Red Hat Enterprise Linux 7
电源管理指南

管理 Red Hat Enterprise Linux 7 的电源消耗

作者：Jacquelynn East 作者：Don Domingo
作者：Rüdiger Landmann
作者：Jack Reed
翻译、校对：潘陈斯梦 – Chensimeng (April) Pan
校对、责任编辑：鄭中 – Chester Cheng

Red Hat Enterprise Linux 7 电源管理指南

管理 Red Hat Enterprise Linux 7 的电源消耗

作者：Jacquelynn East
红帽公司　出版部

作者：Don Domingo
红帽公司　出版部

作者：Rüdiger Landmann
红帽公司　出版部

作者：Jack Reed
红帽公司　出版部

翻译、校对：潘陈斯梦 – Chensimeng (April) Pan
澳大利亚昆士兰大学 笔译暨口译研究生院

校对、责任编辑：鄭中 – Chester Cheng
红帽全球服务部 & 澳大利亚昆士兰大学笔译暨口译研究生院
ccheng@redhat.com, uqcchun1@uq.edu.au

法律通告

摘要

本文档解释了如何在Red Hat Enterprise Linux 7 系统中有效管理您的电源消耗。下面的部分讨论了降低电源消耗的不同技术（服务器和笔记本电脑），以及每个技术对您系统总体性能的影响。

目录

第 1 章 概述

电源管理是 Red Hat Enterprise Linux 7 的改进重点之一。“*绿色 IT*”（环境友好型运算）包含一系列的考量，限制电脑系统耗电是绿色 IT 最重要的方面之一，除此之外还包括可回收材料的使用、硬件制造对环境的影响、以及系统设计和开发中的环保意识。本文件提供了 Red Hat Enterprise Linux 7 中有关电源管理的指南和信息。

1.1. 电源管理的重要性

电源管理的核心是了解如何有效优化每个系统组件的电量消耗。这需要对系统运行的不同任务进行研究，并配置每个组件以确保其性能适用于该任务。

进行电源管理的主要动机是：

- 减少电源总消耗，以便节省成本

正确使用电源管理可产生如下结果：

- 降低服务器和运算中心的温度
- 降低二次成本，包括冷却、空间、电缆、发电机以及“*不间断供电*（UPS）”产生的成本
- 延长笔记本电脑的电池寿命
- 减少二氧化碳排放
- 符合政府法律法规中对绿色 IT 的要求，例如：能耗星级（Energy Star）
- 符合公司对新系统的要求

通常，降低特定组件（或者整个系统）的电量消耗将降低散热，当然也将会降低性能。因此，您应该彻底地研究和测试您进行的任何配置所产生的性能降低，特别是对关键任务系统的配置所产生的性能降低。

通过研究系统执行的不同任务以及每个组件的配置，确认其性能刚好满足该任务的需要，这样就可以节省能源，减少散热，并延长笔记本电脑电池的使用寿命。很多关于电源消耗的系统分析和微调的原则，都与针对性能微调的原则相似。从某种程度来说，电源管理和性能微调是从对立方向进行系统配置，因为系统通常是根据性能或者电源进行优化的。本手册阐述了 Red Hat 提供的工具以及我们开发的技术，来帮助您进行电源管理。

Red Hat Enterprise Linux 7 有了一些默认启用的电源管理新功能。这些特选的功能不会影响特定服务器或者台式电脑的性能。然而，对一些非常具体的情况，比如要求最大吞吐量、最低延迟时间、最高 CPU 性能的情况下，检查默认值设定可能是有必要的。

如需决定您是否应该使用本文件所阐述的技术优化您的电脑，请先问自己几个问题：

问：

我必须优化吗？

答： 电源优化的重要性取决于您的公司是否有要遵循的准则，或者是否有您必须执行的规定。

问：

我要优化到什么程度？

答： 我们提供的几项技术并不需要您对机器进行详细审核和分析，而是提供一组一般性优化方式，来改进电源使用方式。当然它们对系统的优化不如手动审核、优化的系统，但提供了一个好的折衷方案。

问：

优化是否会将系统性能降低到无法接受的程度？

答： 本文件描述的技术，大多数都会对系统性能产生明显的影响。如果您选择使用 Red Hat Enterprise Linux 7 默认值以外的电源管理方式，您应该在电源优化之后对系统性能进行监控，来决定性能损失是否在可接受的范围内。

问：

与消耗的时间和资源相比，优化的结果是否值得？

答： 就消耗的时间和费用来说，按照完整的流程手动优化一个系统通常并不值得，因为这些成本远远高于您在单一系统的生命周期中所能获得的好处。另一方面，如果您在办公室使用 10,000 个配置以设置相同的电脑系统，那么，生成一个优化的设置并应用到所有机器上就是事半功倍了。

下面的章节将解释优化后的硬件性能如何使系统减少电量消耗。

1.2. 电源管理基础

有效电源管理是建立在以下原则上的：

只在需要时唤醒闲置的 CPU

在 Red Hat Enterprise Linux 6 及其之后的版本中，kernel 会运行 "*无计时（tickless）*"，这意味着之前的定期计时中断被替换为按需求中断。因此，在新任务加入队列之前，闲置 CPU 可以一直保持闲置状态，同时 CPU 也可以在省电模式下保持更长时间。然而，如果系统有应用程序会产生不必要的计时事件，这一功能的好处就会被抵消。查询事件，例如检查磁盘卷册的变动或者鼠标活动，是很好的例子。

Red Hat Enterprise Linux 7 提供了一些工具，使用这些工具将能根据 CPU 的使用量对应用程序进行识别和编辑。详情请查看〈*第 2 章 电源管理审核和分析*〉。

完全禁用不使用的硬件和设备

此原则对于带有移动组件的设备（比如硬盘）尤为重要。另外，有些应用程序可能会使未使用但被激活的设备处于开启（open）状态；出现这种情况时，kernel 会假定该设备处于使用状态，这样就会阻止设备进入节电状态。

低活性等于低瓦数

大多数情况下，此原则的实现需要依靠较新的硬件和正确的 BIOS 配置。旧的系统组件经常不支持 Red Hat Enterprise Linux 7 中支持的新功能。请确定您系统使用的是最新的官方固件，且在 BIOS 中的电源管理或者设备配置部分启用了电源管理功能。您需要关注的功能包括：

- SpeedStep
- PowerNow!
- Cool'n'Quiet
- ACPI （C 状态）
- Smart

如果您的硬件支持这些功能，且在 BIOS 中启用，Red Hat Enterprise Linux 7 将默认使用这些功能。

CPU 状态的不同形式及其效果

现代 CPU 与“*高级配置和电源接口*”（Advanced Configuration and Power Interface，ACPI）共同提供不同的电源状态。三种不同的状态是：

- 睡眠（C 状态）
- 频率（P 状态）
- 热输出（T 状态或者 "热状态"）

以最低休眠状态运行的 CPU 可能会消耗最少的电力，但是若要将其从该状态唤醒，也需要相对较长的时间。在极少数情况下，这会导致 CPU 需要在刚刚进入休眠状态后就马上就要被唤醒。这种情况导致 CPU 一直处于忙碌状态，并在已经使用另一种状态时可能无法节电。

关机后，电量使用量最低

很明显，最佳的节电方法就是关闭系统。例如：您的公司可以发展出注重 “绿色 IT” 的企业文化，让员工有意识地在午休时间或者回家前关闭电脑。您还可以将几台实体服务器合并成一个较大的服务器，并使用我们在 Red Hat Enterprise Linux 7 中附带的虚拟化技术将其虚拟化。

第 2 章 电源管理审核和分析

2.1. 审核与分析概述

我们通常不会对单一系统进行详细的手动审核、分析以及微调，因为其花费的时间和成本一般都会超过系统微调所带来的好处。但是，在大量基本一致的系统上，可以为所有系统重复使用相同的设置，这种情况下执行这些任务就是非常有用的。例如：一次部署成千上万个桌面系统，或者机器几乎完全一样的 HPC 群集。另一个执行审核和分析的理由是提供进行比较的基准，在将来用它来识别系统行为退化或者改变。在进行硬件、BIOS 或者软件常规更新，同时还要避免电力消耗异常时，这些分析结果非常有用。通常来说，完整的审核和分析可让您对系统的情况有更好的了解。

审核和分析系统的电力消耗是相当困难的，即使在最先进的系统上也是这样。大多数系统没有提供用于测量电力消耗的软件。不过还是有例外： Hewlett Packard 服务器系统的 ILO 管理控制台有提供电源管理模块，使用者可以通过网页存取。IBM 在 BladeCenter 电源管理模块中提供了类似的解决方案。在一些 Dell 系统上，“IT 助手”提供了电源监控的功能。其他厂商可能也会为他们的服务器平台提供类似的功能。由此可以看出，没有一种解决方案适用于所有的产品。

通常只在尽最大可能节约能耗时，才需要直接测量电源消耗。幸运的是，还有其他方法可度量改变是否有效，以及了解系统的行为方式。本章介绍了一些必需的工具。

2.2. PowerTOP

无计时 kernel 使得 CPU 能够更常进入闲置状态，以此减少电量消耗、改善电源管理。新 **PowerTOP** 工具会识别经常唤醒 CPU 的特定 kernel 组件和使用者空间应用程序。

Red Hat Enterprise Linux 7 提供了 2.x 版本的 **PowerTOP**。此版本完全重写了 1.x 版本的基本代码。此版本的使用者界面以标签页为基础并且更加清晰，还广泛使用了 kernel 的“perf”框架来提供更准确的数据。系统装置的电源操作将会被追踪并且被明显地显示出来，这样一来就能快速找出问题。更具有实验性的是，2.x 版本的基本代码包括了一个电源估算引擎，能够显示各个装置和进程消耗的电量。请参阅图 2.1 “PowerTOP 操作画面”。

若要安装 **PowerTOP** ， 请以 **root** 身份执行下列命令:

```
yum install powertop
```

若要运行 **PowerTOP**，请以 **root** 身份执行下列命令：

```
powertop
```

PowerTOP 能够对系统的电源使用总量进行估算，并显示每个进程、装置、 kernel 工作、计时器以及中断处理程序的耗电量。笔记本电脑在执行这项任务时应该使用电池电源。如果要校准电源估算引擎，请以 **root** 身份执行下列指令：

```
powertop --calibrate
```

校准需要时间。这项程序会执行很多测试，还将进行屏幕亮度的循环测试，并将装置开启和关闭。校准期间请勿操作机器。 校准程序完成后， **PowerTOP** 将正常启动。请让它运行约一小时以搜集数据。搜集到足够的数据时，电量估算数据将会显示在第一列。

如果在笔记本电脑上执行这项命令，请使用电池电源，以便得到所有的数据。

当它运行时，**PowerTOP** 会从系统搜集数据。在“**概览**”标签页，您可以查看最常唤醒 CPU 或者耗电最多的元件列表 （请参阅图 2.1 “PowerTOP 操作画面”）。相邻的信息栏显示了电源估算、资源使用情况、每秒唤醒次数、元件类别（比如进程、设备或者计时器）以及元件的描述。每秒唤醒次数表明 kernel 的服务或者装

置和驱动的效率有多高。唤醒次数越少意味着消耗的电量越低。元件会根据电源使用量能够被优化的程度进行排列。

调试驱动元件通常需要对 kernel 进行改变，这不在本指南的讨论范畴之内。然而，管理传送唤醒信号的使用者空间进程较为容易。首先，请判断服务或应用是否需要在系统上完全运行。如果不需要，请将它关闭。若要永久停用旧的 System V 服务，请运行：

```
systemctl disable servicename.service
```

若要了解关于此程序的详细信息，请以 **root** 身份运行以下指令：

```
ps -awux | grep processname
strace -p processid
```

如果追踪记录显示它正在重复执行，它可能是一个忙碌的循环。修正这种错误通常需要修改元件中的代码。

正如图 2.1 “PowerTOP 操作画面” 所述，电源消耗总量和电池剩余电量（若存在）将会被显示。下面是一个简短的概要，包含了每秒唤醒次数总量、每秒 GPU 操作量，以及每秒虚拟文件系统操作量。屏幕剩下的部分是一个包含进程、中断、设备和其它资源的列表，它们是根据使用量排列的。若经过正确校准，每一个列出的项目的电量消耗估算值也会显示在第一栏。

使用 **Tab** 和 **Shift+Tab** 按键来循环浏览标签页。在 **Idle stats** 页面，会显示所有处理器和核心的 C 状态的使用情况。在 **Frequency stats** 页面，会显示所有处理器和核心的 P-states 使用情况，包括 Turbo 模式（若可用）。CPU 处于 C 状态或者 P 状态的时间越长越好（**C4** 比 **C3**更高）。这能很好的显示 CPU 使用量的优化程度如何。当系统闲置时，C 状态或者 P 状态的停留时间在理想状况下应该是 90% 或者更高。

Device Stats 页面提供了和**概览**页面类似的信息，但它只提供和装置有关的信息。

Tunables 页面包含了关于优化系统以降低电量消耗的建议。请使用 **up** 和 **down** 键浏览建议，使用 **enter** 键将建议切换为开启或关闭。

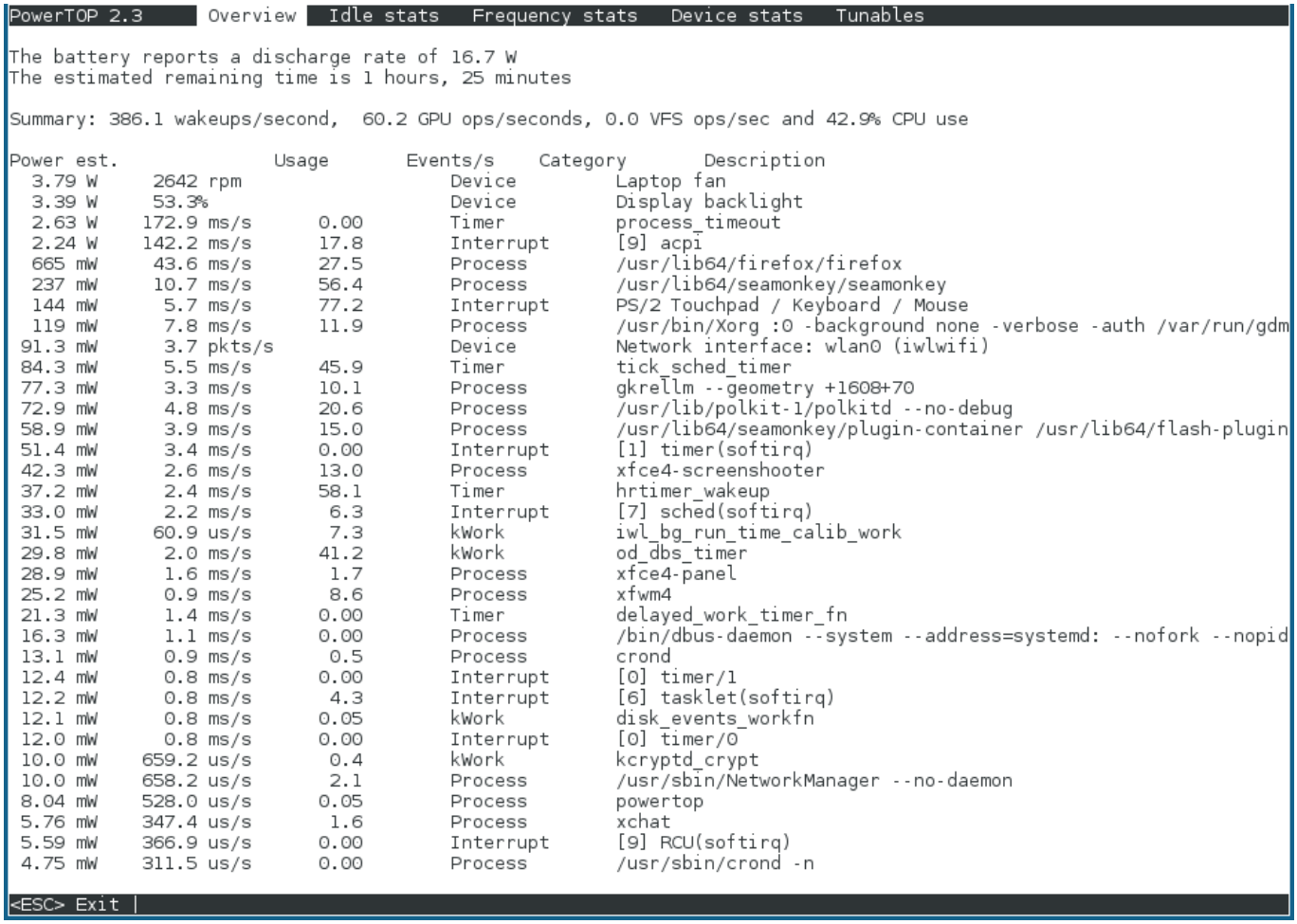

图 2.1. PowerTOP 操作画面

你还可以通过运行 **PowerTOP** 并使用 **--html** 选项来生成 HTML 报告。将 *htmlfile.html* 参数替换为您想要的输出文件名称 。

```
powertop --html=htmlfile.html
```

PowerTOP 默认每 20 秒进行测量，可通过 **--time** 选项改变这一设置：

```
powertop --html=htmlfile.html --time=seconds
```

欲知更多有关 PowerTOP 的信息，请参阅 PowerTOP 主页。

PowerTOP 还可以和 **turbostat** 实用工具搭配使用。 **turbostat** 实用工具是一种报告工具，它能显示 Intel 64 处理器上有关处理器拓扑、频率、闲置状态统计、温度和电源使用状况的信息。欲知更多有关 **turbostat** 实用工具的信息，请参阅 **turbostat**(8) man page，或参阅《性能微调指南》。

2.3. Diskdevstat 和 Netdevstat

Diskdevstat 和 **netdevstat** 都属于 **SystemTap** 工具，它们的功能是搜集系统上运行的所有应用程序的磁盘和网络活动的详细信息。这些工具的灵感来自于 **PowerTOP**，PowerTOP 可显示每个应用程序每秒唤醒 CPU 的次数（请参阅〈第 2.2 节 “PowerTOP”〉）。这些工具收集的统计数据可让您识别那些使用大量的小型 I/O 操作的应用程序，这种应用程序比少量的较大操作更耗电。其它的监控工具只测量传输速度，无法帮助分辨此类使用量。

使用 **SystemTap** 来安装这些工具，请以 **root** 身份执行以下指令：

```
yum install tuned-utils-systemtap kernel-debuginfo
```

使用以下命令运行这些工具：

```
diskdevstat
```

或使用以下命令运行这些工具：

```
netdevstat
```

这两个命令都可以接受三个参数，分别为：

diskdevstat ***update_interval total_duration display_histogram***

netdevstat ***update_interval total_duration display_histogram***

update_interval

显示更新的时间间隔，以秒为单位。默认值：**5**

total_duration

整体运行时间，以秒为单位。默认值：**86400**（一天）

display_histogram

是否在运行结束时将收集的所有数据以柱形图显示的标志。

输出结果和 **PowerTOP** 的结果类似。以下是在使用 KDE 4.2 的 Fedora 10 系统上运行 **diskdevstat** 的输出结果示例：

```
  PID   UID DEV     WRITE_CNT WRITE_MIN WRITE_MAX WRITE_AVG    READ_CNT
READ_MIN  READ_MAX  READ_AVG COMMAND
 2789  2903 sda1          854     0.000   120.000    39.836           0
0.000     0.000     0.000 plasma
15494     0 sda1            0     0.000     0.000     0.000         758
0.000     0.012     0.000 0logwatch
15520     0 sda1            0     0.000     0.000     0.000         140
0.000     0.009     0.000 perl
15549     0 sda1            0     0.000     0.000     0.000         140
0.000     0.009     0.000 perl
15585     0 sda1            0     0.000     0.000     0.000         108
0.001     0.002     0.000 perl
 2573     0 sda1           63     0.033  3600.015   515.226           0
0.000     0.000     0.000 auditd
15429     0 sda1            0     0.000     0.000     0.000          62
0.009     0.009     0.000 crond
15379     0 sda1            0     0.000     0.000     0.000          62
0.008     0.008     0.000 crond
15473     0 sda1            0     0.000     0.000     0.000          62
0.008     0.008     0.000 crond
15415     0 sda1            0     0.000     0.000     0.000          62
0.008     0.008     0.000 crond
15433     0 sda1            0     0.000     0.000     0.000          62
0.008     0.008     0.000 crond
15425     0 sda1            0     0.000     0.000     0.000          62
```

```
0.007     0.007     0.000 crond
15375     0 sda1            0     0.000     0.000     0.000          62
0.008     0.008     0.000 crond
15477     0 sda1            0     0.000     0.000     0.000          62
0.007     0.007     0.000 crond
15469     0 sda1            0     0.000     0.000     0.000          62
0.007     0.007     0.000 crond
15419     0 sda1            0     0.000     0.000     0.000          62
0.008     0.008     0.000 crond
15481     0 sda1            0     0.000     0.000     0.000          61
0.000     0.001     0.000 crond
15355     0 sda1            0     0.000     0.000     0.000          37
0.000     0.014     0.001 laptop_mode
 2153     0 sda1           26     0.003  3600.029  1290.730           0
0.000     0.000     0.000 rsyslogd
15575     0 sda1            0     0.000     0.000     0.000          16
0.000     0.000     0.000 cat
15581     0 sda1            0     0.000     0.000     0.000          12
0.001     0.002     0.000 perl
15582     0 sda1            0     0.000     0.000     0.000          12
0.001     0.002     0.000 perl
15579     0 sda1            0     0.000     0.000     0.000          12
0.000     0.001     0.000 perl
15580     0 sda1            0     0.000     0.000     0.000          12
0.001     0.001     0.000 perl
15354     0 sda1            0     0.000     0.000     0.000          12
0.000     0.170     0.014 sh
15584     0 sda1            0     0.000     0.000     0.000          12
0.001     0.002     0.000 perl
15548     0 sda1            0     0.000     0.000     0.000          12
0.001     0.014     0.001 perl
15577     0 sda1            0     0.000     0.000     0.000          12
0.001     0.003     0.000 perl
15519     0 sda1            0     0.000     0.000     0.000          12
0.001     0.005     0.000 perl
15578     0 sda1            0     0.000     0.000     0.000          12
0.001     0.001     0.000 perl
15583     0 sda1            0     0.000     0.000     0.000          12
0.001     0.001     0.000 perl
15547     0 sda1            0     0.000     0.000     0.000          11
0.000     0.002     0.000 perl
15576     0 sda1            0     0.000     0.000     0.000          11
0.001     0.001     0.000 perl
15518     0 sda1            0     0.000     0.000     0.000          11
0.000     0.001     0.000 perl
15354     0 sda1            0     0.000     0.000     0.000          10
0.053     0.053     0.005 lm_lid.sh
```

这些列代表：

PID

应用程序的进程 ID

UID

运行中的应用程序用户 ID

DEV

发生 I/O 的装置

WRITE_CNT

写入操作总数

WRITE_MIN

两次连续写入所需最短时间（以秒为单位）

WRITE_MAX

两次连续写入所需最长时间（以秒为单位）

WRITE_AVG

两个连续写入操作所需平均时间（以秒为单位）

READ_CNT

读取操作总数

READ_MIN

两次连续读取所需最短时间（以秒为单位）

READ_MAX

两次连续读取所需最长时间（以秒为单位）

READ_AVG

两次连续读取所需平均时间（以秒为单位）

COMMAND

进程名称

在这个示例中，可看到三个非常明显的应用程序：

```
  PID   UID DEV     WRITE_CNT WRITE_MIN WRITE_MAX WRITE_AVG    READ_CNT
READ_MIN  READ_MAX  READ_AVG COMMAND
 2789  2903 sda1          854     0.000   120.000    39.836           0
0.000     0.000     0.000 plasma
 2573     0 sda1            63     0.033  3600.015   515.226           0
0.000     0.000     0.000 auditd
 2153     0 sda1            26     0.003  3600.029  1290.730           0
0.000     0.000     0.000 rsyslogd
```

这三个应用程序的 **WRITE_CNT** 都大于 **0**，这意味着它们在测量期间都执行了一些写入操作。其中，**plasma** 最严重：它执行的写入操作最多，当然写入操作平均间隔时间就最短。如果您想找到最耗电的应用程序，那么 **Plasma** 就是最佳的观察对象。

请使用 **strace** 和 **ltrace** 命令，通过追踪所有给定进程 ID 的系统调用，对应用程序进行进一步检查。在这个示例中，您可以运行：

```
strace -p 2789
```

在这个示例中，**strace** 的输出结果中包含一个每 45 秒重复一次的模式，该模式会打开用户的 KDE 图标缓冲文件，接着写入，然后马上关闭该文件。这导致必需在硬盘上进行物理写入，因为文件原数据已经改变了（更准确地说是时间被修改了）。最终修复是为了避免在没有图标更新时进行不必要的调用。

2.4. 电池寿命工具组件

Red Hat Enterprise Linux 7 添加了**电池寿命工具组件**（Battery Life Tool Kit，简称 BLTK），这是一套用来模仿、分析电池寿命和性能的测试程序。BLTK 通过执行一系列模拟特定用户组的任务，并将结果形成报告，来达到这一目的。尽管 BLTK 是为检测笔记本电脑的性能设计的，如果启动时使用 **-a** 选项，BLTK 也能报告台式电脑的性能。

BLTK 能生成和真实的计算机使用中非常相似的工作负荷。例如：**office** 负荷写入一个文本，再加以校正，并将同样的程序应用到电子表格中。运行带 **PowerTOP** 或者任何其它审核或分析工具的 BLTK，就可以测试当计算机处于活跃状态，而不是仅仅处于闲置状态时，您的优化是否有效。因为您可以在不同设置中多次运行同样的负荷，并比较不同设置下得到的结果。

使用以下命令安装 BLTK：

```
yum install bltk
```

使用以下命令运行 BLTK：

```
bltk workload options
```

例如：运行 **idle**（闲置）负荷 120 秒：

```
bltk -I -T 120
```

默认可用的工作负荷有：

-I, **--idle**

系统闲置，将其作为与其它负载进行比较的基准

-R, **--reader**

模拟读取文件（默认使用 **Firefox**）

-P, **--player**

模拟观看 CD 或者 DVD 驱动器中的多媒体文件（默认使用 **mplayer**）

-O, **--office**

使用 **OpenOffice.org** 套件模拟编辑文件

其它可让您指定的选项：

-a, **--ac-ignore**

忽略 AC 电源是否可用（台式计算机需使用）

-T *number_of_seconds*, **--time** *number_of_seconds*

测试时间（以秒为单位），此选项需与 **idle** 负荷一起使用

-F *filename*, **--file** *filename*

指定特定负荷使用的文件，例如不访问 CD 或者 DVD 驱动器，而是指定别的文件供 **player** 负荷使用

-W *application*, --prog *application*

指定特定负载使用的应用程序，例如：不使用 **Firefox** ，而是指定其它的浏览器供 **reader** 负载使用

BLTK 支持大量更具体的选项。详情请参考 **bltk** 的 man page。

BLTK 会将产生的结果保存在 **/etc/bltk.conf** 配置文件指定的目录中，默认为 **~/.bltk/*workload*.results.*number*/**。例如：**~/.bltk/reader.results.002/** 目录中会保存第三次 **reader** 负荷测试的结果（第一次不计数）。所有的结果会分散存在几个文本文件中。要将这些结果压缩成方便读取的格式，请运行：

```
bltk_report path_to_results_directory
```

结果会显示在结果目录下的 **Report** 文件中。如需在终端模拟器中查看结果，请使用 **-o** 选项：

```
bltk_report -o path_to_results_directory
```

2.5. Tuned

tuned 是一项守护程序，它会使用 udev 来监控联网装置，并且根据选择的配置文件对系统设置进行静态和动态的微调。它有许多为常见使用案例（例如高吞吐量、低延迟或者节电）的预定义配置文件，并且允许用户更改为每个配置文件定义的规则，还可以自定义如何对一个特定的设备进行微调。若要通过某个配置文件还原系统设置的所有更改，您可以切换到另一个配置文件，或者停用 **tuned** 守护程序。

静态微调主要包括预定义的 **sysctl** 和 **sysfs** 设置和对几种配置工具的单次激活，例如 **ethtool**。**tuned** 还会监控系统组件的使用状况，并根据监控的信息动态地微调信息系统设置。动态微调使得在任何给定系统的运行时间内，不同的系统组件能够以不同的方式被使用。例如，在启动和登录过程中会大量使用硬盘驱动器，但是之后用户可能主要使用类似网页浏览器或者电子邮件客户端这类的应用程序，这种情况下就几乎不会使用硬盘驱动器。类似地，不同的时间对 CPU 和网络设备的使用是不同的。**Tuned** 会监控这些组件的活动，并且对它们在使用过程中出现的改变作出反应。

注意

动态微调在 Red Hat Enterprise Linux 中被全局禁用，若要启用，请编辑 **/etc/tuned/tuned-main.conf** 文件， 将 **dynamic_tuning** 修改为 **1**。

以典型的办公室工作站为例。大多数时间里，以太网网络接口是非常不活跃的。这段时间内可能只会偶尔接收和发送一些电子邮件，或者载入一些网页。对这类型的负载，网络接口并不需要一直按默认设置全速运行。**tuned** 有一个针对网络设备的监控和微调插件，能够检测这种低活跃度，然后自动降低该接口的速度，这样通常能够降低电源消耗。如果接口的活跃度在较长的时间内增加（比如因为正在下载一个 DVD 映像，或者打开一个有大附件的电子邮件），**tuned** 会检测到这种情况并将接口速度设置为最大，以便在活跃等级很高的时候提供最佳的性能。此原则也适用于 CPU 和硬盘的其它插件。

2.5.1. 插件

tuned 通常使用两种插件：“*监控插件*”和“*微调插件*”。监控插件是用来获取运行中的系统的信息。目前使用的是下列监控插件：

disk

获取每个设备在每个测量间隔的磁盘负载（ IO 操作的数量）。

net

获取每个网卡在每个测量间隔的网络负载（已传输数据包的数量）。

load

获取每个 CPU 在每个测量间隔的 CPU 负载。

用于动态微调的微调插件可以使用监控插件的输出结果。目前使用的动态微调算法试图在性能和节能之间找到平衡，因此它在性能配置文件中是被禁用的（对单项插件的动态微调能够在 **tuned** 配置文件中启用或禁用）。若任何启用的微调插件需要监控插件的指标，监控插件将会自动实例化。如果两个微调插件需要相同的数据，监控插件只会生成一个实例，两个微调插件共享该数据。

每个微调插件会对一个单独的子系统进行调整，并且获得从 **tuned** 配置文件中导出的参数。每个子系统可以有多个由微调插件的单独实例操作的设备（比如多个 CPU 或者网卡）。同样支持为单独设备进行特定设置。提供的配置文件使用通配符来将所有的设备和独立子系统配对（欲知如何更改，请参阅〈第 2.5.3 节 “自定义配置文件”〉），这使得插件能够根据需要的目标（选定的文件）对子系统进行微调，用户只需要选择正确的 **tuned** 配置文件（欲知如何选择配置文件或获取提供的配置文件列表，请参阅〈第 2.5 节 “Tuned”〉和〈第 2.5.4 节 “Tuned-adm”〉）。目前主要使用下列微调插件（这些插件中只有一部分是用于动态微调，插件所支持的参数同样也已列出）：

cpu

通过 ***governor*** 参数将 CPU 调控器设置为指定值，并且根据 CPU 负载动态地改变 PM QoS CPU DMA 延迟。如果 CPU 负载低于 ***load_threshold*** 参数设定的指定值，延迟便会根据 ***latency_high*** 参数设定的指定值被设置，不然它就会由 ***latency_low*** 参数设定的指定值设置。延迟同样可以被设为一个固定的值，而不被动态更改。这可以通过将 ***force_latency*** 参数设置为需要的延迟值办到。

eeepc_she

根据 CPU 负载动态设置 FSB 速度。一些上网本也具有此功能，这一功能也被称为华硕 Super Hybrid Engine。如果 CPU 负载低于或等于 ***load_threshold_powersave*** 参数设定的指定值， 插件会根据 ***she_powersave*** 参数设置 FSB 速度（请在 kernel 文档中查看关于 FSB 频率极其对应值的详细信息，该文档提供了对大多数用户有用的默认设置）。 如果 CPU 负载高于或等于 ***load_threshold_normal*** 参数设定的指定值，插件会根据 ***she_normal*** 参数设置 FSB 速度。若无法检测到硬件支持，那么就无法支持静态微调，插件也会被禁用。

net

根据 ***wake_on_lan*** 参数设定的指定值对 LAN 唤醒进行配置（它使用的是和 **ethtool** 实用工具相同的语法）。它也会根据接口使用情况动态地更改接口速度。

sysctl

使用插件参数设置不同的 **sysctl** 设置。该语法是 **name=value**，其中 **name** 和 **sysctl** 工具提供的名称相同。如果需要改变其它插件无法更改的设置，请使用本插件（如果其它插件可以更改该设置，最好使用其它特定插件）。

usb

根据 ***autosuspend*** 参数设定的指定值对 USB 设备自动挂起超时进行设置。若该值为 0 意味着禁用自动挂起。

vm

启用或禁用透明 Huge Page 取决于 ***transparent_hugepages*** 参数的布尔值。

audio

根据 ***timeout*** 参数设定的指定值设置音频编码解码器的自动挂起超时。目前支持 **snd_hda_intel** 和 **snd_ac97_codec**。该值为 **0** 意味着自动挂起被禁用。您还可以将 ***reset_controller*** 的布尔参数设置为 **true**，以此来强制重设控制器。

disk

根据 ***elevator*** 参数设定的指定值设置 elevator。它同样会根据 ***alpm*** 参数设置 ALPM （请参阅〈第 3.7 节 “主动连接电源管理”〉），根据 ***aspm*** 参数设置 ASPM （请参阅〈第 3.6 节 “活动状态电源管理”〉），根据 ***scheduler_quantum*** 参数设置计划程序量程，根据 ***spindown*** 参数设置磁盘旋转降速，根据 ***readahead*** 参数设置磁盘 readahead ，根据 ***readahead_multiply*** 参数指定的常量乘以当前的 readahead 值。除此之外，这一插件会动态地更改高级电源管理和磁盘旋转降速超时，根据当前的驱动器使用情况对驱动器进行设置。默认情况下动态微调将被启用，并可以通过 ***dynamic*** 的布尔参数进行控制。

mounts

根据 ***disable_barriers*** 参数的布尔值启用或禁用挂载障碍。

script

此插件能够用来执行一种外置脚本，该脚本在配置文件已加载或未加载时都会运行。此脚本会被 **start** 或 **stop** 参数调用（这取决于在配置文件加载或卸载期间脚本是否被调用）。您可以通过 ***script*** 参数指定脚本文件名。您需要在脚本中正确执行停止行为，并且还原在启动期间改变的所有设置，否则还原将无法运行。为了方便使用，**functions** 文件的 Bash 帮助脚本已经默认安装，您可以导入和使用其中定义的多种函数。此功能主要用于实现后向兼容性，建议您仅在其它插件无法提供需要的设置时使用本功能。

sysfs

通过插件参数设定不同的 **sysfs** 设置。该语法为 **name=value**，其中 **name** 是供 **sysfs** 路径使用的。如需更改其他插件未提供的设置，请使用本插件（若其它插件有提供需要的设置，最好使用该特定插件）。

video

为视频卡设置不同的节电等级（目前仅支持 Radeon 卡）。可使用 ***radeon_powersave*** 参数指定节电等级。支持的值有：**default**、 **auto**、**low**、**mid**、**high** 和 **dynpm**。详情请参阅 http://www.x.org/wiki/RadeonFeature#KMS_Power_Management_Options。请注意此插件正处于试验阶段，未来发行时参数可能有所更改。

2.5.2. 安装和使用

若要安装 *tuned* 软件包，请以 root 身份运行下列指令：

```
yum install tuned
```

安装 *tuned* 软件包还会预先设置对您的系统最佳的配置文件。目前，默认配置文件是根据下列可自定义的规则选择的：

throughput-performance

此配置为 Fedora 操作系统预先选定的计算节点。在该系统上的目标是为了达到最佳吞吐量性能。

virtual-guest

虚拟机会预先选定此配置。若您不想实现最佳性能，请将其改变为 **balanced** 或者 **powersave** 配置文件（如下所示）。

balanced

其他情况下都会预先选定此配置。其目标在于平衡性能和电源消耗。

如需启动 **tuned**，请以 root 身份运行下列指令：

```
systemctl start tuned
```

若要在每次计算机启动时激活 **tuned**，请输入以下指令：

```
systemctl enable tuned
```

其它的 **tuned** 控制，例如配置文件选择等，请使用：

```
tuned-adm
```

此命令需要 **tuned** 服务正在运行。

若要查看可用的已安装配置文件，请运行：

```
tuned-adm list
```

若要查看目前已激活的配置文件，请运行：

```
tuned-adm active
```

若要选择或激活某一配置文件，请运行：

```
tuned-adm profile profile
```

例如：

```
tuned-adm profile powersave
```

此功能是一项试验性的功能，它可能一次选择较多的配置文件。 **tuned** 应用将会试图在加载期间将它们合并。若有冲突，将优先选择最后选定的配置文件的设置。这会自动实现，并且不会检查产生的合并参数是否有意义。如果不加考虑的使用，此功能可能会以完全相反的方式微调某些参数，其结果可能适得其反。例如，使用 **throughput-performance** 配置文件，将磁盘吞吐量设置为 **high**，同时，使用 **spindown-disk** 配置文件，将磁盘旋转降速设置为 **low**。下面的例子是在一台虚拟机上将系统优化为最佳性能，同时将其微调为低能耗，并且以低能耗为优先：

```
tuned-adm profile virtual-guest powersave
```

若要让 **tuned** 推荐最适合您的系统的配置文件，同时不改变任何现有的配置文件，也不使用安装期间使用过的逻辑，请运行以下指令：

```
tuned-adm recommend
```

tuned 自身有额外选项以供手动运行时使用。然而，我们并不建议您这样做，它主要用于 debug 目的。若要查看可用的选项，请使用以下指令：

```
tuned --help
```

2.5.3. 自定义配置文件

特定分配的配置文件被储存在 **/usr/lib/tuned** 目录中。每个配置文件都有它自己的目录。该配置文件可以构成名为 **tuned.conf** 的主要配置文件，也可选择构成其它文件，比如 helper 脚本。不要更改 **/usr/lib/tuned** 中的配置文件。若需要自定义配置文件，请将配置文件目录复制到 **/etc/tuned** 目录。这是自定义配置文件的位置。若存在两个名称相同的配置文件，将会以 **/etc/tuned** 中的配置文件为优先。您可以在 **/etc/tuned** 目录中创建您自己的配置文件，该目录中会有您感兴趣的配置文件，并且只会更改或替代您想要改动的参数。

tuned.conf 文件包含几个小节。其中有一个 **[main]** 节。其它的小节是插件配置实例。所有的节都是可选的，包括 **[main]** 节。该文件同样支持注释。以 # 开头的行就是注释。

[main] 节有如下选项：

include=*profile*

特定的配置文件将会被包含在内，例如，**include=powersave** 将会包括 **powersave** 配置文件。

描述插件的节将会以下列格式编排：

```
[NAME]
type=TYPE
devices=DEVICES
```

NAME 是插件实例在日志中使用的名称。它可以是任意字符。*TYPE* 指的是微调插件的类型。若要查看微调插件的清单和描述，请查看〈第 2.5.1 节 “插件”〉。*DEVICES* 是该插件实例将会管理的装置清单。**devices** 行可能包含一个清单，一个通配符（*），和否定（!）。您还可以将规则结合起来。若没有 **devices** 行，系统中现有的和之后连接的设备，只要符合 **TYPE**，都将被该插件实例管理。这和使用 **devices=*** 的效果是一样的。若插件中没有指定任何实例，插件将不会被启用。若插件支持更多的选项，它们也能在插件节中被指定。如果没有制定选项，默认值将被使用（如果之前没有在已包括的插件中进行指定）。若要查看插件选项列表，请查看〈第 2.5.1 节 “插件”〉。

例 2.1. 插件实例描述

下面的例子将对所有以 **sd** 开头（比如 **sda** 或者**sdb**）的装置进行匹配，同时不禁用障碍：

```
[data_disk]
type=disk
devices=sd*
disable_barriers=false
```

下面的例子将会除了 **sda1** 和 **sda2** 以外的所有装置进行匹配：

```
[data_disk]
type=disk
devices=!sda1, !sda2
disable_barriers=false
```

如果您不需要为插件实例自定义名称，而且您的配置文件中只有一个对该实例的定义，tuned 支持下面的短语法：

```
[TYPE]
devices=DEVICES
```

在这种情况下，可以省略 **type** 行。该实例将会被引用为一个 name，和 type 一样。那么先前的例子就能被改写为：

```
[disk]
devices=sdb*
disable_barriers=false
```

如果同一节被 **include** 选项指定超过一次，那么设置就会被合并。 如果它们因为有冲突而无法合并，最后的冲突定义将会优先于先前的设置。有时候您并不清楚之前有什么样的定义。这种情况下您可以使用 **replace** 布尔选项，将它设置为 **true**。这将会使得先前所有的名称相同的定义都会被覆盖，合并也就不会发生了。

您还可以通过指定 **enabled=false** 选项来禁用插件。这和不定义该实例的效果是一样的。若您想要重新定义 **include** 选项中的定义，并且不想要插件在自定义配置文件中被激活，禁用插件是很有用的。

大多数情况下，装置会被一个插件实例管理。如果装置和多个实例定义匹配，将会报告错误。

下面的例子是一个以 **balanced** 配置文件为基础的自定义配置文件，将它扩展为将所有设备的 ALPM 设置为最大节电模式。

```
[main]
include=balanced

[disk]
alpm=min_power
```

2.5.4. Tuned-adm

通常，具体的系统审核和分析非常耗时，并且这样做也不节能。之前的做法是使用默认设置。因此，Red Hat Enterprise Linux 7 针对两种极端的不同使用方式提供了不同的配置文件以供选择。同时，它还提供了 **tuned-adm** 工具，使得您可以通过命令行在这些配置文件间进行切换。Red Hat Enterprise Linux 7 包含很多适用于典型案例的预定义配置文件，您只要使用 **tuned-adm** 命令即可选择并激活它们，但您也可以自己创建、修改并删除配置文件。

要列出所有可用配置文件并识别目前激活的配置文件，请运行：

```
tuned-adm list
```

要只显示当前激活的配置文件请运行：

```
tuned-adm active
```

要切换到某个可用的配置文件请运行：

```
tuned-adm profile profile_name
```

例如：

```
tuned-adm profile server-powersave
```

要禁用所有微调：

```
tuned-adm off
```

下面是通过基础数据包安装的配置文件清单：

balanced

默认节电配置文件。其目的为在性能和节能之间找到平衡。它试图在任何可能的情况下都使用自动调整和自动微调。它对大多数负载都会产生好的结果。它唯一的缺点是会增加延迟。目前的 **tuned** 使得它能够启用 CPU、磁盘、音频和视频插件，还会激活 **ondemand** 调控器。***radeon_powersave*** 会被设定为 **auto**。

powersave

用于最大化节能效能的配置文件。它能限制效能，以最大限度地减少实际电量消耗。目前的 **tuned** 使得它能够为 SATA 主适配器启用 USB 自动挂起、 WiFi 节能和 ALPM 节能（请参阅〈第 3.7 节 “主动连接电源管理”〉）。它还会以低唤醒率为系统调度多核节能，同时激活 **ondemand** 控制器。它会根据您的系统启用 AC97 音频节电，或者启用每 10 秒超时的 HDA-Intel 节能。以防您的系统支持启用 KMS 的 Radeon 图形卡，它将自己配置为自动启用节能模式。在华硕 Eee PC 上，会启用动态 Super Hybrid Engine。

注意

powersave 配置文件可能不会总是最有效率的。请考虑以下情况：有一个工作量一定的任务需要完成，例如一个视频文件需要转码。若转码是在满电状态下进行的，您的电脑会消耗较少的电能，因为这种情况下任务会很快的完成，计算机将会闲置，并且能自动进入高效节电模式。另一方面如果您使用受限的机器对文件进行转码，在转码过程中会消耗较少的电能，但是转码过程将持续更久，总体的电量消耗可能会更高。这就是为什么 **balanced** 配置文件总体来说是一个更优的选择。

throughput-performance

将服务器向高吞吐量优化的配置文件。它会禁用节电机制，并启用 sysctl 设置，提升磁盘和网络 IO 的吞吐性能，并切换到 **deadline** 计划程序。CPU 调控器被设定为 **performance**。

latency-performance

将服务器向低延迟优化的配置文件。它会禁用节电机制并启用 sysctl 设置，改善延迟。CPU 调控器被设定为 **performance**，CPU 会被锁定到低 C 状态（通过 PM QoS）。

network-latency

用于低延迟网络微调的配置文件。它以 **latency-performance** 配置文件为基础。它还会额外禁用透明 huge page 和 NUMA 平衡，并且微调一些与网络相关的 sysctl 参数。

network-throughput

用于微调网络吞吐量的配置文件。它以 **throughput-performance** 配置文件为基础。此外它还会增加 kernel 网络缓冲区。

virtual-guest

针对虚拟客机设计的配置文件。它基于企业储存配置文件，会降低虚拟内存的 sawp，增加磁盘预读值。它不会禁用磁盘障碍。

virtual-host

基于 **enterprise-storage** 的配置文件，会降低虚拟内存的 sawppiness，增加磁盘预读值，并且启用更积极的脏页（dirty page）回写。

sap

针对 SAP 软件进行最佳性能优化的配置文件。它基于 enterprise-storage 配置文件。sap 配置文件会额外微调有关共享内存和信号量的 sysctl 设置，以及进程可能有的最大内存映射数量。

desktop

基于 **balanced** 文件，针对台式电脑进行优化的配置文件。它会额外启用 autogroups 计划程序，以使交互应用获得更好的回复。

可以在**选用**频道中使用 *tuned-profiles-compat* 软件包安装额外的预定义配置文件。这些配置文件针对的是后向兼容性，并且不再开发。基础数据包中的通用配置文件的效果通常和这些配置文件一样或者更佳。若您没有特别的理由一定要使用这种配置文件，最好使用上面提到的基础数据包中的配置文件。兼容性配置文件如下所示：

default

此配置文件对节电的作用最小，并且只会启用 **tuned** 中的 CPU 和磁盘插件。

desktop-powersave

针对台式计算机系统的节电配置文件。它会为 SATA 主机适配器启用 ALPM 节电 （参阅〈第 3.7 节 “主动连接电源管理”〉），并启用 **tuned** 中的 CPU、以太网和磁盘插件。

server-powersave

针对服务器系统的节电配置文件。它会为 SATA 主机适配器启用 ALPM 节电，并激活 **tuned** 的 CPU 和磁盘插件。

laptop-ac-powersave

针对 AC 运行的笔记本电脑的中等节电配置文件。它会启用针对 SATA 主机适配器的 ALPM，还会启用 Wi-Fi 节能，以及 **tuned** 的 CPU、以太网和磁盘插件。

laptop-battery-powersave

针对使用电池电源的笔记本计算机的高效节电配置文件。目前的 **tuned** 中有一个 **powersave** 配置文件的别名。

spindown-disk

针对使用标准硬盘驱动器的计算机的节电配置文件，使旋转降速时间达到最长。它会禁用 **tuned** 节电机制，禁用 USB 自动挂起、蓝牙、Wi-Fi 节电和日志同步，增加磁盘回写时间，降低磁盘的 swap。所有的分区都通过 **noatime** 选项重新挂载。

enterprise-storage

针对企业级存储的服务器配置文件，使 I/O 吞吐量达到最大。它和 **throughput-performance** 配置文件激活一样的设置，增加预读设置，禁用非 root 分区和非 boot 的分区上的障碍。

2.5.5. Powertop2tuned

powertop2tuned 实用程序是一种允许您根据 **PowerTOP** 的建议创建自定义 **tuned** 配置文件的工具。（欲知关于 **PowerTOP** 的详细信息，请参阅〈第 2.2 节 “PowerTOP”〉）。

若要安装 **powertop2tuned** 应用，请以 root 身份运行以下指令：

```
yum install tuned-utils
```

若要创建自定义配置文件，请以 root 身份运行以下指令：

```
powertop2tuned new_profile_name
```

基于目前选择的 **tuned** 配置文件，它默认将配置文件创建在 **/etc/tuned** 目录中。为了安全，新配置文件中的所有 **PowerTOP** 微调最初都会被禁用。若要启用，请在 **/etc/tuned/*profile*/tuned.conf** 中对您想要启用的微调取消注释。您可以使用 **--enable** 或 **-e** 选项，这些选项将会生成新的配置文件，同时会启用 **PowerTOP** 建议的大部分微调。一些危险的微调（例如 USB 自动挂起）将会被禁用。若您确实需要启用该微调，您需要手动取消注释。新配置文件默认为未激活状态。如需激活，请运行以下指令 ：

```
tuned-adm profile new_profile_name
```

如需查看完整的 **powertop2tuned** 支持的选项清单，请输入以下指令：

```
powertop2tuned --help
```

2.6. UPower

Red Hat Enterprise Linux 6 中的 **DeviceKit-power** 假设电源管理功能是 **HAL** 的一部分。在先前发行的 Red Hat Enterprise Linux 中 ，其中一些功能是 **GNOME 电源管理器** 的一部分（请参阅〈第 2.7 节 “GNOME 电源管理器”〉）。Red Hat Enterprise Linux 7 和 **DeviceKit-power** 是根据 **UPower** 重新命名的。**UPower** 提供了守护程序、 API 和一组命令行工具。系统上每个电量源都代表了一个设备，不论它是不是物理设备。例如，一个笔记本电池和 AC 电源都代表了设备。

若要使用命令行工具，您可以使用 **upower** 命令和以下选项：

--enumerate, -e

显示系统中每个电源设备的对象路径，例如：

```
/org/freedesktop/UPower/devices/line_power_AC
/org/freedesktop/UPower/devices/battery_BAT0
```

--dump, -d

显示系统中所有电源设备的参数。

--wakeups, -w

显示系统中的 CPU 唤醒。

--monitor, -m

监视系统电源更换，例如：连接或者断开交流电源，或者电池耗尽。如需停止监视系统，请按 **Ctrl+C** 键。

--monitor-detail

监视系统电源更换，例如：连接或者断开交流电源，或者电池耗尽。**--monitor-detail** 选项会显示比 **--monitor** 选项更详细的情况。如需停止监视系统，请按 **Ctrl+C** 键。

--show-info *object_path*, -i *object_path*

显示特定对象路径中所有可用的信息。例如，若要获得 **/org/freedesktop/UPower/devices/battery_BAT0** 对象路径中所代表的关于系统电池的信息，请运行：

```
upower -i /org/freedesktop/UPower/devices/battery_BAT0
```

2.7. GNOME 电源管理器

GNOME 电源管理器是一项守护程序，它是 GNOME 桌面环境的一部分。先前版本的 Red Hat Enterprise Linux 中提供的 **GNOME 电源管理器**的许多电源管理功能，在 Red Hat Enterprise Linux 6 中成为了 **DeviceKit-power** 工具的一部分，在 Red Hat Enterprise Linux 7 中被重新命名为 **UPower**（请参阅〈第 2.6 节 “UPower”〉）。但是，**GNOME 电源管理器**仍然是该功能的前端。作为一个系统托盘中的小应用程序，**GNOME 电源管理器**能够通知您系统电源状况的改变，比如说从电池电源转换到 AC 电源。它还能报告电池状态，并且在电量低的时候进行警告。

2.8. 其它审核工具

Red Hat Enterprise Linux7 提供了一些其它用于系统审核和分析的工具，其中大多数工具都能够被用作信息的补充来源，以防您想要核对您已经发现的东西，或是您需要针对某特定部分更详细的信息。这些工具很多也被用来进行性能微调，包括：

vmstat

vmstat 为您提供有关进程、内存、页、块 I/O、陷阱以及 CPU 活动的详细资料。使用此应用可以进一步查看系统在进行什么活动，以及系统什么地方忙碌。

iostat

iostat 与 **vmstat** 类似，但只在块设备中的 I/O 是这样。它还提供更多详细输出和统计。

blktrace

blktrace 是一个非常详细的块 I/O 追踪程序。它将信息截成与应用关联的单一块。
与**diskdevstat** 合并使用时非常有用。

第 3 章 核心基础结构及技巧

重要

若要使用本章论述的 **cpupower** 命令，请确定已安装 *cpupowerutils* 软件包。

3.1. CPU 闲置状态

使用 x86 构架的 CPU 支持不同的状态，在这些状态中部分 CPU 会被停用或者以低性能设置运行。这些状态，也就是我们知道的“*C 状态*”，允许系统通过停用部分不使用的 CPU 达到节能目的。C 状态从 C0 开始用数字编号，数字越大代表 CPU 功能降低越多，也就越节能。虽然给定数字的 C 状态在不同处理器间类似，但为特定处理器或者处理器产品线使用的特定 C 状态的含义是特定的。C 状态 0-3 定义如下：

C0

操作或者运行状态。在这个状态中，CPU 处于工作状态，完全没有闲置。

C1, 挂起

处理器不执行任何指令的状态，但通常不处于较低功率状态。CPU 可继续进行处理而没有延迟。所有提供 C 状态的处理器都需要支持这个状态。奔腾 4 处理器支持改进的 C1 状态，即 C1E，它实际上是一个低能耗状态。

C2, 时钟停止

在这个状态中，处理器会停止时钟，但它让其暂存器和缓冲保持完整状态，因此重新启动时钟后，它可以立即重新启动处理进程。这是一个可选状态。

C3, 休眠

处理器真正进入睡眠状态且不需要一直更新缓冲。因此从这个状态唤醒的时间要大大长于从 C2 唤醒的时间。这也是一个可选状态。

要查看可用闲置状态以及其他 CPU 闲置驱动程序的统计数据，请运行以下指令：

```
cpupower idle-info
```

最近使用 "Nehalem" 微构架的英特尔 CPU 有新的 C 状态，即 C6 状态。它可将供应 CPU 的电压降低到 0，但通常的节能率在 80% 到 90% 之间。Red Hat Enterprise Linux 7 中的 kernel 包括对这个新 C 状态的优化。

3.2. 使用 CPUfreq 调控器

减少系统电力消耗和发热量的最有效的方法之一就是使用 CPUfreq。CPUfreq，也被称为 CPU 速度计，它允许随时调整处理器的时钟速度。这让系统可在降低的时钟速度下运行以便节电。CPUfreq 调控器中定义了更改频率的规则，无论是加快还是减慢时钟速度，以及何时更改频率。

调控器定义了系统的电源属性，它可影响 CPU 性能。每个调控器有其自身独特的负载行为、目的和实用性。这部分描述了如何选择和配置 CPUfreq 调节器，每个调节器的属性以及每个调节器适用的负载种类。

3.2.1. CPUfreq 调控器类型

本节列出了 Red Hat Enterprise Linux 7 中可用的 CPUfreq 调控器的不同类型。

cpufreq_performance

性能调控器会强制 CPU 使用最高时钟频率。这个频率是静态设置的，不会改变。因此，这个特定的调节器“*不提供节电效益*”。它只适用于几个小时的高负载，而且即使在那种情况下也只可用于 CPU 几乎不（或者从不）闲置的时候。

cpufreq_powersave

相反，节电调控器会强制 CPU 使用最低时钟频率。这个频率将被静态设置，且不会更改。因此，这个特定调节器提供最大节能效益，但这是以“*最低 CPU 性能*”为代价的。

这里“节电”一词有时是不正确的，因为（原则上）满负载但低速运行的 CPU 消耗的电量比没有负载但高速运行的 CPU 要多。因此，尽管我们可能建议在需要低性能时设定 CPU 使用节电调控器，但是在这期间意外的高负载可能会导致系统实际消耗了更多的电量。

简单地说，节电调控器对 CPU 更象是"限速器"而不是"节能器"。它在过热时会出问题的系统和环境中最有用。

cpufreq_ondemand

按需调控器是一个动态调控器，它允许 CPU 在系统负载高时达到最大时钟频率，还允许系统处于闲置时使用最低时钟频率。虽然这允许系统根据系统负载调整电源消耗，但也确实要承受“*频率切换间造成的延迟*”。因此，如果系统在闲置和高负载间切换过于频繁，那么延迟可抵消任何按需调控器带来的性能/节能优势。

对大多数系统来说，按需调控器可在散热、电源消耗、性能以及管理性间提供最佳折中方案。若系统只在每天的某个具体时间繁忙，按需调控器将根据负载自动在最大和最小频率间切换而无须进一步操作。

cpufreq_userspace

用户空间调控器允许用户空间程序，或者任何以 root 身份运行的进程，来设置频率。所有的调控器当中，用户空间调控器是最能够自定义的。根据它的配置，它能够为您的系统提供最佳的性能和能耗的折中方案。

cpufreq_conservative

与按需调控器类似，传统调控器也会根据用量调整时钟频率（类似按需调控器）。但是按需调控器更极端（从最大到最小，再返回），传统调控器则在更接近的频率间进行切换。

这意味着传统调控器会将时钟频率调整为它认为适合负载的频率，而不是简单的在最大和最小频率间选择。虽然这样可以极大地节省电量消耗，但它的代价是产生比按需调控器“*更多的延迟*”。

注意

您可以使用 **cron** 指令启用调控器。这允许您在每天的特定时间自动设定某特定调节器。因此，您可以在闲置时指定低频率调控器（例如工作之余），并在高负载时返回高频率调控器。

若要查看如何启用特定调控器的操作指南，请参阅〈第 3.2.2 节 “CPUfreq 设置”〉。

3.2.2. CPUfreq 设置

所有的 CPUfreq 调控器都被内置于 kernel 工具包中，并且会被自动选定，因此若要设置 CPUfreq，您需要选择一个调控器。

您可以通过下列指令查看特定 CPU 可以使用的调控器：

```
cpupower frequency-info --governors
```

之后，您可以通过下列指令在所有 CPU 上启用这些调控器：

```
cpupower frequency-set --governor [governor]
```

若要只在特定核上启用某个调控器，请使用 **-c** 指令和 CPU 数的范围或是逗号分隔的清单。例如，若要为 CPU 1-3 和 5 启用用户空间调控器，指令为：

```
cpupower -c 1-3,5 frequency-set --governor cpufreq_userspace
```

3.2.3. 微调 CPUfreq 策略和速度

选择适当的 CPUfreq 调控器后，您就可以使用 **cpupower frequency-info** 指令查看 CPU 速度和策略的信息，并可以使用 **cpupower frequency-set** 指令的选项进一步微调每一个 CPU 的速度。

cpupower frequency-info 有以下可用选项：

- **--freq** — 根据 CPUfreq 核显示该 CPU 的当前速度，单位为 KHz。
- **--hwfreq** — 根据硬件显示 CPU 的当前速度，单位为 KHz（仅限 root 用户）。
- **--driver** — 显示这个 CPU 中用来设定频率的 CPUfreq 驱动器。
- **--governors** — 显示此 kernel 上可用的 CPUfreq 调控器。若您想要使用此文件中未列出的 CPUfreq 调控器，请查看〈第 3.2.2 节 “CPUfreq 设置”〉。
- **--affected-cpus** — 列出需要频率协调软件的 CPU。
- **--policy** — 显示当前 CPUfreq 策略范围，单位为 KHz，以及当前活跃的调控器。
- **--hwlimits** — 列出该 CPU 的可用频率，单位为 KHz。

cpupower frequency-set 有以下可用选项：

- **--min <*freq*>** 和 **--max <*freq*>** — 设定 CPU 的 “*策略限制*”，单位为 KHz。

重要

当设定策略限制时，您应该在设定 **--min** 前设定 **--max**。

- **--freq <*freq*>** — 为 CPU 设定具体时钟速度，单位为 KHz。您只能选择一个在 CPU 策略限制范围内的速度（即 **--min** 和 **--max** 之间的数值）。
- **--governor <*gov*>** — 设定新的 CPUfreq 调控器。

注意

如果您没有安装 *cpupowerutils* 软件包，则可在 **/sys/devices/system/cpu/*[cpuid]*/cpufreq/** 的可微调部分查看 CPUfreq 设置。通过写入这些 tunables 即可更改设置和数值。例如：要将 cpu0 的最低时钟速度设定为 360 KHz，请使用：

```
echo 360000 >
/sys/devices/system/cpu/cpu0/cpufreq/scaling_min_freq
```

3.3. CPU 监视器

cpupower 包括一组提供闲置和休眠状态统计和频率信息的监视程序，同时还报告处理器拓扑。有些监视程序是具体应用到处理器中，而另一些可与其他处理器兼容。有关监视程序测量哪些处理器及其兼容的系统，请参考 **cpupower-monitor** man page。

请在 **cpupower-monitor** 命令中使用以下选项：

- **-l** — 列出您系统中的所有可用监视程序。
- **-m *<monitor1>, <monitor2>*** — 显示具体的监视程序。可运行 **-l** 命令查找其识别符号。
- **command** — 显示闲置统计数据以及 CPU 所需的具体命令。

3.4. CPU 节电策略

cpupower 提供管理处理器节电策略的方法。

请在 **cpupower-set** 命令中使用以下选项：

--perf-bias *<0-15>*

可让所支持的英特尔处理器中的软件更多参与如何确定最佳性能和节电之间的平衡。这不会覆盖节电策略。可分配数值的范围为 0 到 15，其中 0 为最佳性能，15 为最佳用电效率。

默认在所有核中应用这个选项。要只在某一个核中应用它，请添加 **--cpu *<cpulist>*** 选项。

--sched-mc <0|1|2>

在从其他 CPU 软件包中提取前，将系统进程使用的电源限制在一个 CPU 软件包的核中。0 代表没有限制，1 代表开始只使用单一 CPU 软件包，2 代表除了使用单一 CPU 软件包之外，另外还使用半闲置 CPU 软件包处理唤醒的任务。

--sched-smt <0|1|2>

在从其它的核中提取前，将系统进程使用的电源限制在一个 CPU 核的同级线程中。0 代表没有限制，1 代表开始只使用单一 CPU 软件包，2 代表除了使用单一 CPU 软件包之外，另外还使用半闲置 CPU 包处理唤醒的任务。

3.5. 挂起和恢复

当系统挂起时，kernel 会调用驱动程序保存其状态然后将其卸载。当系统恢复时，它会载入这些试图重新编程其设备的驱动程序。驱动程序完成这个任务的能力决定了系统是否可以被成功恢复。

就这一点而言，视频驱动程序是最成问题的，这是因为“*高级配置和电源界面*”（ACPI）规范不要求系统固件能够重新编程视频硬件。因此，除非视频驱动程序可以从完全非启动状态编程硬件，否则它们可能会阻止系统恢复。

Red Hat Enterprise Linux 7 包括对新图形芯片组的更大支持，这个支持可确保挂起和恢复功能在大量平台中正常工作。此版本特别对 NVIDIA 芯片组进行了大量改进，也特别针对 GeForce 8800 系列进行了改进。

3.6. 活动状态电源管理

ASPM（Active-State Power Management，活动状态电源管理）能节省 *PCI Express*（PCIe，Peripheral Component Interconnect Express）子系统的电量，其原理为当 PCIe 连接没有处于使用状态时将其设定为低功率状态。ASPM 可以同时控制连接两端的电源状态，并且在连接终端的设备处于满电状态的情况下，仍然可以节电。

启用 ASPM 时，在不同电源状态间转换连接时需要时间，因此会增加设备延迟。ASPM 有三种决定电源状态的策略：

默认（default）

根据系统固件（例如：BIOS）指定的默认设置设定 PCIe 连接的电源状态。这是 ASPM 的默认状态。

节电（powersave）

将 ASPM 设定为尽可能节电，不考虑性能损失。

性能（performance）

禁用 ASPM 使 PCIe 链接以最佳性能操作。

使用 ***pcie_aspm*** kernel 参数可以启用或者禁用 ASPM，其中 **pcie_aspm=off** 会禁用 ASPM，而 **pcie_aspm=force** 会启用 ASPM，即使在不支持 ASPM 的设备中也可以。

ASPM 策略在 **/sys/module/pcie_aspm/parameters/policy** 中设置，但还可以使用 ***pcie_aspm.policy*** kernel 参数在启动时指定，其中 **pcie_aspm.policy=performance** 将设定 ASPM 性能策略。

警告

如果设定了 **pcie_aspm=force**，不支持 ASPM 的硬件可导致系统停止响应。请在设定 **pcie_aspm=force** 前确定系统中的所有 PCIe 硬件都支持 ASPM。

3.7. 主动连接电源管理

ALPM（Aggressive Link Power Management，主动连接电源管理）是一项节能技术，其原理为通过在闲置时（即没有 I/O 的时候）将连接到硬盘的 SATA 连接设定为省电模式，以帮助硬盘节电。在连接中有 I/O 请求队列时，ALPM 会自动将 SATA 连接恢复为活跃电源状态。

ALPM 使用的节电技术是以硬盘延迟为代价的。因此，只有在系统会长时间处于 I/O 闲置状态时才使用 ALPM。

ALPM 只适用于使用“*高级主机控制器接口*”（AHCI）的 SATA 控制器。如需了解更多关于 AHCI 的信息，请浏览 http://www.intel.com/technology/serialata/ahci.htm。

电脑支持 ALPM 时，将默认启用 ALPM。ALPM 有三种模式：

min_power

此模式会在硬盘没有 I/O 的情况下，将连接设定为其最低功率状态（SLUMBER）。这个模式在闲置时间较长时会很有帮助。

medium_power

此模式会在硬盘没有 I/O 的情况下，将连接设定为第二低功率状态（PARTIAL）。这个模式在尽量不影响性能的条件下可实现电源状态间的转换（例如：在中等 I/O 负载和闲置 I/O 时）。

medium_power 模式允许连接根据负载情况在 PARTIAL 和满电（即 ACTIVE）状态间进行转换。请注意不可以直接从 PARTIAL 转换到 SLUMBER 然后再转回来。这种情况下，这两种电源状态都需要首先转换成 ACTIVE 状态，然后方可转换到另一个状态。

max_performance

禁用 ALPM。当硬盘中没有 I/O 时，连接也不会进入任何低功率状态。

要查看您的 SATA 主机适配器是否支持 ALPM，请查看文件 **/sys/class/scsi_host/host*/link_power_management_policy** 是否存在。要更改设置，只要在这些文件中写入本章节中描述的值，或者打开文件检查当前设置即可。

重要

将 ALPM 设定为 **min_power** 或者 **medium_power** 将自动禁用“热插拔”特性。

3.8. Relatime 驱动器访问优化

POSIX 标准要求操作系统维护记录每个文件最后一次被访问的文件系统元数据。这个时间戳被称为 **atime**，维护它需要一个重复的对存储的写入操作。这些写入操作让存储是设备及其连接保持忙碌和通电状态。因为很少应用程序会使用 **atime** 数据，所以这个存储设备活动是在浪费电力。值得注意的是，即使不从存储中读取文件而是从缓存中读取文件，也会发生写入存储的事件。有时，Linux kernel 还支持 **mount** 的 **noatime** 选项，并不在使用此选项挂载的文件系统中写入 **atime**。但是只是关闭这个特性是有问题的，因为有些应用程序会依赖 **atime** 数据，并在此数据不可用时失败。

Red Hat Enterprise Linux 7 使用的 kernel 支持另一个可替换选项 — **relatime**。**relatime** 维护 **atime** 数据，但不是每次访问该文件时都更改。启用这个选项，则只在上次更新 **atime**（**mtime**）后修改该文件时，或者最后一次访问该文件是在相当长一段时间前（默认为一天）时才会将 **atime** 数据写入磁盘。

现在，默认情况下会使用启用的 **relatime** 挂载所有文件系统。您可以将某个文件系统排除在外，而使用 **norelatime** 挂载该文件系统。

3.9. 功率封顶

Red Hat Enterprise Linux 7 支持最近在硬件中使用的功率封顶，比如 HP 的“*动态功率封顶*”（DPC，Dynamic Power Capping）以及英特尔的节点管理器（NM，Node Manager）技术。功率封顶允许管理员使用服务器限制功率消耗，但它还可允许管理器更有效地规划数据中心，因为极大降低了现有电源供应的超载风险。管理器可在同一实体机中放置更多的服务器并确定如果服务器电源消耗封顶，在高负载时对电源的需求不会超出可用的电源。

HP 动态功率封顶

动态功率封顶这一功能在选择 ProLiant 和刀片系统服务器时可获得，它可允许系统管理员对一个服务器或者一组服务器的电源消耗封顶。这个封顶是一个绝对限制，无论其当前工作负载如何，服务器将无法超过该限制。这个封顶只在服务器达到其电源消耗限制时才生效。此时某个管理进程会调整 CPU 的 P 状态和时钟限制来限制电量消耗。

动态功率封顶会修改独立操作系统的 CPU 行为，但是 HP 的“*集成 Lights-Out 2*”（iLO2）固件允许操作系统访问管理处理器，因此用户空间中的应用程序可查询管理处理器。Red Hat Enterprise Linux 7 中使用的 kernel 包括用于 HP iLO 和 iLO2 固件的驱动程序，它们可允许程序查询 **`/dev/hpilo/dXccbN`** 中的管理处理器。该 kernel 还包括 **`hwmon sysfs`** 接口的扩展来支持功率封顶特性，以及针对使用 **`sysfs`** 接口的 ACPI 4.0 计量程序的 **`hwmon`** 驱动程序。这些特性允许操作系统和用户空间工具共同读取为功率封顶配置的值以及系统的当前电源用量。

有关 HP 动态功率封顶详情请参考《*HP 功率封顶以及用于 ProLiant 服务器的 HP 动态功率封顶*》，请查看：http://h20000.www2.hp.com/bc/docs/support/SupportManual/c01549455/c01549455.pdf。

Intel 节点管理器

Intel 节点管理器在系统中强制使用功率封顶，使用处理器 P 状态和 T 状态限制 CPU 性能，并因此节能。通过设置电源管理策略，管理员可将系统配置为在系统负载低时，比如夜间或者周末，消耗较少的电力。

Intel 节点管理器使用“*直接操作系统配置和电源管理*”（OSPM，Operating System-directed configuration and Power Management）通过标准“*高级配置和电源接口*”调整 CPU 性能。当 Intel 节点管理器通知 OSPM 驱动程序更改到 T 状态时，该驱动程序会响应并更改处理器 P 状态。同样，当 Intel 节点管理器通知 OSPM 驱动程序更改到 P 状态时，该驱动程序也会相应更改 T 状态。这些更改会自动进行且不需要操作系统有进一步的输入。管理员使用“*Intel 数据中心管理器*”（DCM）软件配置并监控 Intel 节点管理器。

有关 Intel 节点管理器详情请参考《*节点管理器 — 动态管理数据中心电源*》，请查看：http://communities.intel.com/docs/DOC-4766。

3.10. 改进的图形电源管理

Red Hat Enterprise Linux 7 通过删除不必要的资源消耗在图形和显示设备中节能。

LVDS 重新计时

“*低压差分信号传输*”（Low-voltage differential signalling，LVDS）是使用铜线承载电信号的系统。一个主要的应用是将像素信息传输到笔记本电脑的*液晶显示*（LCD）屏幕中。所有显示都有“*刷新率*” — 即它们从图形控制器接受新数据并在屏幕中重新成像的频率。通常屏幕每秒接受 60 次新数据（即频率为 60 Hz）。当屏幕和图形控制器是以 LVDS 连接时，LVDS 系统在每次刷新时都会消耗能量。当闲置时，很多 LCD 屏幕的刷新率都会下降到 30 Hz，且不会产生明显的影响［与“*阴极射线管*”（CRT）显示器不同，后者在降低刷新率时会产生闪烁现象］。Red Hat Enterprise Linux 7 中 kernel 使用的 Intel 图形适配器的驱动程序可自动执行这个“*降频*”，并在屏幕闲置时节约 0.5 W 左右的电力。

启用内存自动刷新

“*同步动态随机访问内存*”（SDRAM） - 用于图形适配器的视频内存，每秒会重复充电上千次，使得每个内存单元可保留保存在其中的数据。除了管理数据的主要功能外，因为有数据流入或者流出内存，所以内存控制器通常负责初始化这些刷新循环。但是 SDRAM 还有一个低功率“*自动刷新*”模式。在这个模式中，内存使用内部计时器生成其自身刷新循环，它可允许系统在不损害当前内存数据的情况下关闭内存控制器。Red Hat Enterprise Linux 7 使用的 kernel 可在 Intel 图形适配器处于闲置状态时触发内存自动刷新，并可节约 0.8 W 左右的电力。

降低 GPU 时钟频率

标准图形处理单元（GPU）带有内部时钟，用于管理其内部电路的不同部分。Red Hat Enterprise Linux 7 使用的 kernel 可降低部分 Intel 和 ATI GPU 的内部时钟频率。减少 GPU 组件在给定时间内执行循环的次数，可以避免组件执行不必要的循环，以达到节约电量的目的。当 GPU 闲置时，kernel 可自动降低这些时钟的速度；同时当 GPU 活性增强时会提高其时钟速度。降低 GPU 时钟循环周期最多可节省 5 W 电力。

GPU 关闭

Red Hat Enterprise Linux 7 中使用的 Intel 和 ATI 图形驱动程序可探测到什么时候适配器中没有连接显示器，并完全关闭 GPU。这个功能对不经常连接显示器的服务器尤为重要。

3.11. RFKill

很多计算机系统包含无线电传输，其中包括 Wi-Fi、蓝牙和 3G 设备。这些设备消耗电源，在不使用这些设备时是一种浪费。

RFKill 是 Linux kernel 中的一个子系统，它可提供一个界面，在此界面中可查询、激活并取消激活计算机系统中的无线电传输。当取消激活传输时，可使其处于可被软件重新激活的状态（即“*软锁定*”）或者将其放在软件无法重新激活的位置（即“*硬锁定*”）。

RFKill 核为子系统提供应用程序编程界面（API）。kernel 驱动程序被设计为支持 RFKill 使用这个 API 注册 kernel，并包含启用和禁用这个设备的方法。另外，RFKill 核提供用户程序可解读的通知以及用户程序查询传输状态的方法。

RFKill 界面位于 **/dev/rfkill**，其中包含系统中所有无线电传输的当前状态。每个设备都在 **sysfs** 中注册当前 RFKill 状态。另外，在启用了 RFKill 的设备中每当状态更改时，RFKill 会发出 *uevents*。

Rfkill 是一个命令行工具，您可使用它查询和更改系统中启用了 RFKill 的设备。要获得这个工具，请安装 *rfkill* 软件包。

使用命令 **rfkill list** 获得设备列表，每个都包含与之关联的“*索引号*”，从 **0** 开始。您可以使用这个索引号让 **rfkill** 停止使用或者使用某个设备，例如：

```
rfkill block 0
```

停用系统中第一个启用 RFKill 的设备。

您还可以使用 **rfkill** 阻断某一类设备，或者所有已启用 RFKill 的设备。例如：

```
rfkill block wifi
```

停用系统中的所有 Wi-Fi 设备。要停用所有已启用 RFKill 的设备，请运行：

```
rfkill block all
```

要重新使用设备，请运行 **rfkill unblock**，而不是 **rfkill block**。要获得 **rfkill** 可停用的完整设备类别列表，请运行 **rfkill help**。

第 4 章 使用案例

本章描述了两类使用案例演示本指南中描述的分析和配置方法。第一个示例说的是典型服务器，第二个是典型笔记本电脑。

4.1. 示例 — 服务器

现在典型的标准服务器基本都包含 Red Hat Enterprise Linux 7 中支持的所有所需硬件功能。您的首要考虑是该服务器主要使用的负载类型。根据这个信息您可以决定要优化哪些组件节能。

不论服务器类型如何，通常都不需要图像性能。因此可打开 GPU 节能。

网页服务器

网页服务器需要网络和磁盘 I/O。根据外部连接速度，100 Mbit/s 应该足够了。如果该机器大多数提供的是静态页面，CPU 性能则并不重要。因此电源管理选项应包括：

- 无 **tuned** 的磁盘或者网络插件。
- 打开 ALPM。
- 打开 **ondemand**（按需）调节器。
- 网卡限制为 100 Mbit/s。

计算服务器

计算服务器主要是 CPU。电源管理选择可能包括：

- 根据任务以及出现数据存储的位置，激活 **tuned** 的磁盘或者网络插件；或者在批处理系统中完全激活 **tuned**。
- 根据应用，可能是 **performance**（性能）调节器。

邮件服务器

邮件服务器主要需要磁盘 I/O 和 CPU。电源管理选择应包括：

- 打开 **ondemand** 调节器，因为 CPU 最后的几个百分比并不重要。
- 无 **tuned** 的磁盘或者网络插件。
- 不应该限制网络速度，因为邮件通常是内部的，并可因此从 1 Gbit/s 或者 10 Gbit/s 连接中获益。

文件服务器

文件服务器的需求与邮件服务器类似，但根据所用协议，可能需要更多的 CPU 性能。通常，基于 Samba 的服务器需要的 CPU 比 NFS 多，而 NFS 又比 iSCSI 需要更多的 CPU。即使如此，您应可以使用 **ondemand** 调节器。

目录服务器

目录服务器通常对磁盘 I/O 的要求较低，特别是在有足够 RAM 的情况下。虽然网络 I/O 没有那么重要，网络延迟问题却很重要。您可以考虑使用较低连接速度的延迟网络调节，但您应该为具体网络进行细心的测试。

4.2. 示例 — 笔记本电脑

另一个电源管理和节能通常起作用的示例就是笔记本电脑。因为笔记本电脑一般已经被设计为比工作站或者服务器节省很多电力。当使用电池模式时，节能可让您的电池使用时间多几分钟。虽然这部分着重阐述笔记本电脑的电池模式，但您仍然可在使用交流电供电时也使用全部或者部分微调。

在笔记本电脑中单一组件中的节能通常要比在工作站中更明显。例如：1 Gbit/s 网络接口以 100 Mbits/s 运行时可节电大约 3-4 瓦。对于总耗电 400 瓦的典型服务器来说，这个节能大概是在 1 %。在总耗电 40 瓦的典型笔记本电脑中，这一个组件的节能就是总耗电量的 10 %。

典型笔记本电脑中的具体节能优化包括：

- 将系统 BIOS 配置为禁用所有您不使用的硬件。例如：并口或者串口、读卡器、摄像头、WiFi 以及蓝牙，这里只给出一些可能的硬件。
- 在较暗的环境中，您不需要使用最大亮度就可舒服地阅读屏幕中的内容，此时可调暗显示屏。请在 GNOME 桌面中使用「**系统**」+「**首选项**」 → 「**电源管理**」，在 KDE 桌面中使用 「**Kickoff Application Launcher**」+「**计算机**」+「**系统设置**」+「**高级**」 → 「**电源管理**」，或者在命令行中使用 **gnome-power-manager** 或者 **xbacklight**，或者您笔记本电脑中的功能键。

作为补充或者替代，您还可以对不同的系统设定进行许多小的调整：

- 请使用 **ondemand** 调控器（Red Hat Enterprise Linux 7 中默认启用）
- 启用 AC97 音频节能（Red Hat Enterprise Linux 7 中默认启用）：

  ```
  echo Y > /sys/module/snd_ac97_codec/parameters/power_save
  ```

- 启用 USB 自动挂起：

  ```
  for i in /sys/bus/usb/devices/*/power/autosuspend; do echo 1 > $i;
  done
  ```

  请注意：USB 自动挂起并不适用于所有 USB 设备。

- 使用 relatime 装载文件系统（Red Hat Enterprise Linux 7 中默认启用）：

  ```
  mount -o remount,relatime mountpoint
  ```

- 将屏幕亮度降低至 **50** 或更小，例如：

  ```
  xbacklight -set 50
  ```

- 为屏幕闲置激活 DPMS：

  ```
  xset +dpms; xset dpms 0 0 300
  ```

- 取消激活 Wi-Fi：

  ```
  echo 1 > /sys/bus/pci/devices/*/rf_kill
  ```

附录 A. 开发者小贴示

每本优秀的编程课本都包含内存分配以及具体功能性能的问题。当您开发自己的软件时，请注意可能在运行该软件的系统中增加电源消耗的问题。虽然这些考虑不会影响每一行代码，但您可以优化那些经常成为性能瓶颈部分的代码。

经常会出问题的技术包括：

- 使用线程。
- 不必要的唤醒 CPU 或者未有效使用唤醒。如果您必须执行唤醒，尽快一次做完所有的事（迅速返回闲置状态）。
- 不必要的使用 **[f]sync()**。
- 不必要的活跃调用或者使用简短常规超时（使用响应事件）。
- 未有效使用唤醒。
- 低效磁盘访问。使用大量缓冲来避免频繁的磁盘访问。一次写入大块信息。
- 低效使用计时器。可能时使用跨应用程序（甚至跨系统）的组群计时器。
- 过量的 I/O、电源消耗或者内存使用（包括内存泄露）。
- 执行不必要的计算。

下面的部分将对这些方面进行更详细地阐述。

A.1. 使用线程

大家普遍认为使用线程能够更好且更迅速地执行应用，但情况并不总是如此。

Python

Python 使用全局锁定解码器 [1]，因此使用线程只能在有大量 I/O 操作时受益。**Unladen-swallow** [2] 是一个 Python 快速部署，您可用它来优化您的代码。

Perl

Perl 线程原是用于系统中不使用分叉技术的应用程序的（比如使用 32 位 Windows 操作系统的系统）。在 Perl 线程中会为每个单一线程复制数据（写时复制）。数据不是默认共享的，因为用户应该可以定义数据共享等级。必须包括共享 **threads::shared** 模块的数据。但是数据不仅仅是被复制（写时复制），该模块还为这些数据生成了捆绑变量，这就需要更多的时间，且速度更慢。 [3]

C

C 线程共享同一内存，每个线程都有自己的层叠，同时 kernel 不一定要生成新的文件描述符并分配新的内存空间。C 可以真正在更多线程中使用更多 CPU 支持。因此要最大化您的线程性能，请使用低级语言，比如 C 或者 C++。如果您使用脚本语言，请考虑写入一个 C 绑定。请使用分析器识别不能很好执行的代码。 [4]

A.2. 唤醒

很多应用程序都会扫描配置文件中的变更。在很多情况下，这种扫描的时间间隔是固定的，例如：每分钟。这可能是个问题，因为它强制将磁盘从低转速状态唤醒。最佳解决方案是找到合理的时间间隔，好的检查机制或者使用 **inotify** 检查并响应每个事件。**inotify** 可查看文件或者目录中的各种更改。

例如：

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/time.h>
#include <sys/types.h>
#include <sys/inotify.h>
#include <unistd.h>

int main(int argc, char *argv[]) {
  int fd;
  int wd;
  int retval;
  struct timeval tv;

  fd = inotify_init();

  /* checking modification of a file - writing into */
  wd = inotify_add_watch(fd, "./myConfig", IN_MODIFY);
  if (wd < 0) {
    printf("inotify cannot be used\n");
    /* switch back to previous checking */
  }

  fd_set rfds;
  FD_ZERO(&rfds);
  FD_SET(fd, &rfds);
  tv.tv_sec = 5;
  tv.tv_usec = 0;
  retval = select(fd + 1, &rfds, NULL, NULL, &tv);
  if (retval == -1)
    perror("select()");
  else if (retval) {
    printf("file was modified\n");
  }
  else
    printf("timeout\n");

  return EXIT_SUCCESS;
}
```

此方法的优点是您可执行不同的检查。

主要的局限是一个系统中可查看的次数是有限的。次数可在 **/proc/sys/fs/inotify/max_user_watches** 中获得，虽然该数字是可以更改的，但并不建议更改。此外，**inotify** 失败时，该代码必须返回不同的检查方法，通常意味着在源代码中会有很多 **#if #define**。

欲知有关 **inotify** 的详细信息，请参阅 inotify(7) man page。

A.3. Fsync

fsync 被视为大量消耗 I/O 的操作，但这并不完全正确。

Firefox 原来在用户每次点击一个链接时都调用 **sqlite** 程序库进入新的页面。**sqlite** 调用 **fsync**，且由于文件系统设置（主要使用数据排序模式的 ext3），什么都不发生时会有一个长时间延迟。如果另一个进程同时正在复制一个大文件，这就需要很长的时间（最长可达 30 秒）。

可是在其它情况下，若完全不使用 **fsync**，转换到 ext4 文件系统时就会出现问题。ext3 被设定为数据排序模式，此模式每隔几秒就会刷新内存并将其储存在磁盘上。但是若使用 ext4 和 laptop_mode，储存的间隔会变长，若系统意外关闭可能造成数据丢失。现在 ext4 进行了修补，但是我们仍然需要仔细考虑应用的设计，并且适当地使用 **fsync** 。

下面读取和写入配置文件的简单示例演示了如何备份文件或者数据是怎么丢失的：

```
/* open and read configuration file e.g. ./myconfig */
fd = open("./myconfig", O_RDONLY);
read(fd, myconfig_buf, sizeof(myconfig_buf));
close(fd);
...
fd = open("./myconfig", O_WRONLY | O_TRUNC | O_CREAT, S_IRUSR |
S_IWUSR);
write(fd, myconfig_buf, sizeof(myconfig_buf));
close(fd);
```

更好的方法可能是：

```
/* open and read configuration file e.g. ./myconfig */
fd = open("./myconfig", O_RDONLY);
read(fd, myconfig_buf, sizeof(myconfig_buf));
close(fd);
...
fd = open("./myconfig.suffix", O_WRONLY | O_TRUNC | O_CREAT, S_IRUSR |
S_IWUSR
write(fd, myconfig_buf, sizeof(myconfig_buf));
fsync(fd); /* paranoia - optional */
...
close(fd);
rename("./myconfig", "./myconfig~"); /* paranoia - optional */
rename("./myconfig.suffix", "./myconfig");
```

[1] http://docs.python.org/c-api/init.html#thread-state-and-the-global-interpreter-lock

[2] http://code.google.com/p/unladen-swallow/

[3] http://www.perlmonks.org/?node_id=288022

[4] http://people.redhat.com/drepper/lt2009.pdf

附录 B. 修订历史

修订 2.0-0.1 **Mon Feb 15 2016** **Chester Cheng**

说明：翻译、校对完成。
翻译、校对：潘陈斯梦。
校对、责任编辑：鄭中。
附注：本简体中文版来自「**Red Hat 全球服务部**」与「**澳大利亚昆士兰大学笔译暨口译研究生院**」之产学合作计划。若有疏漏之处，盼各方先进透过以下网址，给予支持指正：https://bugzilla.redhat.com/。

修订 2.0-0 **Wed 18 Feb 2015** **Jacquelynn East**

7.1 GA 版本

修订 1.1-0 **Thu Dec 4 2014** **Jacquelynn East**

7.1 Beta 版本

修订 1.0-9 **Tue Jun 9 2014** **Yoana Ruseva**

7.0 GA 发行版本

修订 0.9-1 **Fri May 9 2014** **Yoana Ruseva**

因风格变化而重新编写

修订 0.9-0 **Wed May 7 2014** **Yoana Ruseva**

为了审核目的发布 Red Hat Enterprise Linux 7.0 指南。

修订 0.1-1 **Thu Jan 17 2013** **Jack Reed**

衍生自 Red Hat Enterprise Linux 6 版本的文件