ktune** for RHEL5.5; **tuned** for RHEL 6.0
- Transparent Huge Pages (THP) disabled for RHEL 6.0
- I/O barriers disabled

The left most part of each bar is the sum in seconds of the 34 SAS jobs wall clock run times. The bright red section is the sum in seconds of the 34 SAS jobs system time.



**Figure 1: Bare Metal Performance Comparing RHEL5.5 vs RHEL6.0**



## 5.1 File systems

Analysis of the results showed some file systems did significantly better than others, sometimes only one one family of RHEL.

- **ext3** performance, regardless of which version of RHEL is being used, simply is not an appropriate file system to use with the file sizes and access patterns of SAS.

One of the primary reasons is ext3 file system allocation is not being extent based. File system data gets fragmented all over the volume, and file deletion times of large files are excessive.

Numerous large files are created and deleted in the saswork directory.

- **ext4** performance in RHEL 5.5 was worse than ext3. It is still in technical preview for this version of RHEL and not recommended for production.

For RHEL 6.0, ext4 is the default file system and provides excellent performance, nearly on par with xfs except that it requires more system CPU cycles.

- **xfs** is the best fully supported performer on RHEL 5.5 and performs even better on RHEL 6.0. In both cases, xfs uses nearly 20% less system CPU resources than ext4.
- **gfs2** performed poorly in RHEL 5.5 than all other file systems. Additionally, RHEL 5.5 gfs2 does not support disabling barriers as noted in the graph.

In RHEL 6.0 it performed better than ext3. Additional testing with Intel hyper-threading technology enabled showed that gfs2 performance improved an additional 15%.

We hope to do characterize SAS in a RHEL6 cluster when resources become available.

## 5.2 I/O Barriers

Runs with I/O barriers disabled typically ran better than with them enabled. In our configuration it was safe to run without these barriers since we had enterprise storage with battery backed storage controller caches.



## 5.3 Transparent Huge Pages

In virtually all cases, the use of Transparent Huge Pages significantly reduced SAS performance. This was expected since this feature is designed to merge/coalesce anonymous pages and SAS relies heavily on the page cache for data manipulation. The performance hit in some cases was as high as 25%.

This feature is enabled when tuned starts up with the 'enterprise-storage' profile. We recommend Transparent Huge Pages be disabled via:

```
# echo never > /sys/kernel/mm/redhat_transparent_hugepage/enabled
```

Since **tuned** enables Transparent Huge Pages, one way to ensure this feature is disabled at boot-up time is to add the command to a run level init script. For example the command could be added to `/etc/rc3.d/S99local` to ensure it is disabled.

## 5.4 Best Practices

The configuration producing the best results in the RHEL 5.5 environment was clearly xfs with I/O barriers disabled. Ext4 is still in Technical Preview in RHEL 5.5. The system should be tuned via **ktune** (RHEL 5).

For RHEL 6.0, using either xfs or ext4, with I/O barriers and Transparent Huge Pages both disabled is best. Xfs performs a little better than ext4 in that it used 20% less system CPU resources. If there are more compute based SAS jobs running, ext4's extra CPU requirements could reduce performance.

The system should be tuned via **tuned [enterprise-storage]** (RHEL 6). Since Transparent Huge Pages is enabled by tuned, we recommend creating a run level init script to disable it after the tuned service has started.

Read ahead tuning for all LUNs and logical volumes which the SAS file systems directly mount need to be elevated. Even though RHEL6 **tuned** elevates these values, they are typically not large enough for SAS heavy I/O workloads. The best way to tune them is with the **blockdev** command. This is not persistent between boots so we recommend creating a run level init script to disable it after the tuned service has started.



## 6 Virtualization Testing

With the ever increasing needs for basic virtualization and cloud computing for provisioning system resources, it became clear quite early in this work that virtualization characterization was needed as well. Since we had limited hardware resources and the SAS workload requires a large amount of compute, memory, and storage bandwidth, we only focused on a single large virtualized KVM guest.

Since we are also interested in a RHEL5.5 and RHEL 6.0 host, the following configurations were tested.

- RHEL5.5 host RHEL5.5 KVM guest
- RHEL5.5 host RHEL6.0 KVM guest
- RHEL6.0 host RHEL6.0 KVM guest

A RHEL5.5 hosting a RHEL6.0 guest was not tested.

### 6.1 Host Configuration

The host operating system configuration was the one used for the bare metal testing. Thus for RHEL 5.5, **ktune** was still configured, and for RHEL 6.0, **tuned** was configured with the **enterprise-storage** profile.

The major difference was tests were executed with Transparent Huge Pages (THP) enabled and disabled from the host perspective.

### 6.2 KVM Guest Configuration

The image for the KVM guest was a simple 32 GB file. The resources allocated to the guest were:

- **8 CPUS** - The entire host
- **44 GB of RAM** - 4GB of total memory reserved for the host kernel needs and KVM qemu process resources like I/O buffers
- **LVM logical volumes** for SAS data processing including the archive volume were passed through to the guest via the virtIO driver with **cache=none** policy.



### 6.2.1 RHEL 5.5 guest configuration

RHEL 5.5 was installed along with **ktune**. The LVM volumes managed by virtIO were presented to the test harness so all recreation and mounting of the file systems was performed inside the guest.

### 6.2.2 RHEL 6.0 guest configuration

RHEL 6.0 was installed along with **tuned**. The **enterprise-storage** profile was enabled. The LVM volumes managed by virtIO were presented to the test harness so all recreation and mounting of the file systems was performed inside the guest.

## 6.3 Support Script Changes

The support scripts created to build the file systems, perform mounts, toggle Transparent Huge Pages (THP), and I/O barriers, was modified to handle the the different device names (vd\* naming). Additionally, since it was clear that THP did not help the SAS environment, it was disabled. No other changes were required.

## 6.4 Virtualization Results

Below is a graph (Figure 2) of the configuration which produced the best results. This configuration was:

### Host

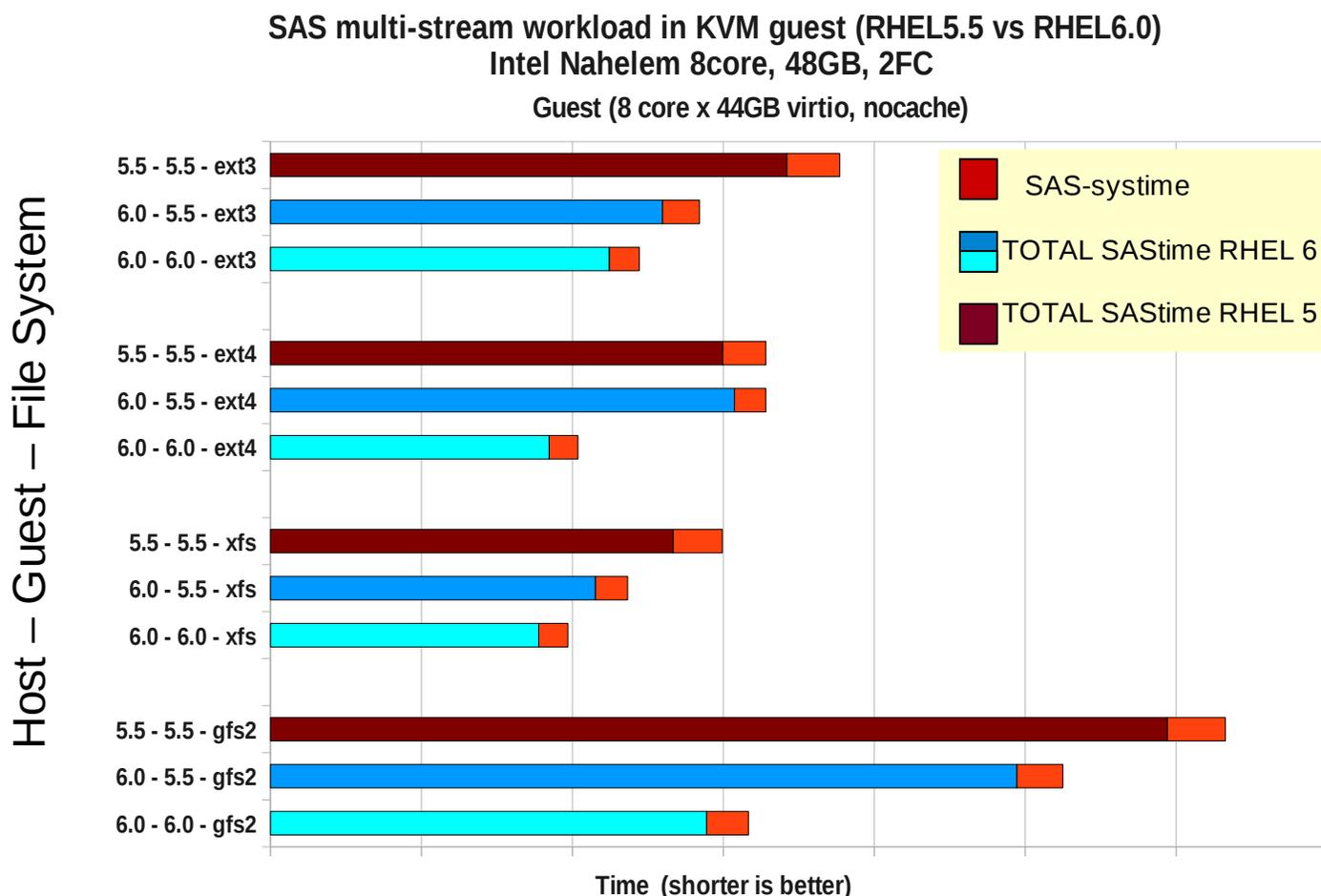
- tuned with **ktune** (for RHEL5.5) **tuned** for (RHEL 6.0)
- Transparent Huge Pages (THP) **enabled**

### Guest

- tuned with **ktune** (for RHEL5.5) **tuned** for (RHEL 6.0)
- Transparent Huge Pages (THP) **disabled** for RHEL 6.0
- I/O barriers disabled



The config names `H:rhel-G:rhel_*` translates to `Host:version-Guest:version_filesystem`. The left most part of each bar is the sum in seconds of the 34 SAS jobs wall clock run times. The bright red section is the sum in seconds of the 34 SAS jobs system time.



**Figure 2: SAS Results in KVM virtual machines**

## 6.5 Host and Guest by Version

One of the first observations from the data, is that the more RHEL6 you use, the better your performance. This is noted by the dark red bars which is a total RHEL5.5 solution, compared to the blue bar which is a RHEL6 host and RHEL5.5 guest, compared to the light blue bar which is a RHEL6.0 solution.

In **most** cases there is a significant performance improvement with a RHEL6 host.

In **all** cases there is a significant performance improvement using a RHEL6 guest.



## 6.6 File systems

Relative file system performance compared to bare metal shows that only gfs2 didn't run as expected in a virtualized environment.

- **ext3** performance, regardless of which version of RHEL is being used, simply is not an appropriate file system to use with the file sizes and access patterns of SAS.
- **ext4** performance with a RHEL6.0 host and RHEL 5.5 guest was worse than expected. Significant performance and scalability enhancements went into ext4 for RHEL6 which have yet to be back ported to RHEL5. Additionally ext4 is still in technical preview for this version of RHEL and not recommended for production.

For the full RHEL 6.0 environment, ext4 is the default file system and provides excellent performance, nearly on par with xfs.

- **xfs** is the best fully supported performer on RHEL 5.5 and performs even better on RHEL 6.0.
- **gfs2** still performed poorly in RHEL 5.5 than all other file systems. Additionally, RHEL 5.5 gfs2 does not support disabling barriers as noted in the graph.

In RHEL 6.0 it performed much better but no longer outperformed ext3.

## 6.7 I/O Barriers

SAS runs with I/O barriers disabled typically ran identically or a little better in a guest.

## 6.8 Transparent Huge Pages in the Host

We already learned from the bare metal runs that Transparent Huge Pages (THP) interferes significantly with performance in the SAS environment. Inside the guest, we recommend disabling THP via:

```
# echo never > /sys/kernel/mm/redhat_transparent_hugepage/enabled
```



Since **tuned** enables Transparent Huge Pages, one way to ensure this feature is disabled at boot-up time is to add the command to a run level init script. For example the command could be added to `/etc/rc3.d/S99local` to ensure it is disabled.

We tested leaving THP enabled (default) in the RHEL 6 host to understand it's effect on the KVM guest process and resources.

In all cases, SAS performance improved with THP enabled in the host. Not only was the SAS wall clock time better, but the SAS system time was upwards of 2.5 times lower. This is probably attributed to less cache misses with the processors Translation Lookaside Buffer (TLB).

## 6.9 Best Practices

To obtain the best performance in a virtualized environment, we recommend using RHEL 6 in both the host and guest. SAS performed significantly better this way.

### Host

- The system should be tuned via **ktune** (RHEL 5) and **tuned [enterprise-storage]** (RHEL 6).
- Transparent Huge Pages (THP) enabled (default) in the RHEL 6 host improves performance as well. SAS wall clock and especially system time is dramatically reduced.

### Guest

- The system should be tuned via **ktune** (RHEL 5) and **tuned [enterprise-storage]** (RHEL 6).
- xfs is the ideal file system for either RHEL 5 or RHEL 6. Ext4 can be used with a RHEL 6 guest.
- I/O barriers should be disabled.
- Transparent Huge Pages should be disabled.
- **Read ahead** tuning for all LUNs and logical volumes which the SAS file



systems directly mount need to be elevated. Even though RHEL6 **tuned** elevates these values, they are typically not large enough for SAS heavy I/O workloads. The best way to tune them is with the **blockdev** command. This is not persistent between boots so we recommend creating a run level init script to disable it after the tuned service has started.