Red Hat Enterprise MRG 1.1

Realtime Installation Guide

Installation information for the Realtime component of Red Hat Enterprise MRG

Lana Brindley

Red Hat Enterprise MRG 1.1 Realtime Installation Guide
Installation information for the Realtime component of Red Hat Enterprise MRG
Edition 0

Author Lana Brindley lbrindle@redhat.com

The GPG fingerprint of the security@redhat.com key is:

CA 20 86 86 2B D6 9D FC 65 F6 EC C4 21 91 80 CD DB 42 A6 0E

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

This book will show you how to download and install the MRG Realtime component of the Red Hat Enterprise MRG distributed computing platform. For detailed information on tuning MRG Realtime, see the MRG Realtime Reference Guide

Preface

Red Hat Enterprise MRG

This book contains basic installation and tuning information for the MRG Realtime component of Red Hat Enterprise MRG. Red Hat Enterprise MRG is a high performance distributed computing platform consisting of three components:

1. *M*essaging — Cross platform, high performance, reliable messaging using the Advanced Message Queuing Protocol (AMQP) standard.
2. *R*ealtime — Consistent low-latency and predictable response times for applications that require microsecond latency.
3. *G*rid — Distributed High Throughput (HTC) and High Performance Computing (HPC).

All three components of Red Hat Enterprise MRG are designed to be used as part of the platform, but can also be used separately.

MRG Realtime

Many industries and organizations need extremely high performance computing and may require low and predictable latency, especially in the financial and telecommunications industries. Latency, or response time, is defined as the time between an event and system response and is generally measured in microseconds (µs). For most general applications running under a Linux environment, basic performance tuning can improve latency sufficiently. For those industries where latency not only needs to be low, but also accountable and predictable, Red Hat have now developed a 'drop-in' kernel replacement that provides this. MRG Realtime is distributed as part of Red Hat Enterprise MRG and provides seamless integration with Red Hat Enterprise Linux 5.2. MRG Realtime offers clients the opportunity to define, measure, configure and record latency times across their organization.

1. Document Conventions

This manual uses several conventions to highlight certain words and phrases and draw attention to specific pieces of information.

In PDF and paper editions, this manual uses typefaces drawn from the *Liberation Fonts*[1] set. The Liberation Fonts set is also used in HTML editions if the set is installed on your system. If not, alternative but equivalent typefaces are displayed. Note: Red Hat Enterprise Linux 5 and later includes the Liberation Fonts set by default.

1.1. Typographic Conventions

Four typographic conventions are used to call attention to specific words and phrases. These conventions, and the circumstances they apply to, are as follows.

`Mono-spaced Bold`

Used to highlight system input, including shell commands, file names and paths. Also used to highlight key caps and key-combinations. For example:

[1] https://fedorahosted.org/liberation-fonts/

> To see the contents of the file **`my_next_bestselling_novel`** in your current working directory, enter the **`cat my_next_bestselling_novel`** command at the shell prompt and press **`Enter`** to execute the command.

The above includes a file name, a shell command and a key cap, all presented in Mono-spaced Bold and all distinguishable thanks to context.

Key-combinations can be distinguished from key caps by the hyphen connecting each part of a key-combination. For example:

> Press **`Enter`** to execute the command.
>
> Press **`Ctrl`**-**`Alt`**-**`F1`** to switch to the first virtual terminal. Press **`Ctrl`**-**`Alt`**-**`F7`** to return to your X-Windows session.

The first sentence highlights the particular key cap to press. The second highlights two sets of three key caps, each set pressed simultaneously.

If source code is discussed, class names, methods, functions, variable names and returned values mentioned within a paragraph will be presented as above, in **`Mono-spaced Bold`**. For example:

> File-related classes include **`filesystem`** for file systems, **`file`** for files, and **`dir`** for directories. Each class has its own associated set of permissions.

Proportional Bold

This denotes words or phrases encountered on a system, including application names; dialogue box text; labelled buttons; check-box and radio button labels; menu titles and sub-menu titles. For example:

> Choose **System > Preferences > Mouse** from the main menu bar to launch **Mouse Preferences**. In the **Buttons** tab, click the **Left-handed mouse** check box and click **Close** to switch the primary mouse button from the left to the right (making the mouse suitable for use in the left hand).
>
> To insert a special character into a **gedit** file, choose **Applications > Accessories > Character Map** from the main menu bar. Next, choose **Search > Find…** from the **Character Map** menu bar, type the name of the character in the **Search** field and click **Next**. The character you sought will be highlighted in the **Character Table**. Double-click this highlighted character to place it in the **Text to copy** field and then click the **Copy** button. Now switch back to your document and choose **Edit > Paste** from the **gedit** menu bar.

The above text includes application names; system-wide menu names and items; application-specific menu names; and buttons and text found within a GUI interface, all presented in Proportional Bold and all distinguishable by context.

Note the **>** shorthand used to indicate traversal through a menu and its sub-menus. This is to avoid the difficult-to-follow 'Select **Mouse** from the **Preferences** sub-menu in the **System** menu of the main menu bar' approach.

`Mono-spaced Bold Italic` or ***`Proportional Bold Italic`***

Whether Mono-spaced Bold or Proportional Bold, the addition of Italics indicates replaceable or variable text. Italics denotes text you do not input literally or displayed text that changes depending on circumstance. For example:

> To connect to a remote machine using ssh, type **ssh *username*@*domain.name*** at a shell prompt. If the remote machine is **example.com** and your username on that machine is john, type **ssh john@example.com**.
>
> The **mount -o remount *file-system*** command remounts the named file system. For example, to remount the **/home** file system, the command is **mount -o remount /home**.
>
> To see the version of a currently installed package, use the **rpm -q *package*** command. It will return a result as follows: ***package-version-release***.

Note the words in bold italics above — username, domain.name, file-system, package, version and release. Each word is a placeholder, either for text you enter when issuing a command or for text displayed by the system.

Aside from standard usage for presenting the title of a work, italics denotes the first use of a new and important term. For example:

> When the Apache HTTP Server accepts requests, it dispatches child processes or threads to handle them. This group of child processes or threads is known as a *server-pool*. Under Apache HTTP Server 2.0, the responsibility for creating and maintaining these server-pools has been abstracted to a group of modules called *Multi-Processing Modules* (*MPMs*). Unlike other modules, only one module from the MPM group can be loaded by the Apache HTTP Server.

1.2. Pull-quote Conventions

Two, commonly multi-line, data types are set off visually from the surrounding text.

Output sent to a terminal is set in `Mono-spaced Roman` and presented thus:

```
books        Desktop   documentation  drafts  mss    photos   stuff  svn
books_tests  Desktop1  downloads      images  notes  scripts  svgs
```

Source-code listings are also set in `Mono-spaced Roman` but are presented and highlighted as follows:

```
package org.jboss.book.jca.ex1;

import javax.naming.InitialContext;

public class ExClient
{
   public static void main(String args[])
       throws Exception
   {
      InitialContext iniCtx = new InitialContext();
      Object         ref    = iniCtx.lookup("EchoBean");
      EchoHome       home   = (EchoHome) ref;
      Echo           echo   = home.create();
```

```
        System.out.println("Created Echo");

        System.out.println("Echo.echo('Hello') = " + echo.echo("Hello"));
    }

}
```

1.3. Notes and Warnings

Finally, we use three visual styles to draw attention to information that might otherwise be overlooked.

Note

A note is a tip or shortcut or alternative approach to the task at hand. Ignoring a note should have no negative consequences, but you might miss out on a trick that makes your life easier.

Important

Important boxes detail things that are easily missed: configuration changes that only apply to the current session, or services that need restarting before an update will apply. Ignoring Important boxes won't cause data loss but may cause irritation and frustration.

Warning

A Warning should not be ignored. Ignoring warnings will most likely cause data loss.

2. We Need Feedback

If you find a typographical error in the *MRG Realtime MRG Realtime Installation Guide*, or if you have thought of a way to make this manual better, we would love to hear from you! Please submit a report in Bugzilla: *http://bugzilla.redhat.com/bugzilla/* against the product *Red Hat Enterprise MRG* and version *1.1*.

When submitting a bug report, be sure to mention the manual's identifier:

```
Real_Time_Installation_Guide - MRG Realtime Installation Guide
```

If you have a suggestion for improving the documentation, try to be as specific as possible when describing it. If you have found an error, please include the section number and some of the surrounding text so we can find it easily.

Why Use MRG Realtime to Optimize Latency?

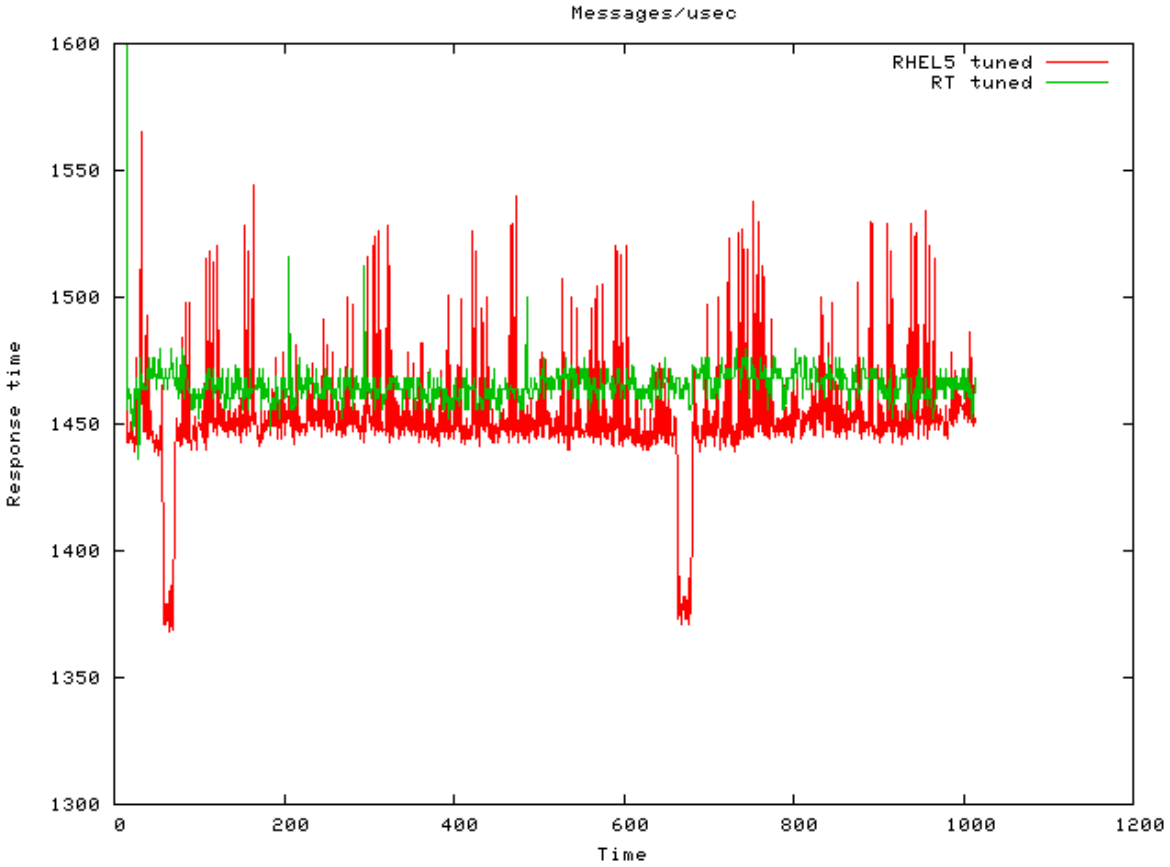
For anyone considering evaluating the performance benefits of the MRG Realtime kernel, it is crucial to understand both the importance of tuning and how to establish the right performance expectations.

MRG Realtime is designed to be used on well-tuned systems for applications with extremely high determinism requirements. Kernel system tuning offers the vast majority of the improvement in determinism. For example, in many workloads thorough system tuning improves consistency of results by around 90%. This is why we typically recommend that customers first perform the suggested system tuning of standard Red Hat Enterprise Linux to see if it meets their objectives, before using MRG Realtime.

System tuning is just as important when using the MRG Realtime kernel as it is for standard Red Hat Enterprise Linux. In fact, if you simply took an untuned system running standard Red Hat Enterprise Linux and substitute the MRG Realtime kernel for the stock kernel supplied as part of the Red Hat Enterprise Linux release, you are unlikely to notice any benefit. Standard tuning will yield 90% of the determinism gains. The MRG Realtime kernel provides the last 10% of determinism required by the most demanding workloads.

Establishing the right performance expectations refers to the fact that the MRG Realtime kernel is not a panacea. Its objective is consistent, low-latency determinism offering predictable response times. There is some additional kernel overhead associated with the MRG Realtime kernel. This is due primarily to handling hardware interrupts in separately scheduled threads. The increased overhead in some workloads results in some degradation in overall throughput. The exact amount is very workload dependent, ranging from 0% to 30%. However, it is the cost of determinism.

For typical workloads with kernel latency requirements in the millisecond (ms) range, then the standard Red Hat Enterprise Linux 5.2 kernel is sufficient. If your workload has stringent low-latency determinism requirements for core kernel features such as interrupt handling and process scheduling in the microsecond (µs) range, then the MRG Realtime kernel is for you.



The red line in this graph represents the system response time (in µs) of a tuned Red Hat Enterprise Linux 5 kernel running a client/server messaging workload. The green line represents the system response time of a tuned MRG Realtime kernel. It is clear from this graph that the response time of the MRG Realtime kernel is very consistent, in contrast to the Red Hat Enterprise Linux 5 case which has greater variability and more spikes.

Download

System Requirements

A prerequisite of installing the MRG Realtime kernel is that the user or administrator first perform a fresh installation of Red Hat Enterprise Linux 5.2. It is important to note, however:

- Red Hat Enterprise Linux 4 or 5.0 update installations are not tested or supported. A full system installation of Red Hat Enterprise Linux 5.2 is required
- A mix of Red Hat Enterprise Linux 4 and 5 components is not supported. You will not be able to drop the MRG Realtime kernel and **`glibc`** into Red Hat Enterprise Linux 4 because of dependencies.
- Red Hat Enterprise Linux 5.0 is not supported for use with MRG Realtime.
- When doing the initial operating system installation, you should select the set of packages that meet the needs of your application environment. Nothing MRG Realtime specific is required on the initial install.

Once you have installed Red Hat Enterprise Linux 5.2 follow the instructions given at *Section 2.2, "Installing MRG Realtime using Yum"* to add the MRG Realtime specific packages from a separate yum repository.

Differences Between MRG Realtime and the Standard Kernel

MRG Realtime differs substantially from the standard Red Hat Enterprise Linux 5.2 kernel. The following list itemizes several differences:

- *Physical Address Extension (PAE) is configured in the x86 MRG Realtime kernel*
  - Red Hat Enterprise Linux 5.2 for x86 (32 bit) includes the following kernel variants (using example version numbers which will vary)
    - kernel-2.6.18-8.el5 - this kernel does not have PAE enabled.
    - kernel-PAE-2.6.18-8.el5 - this kernel does have PAE enabled - allowing it to access more than 4GB of memory
  - In the MRG Realtime x86 kernel PAE is configured as we anticipate that most systems requiring RT capability will have at least 4GB of memory.

Note

Note: the maximum memory supported on x86 systems is 16GB (which is the same limit as standard Red Hat Enterprise Linux 5.2). Larger memory configurations are supported on the x86-64 kernel.

- *Third-party kernel modules are incompatible with standard Red Hat Enterprise Linux 5.2*
  - Kernel modules are inherently specific to the kernel they are built for. Since the MRG Realtime kernel is substantially different from the standard Red Hat Enterprise Linux 5.2 kernel, kernel modules are incompatible. In other words, you can't take third-party driver modules from Red Hat Enterprise Linux 5.2 and use them as-is on MRG Realtime.

- The following are some example third-party drivers which ship for standard Red Hat Enterprise Linux 5.2 which do not currently have a MRG Realtime build:
  - EMC Powerpath
  - NVidia graphics
  - Advanced storage adapter configuration utilities from Qlogic

Important

The user space **`syscall`** interface *is* compatible with standard Red Hat Enterprise Linux 5.2. These compatibility restrictions pertain *only* to kernel modules not supplied by Red Hat.

2.1. MRG Realtime Kernel Variants

There are numerous kernel variants provided. Each variant is simply a version of the MRG Realtime kernel compiled with support for different configuration options. In this case, the variants offer differing diagnostic capabilities. The set of kernels is separately provided for x86 (32-bit) and x86-64 (64-bit) systems. Select either set based on whether you intend to run in 32-bit or 64-bit mode.

The main deployment MRG Realtime kernel is identified below as *Production*. Additionally there are several debug kernels which have progressively more diagnostic code compiled in. The reason for doing this is that as the amount of debug code is increased, so does the overhead. **`kernel-rt-trace`** has less overhead than **`kernel-rt-debug`**.

Finally, the **`Vanilla`** kernel does not include the MRG Realtime features. This is used to help distinguish whether bugs were introduced in the MRG Realtime features, or are inherent bugs in the baseline kernel. See *Section 4.1, "Reporting Bugs"* for more information.

Variant	Intended Usage	Notes
`x86` (kernel-rt)	*Production* - Standard 32-bit production realtime kernel	x86 version with Physical Address Extension (PAE) enabled to allow for up to 16GB memory
`x86 tracing` (kernel-rt-trace)	*Debugging* - 32-bit trace kernel	Latency tracer enabled - Used to locate latency hotspots
`x86 debug` (kernel-rt-debug)	*Debugging* - 32-bit debugging kernel	Includes debugging options, with latency tracer disabled - Used to debug the MRG Realtime kernel
`x86 vanilla` (kernel-rt-vanilla)	*Debugging* - 32-bit base kernel	No MRG Realtime features, used for comparison

Table 2.1. MRG Realtime Kernel Variants for x86 systems

Variant	Intended Usage	Notes
`x86_64` (kernel-rt)	*Production* - Standard 64-bit production kernel	

Variant	Intended Usage	Notes
x86_64 tracing (kernel-rt-tracing)	*Debugging* - 64-bit trace kernel	Latency tracer enabled - Used to locate latency hotspots
x86_64 debug (kernel-rt-debug)	*Debugging* - 64-bit debugging kernel	Includes debugging options, with latency tracer disabled - Used to debug the MRG Realtime kernel
x86_64 vanilla (kernel-rt-vanilla)	*Debugging* - 64-bit base kernel	No MRG Realtime features, used for comparison

Table 2.2. MRG Realtime Kernel Variants for AMD64 and Intel 64 systems

2.2. Installing MRG Realtime using Yum

The best strategy for installing MRG Realtime components is to use **yum**.

Important

Before you install Red Hat Enterprise MRG check that your hardware and platform is supported. A complete list is available on the *Red Hat Enterprise MRG Supported Hardware Page*[1].

1. Use the **yum** command to install the MRG Realtime group:

```
# yum groupinstall "MRG Realtime"
```

   The **MRG Realtime** group installs five packages:

   - **rt-setup** sets up the basic environment required by MRG Realtime.
   - **kernel-rt** is the standard MRG Realtime kernel package.
   - **rtctl** is a startup script that sets the priorities of the various kernel threads.
   - **rtcheck** is a program that tests the running system for MRG Realtime capabilities.
   - **tuna** is a graphical tool used to manage your MRG Realtime application. For information on running Tuna, see the *MRG Realtime Tuning Guide*.

2. You can check the installation location and that the components have been installed successfully by using the **rpm -ql** command.

```
# rpm -ql rt-setup
/etc/security/limits.d/realtime.conf

# rpm -ql kernel-rt
/boot/System.map-2.6.21-43.el5rt
/boot/config-2.6.21-43.el5rt
/boot/vmlinuz-2.6.21-43.el5rt
/lib/modules/2.6.21-43.el5rt
```

```
/lib/modules/2.6.21-43.el5rt/build
/lib/modules/2.6.21-43.el5rt/extra
/lib/modules/2.6.21-43.el5rt/kernel
...
[output truncated]
```

Note

See *Section 4.2, "Further Reading"*, for places to turn for help if you have trouble with installing the MRG Realtime kernel

2.3. Available Packages — RPM

This section lists the RPM packages available in the repository for MRG Realtime.

The MRG Realtime specific column indicates if the RPM differs from the standard Red Hat Enterprise Linux 5.2 maintenance stream, or is not applicable to the standard kernel. A No in this column indicates that the RPM performs equivalently on Red Hat Enterprise Linux 5.2.

The Required column indicates whether or not the package is mandatory for correct MRG Realtime behaviour. A No in this column indicates that usage is optional.

RPM Package Name	Description	MRG Realtime Specific?	Required?
kernel-rt	Low latency and pre-emption functionality	Yes	Yes
rtctl	System start-up script used to configure the default MRG Realtime scheduling priorities of kernel threads	Yes	Yes
dslimit	Shell tool to run a command with a soft CPU limit	No	No

Table 2.3. Basic MRG Realtime Kernel Packages

The following packages contain test programs for use with MRG Realtime.

RPM Package Name	Description
kernel-rt-devel	Headers and libraries for kernel development
kernel-rt-trace	MRG Realtime kernel with tracing functions compiled in
kernel-rt-trace-devel	Headers and libraries for development on trace kernel
kernel-rt-debug	MRG Realtime kernel with debugging functions compiled in (slow)
kernel-rt-debug-devel	Headers and libraries for development on debug kernel

RPM Package Name	Description
kernel-rt-vanilla	Base kernel for comparisons
kernel-rt-vanilla-devel	Headers and libraries for development on vanilla kernel
rt-tests	Utilities for measuring system latencies and for proving that priority-inheritance mutexes function properly.
rt-watchdog	When setting kernel and MRG Realtime application priorities it is possible to incorrectly elevate application processes too high. This can result in starvation of kernel threads - leading to unexpected or hung systems. This is a tool used to detect such hung systems and provide a means to break out and diagnose them
rtcheck	A program that tests the running system for MRG Realtime capabilities.

Table 2.4. MRG Realtime Test Packages

The following set of packages are provided for use with **oprofile**, **systemtap** and the crash utility for analyzing kernel crashdumps. The debugging packages consist of symbol tables and are quite large. For this reason, they are separately delivered from the other MRG Realtime packages.

RPM Package Name	Description
kernel-rt-debuginfo	Symbols for profiling and debugging use, such as **oprofile** or **systemtap**
kernel-rt-trace-debuginfo	Symbols for profiling and tracing
kernel-rt-debug-debuginfo	Symbols for profiling and tracing
kernel-rt-vanilla-debuginfo	Symbols for profiling and tracing

Table 2.5. MRG Realtime Debugging Packages

Important

The packages in *Table 2.4, "MRG Realtime Test Packages"* and *Table 2.5, "MRG Realtime Debugging Packages"* are not essential in order to run MRG Realtime. They are provided as diagnostic tools only and should not be run as a matter of course. To do so will negatively impact performance and could render any benefit from the use of the MRG Realtime kernel negligible.

2.4. Post-Installation Instructions

The MRG Realtime kernel is not automatically specified as the default boot kernel during the installation process. The recommended approach after installing the kernel replacement is to reboot, then manually select the MRG Realtime kernel in the grub menu.

1. Once you know that the MRG Realtime kernel is fully operational on your system you can modify **grub.conf** to make it the default boot kernel. The **grub.conf** file is located in **/boot/grub/grub.conf**. View the file using your preferred text editor. It should look similar to the following:

```
default=1
timeout=5
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
title Red Hat Enterprise Linux (realtime) (2.6.24.7-81.el5rt)
 root (hd0,0)
 kernel /vmlinuz-2.6.24.7-81.el5rt ro root=/dev/Root rhgb quiet
 initrd /initrd-2.6.24.7-81.el5rt.img
title Red Hat Enterprise Linux Client (2.6.24.7-81.el5)
 root (hd0,0)
 kernel /vmlinuz-2.6.24.7-81.el5 ro root=/dev/Root rhgb quiet
 initrd /initrd-2.6.24.7-81.el5.img
```

2. In the example, the MRG Realtime kernel is listed first as **(hd0,0)**. Change the value of **default=** to *0* as follows and save your changes.

```
default=0
timeout=5
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
title Red Hat Enterprise Linux (realtime) (2.6.24.7-81.el5rt)
 root (hd0,0)
 kernel /vmlinuz-2.6.21-43.el5rt ro root=/dev/Root rhgb quiet
 initrd /initrd-2.6.21-43.el5rt.img
title Red Hat Enterprise Linux Server (2.6.24.7-81.el5)
 root (hd0,0)
 kernel /vmlinuz-2.6.18-53.el5 ro root=/dev/Root rhgb quiet
 initrd /initrd-2.6.18-53.el5.img
```

3. You will also need to edit **/etc/sysconfig/kernel**. This will ensure that the changes you made to the grub file will remain as the default when you perform a system upgrade. To edit this file, open it in any text editor. It should look similar to this:

```
# UPDATEDEFAULT specifies if new-kernel-pkg should make
# new kernels the default
UPDATEDEFAULT=yes

# DEFAULTKERNEL specifies the default kernel package type
DEFAULTKERNEL=kernel
```

4. Simply change the **DEFAULTKERNEL=** parameter to read *kernel-rt*.

```
# UPDATEDEFAULT specifies if new-kernel-pkg should make
# new kernels the default
UPDATEDEFAULT=yes
```

```
# DEFAULTKERNEL specifies the default kernel package type
DEFAULTKERNEL=kernel-rt
```

5. You will now be able to confirm that your system is running the MRG Realtime kernel, by running the **uname** command as the root user at the shell prompt. Check the output for the *RT* designation. If it appears, the MRG Realtime kernel is running.

```
# uname -a
Linux server01 2.6.21-43.el5rt #1 SMP PREEMPT RT Tue Oct 16 11:05:05
 EDT 2007 x86_64 x86_64 x86_64 GNU/Linux
```

Note

MRG Realtime can be configured to provide crash dump information by enabling **kexec/kdump**. Further information and instructions on how to configure your system to obtain kernel crash information can be found in the *MRG Realtime Tuning Guide*.

MRG Realtime Kernel Tuning

The MRG Realtime kernel offers many performance tuning parameters not otherwise available in Red Hat Enterprise Linux 5.2. In order to achieve optimal low-latency determinism it is necessary to perform MRG Realtime specific system tuning.

Note

For comprehensive tuning information, see the *MRG Realtime Tuning Guide*.

Tuna

The primary diagnostic facility provided with the MRG Realtime kernel is Tuna. Tuna provides both a command line tool and a graphical interface that can be used to change attributes of threads (scheduling policy, scheduler priority and processor affinity) and interrupts (processor affinity). The tool is designed to be used on a running system, and changes take place immediately. This allows any application-specific measurement tools to see and analyze system performance immediately after the changes have been made.

Latency Tracer

One of the diagnostic facilities provided with the MRG Realtime kernel is the latency tracer. The latency tracer is a peak detector which is used to identify the longest running non-preemptable kernel codepaths. This is particularly useful for identifying whether non-deterministic performance results are attributable to the kernel or to user space components. In customer deployments, the tool is most useful to differentiate whether delays are in the kernel or the application.

Direct Memory Access with `rmem.ko`

MRG Realtime includes a kernel module called **`rmem.ko`**. This module is not loaded by default, and is provided only to meet the realtime Java conformance tests - specifically the Technology Compatability Kit (TCK) test in the Real Time Specification for Java's (*RTSJ*[1]) conformance suite. It is an RTSJ requirement of this conformance test that Java programs have direct access to physical memory. This **`/dev/rmem`** capability allows user applications to map any arbitrary memory region.

The capability can only be enabled as a result of direct root system administrator action. To further prevent accidental usage, the capability is disabled unless *`unprotected_address_spaces=1`* is specified as a boot option in the **`grub.conf`** file. If the **`rmem.ko`** kernel module is explicitly loaded, the kernel will have a *`tainted`* flag. In this case, Red Hat Global Support may require you to reproduce any problems without the presence of **`rmem.ko`**.

Warning

Enabling **`rmem.ko`** gives any application direct access to physical memory. Many normal security mechanisms are bypassed and the system becomes much more vulnerable to attacks by malicious users. For this reason, it is strongly suggested that this feature be used *exclusively* for RTSJ certification purposes only. *DO NOT* use the **`rmem.ko`** module for production deployment.

[1] http://www.rtsj.org/

There is another related boot option for this feature. This boot option reserves contiguous physical kernel memory at boot time for later usage by Java runtime. This memory is allocated at boot prior to the system memory becoming fragmented. The purpose of this is to avoid allocation failures for large contiguous memory. The line that would need to be added to the **`grub.conf`** file is the following, where the *`memsize`* parameter is expressed in bytes.

```
alloc_rtsj_mem.size=memsize
```

More Information

4.1. Reporting Bugs

Important

An up-to-date listing of known issues can be found on the *MRG Realtime Wiki: Known Bugs*[1]. Always check this list before reporting a new bug.

Diagnosing a Bug

Before you a file a bug report, follow these steps to diagnose where the problem has been introduced. This will greatly assist in rectifying the problem.

1. Try reproducing the problem with the standard kernel. Check that you have the latest version of the Red Hat Enterprise Linux 5.2 kernel, then boot into it from the grub menu. Try reproducing the problem. If the problem still occurs with the standard kernel, report a bug against Red Hat Enterprise Linux 5.2 *NOT* MRG Realtime.

2. If the problem does not occur when using the standard kernel, then the bug is probably the result of changes introduced in either:

   a. The upstream kernel on which MRG Realtime is based. For example, Red Hat Enterprise Linux 5.2 is based on 2.6.18 and MRG Realtime is based on 2.6.24.7

   b. MRG Realtime specific enhancements Red Hat has applied on top of the baseline (2.6.24.7) kernel

   To determine the problem, it is helpful to see if you can reproduce the problem on an unmodified upstream 2.6.24.7 kernel. For this reason, in addition to providing the MRG Realtime kernel, we also provide a **`vanilla`** kernel variant. The **`vanilla`** kernel is the unmodified upstream kernel build without the MRG Realtime additions.

Reporting a Bug

If you have determined that the bug is specific to MRG Realtime follow these instructions to enter a bug report:

1. You will need a *Bugzilla*[2] account. You can create one at *Create Bugzilla Account*[3].

2. Once you have a Bugzilla account, log in and click on *Enter A New Bug Report*[4].

3. You will need to identify the product the bug occurs in. MRG Realtime appears under Red Hat Enterprise MRG in the Red Hat products list. It is important that you choose the correct product that the bug occurs in.

4. Continue to enter the bug information by designating the appropriate component and giving a detailed problem description. When entering the problem description be sure to include details of whether you were able to reproduce the problem on the standard Red Hat Enterprise Linux 5.2 or the supplied **`vanilla`** kernel.

4.2. Further Reading

- Red Hat Enterprise MRG Product Information
  - *http://www.redhat.com/mrg*
- MRG Realtime Tuning Guide and other Red Hat Enterprise MRG documentation
  - *http://redhat.com/docs/en-US/Red_Hat_Enterprise_MRG*
- Mailing List
  - To post to the list, send mail to `rhemrg-users-list@redhat.com`
  - Subscribe to the mailing list at: *http://post-office.corp.redhat.com/mailman/listinfo/rhemrg-users-list*

Appendix A. Revision History

Revision History

Revision 0.3 Fri Nov 21 2008 LanaBrindley*lbrindle*

Minor updates prior to releasing document to Quality Engineering

Revision 0.2 Thu Oct 30 2008 LanaBrindley*lbrindle*

Minor updates and changes in preparation for technical review

Revision 0.1 Thu Oct 30 2008 LanaBrindley*lbrindle*

Ported information from Deployment Guide to Installation Guide