Red Hat Enterprise Linux 6

電力管理ガイド

Red Hat Enterprise Linux 6 での電力消費量の管理

Last Updated: 2017-10-12

Red Hat Enterprise Linux 6 電力管理ガイド

Red Hat Enterprise Linux 6 での電力消費量の管理

Don Domingo
Red Hat Engineering Content Services

Rüdiger Landmann
Red Hat Engineering Content Services

Jack Reed
Red Hat Engineering Content Services
jreed@redhat.com

Red Hat Inc.

法律上の通知

概要

本ガイドでは、Red Hat Enterprise Linux 6 システムで効果的に電力消費量を管理する方法を説明しています。以下のセクションでは、電力消費量を低減する様々な技術 (サーバー向けとノート PC 向けの両方)、そしてその各技術がどのようにシステムの全体的なパフォーマンスに影響を与えるかについて説明しています。

目次

第1章 概要 **3**
1.1. 電力管理の重要性 3
1.2. 電力管理の基礎 4

第2章 電力管理の監査と分析 **6**
2.1. 監査と分析の概要 6
2.2. POWERTOP 6
2.3. DISKDEVSTAT と NETDEVSTAT 8
2.4. BATTERY LIFE TOOL KIT 12
2.5. TUNED および KTUNE 13
2.5.1. tuned.conf ファイル 15
2.5.2. Tuned-adm 16
2.6. DEVICEKIT-POWER と DEVKIT-POWER 19
2.7. GNOME の電源管理 20
2.8. 他の監査方法 20

第3章 中核となるインフラストラクチャとメカニズム **22**
3.1. CPU のアイドル状態 22
3.2. CPUFREQ ガバナーの使用 22
3.2.1. CPUfreq ガバナーのタイプ 23
3.2.2. CPUfreq の設定 24
3.2.3. CPUfreq ポリシーおよび速度のチューニング 25
3.3. CPU モニター 26
3.4. CPU 節電ポリシー 26
3.5. サスペンドと復帰 27
3.6. TICKLESS KERNEL 27
3.7. ACTIVE-STATE POWER MANAGEMENT 27
3.8. AGGRESSIVE LINK POWER MANAGEMENT 28
3.9. RELATIME ドライブアクセス最適化 29
3.10. パワーキャッピング (POWER CAPPING) 29
3.11. 拡張グラフィックス電力管理 30
3.12. RFKILL 31
3.13. ユーザースペースの最適化 32

第4章 使用例 **33**
4.1. 例 – サーバー 33
4.2. 例 – ノート PC 34

付録A 開発者へのヒント **37**
A.1. スレッドの使用 37
A.2. ウェイクアップ (WAKE-UPS) 38
A.3. FSYNC 38

付録B 改訂履歴 **40**

第1章 概要

Red Hat Enterprise Linux 6 の改良過程におけるフォーカスのひとつが電力管理です。コンピュータシステムが使用する電力の制限は、 *グリーン IT*(環境に優しいコンピューティング) という側面からも最も重要な課題のひとつとなります。このグリーン IT は、リサイクル可能な資源の利用、ハードウェアの製造により環境に及ぼす影響、システム設計と導入における環境意識などの点についても配慮しています。本ガイドでは、 Red Hat Enterprise Linux 6 が稼働するシステムでの電力管理について説明しています。

1.1. 電力管理の重要性

電力管理の中核は、各システムコンポーネントによるエネルギー消費を効果的に最適化する方法を理解するところにあります。そのためには、システムが実行するさまざまなタスクを調査し、そのパフォーマンスがジョブに適したものになるよう各コンポーネントの設定を行う必要があります。

電力管理を行う主な要因は、

- 全体的な消費電力を抑えてコストを削減することです。

電力管理を適切に活用すると、以下のような結果が得られます。

- サーバーおよびコンピューティングセンターの冷却
- 冷却、空間、ケーブル、発電機、*無停電電源装置* (UPS) などにかかる二次コストの削減
- ノート PC のバッテリー寿命の延長
- 二酸化炭素排出量の低減
- エナジースター (Energy Star) などグリーン IT に関する政府の規則、または法的要件への適合
- 新システムにおける企業のガイドラインへの適合

通常、あるコンポーネント (またはシステム全体) の電力消費量を抑えようとすると、発生熱量が低下するため、必然的にパフォーマンスの低下につながります。そのため、特にミッションクリティカルなシステムの場合、設定変更でもたらされるパフォーマンスの低下については十分な調査と検証を行なってください。

システムが実行する様々なタスクを調査し、そのパフォーマンスがジョブに適したパフォーマンスとなるよう各コンポーネントを設定することで、エネルギーを節約し、発生熱を低減し、ノート PC のバッテリー寿命を最適化することができます。電力消費量に関するシステムの分析とチューニングに関する原則の多くは、パフォーマンスのチューニングの原則と同様のものです。システムの最適化は通常、パフォーマンスまたは電力のいずれかに対して行なわれるため、電力管理とパフォーマンスのチューニングは、ある程度、システム構成の上で正反対となるアプローチになると言えます。本ガイドでは、電力管理で役に立つ Red Hat 提供のツール、そして Red Hat で開発された技術について説明していきます。

Red Hat Enterprise Linux 6 には、デフォルトで有効になっている多くの新しい電力管理機能が既に同梱されています。これらはすべて、サーバーやデスクトップの標準的な使用ではパフォーマンスに影響を及ぼさないものとして選択されています。しかし、最大のスループットや最小限の遅延、最大の CPU パフォーマンスなど、非常に特殊な使用が絶対的に必要とされるような場合は、デフォルト設定の再検討が必要となる可能性があります。

以下の質問とその答えをお読みいただいた上で、 本ガイドで説明している技術を使用してマシンを最適化すべきかどうかを判断してください。

問： マシンを最適化すべきですか？

答：電力を最適化する重要性は、お客様に従うべきガイドラインがあるか、または順守すべき規則があるかによって変わってきます。

問： どの程度、最適化する必要がありますか？

答： ここで紹介している技術の中には、マシンを詳しく監査、分析する行程すべてを通して行なう必要がなく、その代わりに電力の使用を標準的に改善する全般的な最適化を行なえばよいものがあります。もちろん手作業で行うシステム監査と最適化ほど優れてはいませんが、適度な効果はあります。

問： 最適化によりシステムパフォーマンスが許容範囲を下回りませんか？

答： 本ガイドで説明している技術のほとんどがシステムパフォーマンスに明らかな影響を与えます。Red Hat Enterprise Linux 6 に既に設定されているデフォルトを越える電力管理を実装させる場合は、電力最適化の実行後、システムパフォーマンスを監視してパフォーマンスの低下が許容範囲内であるか判断する必要があります。

問： 最適化に時間とリソースを費した場合、そこから得られる結果より負担の方が大きくなってしまいませんか？

答： 1 台のシステムに対して全行程を手作業で行なっていく最適化については、費される時間とコストが 1 台のマシンの寿命が尽きるまでに得られるであろう恩恵をはるかに上回ってしまうため、一般的には意味がありません。一方、例えば 1 万台のデスクトップシステムに同じ構成と設定を持たせて、複数のオフィスへの実装を展開する場合には、最適化した設定をひとつ構成してそれを 1 万台すべてのマシンに適用していけば十分に役立つ可能性が高くなります。

次のセクションでは、最適なハードウェアのパフォーマンスがエネルギー消費の観点でどのようにシステムに恩恵をもたらすのかについて解説していきます。

1.2. 電力管理の基礎

効率的な電力管理は以下の原則の上に成り立っています。

アイドル状態の CPU は必要な時にウェイクアップする

Red Hat Enterprise Linux 5 のカーネルは、各 CPU に対して定期的なタイマーを使用していました。このタイマーにより、CPU がアイドル状態になるのを防いでいます。プロセスが実行中であるかどうかに関係なく、CPU は毎回のタイマーイベント (設定により数ミリ秒毎に発生) を処理する必要があります。効率的な電力管理を行うための主なポイントは、CPU がウェイクアップする頻度を少なくすることです。

このため、Red Hat Enterprise Linux 6 の Linux カーネルには、定期的なタイマーがありません。その結果、CPU の アイドル状態は *ティックレス (tickless)* になります。これにより、CPU がアイドル状態の時には不要な電力を消費させないようにします。ただし、システムに不要なタイマーイベントを作成するアプリケーションがあると、この機能がもたらす利点が相殺されてしまう恐れがあります。こうした不要なタイマーイベントの例としてはポーリングイベントなどがあります (ボリューム変更の確認、マウス動作の確認など)。

Red Hat Enterprise Linux 6 には、CPU の使用量でアプリケーションを識別し、監査を行なうことができるツールが同梱されています。詳細は、2章 *電力管理の監査と分析* を参照してください。

使用していないハードウェアとデバイスは完全に無効にする

これは、可動パーツを持つデバイス (例えば、ハードディスク) に特に当てはまります。さらに、一部のアプリケーションは、使用していないが有効なデバイスを「オープン」状態にすることがあります。これが起こると、カーネルはデバイスが使用中だと想定し、デバイスが節電状態に入ることを阻止する可能性があります。

動作が少ないということは消費電力も少ない

ただし多くの場合、これは最新のハードウェアと正しい BIOS 設定によって異なります。現在 Red Hat Enterprise Linux 6 では対応できるようになった新しい機能の一部は、旧式のシステムコンポーネントでは対応していないことが多々あります。システムに最新の公式のファームウェアが使用されていること、また BIOS の電力管理またはデバイス設定のセクションで電力管理の機能が有効になっていること、を確認してください。確認する機能は以下のとおりです。

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

上記の機能がハードウェアでサポートされ、BIOS で有効になっている場合、Red Hat Enterprise Linux 6 ではその機能がデフォルトで使用されます。

CPU の各種状態とその効果

ACPI (*電力制御インタフェース：Advanced Configuration and Power Interface*) を搭載する最新の CPU は、以下の 3 種類の電力状態を提供します。

- Sleep (C 状態)
- Frequency (P 状態)
- Heat output (T 状態、または「温度状態」)

最小限のスリープ (sleep) 状態で稼働している CPU は、最小限のワット数を消費しますが、必要なときにこの状態から復帰するには相当な時間がかかります。非常に稀なケースですが、CPU がスリープに入る度に直ちに復帰する必要がある場合もあります。この場合、事実上 CPU が常にビジーな状況を引き起こし、別の状態を使用していたら実現できた節電効果が得られなくなってしまうことになります。

電源がオフになっているマシンの消費電力は最小となる

当たり前かもしれませんが、実際に節電を行う最善策の１つは、システムの電源を切ることです。例えば、「グリーン IT」の意識に焦点を置いた企業文化を育み、昼休みや帰宅時にはマシンの電源を切るガイドラインを設けるのも一案です。また、数台の物理サーバーを大きなサーバー 1 台に統合し、Red Hat Enterprise Linux 6 で配布されている仮想化技術を使用して仮想化することもできます。

第2章 電力管理の監査と分析

2.1. 監査と分析の概要

たった一台のシステムに対して監査や分析、チューニングを細かく手作業で行うことは、通常、例外的です。こうしたシステムのチューニングの作業にかかる時間やコストが、一般的にはその作業から得られる恩恵を上回ってしまうためです。しかし、この作業を一度だけ行い、同じ設定をほぼ同一構成となる大量のマシンに再利用できる場合には、非常に便利です。たとえば、数千に及ぶデスクトップシステムの導入、あるいはほぼ同一構成の複数マシンから成る HPC クラスタの導入などを考えてみてください。監査や分析を行うもうひとつの理由は、将来的にシステムの動作に起こる後退や変化を特定できるよう比較対象となる基準を設けるということです。ハードウェアや BIOS、ソフトウェアなどの定期更新によって予想以上の電力消費が発生するのを避けたい場合などに、この分析結果が非常に役立ちます。一般的には、徹底的な監査や分析を行うことで、特定のシステムで発生している現状を把握できるようになります。

電力消費に関する監査と分析は、最新システムを使用しても比較的難しいものです。ほとんどのシステムには、ソフトウェアを介する電力使用量を測定するために必要な手段が搭載されていません。ただし、例外はあります。Hewlett Packard サーバーシステムの ILO 管理コンソールには、ウェブ経由でアクセスできる電力管理モジュールが備わっています。IBM では、BladeCenter 電力管理モジュールで同様のソリューションを提供しています。Dell システムの一部でも、IT Assistant 機能により電力監視機能が提供されています。他のベンダーでもサーバープラットフォーム向けには似たような機能を提供している可能性はありますが、すべてのベンダーで対応しているソリューションは存在しません。システムに電力消費量を測定するメカニズムが組み込まれていない場合は、選択肢がいくつかあります。そのひとつは、USB ポート経由で電力消費情報を提供する特殊な電力供給装置をインストールすることです。さらに、USB で接続するタイプの Watts up? PRO など、外付けのワットメーターなどがあります。

電力消費量を測定する直接的な方法としては多くの場合、できるだけ節電に最大限の努力をすることしかありません。変更が反映されているか、システムがどのように動作しているか査定する他の方法もあります。この章では、そのために必要なツールについて説明します。

2.2. POWERTOP

Red Hat Enterprise Linux 6 ではティックレスカーネル (「Tickless Kernel」 を参照) を採用することで、CPU がより頻繁にアイドル状態に入るようになり、電力消費量が抑えられるため、電力管理が向上しています。新しいツールとなる **PowerTOP** は、CPU を頻繁にウェイクアップさせるカーネルとユーザースペースのアプリケーションの特定コンポーネントを識別します。**PowerTOP** は、「ユーザースペースの最適化」 に説明されている監査を行う開発段階で使用されていました。それにより、このリリースで多くのアプリケーションのチューニングが行われ、不必要に CPU がウェイクアップする率が約 10分の１に低下しました。

Red Hat Enterprise Linux 6 には **PowerTOP** のバージョン **2.x** が装備されています。これは、**1.x** コードベースの完全な書き換えです。これはより明確なタブベースのユーザーインターフェイス機能があり、カーネルの **"perf"** インフラストラクチャーを徹底的に使用してより正確なデータを提供します。システムデバイスの電力動作が追跡され、明確に表示されることで、問題を迅速に指摘することが可能です。**2.x** コードベースにはより実験的に、電力予測エンジンが含まれています。これは、個別のデバイスおよびプロセスが消費している電力を示すことができます。図2.1「実行中の PowerTOP」 を参照してください。

PowerTOP をインストールするには、**root** で以下のコマンドを実行します。

```
yum install powertop
```

PowerTOP を実行するには、**root** で以下のコマンドを実行します。

```
powertop
```

PowerTOP はシステム全体の電力使用量の予測を提供し、各プロセス、デバイス、カーネル作業、タイマー、割り込みハンドラーの電力使用量を表示することができます。このタスク中は、ノート PC をバッテリー電源で稼働してください。電力予測エンジンを調整するには、**root** で以下のコマンドを実行します。

```
powertop --calibrate
```

調整には時間がかかります。このプロセスでは様々なテストが実行され、輝度レベルおよびスイッチデバイスのオンとオフが繰り返されます。調整中にはマシンに触れないでください。調整プロセスが終わると、**PowerTOP** が正常に開始されます。データ収集をおよそ 1 時間実行させます。十分なデータが収集されると、最初のコラムに電力予測の数字が表示されます。

ノート PC でこのコマンドを実行する場合は、バッテリー電源で稼働することで利用可能なデータすべてが提供されます。

PowerTOP は実行中にシステムから統計数字を収集します。**Overview** タブでは、CPU にウェイクアップを最も頻繁に送信するコンポーネントまたは最も電力を消費しているコンポーネントのリストが表示されます (図2.1「実行中の PowerTOP」 を参照)。その横のコラムでは、電力消費予測、リソースの使用方法、1 秒あたりのウェイクアップ、プロセスやデバイス、タイマーなどコンポーネントの分類、およびコンポーネントの説明が表示されます。1 秒あたりのウェイクアップは、サービスまたはカーネルのデバイスおよびドライバーのパフォーマンスの効率性を示します。ウェイクアップが少ないと消費電力も少ないことになります。コンポーネントは、電力使用量の最適化がさらに実行可能な度合いで並んでいます。

ドライバーコンポーネントのチューニングは通常、カーネルの変更を必要とし、これは本ガイドの対象外となります。しかし、ウェイクアップを送信するユーザースペースのプロセスは、管理がより簡単です。最初に該当するサービスまたはアプリケーションがシステム上で実行する必要があるかどうかを判断します。不要な場合は、単にこれを停止します。古い System V サービスを永続的にオフにするには、以下を実行します。

```
chkconfig off servicename off
```

このプロセスについてより詳細な情報を得るには、**root** で以下のコマンドを実行します。

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

トレースが繰り返し行われているように見える場合は、恐らくビジーループが発生しています。このようなバグを修復するには通常、そのコンポーネントでコードを変更する必要があります。

図2.1「実行中の PowerTOP」 では、消費電力量の合計と、該当する場合はバッテリーの残量が表示されます。これらの下には、1 秒あたりのウェイクアップ合計、1 秒あたりの GPU 操作、および 1 秒あたりの仮想ファイルシステム操作の概要があります。画面の残りには、使用量にしたがってプロセス、割り込み、デバイス、およびリソースの一覧が表示されます。適切に調整されると、一覧の各アイテムの最初のコラムに電力消費量の予測も表示されます。

タブを移動するには、**Tab** および **Shift+Tab** キーを使用します。**Idle stats** タブでは、すべてのプロセッサーおよびコアの C 状態の使用が表示されます。**Frequency stats** タブでは、Turbo モード (該当する場合) を含む全プロセッサーおよびコアの P 状態の使用が表示されます。CPU がより高い C または P 状態に長くいればいるほど、よいことになります (**C4** の方が **C3** よりも高い)。これは、CPU 使用率がどの程度うまく最適化されているかを示す指標になります。システムのアイドル中の理想状態は、最高の C または P 状態が 90% 以上を維持していることです。

Device Stats タブは **Overview** タブと同様の情報を表示しますが、デバイスに限定されます。

Tunables タブには、システムの消費電力量を低減させるための提案が含まれています。**up** および **down** キーを使って各提案に移動し、**enter** キーでそれらのオン/オフを切り替えます。

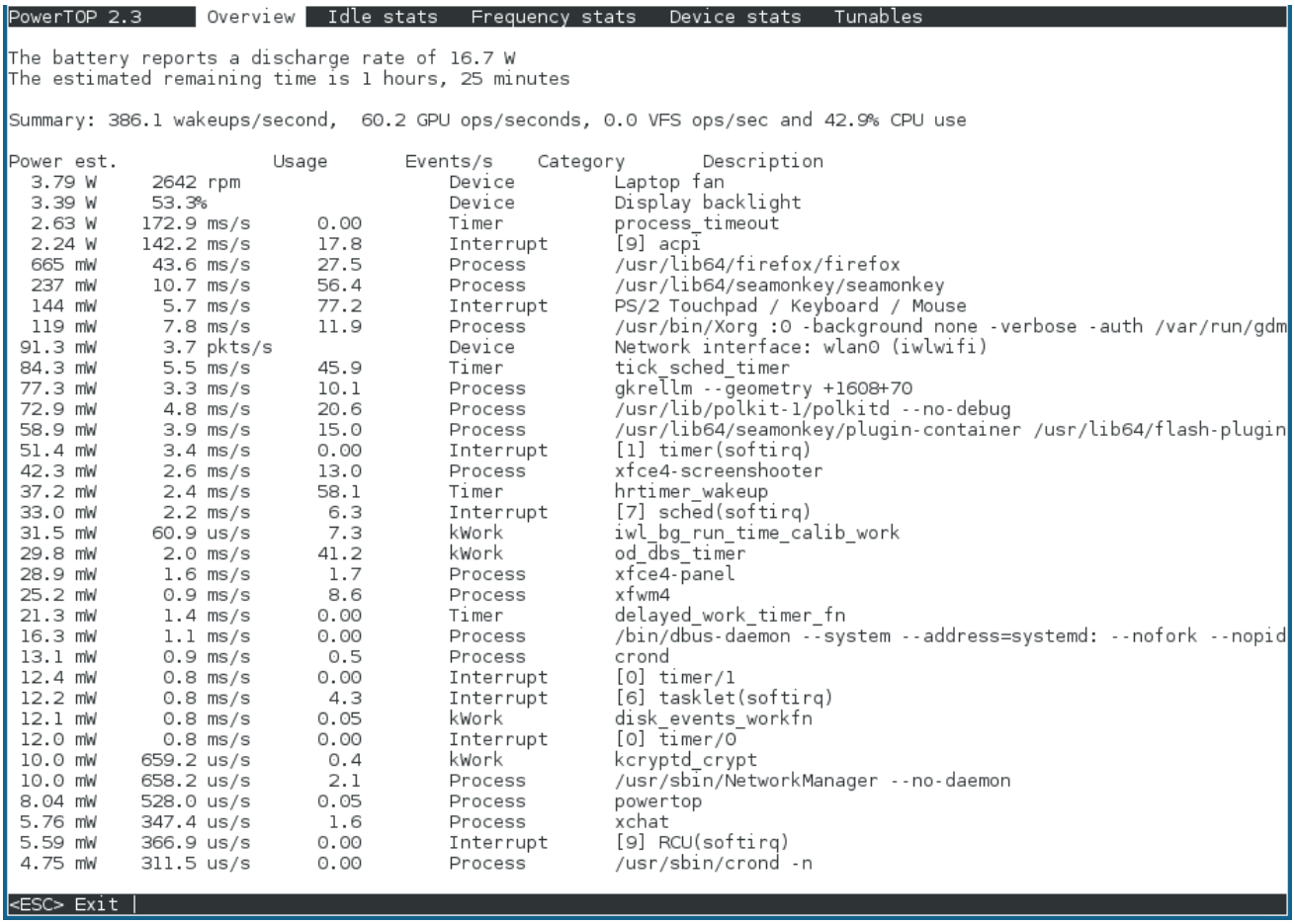


図2.1 実行中の PowerTOP

PowerTOP を **--html** オプションで実行すると、HTML レポートを生成することもできます。*htmlfile.html* パラメーターを希望する出力ファイル名に置き換えます。

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

デフォルトでは、**PowerTOP** は 20 秒間隔で測定を行いますが、**--time** オプションを使うとこれを変更することもできます。

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

PowerTOP プロジェクトの詳細情報については、プロジェクトのホームページ (英語) を参照してください。

PowerTOP は **turbostat** ユーティリティーと併用することもできます。**turbostat** ユーティリティーはレポーティングツールで、Intel 64 プロセッサー上のプロセッサートポロジー、周波数、アイドル状態の電力状態、温度、および電力使用量を表示します。**turbostat** ユーティリティーについての詳細は、**turbostat**(8) man ページまたはパフォーマンスチューニングガイドを参照してください。

2.3. DISKDEVSTAT と NETDEVSTAT

Diskdevstat と **netdevstat** は、システム上で稼働しているすべてのアプリケーションのディスク活動

とネットワーク活動の詳細情報を収集する **SystemTap** ツールです。これらのツールは、あらゆるアプリケーションで引き起こされる CPU のウェイクアップ回数を秒単位で示す **PowerTOP** から発想を得ています (「PowerTOP」 を参照)。これらのツールが収集する統計により、小規模な I/O 動作を数多く行なうことで電力を無駄にしているアプリケーションを特定することができるようになります。転送速度のみを測定する他の監視ツールでは、このタイプの使用量を特定できません。

SystemTap のこれらのツールをインストールするには、**root** で以下のコマンドを実行します。

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

次のコマンドでツールを実行します。

```
diskdevstat
```

あるいは、以下のコマンドを実行します。

```
netdevstat
```

これらコマンドは両方とも以下のように、最大 3 つのパラメータを取ります。

diskdevstat *update_interval total_duration display_histogram*

netdevstat *update_interval total_duration display_histogram*

update_interval

表示が更新される秒単位の間隔。デフォルト:**5**

total_duration

実行完了にかかる秒単位の時間。デフォルト:**86400** (1 日)

display_histogram

実行完了時に全収集データによる度数分布図 (柱状グラフ) を作成するかどうかを指定するフラグ。

以下の出力は **PowerTOP** の出力に似ています。以下に KDE 4.2 を稼働している Fedora 10 システム上の長期の **diskdevstat** 実行からのサンプル出力を示します。

```
  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
```

各コラムの表示内容は、以下のとおりです。

PID
アプリケーションのプロセス ID

UID
アプリケーションの実行元となるユーザー ID

DEV
I/O が発生したデバイス

WRITE_CNT
書き込み動作回数の合計

WRITE_MIN
2 回の連続書き込みに要した最短時間 (秒)

WRITE_MAX
2 回の連続書き込みに要した最長時間 (秒)

WRITE_AVG
2 回の連続書き込みに要した平均時間 (秒)

READ_CNT
読み込み動作回数の合計

READ_MIN
2 回の連続読み込みに要した最短時間 (秒)

READ_MAX
2 回の連続読み込みに要した最長時間 (秒)

READ_AVG
2 回の連続読み込みに要した平均時間 (秒)

COMMAND
プロセスの名前

この例には、非常に目立つアプリケーションが 3 つあります。

```
  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
```

これらの３つのアプリケーションには、**0** 以上の **WRITE_CNT** があり、これは測定中になんらかの書き込みを実行したことを意味します。その中でも、**plasma** が他と大差をつけて一番高い数値を示しています。最多の書き込み動作を実行しているため、当然書き込みの平均時間は最短となります。そのため、電力効率の悪いアプリケーションに懸念がある場合には、**Plasma** が調査の最大のターゲットとなります。

strace コマンドと **ltrace** コマンドを使用して、所定のプロセス ID のすべてのシステムコールを追跡することによりアプリケーションをもっと詳しく検査できます。この例では、以下を実行することが可能です。

```
strace -p 2789
```

この例では、**strace** の出力には、ユーザーの KDE アイコンのキャッシュファイルを書き込みのため開き、直後にそのファイルを再び閉じるという動作が 45 秒毎に繰り返されるパターンが含まれていました。これにより、ファイルのメタデータ (特に修正時間) が変更されたため、それに必要な物理的な書き込みがハードディスクに行なわれました。最終的な修正では、アイコンに更新が加えられなかった場合には、こうした不要なコールが発生しないようになりました。

2.4. BATTERY LIFE TOOL KIT

Red Hat Enterprise Linux 6 では、BLTK (**Battery Life Tool Kit**) が採用されています。バッテリーの寿命とパフォーマンスをシミュレートして解析するテストスイートです。BLTK は、特定のユーザーグループをシミュレートするタスクセットを実行して、その結果を報告します。特にノート PC のパフォーマンスをテストする目的で開発されましたが、BLTK は **-a** を付けて起動すると、デスクトップ PC のパフォーマンスも報告できます。

BLTK を使用すると、実際にマシンを使用しているのと同程度の再現可能な作業負荷を生成できるようになります。例えば、**office** の作業負荷はテキストを書き込み、その中で修正を行います。同じ作業を表計算でも行ないます。BLTK を **PowerTOP** や他の監査用ツール、解析用ツールなどと併用することで、実施した最適化がアイドル状態の時だけでなく、マシンが頻繁に使用されている時にも効果を発揮しているかどうかを検証することができます。全く同じ作業負荷を異なる設定で複数回実行できるため、異なる設定における結果を比較することができます。

以下のコマンドを使用して、BLTK をインストールします。

```
yum install bltk
```

以下のコマンドで、BLTK を実行します。

```
bltk workload options
```

例えば、**idle** の作業負荷を 120 秒間実行するには、以下を実行します。

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

デフォルトで使用できる作業負荷は次の通りです。

-I, --idle

システムはアイドル状態です。他の作業負荷と比較する場合に基準値として使用します。

-R, --reader

ドキュメントを読み込むシミュレートを行います (デフォルトで、**Firefox** を使用)。

-P, --player

CD または DVD ドライブのマルチメディアファイルを見ているシミュレートを行います (デフォルトでは **mplayer** を使用)。

-O, --office

OpenOffice.org スイートを使ったドキュメント編集のシミュレートを行います。

指定できる他のオプションは以下のとおりです。

-a, --ac-ignore

AC 電源が使用可能かどうか無視します (デスクトップで必要)。

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

テストを実行する期間 (秒)；**idle** 作業負荷を使ってこのオプションを使用します。

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

特定の作業負荷で使用されるファイルを指定します。例えば、CD か DVD ドライブにアクセスする代わりに、**player** の作業負荷で再生するファイルです。

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

特定の作業負荷で使用されるアプリケーションを指定します。例えば、**reader** の作業負荷で **Firefox** 以外のブラウザを指定します。

BLTK は、数多くの特殊化したオプションをサポートします。詳細情報は**bltk** man ページを参照してください。

BLTK は、生成する結果を **/etc/bltk.conf** 設定ファイルで指定されたディレクトリーに保存します。デフォルトでは **~/.bltk/*workload*.results.*number*/** です。例えば、**~/.bltk/reader.results.002/** ディレクトリーは **reader** の作業負荷の 3 つ目のテスト結果を保持します (ひとつ目のテストは番号なし)。結果はいくつかのテキストファイルに分散されます。これらの結果を読み取りやすい形式に簡略化するには、以下を実行します。

```
bltk_report path_to_results_directory
```

これにより、結果のディレクトリーに **Report** というテキストファイルでその結果が表示されます。代わりにターミナルエミュレーターでその結果を閲覧するには、**-o** オプションを使用します。

```
bltk_report -o path_to_results_directory
```

2.5. TUNED および KTUNE

Tuned はシステムコンポーネントの使用を監視して、その監視情報を基にして動的にシステム設定をチューニングするデーモンです。動的なチューニングでは、所定のシステムの稼働時間中に各種システムコンポーネントを異なる方法で使用する方法を考慮します。例えば、ハードドライブは起動時とログイン時に大いに使用されますが、主に OpenOffice や 電子メールクライアントのようなアプリケーションを使う時にはほとんど使用されません。同様に CPU とネットワークデバイスは、異なる時期に異なる方法で使用されます。**Tuned** はこうしたコンポーネントの動作を監視して、それらの使用の変化に対応します。

実際の例として、標準的なオフィスにあるワークステーションを考慮してみましょう。ほとんどの時間帯でイーサネットのネットワークインターフェースがほぼ使用されていない状態とします。ほんの数件の電子メールの着信や発信がある程度、またはウェブページが数ページ読み込まれる程度だとします。この程度の負荷の場合、ネットワークインターフェースはデフォルト設定のように常にフルスピードで動作する必要はありません。**Tuned** には、ネットワークデバイスを監視してチューニングを行なうプラグインがあり、ネットワークインターフェースの利用率が低くなると、それを検出して自動的にそのインターフェースの速度を落とすことで電力の使用を節約します。例えば、DVD 画像をダウンロードしたり、重い添付ファイルが付いた電子メールを開いたためにインターフェースでの動作が長期に渡って劇的に増加すると、**tuned** はこれを検出して、動作のレベルが非常に高い間は最善のパフォーマンスを提供できるようインターフェース速度を最大に設定します。この原理は CPU やハードディスクなどの他のプラグインにも使用されています。

ネットワークデバイスは、デフォルトではこのように動作するようには設定されていません。速度変更が反映され、ユーザーの使用感に直接、目に見えるほどの影響を与えるまでには数秒かかるためです。CPU やハードドライブのチューニング用プラグインにも同様のことが言えます。ハードドライブが減速すると、再び加速するには数秒はかかるため、この間はシステムからの応答なしと見られてしまいます。この時間差は CPU プラグインの場合がもっとも短時間となりますが、ユーザーにはほとんど感知できない長さとはいえ、少なくとも測定可能な範囲となります。

tuned と並行して、今回は **ktune** も提供しています。**Ktune** は、特定の使用を目的としてマシンのパフォーマンスを最適化するフレームワークおよびサービスとして Red Hat Enterprise Linux 5.3 で導入されました。それ以降、**ktune** は改良を重ねて、現在では一般的なチューニングのフレームワークの固定した部分として使用できるようになりました。**ktune** は、「Tuned-adm」 で説明されているような定義済みの各種のプロファイルで主に使用します。

以下のコマンドを使用して tuned パッケージと関連の **systemtap** スクリプトをインストールします。

```
yum install tuned
```

tuned パッケージをインストールすると、**/etc/tuned.conf** にサンプルの設定ファイルが設定され、デフォルトのプロファイルがアクティベートされます。

以下を実行して **tuned** を開始します。

```
service tuned start
```

マシンを起動する度に **tuned** が開始されるよう以下を実行します。

```
chkconfig tuned on
```

Tuned には、手動で実行する場合に使用できる追加オプションがあります。オプションは以下の通りです。

-d, --daemon

フォアグラウンドではなく、デーモンとして **tuned** を開始します。

-c, --conffile

ファイル名とパスを指定して設定ファイルを使用します。例えば、**--conffile=/etc/tuned2.conf** です。デフォルトでは **/etc/tuned.conf** となります。

-D, --debug

ロギングの最高レベルを使用します。

2.5.1. tuned.conf ファイル

tuned.conf ファイルには、**tuned** の構成を設定する機能があります。デフォルトでは、ファイルは **/etc/tuned.conf** に置かれていますが、**tuned** に **--conffile** オプションを付けて開始すると異なる名前と場所を指定することができます。

設定ファイルには、**tuned** の一般的なパラメータを定義する **[main]** セクションを常に含める必要があります。ファイルには各プラグイン用のセクションがあります。

[main] セクションには、以下のオプションがあります。

interval

tuned がシステムを監視してチューニングする間隔を秒単位で表示します。デフォルト値は**10** です。

verbose

出力を詳細に表示するかどうか指定します。デフォルト値は **False** です。

logging

ログされるメッセージの最低優先度を指定します。使用可能な値は降順で、**critical**、**error**、**warning**、**info**、および **debug** となります。デフォルト値は **info** です。

logging_disable

ログされるメッセージの最高優先度を指定します。この優先度、またはそれ以下の優先度を持つメッセージはログされません。使用可能な値は降順で、**critical**、**error**、 **warning**、**info**、および **debug** となります。値 **notset** はこのオプションを無効にします。

プラグインにはそれぞれセクションがあり、プラグインの名前は角括弧で指定されています。例えば: **[CPUTuning]** です。プラグインにはそれぞれオプションがありますが、以下はすべてのプラグインに適用されます。

enabled

プラグインが有効かどうか指定します。デフォルト値は **True** です。

verbose

出力を詳細に表示するかどうか指定します。指定がないと、**[main]** からの値が継承されます。

logging

ログされるメッセージの最低優先度を指定します。指定がないと、**[main]** からの値が継承されます。

サンプルの設定ファイルは以下のようになります。

```
[main]
interval=10
pidfile=/var/run/tuned.pid
logging=info
logging_disable=notset
```

```
# Disk monitoring section

[DiskMonitor]
enabled=True
logging=debug

# Disk tuning section

[DiskTuning]
enabled=True
hdparm=False
alpm=False
logging=debug

# Net monitoring section

[NetMonitor]
enabled=True
logging=debug

# Net tuning section

[NetTuning]
enabled=True
logging=debug

# CPU monitoring section

[CPUMonitor]
# Enabled or disable the plugin. Default is True. Any other value
# disables it.
enabled=True

# CPU tuning section

[CPUTuning]
# Enabled or disable the plugin. Default is True. Any other value
# disables it.
enabled=True
```

2.5.2. Tuned-adm

システムを詳細に監査、分析することは、非常に時間のかかる作業であり、数ワットの節電のためだけに行う価値はないかもしれません。今までは、こうした煩雑な作業の代わりになる方法は、単にデフォルトを使用することだけでした。そのため、Red Hat Enterprise Linux 6 では、こうした両極端な方法の代替案となるプロファイルを複数用意し、特定の使用例に対応しています。さらには、コマンドラインで複数のプロファイルを簡単に切り替えることができる **tuned-adm** ツールも提供しています。Red Hat Enterprise Linux 6 には、一般的な使用例に則した定義済みのプロファイルが同梱されているので、**tuned-adm** コマンドで選択してアクティベートするだけで使用できるようになります。ただし、プロファイルをユーザー自身で作成したり、修正を行なうほか、削除することも可能です。

利用可能な全プロファイルを一覧表示して、現在アクティブなプロファイルを特定するには、以下を実行します。

```
tuned-adm list
```

現在アクティブなプロファイルだけを表示する場合は、以下を実行します。

```
tuned-adm active
```

別のプロファイルに切り替える場合は、以下を実行します。

```
tuned-adm profile profile_name
```

例を示します。

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

すべてのチューニングを無効にする場合は、以下を実行します。

```
tuned-adm off
```

tuned を初めてインストールする際には、**default** プロファイルはアクティブになっています。Red Hat Enterprise Linux 6 には以下の定義済みプロファイルも含まれています。

デフォルト

デフォルトの節電プロファイルです。利用可能なプロファイルの中で節電への影響が最も少なく、**tuned** の CPU とディスクのプラグインを有効にするだけです。

desktop-powersave

デスクトップシステム向けの節電プロファイルです。SATA ホストアダプター用の ALPM 節電 (「Aggressive Link Power Management」 参照) と **tuned** の CPU、イーサネット、およびディスクプラグインを有効にします。

server-powersave

サーバーシステム向けの節電プロファイルです。SATA ホストアダプター用の ALPM 節電を有効にして、**HAL** による CD-ROM ポーリングを無効にします (hal-disable-polling man ページ参照)。 また、 **tuned** の CPU とディスクプラグインをアクティベートします。

laptop-ac-powersave

AC 電源で稼働中のノート PC 向け節電プロファイルで、中程度の影響があります。SATA ホストアダプター用の ALPM 節電、WiFi 節電、および **tuned** の CPU、イーサネット、ディスクプラグインを有効にします。

laptop-battery-powersave

バッテリーで稼働中のノート PC 向け節電プロファイルで、大きな影響があります。上記プロファイルの全節電メカニズムをアクティベートし、さらにウェイクアップの少ないシステムのマルチコア節電スケジューラーを有効にします。また、 **ondemand** ガバナーをアクティブにし、AC97 オーディオ節電も有効にします。このプロファイルを使用すると、バッテリーで駆動しているノート PC だけでなく、あらゆるシステムで最大限の節電が可能になります。このプロファイルのトレードオフは、 パフォーマンス、特にディスクとネットワーク I/O の待ち時間に顕著な影響がある点です。

spindown-disk

高い節電率を期待できる標準的なハードディスクを持つマシン向けプロファイルです。ディスクの書き戻し値を増やし、ディスクの **swappiness** 設定値を低くし、またログの同期を無効にすることで積極的なディスク回転の減速を行ないます。パーティションはすべて **noatime** オプションで再マウントされます。**tuned** のプラグインはすべて無効になります。

throughput-performance

標準的なスループットのパフォーマンスチューニング用のサーバープロファイルです。これは **tuned** と **ktune** の節電メカニズムを無効にし、ディスクとネットワーク I/O のスループットのパフォーマンスを向上させる **sysctl** 設定を有効にします。また、**deadline scheduler** に切り替えます。

latency-performance

標準的な遅延パフォーマンスチューニング用のサーバープロファイルです。このプロファイルは、動的チューニングメカニズムと transparent hugepages を無効にします。これは、**cpuspeed** で p 状態用に **performance** ガバナーを使用し、I/O スケジューラーを **deadline** に設定します。また Red Hat Enterprise Linux 6.5 およびそれ以降では、プロファイルは ***cpu_dma_latency*** の値 **1** を要求します。Red Hat Enterprise Linux 6.4 およびそれ以前では、**cpu_dma_latency** は **0** の値を要求していました。

enterprise-storage

企業規模のサーバー構成でスループットパフォーマンスを向上させるサーバー向けプロファイルです。 **deadline scheduler** に切り替わり、 特定の I/O バリアを無効にすることでスループットを大幅に向上させます。

virtual-guest

このプロファイルは、仮想マシン用に最適化されています。enterprise-storage プロファイルをベースとしていますが、仮想メモリーの swap も低減します。このプロファイルは Red Hat Enterprise Linux 6.3 以降で利用可能です。

virtual-host

enterprise-storage プロファイルに基づいている virtual-host は、仮想メモリーの swap を減らし、ダーティーページのより積極的な書き戻しを可能にします。**barrier=0** で root ファイルシステムおよび起動ファイルシステム以外のファイルシステムがマウントされます。さらに Red Hat Enterprise Linux 6.5 では、***kernel.sched_migration_cost*** パラメーターが 5 ミリ秒に設定されます。Red Hat Enterprise Linux 6.5 より前では、***kernel.sched_migration_cost*** はデフォルト値の 0.5 ミリ秒を使用していました。

sap

SAP ソフトウェアのパフォーマンスを最大化するように最適化されたプロファイルです。enterprise-storage プロファイルをベースにしています。sap プロファイルは、共有メモリー、セマフォ、プロセスが所有するメモリーマップの最大数に関する sysctl 設定も調整します。

すべてのプロファイルは **/etc/tune-profiles** の下の個別のサブディレクトリーに保管されています。そのため、 **/etc/tune-profiles/desktop-powersave** には、そのプロファイルに関するすべての必要なファイルと設定が含まれています。これらのディレクトリーにはそれぞれ最大 4 つの以下のようなファイルがあります。

tuned.conf

このプロファイルに対し tuned サービスをアクティブする設定です。

sysctl.ktune

ktune で使用される **sysctl** 設定です。この形式は **/etc/sysconfig/sysctl** ファイルの形式と同じです (sysctl および sysctl.conf man ページを参照)。

ktune.sysconfig

ktune の設定ファイルです。通常は **/etc/sysconfig/ktune** です。

ktune.sh

ktune サービスで使用される **init** スタイルのシェルスクリプトです。システム起動時に、チューニングを行なうための特定のコマンドを実行することができます。

新しいプロファイルを開始する最も簡単な方法は既存のプロファイルをコピーすることです。**laptop-battery-powersave** プロファイルには、既に非常に豊富なチューニングのグループがあるため、ここから始めるのが現実的です。以下のように、ディレクトリー全体を新しいプロファイル名にコピーするだけです。

```
cp -a /etc/tune-profiles/laptop-battery-powersave/ /etc/tune-
profiles/myprofile
```

個々のニーズに合うように、新しいプロファイルのファイルのいずれかを修正します。例えば、CD の変更を検出したい場合は、以下のように **ktune.sh** スクリプト内の該当する行をコメントアウトすることにより、その最適化を無効にすることができます。

```
# Disable HAL polling of CDROMS
# for i in /dev/scd*; do hal-disable-polling --device $i; done > /dev/null
2>&1
```

2.6. DEVICEKIT-POWER と DEVKIT-POWER

Red Hat Enterprise Linux 6 では、**DeviceKit-power** は、**HAL** の一部だった電力管理機能と Red Hat Enterprise Linux の以前のリリースの **GNOME Power Manager** の一部だった電力管理機能を引き継ぎます (「GNOME の電源管理」 も参照してください) 。**DeviceKit-power** はデーモン、API や一連のコマンドラインのツールを提供します。物理デバイスかどうかに関係なく、システム上の電源はデバイスとして表示されます。例えば、ノート PC のバッテリーと AC 電源は両方ともデバイスとして表示されます。

コマンドラインツールにアクセスするには、以下のオプションを付けた **devkit-power** コマンドを使用します。

--enumerate, -e

システム上の電源デバイス用のオブジェクトパスを表示します。例えば以下のとおりです。

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

--dump, -d

システム上のすべての電源デバイス用のパラメータを表示します。

--wakeups, -w

システムの CPU のウェイクアップ を表示します。

--monitor, -m

AC 電源の接続や切断、あるいはバッテリーの低下などの電源デバイスの変化についてシステムを監視します。システムの監視を止めるには、**Ctrl+C** を押します。

--monitor-detail

AC 電源の接続や切断、あるいはバッテリの低下などの電源デバイスの変化についてシステムを監視します。**--monitor-detail** オプションでは、**--monitor** オプションよりも詳細を提供します。システムの監視を止めるには、**Ctrl+C** を押します。

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

特定のオブジェクトパスに関して利用可能なすべての情報を表示します。例えば、オブジェクトパス **/org/freedesktop/UPower/DeviceKit/power/battery_BAT0** で表示されるシステムのバッテリーに関する情報を取得するには、以下を実行します。

```
devkit-power -i /org/freedesktop/UPower/DeviceKit/power/battery_BAT0
```

2.7. GNOME の電源管理

GNOME 電源管理 は、GNOME デスクトップの一部としてインストールされているデーモンです。Red Hat Enterprise Linux の以前のバージョンで **GNOME 電源管理** が提供した電力管理機能の大部分は、Red Hat Enterprise Linux 6 の **DeviceKit-power** の構成要素となりました (「DeviceKit-power と devkit-power」 参照)。ただし、**GNOME 電源管理** はその機能のフロントエンドとして残ります。システムトレイのアプレットを介して **GNOME 電源管理** は、バッテリーから AC 電源への切り替えなど、システムの電源状態の変化を通知します。また、バッテリーの状態を報告すると共に、バッテリーの電力が低くなると警告を出します。

また、**GNOME 電源管理** により、基本的な電力管理設定が可能になります。これらの設定にアクセスするには、システムトレイの **GNOME 電源管理** アイコンをクリックし、**設定** をクリックします。

電源管理の設定 画面にはタブが 2 つあります。

- **AC 電源使用時**
- **全般**

ノート PC の場合、 **電源管理の設定** 画面のタブは 3 つになります。

- **バッテリー使用時**

AC 電源使用時 のタブと **バッテリー使用時** のタブを使用して、システムが非アクティブ時にディスプレイがオフになるまでの経過時間、非アクティブなシステムがスリープになるまでの経過時間、および使用していない時にハードディスクの回転を停止するかどうか指定します。**バッテリ使用時** のタブでは、ディスプレイの輝度の設定、バッテリが非常に低下したシステムでの動作の選択できます。例えば、**GNOME 電源管理** のデフォルト設定では、バッテリーが非常に低くなるとシステムは休止状態になります。**全般** のタブを使用すると、システム上の (物理的な) 電源ボタンとサスペンドボタンの動作を設定でき、**GNOME 電源管理** アイコンがシステムトレイに出現すべき状態を指定できます。

2.8. 他の監査方法

Red Hat Enterprise Linux 6 は、システムの監査と分析を実行するためのツールを他にもかなり多く提供しています。それらの多くは、既に発見した事項を検証する場合、あるいは特定の部分に関するより詳しい情報が必要な場合に補助の情報源として使用できるものです。これらのツールの多くはパフォーマンスチューニングにも使用されます。以下のとおりです。

vmstat

vmstat はプロセス、メモリ、ページング、ブロック I/O、トラップ、および CPU 活動について詳細情報を提供します。システム全体で実行している動作やビジーな部分を詳しく見るために使用します。

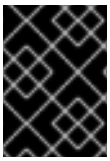
iostat

iostat は **vmstat** と似ていますが、ブロックデバイスの I/O 専用です。詳細な出力と統計も提供します。

blktrace

blktrace は、非常に詳細に渡るブロック I/O のトレースプログラムです。情報をアプリケーションに関連した 1 つずつのブロックに分割します。**diskdevstat** と併せて使用すると大変役立ちます。

第3章 中核となるインフラストラクチャとメカニズム



重要

本章で解説している **cpupower** コマンドを使用する場合は、 **cpupowerutils** パッケージがインストールされていることを確認してください。

3.1. CPU のアイドル状態

x86 アーキテクチャの CPU は、CPU の一部を停止させる設定や低いパフォーマンスで実行させる設定など、様々な状態に対応します。こうした状態は *C 状態* と呼ばれ、使用されていない CPU を部分的に停止させることで節電を可能にしています。C 状態は番号付けされ、C0 から始まり数値が増えていきます。大きな数字ほど、CPU の機能性は低く節電率は高くなります。特定の番号が付いた C 状態は、プロセッサー間でさほど違いませんが、特定の機能に関しては正確にはプロセッサーファミリー間で異なる場合があります。C 状態 0 から 3 は 以下のように定義されています。

C0

稼働中、または実行中の状態。この状態では、CPU は完全に動作中であり、 アイドル状態の部分はありません。

C1, 停止

プロセッサーが何の指示も実行していない状態ですが、一般的には電力が低い状態でもありません。CPU は実質的に遅延なく処理を継続できます。C 状態を提供するプロセッサーはすべて、この状態に対応できなければなりません。Pentium 4 のプロセッサーは、実際には電力消費が低い状態の C1E と呼ばれる拡張型 C1 状態に対応します。

C2, クロック停止

このプロセッサーのクロックが停止している状態ですが、そのレジスタとキャッシュの状態は完全な状態で保持しているため、クロックを再開させると直ちに処理を再開することができます。オプションの状態になります。

C3, スリープ

プロセッサーが実際にスリープ状態に入り、キャッシュの更新をする必要がない状態です。この状態から復帰するには、C2 状態からの復帰に比べ、かなり長い時間がかかります。この状態もオプションになります。

利用可能なアイドル状態および CPUidle ドライバーの統計値を表示させるには、次のコマンドを実行します。

```
cpupower idle-info
```

Nehalem マイクロアーキテクチャを搭載する近年の Intel CPU の特徴は、新しい C 状態である C6 です。これにより、CPU の電圧供給をゼロにまで下げることが可能ですが、通常は 80% から 90% 電力消費量を低減します。Red Hat Enterprise Linux 6 のカーネルには、この新しい C 状態に対する最適化が含まれています。

3.2. CPUFREQ ガバナーの使用

ご使用のシステムで電力消費と発生熱量を低減する効果的な方法の 1 つは、CPUfreq を使用することです。CPU 速度スケールとも呼ばれる CPUfreq により、プロセッサーのクロック速度をオンザフライで

調節できます。これにより、システムは減速したクロック速度で稼働し、節電します。CPUfreq ガバナーが、クロック速度の変更、周波数を変換する時期の決定といった周波数の変換に関する規則を定義します。

ガバナーは、システムの CPU の電力特性を定義し、これは結果的に CPU のパフォーマンスに影響を与えます。ガバナーには作業負荷に関してそれぞれ特有の動作、目的、および適合性があります。このセクションでは、CPUfreq ガバナーの選択および設定方法、各ガバナーの特性、およびガバナーに適している作業負荷の種類について説明します。

3.2.1. CPUfreq ガバナーのタイプ

このセクションでは、Red Hat Enterprise Linux 6 で利用できる様々な種類の CPUfreq ガバナーについて説明しています。

cpufreq_performance

Performance ガバナーは、CPU が最高クロック周波数を使用するように強制します。この周波数は静的に設定され、変化しないため、このガバナーでは、**節電する利点はありません。**このガバナーは、何時間にも渡るような作業負荷が大きい時だけ、しかも CPU がアイドル状態になることがほとんどない (もしくはまったくならない) 時のみに適しています。

cpufreq_powersave

一方、Powersave ガバナーは、CPU が最低クロック周波数を使用するように強制します。この周波数は静的に設定され変化しないため、このガバナーでは最大の節電を実現しますが、**CPU パフォーマンスが一番低く** なってしまいます。

しかし「節電 (Powersave) 」という用語は時に誤解を招きます。全負荷で遅い CPU は (原則として) 、負荷がない高速の CPU よりも多くの電力を消費します。そのため、低活動が予期できる時には Powersave ガバナーを使用するよう CPU を設定することが推奨されますが、この期間中に予期しない高負荷が発生するとシステムは実際にはより多くの電力を消費することがあります。

Powersave ガバナーは簡単にいうと、CPU にとっては「節電」よりも「スピードリミッター」の意味を持ちます。これは、過熱が問題となる恐れがあるシステムや環境で最も役立ちます。

cpufreq_ondemand

Ondemand ガバナーは動的なガバナーです。システム負荷が大きい時は、CPU は最高クロック周波数を実現し、システムがアイドル状態の時には、CPU は最低クロック周波数を実現します。これにより、システム負荷に対してシステムは電力消費量を適宜調節できますが、そうすることで **周波数変換の間の遅延** が発生してしまいます。そのため、システムがアイドル状態と高負荷の間で頻繁に替わりすぎると、遅延により Ondemand ガバナーが実現できるパフォーマンスおよび/または節電の利点が少なくなる恐れがあります。

ほとんどのシステムでは、Ondemand ガバナーは熱の放出、消費電力、パフォーマンス、および管理のしやすさの間で、最良の妥協策を提供します。1 日の中で特定の時間帯にのみシステムがビジーになる場合は、Ondemand ガバナーはそれ以上介入せずに、負荷に応じて最高周波数と最低周波数の間で自動的に切り替わります。

cpufreq_userspace

Userspace ガバナーにより、ユーザースペースプログラム (または root で実行しているプロセス) は周波数を設定できます。このガバナーは通常、**cpuspeed** デーモンと併せて使用されます。Userspace ガバナーは、すべてのガバナーの中で最もカスタマイズ可能であり、設定によってはご使用のシステムにパフォーマンスと電力消費との間で最良のバランスを実現します。

cpufreq_conservative

Ondemand ガバナーと同様に、Conservative ガバナーも使用量に応じてクロック周波数を調節します

(Ondemand ガバナーと同様です) 。ただし、Ondemand ガバナーがより積極的にクロック周波数を調節するのに対し (最高周波数から最低周波数、そして最高周波数に戻る) 、Conservative ガバナーはもっとゆっくりと調節を行います。

これが意味しているのは、Conservative ガバナーは単に最高と最低の周波数を選択するのではなく、負荷に対して適切と判断するクロック周波数に合わせるということです。これは電力消費に著しく貢献する可能性がありますが、Ondemand ガバナーよりも **長い遅延で** 行います。

注記

cron ジョブを使用してガバナーを有効にできます。これにより、1 日のある時間帯にあるガバナーを自動的に設定することができます。そのため、アイドル状態 (例えば終業後) の時には、低周波数のガバナーを指定し、高負荷となる時間帯には高周波数に戻るように設定できます。

特定のガバナーを有効にする方法については、「CPUfreq の設定」 の 手順 3.2「CPUfreq ガバナーの有効化」 を参照してください。

3.2.2. CPUfreq の設定

CPUfreq ガバナーを選択して設定する前に、最初に適切な CPUfreq ドライバーを追加する必要があります。

手順3.1 CPUfreq ドライバーの追加方法

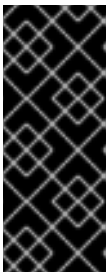
1. 以下のコマンドを使用すると、ご使用のシステムに使用可能な CPUfreq ドライバーが表示されます。

```
ls /lib/modules/[kernel version]/kernel/arch/[architecture]/kernel/cpu/cpufreq/
```

2. **modprobe** を使用して、適切な CPUfreq ドライバーを追加します。

```
modprobe [CPUfreq driver]
```

上記のコマンドを使用する時、ファイル名の接尾辞 **.ko** を忘れずに削除してください。



重要

適切な CPUfreq ドライバーを選択する場合は、常に **p4-clockmod** より **acpi-cpufreq** を選択してください。**p4-clockmod** ドライバーを使用すると、CPU のクロック周波数は低くなりますが、電圧は低下しません。その一方、**acpi-cpufreq** を使用すると CPU のクロック周波数とともに電圧も低くなり、パフォーマンスの低下により電力消費と発生熱量が低くなります。

また以下を実行すると、 特定の CPU に使用できるガバナーを表示させることもできます。

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

一部の CPUfreq ガバナーは、利用できないことがあります。その場合は、**modprobe** を実行して、使用したい CPUfreq ガバナーを有効にするために必要なカーネルモジュールを追加します。これらのカーネルモジュールは、**/lib/modules/*[kernel version]*/kernel/drivers/cpufreq/** で利用

可能です。

手順3.2 CPUfreq ガバナーの有効化

1. ご使用の CPU に利用できないガバナーがある場合は、 **modprobe** を実行して、使用したいガバナーを有効にします。例えば、ご使用の CPU 用に **ondemand** ガバナーが利用できない場合は、以下のコマンドを使用します。

```
modprobe cpufreq_ondemand
```

2. ガバナーが CPU で利用可能と表示されたら、 以下を実行して有効にします。

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

3.2.3. CPUfreq ポリシーおよび速度のチューニング

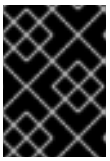
該当する CPUfreq ガバナーを選択すると、 **cpupower frequency-info** コマンドでCPU 速度とポリシー情報を表示させることができるようになります。 さらに、 **cpupower frequency-set** のオプションを使うと各 CPU の速度を調整することができます。

cpupower frequency-info には、 以下のようなオプションを使用することができます。

- **--freq** – CPUfreq コアに準じて現在の CPU の速度を KHz 単位で表示します。
- **--hwfreq** – ハードウェアに準じて現在の CPU の速度を KHz 単位で表示します (root による実行でのみ可)。
- **--driver** – この CPU で周波数の設定に使用している CPUfreq ドライバーを表示します。
- **--governors** – このカーネルで使用できる CPUfreq ガバナーを表示します。このファイルには表示されていない CPUfreq ガバナーを使用したい場合は、 「CPUfreq の設定」 の 手順 3.2「CPUfreq ガバナーの有効化」 で使い方を参照してください。
- **--affected-cpus** – 周波数調整ソフトウェアを必要とする CPU を一覧表示します。
- **--policy** – 現在の CPUfreq ポリシーの範囲 (KHz 単位) と現在アクティブなガバナーを表示します。
- **--hwlimits** – CPU 使用できる周波数を KHz 単位で一覧表示します。

cpupower frequency-set では、以下のオプションを使用することができます。

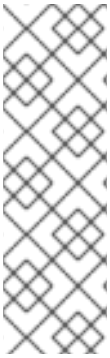
- **--min <*freq*>** と **--max <*freq*>** – CPU の *ポリシーの限界* を KHz 単位で設定します。



重要

ポリシーの限界を設定する場合は、 **--max** を先に設定してから **--min** を設定してください。

- **--freq <*freq*>** – CPU に特定のクロック速度を KHz 単位で設定します。設定できる速度は CPU のポリシーの限界範囲内に限られます (**--min** と **--max**)。
- **--governor <*gov*>** – 新しい CPUfreq ガバナーを設定します。



注記

cpupowerutils パッケージをインストールしていない場合は、CPUfreq の設定は **/sys/devices/system/cpu/*[cpuid]*/cpufreq/** 内にある調節可能値で確認することができます。たとえば、cpu0 の最小クロック速度を 360 KHz に設定する場合は次のコマンドを使用します。

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

3.3. CPU モニター

cpupower には、 アイドルとスリープ状態の統計値および周波数情報を提供しプロセッサーのトポロジーに関してレポートを行なう各種のモニター機能が備わっています。プロセッサー固有のモニターもあれば、あらゆるプロセッサーに互換性があるものもあります。各モニターの測定対象や互換性のあるシステムなどに関する詳細については、**cpupower-monitor** の man ページをご覧ください。

次のオプションは、**cpupower monitor** コマンドに付けて使用します。

- **-l** – システムで使用できる全モニターを一覧表示します。
- **-m *<monitor1>, <monitor2>*** – 特定のモニターを表示します。識別子については **-l** を実行して確認します。
- **command** – 特定コマンドに関する CPU の需要とアイドル統計値を表示します。

3.4. CPU 節電ポリシー

cpupower を使うと、プロセッサーの節電ポリシーが調整できます。

次のオプションは **cpupower set** コマンドに付けて使用します。

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

対応している Intel プロセッサ上のソフトウェアがよりアクティブに、最適なパフォーマンスと節電とのバランスを確定できるようにします。このオプションは他の節電ポリシーを上書きするものではありません。割り当てる値は 0 から 15 の間で、0 が最適なパフォーマンスとなり、15 なら最適な電力効率となります。

デフォルトでは、このオプションはすべてのコアに適用されます。コア別に適用する場合は、 **--cpu *<cpulist>*** オプションを追加します。

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

他の CPU パッケージが選ばれるまで、一つの CPU パッケージ内のコアに対するシステムプロセッサの電力使用を制限します。0 は制限なし、1 は最初は CPU パッケージをひとつだけ採用、2 は 1 に加えてタスクの復帰を処理する場合にセミアイドルの CPU パッケージを優先します。

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

他の コアが選ばれるまで、一つの CPU コアの複数スレッドに対するシステムプロセッサの電力使用を制限します。0 は制限なし、1 は最初は CPU パッケージをひとつだけ採用、2 は 1 に加えてタスクの復帰を処理する場合にセミアイドルの CPU パッケージを優先します。

3.5. サスペンドと復帰

システムがサスペンド状態になると、カーネルはドライバーを呼び出してその状態を保存し、それからドライバーをアンロードします。システムが復帰する時には、ドライバーを再読み込みし、デバイス群を再プログラムします。このタスクを遂行するドライバーにより、システムが正常に復帰できるかどうかが決まります。

この点では、ビデオドライバーが特に問題です。その理由は、ACPI (*電力制御インタフェース：Advanced Configuration and Power Interface*) 規格では、システムファームウェアがビデオハードウェアを再プログラムできる必要がないためです。そのため、ビデオドライバーがハードウェアを完全な未初期化の状態からプログラムできない限りは、システムは復帰できないことがあります。

Red Hat Enterprise Linux 6 では、新しいグラフィックスチップセットをより強力にサポートしています。これにより、サスペンドと復帰は以前より多くのプラットフォームで機能します。特に、NVIDIA チップセットに対するサポートは格別に向上しており、GeForce 8800 シリーズでは特に改善されています。

3.6. TICKLESS KERNEL

Linux カーネルは、以前はプラットフォームに応じて所定の周波数 – 100 Hz、250 Hz、 あるいは 1000 Hz で、システムの各 CPU の割り込みが定期的に行われていました。カーネルは CPU が実行しているプロセスについてクエリを出し、その結果をプロセスアカウンティングとロードバランシングに使用していました。CPU の電力状態に関係なく、カーネルは *タイマーティック (timer tick)* として知られるこの割り込みを実行していました。そのため、アイドル状態の CPU でも最大で毎秒 1000 ものこうしたリクエストに応答していたのです。アイドル状態の CPU に省電力機能を実装しているシステムでは、こうした節電による利点を得られるようにタイマーティックは CPU が長い間アイドル状態にならないようにしていました。

Red Hat Enterprise Linux 6 のカーネルは、*ティックレス (tickless)* を実行します。つまり、旧式の定期タイマーの割り込みが、オンデマンド型の割り込みに取って代わられます。そのため、アイドル状態の CPU は、新しいタスクがプロセスにキューされるまではアイドル状態を維持します。そして電力消費が低い状態にある CPU はその状態をより長く維持できます。

3.7. ACTIVE-STATE POWER MANAGEMENT

Active-State Power Management (ASPM) は、接続するデバイスが使用中でない時に PCIe リンク用に電力状態を低く設定することで *Peripheral Component Interconnect Express* (PCI Express または PCIe) サブシステムで電力を節約します。ASPM はリンクの両端で電力状態を制御して、リンク末端のデバイスで電力が最大の場合でもリンク内で電力を節約します。

ASPM が有効な時には、異なる電力状態の間でのリンクを切り替えるために時間が必要なため、デバイスの遅延は大きくなります。ASPM には、電力状態を決定する以下の 3 つのポリシーがあります。

デフォルト

システムのファームウェア (例えば、BIOS) で指定されたデフォルトに従って、PCIe リンクの電力状態を設定します。これが ASPM のデフォルト状態です。

powersave

パフォーマンスの低下に関係なく、できる限り電力を節約するように ASPM を設定します。

performance

PCIe リンクが最大パフォーマンスで稼働できるように ASPM を無効にします。

APSM のサポートは ***pcie_aspm*** カーネルパラメータで有効にしたり無効にしたりすることができます。 **pcie_aspm=off** を使うと ASPM は無効になり、 **pcie_aspm=force** にすると ASPM は有効になります。ASPM に対応していないデバイス上でも使用できます。

APSM のポリシーは **/sys/module/pcie_aspm/parameters/policy** で設定しますが、 ***pcie_aspm.policy*** カーネルパラメーターを使って起動時に指定することも可能です。 例えば、 **pcie_aspm.policy=performance** を使用すると ASPM の performance ポリシーに設定されます。

警告

pcie_aspm=force を設定すると、ASPM をサポートしていないハードウェアでは、システムが反応しなくなる恐れがあります。**pcie_aspm=force** を設定する前に、システム上のすべての PCIe ハードウェアが ASPM をサポートすることを確認してください。

3.8. AGGRESSIVE LINK POWER MANAGEMENT

Aggressive Link Power Management(ALPM) は、アイドル時 (I/O が存在しない時) にディスクへの SATA リンクを低電力に設定することにより、ディスクの節電を促進する節電技術です。I/O リクエストがそのリンクにキューされると、ALPM は SATA リンクを自動的にアクティブな電力状態に設定し直します。

ALPM で導入された節電には、ディスクの遅延が伴います。そのため、アイドル状態の I/O 時間が長くなると思われる場合にのみ ALPM を使用するべきです。

ALPM は、*Advanced Host Controller Interface*(AHCI) を使用する SATA コントローラ上でのみ利用できます。AHCI についての詳細情報は、http://www.intel.com/technology/serialata/ahci.htm を参照してください。

利用可能時には、ALPM はデフォルトで有効になっています。ALPM には以下の 3 つのモードがあります。

min_power

このモードは、ディスクに I/O がない時に最小電力状態 (SLUMBER) へのリンクを設定します。このモードはアイドル時間が長くなると思われる場合に役立ちます。

medium_power

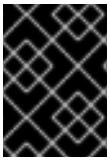
このモードは、ディスク上に I/O がない時に 2 番目に電力が低い状態へのリンクを設定します。このモードでは、パフォーマンスへの影響ができるだけ少なくなるように、リンクの電力状態を切り替えることができます (例えば、一時的に I/O が多くなりまたアイドルになる間) 。

medium_power モードでは、負荷に応じてリンクが PARTIAL と電力が最大の ACTIVE 状態の間で切り替え可能になります。PARTIAL から SLUMBER に、そして PARTIAL に戻るリンクを直接切り替えることはできません。このような場合、どの電力状態も最初に ACTIVE 状態を経由せずに、他に切り替わることはできません。

max_performance

ALPM は無効です。ディスクに I/O がない時は、リンクは低電力状態になりません。

ご使用の SATA ホストアダプタが実際に ALPM に対応しているかどうか確認するには、**/sys/class/scsi_host/host*/link_power_management_policy** ファイルが存在するかどうか確認します。設定を変更するには、このセクションに記載してある値をこのファイルに書き込むか、あるいはファイルを表示して現在の設定を確認します。



重要

ALPM を **min_power**、または **medium_power** に設定すると、自動的に「ホットプラグ」機能を無効にします。

3.9. RELATIME ドライブアクセス最適化

POSIX 基準では、各ファイルが最後にアクセスされた時間を記録するファイルシステムのメタデータがオペレーティングシステムによって維持されていなければなりません。このタイムスタンプは **atime** と呼ばれ、これを維持するにはストレージに常時書き込みをする動作が必要になります。これらの書き込みにより、ストレージデバイスとそのリンクに常に電源が投入され、ビジー状態になります。**atime** データを使用するアプリケーションは少ないため、このストレージデバイスの動作が電力を浪費していることになります。重要なことは、ストレージへの書き込みは、ファイルがストレージからではなくキャッシュから読み込まれた場合でも発生する点です。これまで、Linux カーネルでは **mount** 用の **noatime** オプションに対応してきたため、このオプションでマウントされたファイルシステムには **atime** データを書き込んでいませんでした。しかし、単純に**atime** データを使用しないことにも問題があります。一部のアプリケーションは **atime** データに依存しているため、これが利用できないと機能しないためです。

Red Hat Enterprise Linux 6 で使用しているカーネルは、代替となる **relatime** に対応しています。**Relatime** では **atime** データを維持しますが、ファイルがアクセスされる度の書き込み動作はしません。このオプションを有効にすると、ファイルが変更された、つまり **atime** が更新された (**mtime**) 場合、またはファイルが最後にアクセスされてから一定以上の時間 (デフォルトでは 1 日) が経過している場合に限り、**atime** データがディスクに書き込まれます。

デフォルトでは、**relatime** が有効な状態ですべてのファイルシステムがマウントされるようになります。特定のファイルシステムに対してこのオプションを無効にしたい場合には、そのファイルシステムをマウントする際に **norelatime** オプションを使用します。

3.10. パワーキャッピング (POWER CAPPING)

Red Hat Enterprise Linux 6 では、HPの *Dynamic Power Capping* (DPC) や Intel Node Manager (NM) テクノロジーなど、最近のハードウェアに見られるパワーキャッピング (電力制限) 機能を利用しています。パワーキャッピングにより、管理者はサーバーによる電力消費の上限を設定できるだけでなく、より効率的にデータセンターを計画できます。その理由は、既存の電力供給装置に過負荷をかけるリスクが大幅に減少するためです。また、管理者はさらに多くのサーバー群を同じ物理フットプリント (physical footprint) に配置でき、サーバーの電力消費が制限されると、確実に高負荷時に電力需要が利用可能な電力を超えないようにします。

HP Dynamic Power Capping

Dynamic Power Capping は、選ばれた ProLiant と BladeSystem のサーバーで利用できる機能であり、システム管理者が 1 つのサーバー、あるいはサーバーのグループの電力消費量を制限できるようにします。キャップとは、現時点の作業負荷に関係なく、サーバーが超過しない確実な上限のことです。キャップには、サーバーがその消費電力の上限に到達するまでは何の効果もありません。到達した時点で、管理プロセッサーは CPU P状態 とクロックスロットル (clock throttling) を調節して消費電力を制限します。

Dynamic Power Capping は、オペレーティングシステムから独立して CPU の動作を個別に修正します

が、HP の *integrated Lights-Out 2*(iLO2) ファームウェアにより、オペレーティングシステムは管理プロセッサーにアクセスでき、その結果ユーザースペースのアプリケーションは管理プロセッサーにクエリできます。Red Hat Enterprise Linux 6 で使用されているカーネルには HP iLO と iLO2 の ファームウェア用のドライバーが含まれており、プログラムが **/dev/hpilo/dXccbN** で管理プロセッサーにクエリできるようにします。カーネルには、パワーキャッピング機能をサポートするための **hwmon sysfs** インターフェースの拡張と、**sysfs** インターフェースを使用する ACPI 4.0 パワーメーター用の **hwmon** ドライバーが含まれています。これらの機能が一緒になって、オペレーティングシステムとユーザースペースのツールがパワーキャップ用に設定された値とシステムの現在の電力消費量を読み込めるようになります。

HP Dynamic Power Capping についての詳細情報は、http://h20000.www2.hp.com/bc/docs/support/SupportManual/c01549455/c01549455.pdf にある 『HP Power Capping and HP Dynamic Power Capping for ProLiant Servers 』 を参照してください。

Intel Node Manager

Intel Node Manager は、CPU パフォーマンス、ひいては電力消費量を制限するためにプロセッサーの P 状態と T 状態を使用して、システムにパワーキャップをかけます。電源管理ポリシーを設定することにより、管理者は、例えば夜間や週末などのシステムの負荷が低い時に電力消費が低くなるよう設定することができます。

Intel Node Manager は、標準の *電力制御インタフェース (Advanced Configuration and Power Interface)* を通じて、OSPM *オペレーティングシステム向け構成および電力管理 (Operating System-directed configuration and Power Management)* を使用することで CPU のパフォーマンスを調整します。Intel Node Manager が OSPM ドライバーに T 状態への変更を通知すると、そのドライバーは P 状態に対応する変更を加えます。同様に Intel Node Manager が OSPM ドライバーに P 状態への変更を通知すると、ドライバーはそれに応じて T 状態を変更します。こうした変更は自動的に発生し、オペレーティングシステムの介入を必要としません。管理者は *Intel Data Center Manager*(DCM) ソフトウェアを使用して Intel Node Manager の設定と監視を行います。

Intel Node Manager についての詳細情報は、http://communities.intel.com/docs/DOC-4766 にある 『Node Manager – A Dynamic Approach To Managing Power In The Data Center 』 を参照してください。

3.11. 拡張グラフィックス電力管理

Red Hat Enterprise Linux 6 は不必要な消費が発生するソースを取り除くことにより、グラフィックスおよびディスプレイデバイスの節電を行います。

LVDS 再クロック

LVDS *低電圧差動信号 (Low-voltage differential signalling)*とは、電子信号を銅線上で伝えるシステムです。このシステムが応用されている重要な例の 1 つは、ピクセル情報をノート PC の *液晶ディスプレイ* (LCD) 画面に送信することです。すべてのディスプレイには *リフレッシュレート* があります。これはディスプレイがグラフィックコントローラから新しいデータを受け取り、画像を画面に再表示する頻度です。通常、画面は毎秒 60 回新しいデータを受信します (60 Hz の周波数) 。画面とグラフィックコントローラが LVDS でリンクされている時は、LVDS システムはリフレッシュのたびに電力を使用します。アイドル状態の時、多くの LCD 画面のリフレッシュレートは、目立った変化なく 30 Hz まで低下することがあります (リフレッシュレートが低下すると特有のフリッカーが起こる *ブラウン管*(CRT) モニターとは異なります) 。Red Hat Enterprise Linux 6 のカーネルに組み込まれている Intel グラフィックスアダプタ用のドライバーは、自動的にこの *ダウンクロック (downclocking)*を実行し、画面がアイドル状態の時には約 0.5 W の節電をします。

メモリのセルフリフレッシュの有効化

SDRAM *Synchronous dynamic random access memory*– これは、グラフィックスアダプタのビデオメモ

リに使用されます。毎秒何千回もリチャージされるため、個々のメモリセルは保管されているデータを保持します。データはメモリの内外へと移動するためそのデータを管理するその主要機能の他に、メモリコントローラには通常これらのリフレッシュサイクルを開始する役割があります。一方、SDRAM には低電力の *セルフリフレッシュ* モードもあります。このモードでは、メモリは内部タイマーを使用して、そのリフレッシュサイクルを生成します。これにより、現在メモリに保存されているデータを危険にさらすことなく、システムはメモリコントローラをシャットダウンできます。Red Hat Enterprise Linux 6 で使用されているカーネルは、アイドル状態の時に Intel グラフィックスアダプタのメモリにセルフリフレッシュをさせることがあります。これにより約 0.8 W の節電ができます。

GPU クロックの低減

標準的なグラフィカルプロセッシングユニット (GPU) には、その内部回路の各種パーツを制御する内部クロックが含まれています。Red Hat Enterprise Linux 6 で使用されているカーネルは、Intel および ATI の GPU 内の内部クロックの一部の周波数を低くすることができます。GPU コンポーネントが所定時間内に実行するサイクル数を低減すると、それらが実行する必要がなかったサイクルで消費されていたであろう電力を節減します。GPU がアイドル状態の時には、カーネルは自動的にそうしたクロックの速度を遅くし、GPU の活動が増加すると速めます。GPU のクロックサイクルを低下させることで、最大で約 5 W の節電ができます。

GPU の電源オフ

Red Hat Enterprise Linux 6 の Intel と ATI グラフィックスドライバーは、アダプタにモニターが接続されていない時を検出できるため、GPU を完全にシャットダウンすることができます。この機能は、常時モニターを接続していないサーバーで特に重要です。

3.12. RFKILL

多くのコンピュータシステムには、Wi-Fi、Bluetooth、および 3 G デバイスを含む無線送信器が搭載されています。これらのデバイスは電力を消費し、使用していない時には無駄になります。

RFKill は、コンピュータシステムの無線送信器が、クエリ、アクティベート、非アクティブ化されるインターフェースを提供する Linux カーネルのサブシステムです。無線送信器が非アクティブ化されると、それらはソフトウェアが再アクティベートできる状態 (*ソフトブロック*) に置かれるか、またはソフトウェアが再アクティベートできない状態 (*ハードブロック*) に置かれます。

RFKill コアは、サブシステムにアプリケーションプログラミングインターフェース (API) を提供します。RFkill をサポートするように設計されているカーネルドライバーは、この API を使用してカーネルに登録します。また、デバイスを有効および無効にする方法を含んでいます。さらに RFKill コアは、ユーザーアプリケーションが解釈できる通知と、ユーザーアプリケーションが送信器の状態をクエリする方法を提供します。

RFKill インターフェースは **/dev/rfkill** にありますが、システムのすべての無線送信器の現在の状態が含まれています。各デバイスの現在の RFKill の状態は、**sysfs** に登録されています。また、RFKill は RFKill 対応のデバイス内の状態の変化について *uevents* を発行します。

Rfkill は、システム上の RFKill 対応のデバイスをクエリ、変更できるコマンドラインツールです。このツールを取得するには、**rfkill** パッケージをインストールしてください。

コマンド **rfkill list** を使用すると、デバイスの一覧が取得できます。それぞれのデバイスにはそれに関連した **0** から始まる*インデックス番号*があります。このインデックス番号を使用して**rfkill** に対してデバイスのブロックとブロック解除を指示します。例を示します。

```
rfkill block 0
```

上記は、システムの最初の RFKill 対応デバイスをブロックします。

また、**rfkill** を使用してデバイスの特定のカテゴリ、またはすべての RFKill 対応のデバイスもブロック

できます。例を示します。

```
rfkill block wifi
```

システムのすべての Wi-Fi デバイスをブロックします。すべての RFKill 対応デバイスをブロックするには、以下を実行します。

```
rfkill block all
```

デバイスをブロック解除するには、**rfkill block** の代わりに **rfkill unblock** を実行します。**rfkill** がブロックできるデバイスカテゴリの全一覧を取得するには、**rfkill help** を実行してください。

3.13. ユーザースペースの最適化

システムハードウェアで実行される作業量を低減することは、節電における基本となります。3章*中核となるインフラストラクチャとメカニズム* で説明された変更により、消費電力が低い様々な状態でシステムが動作するようにします。しかし、システムハードウェアから不要な作業を要求するユーザースペースのアプリケーションにより、ハードウェアがこうした状態に入らないようにします。Red Hat Enterprise Linux 6 の開発段階では、ハードウェア上の不要な要求を少なくするために以下の領域で監査を実行しました。

ウェイクアップの低減

Red Hat Enterprise Linux 6 は、*ティックレスカーネル*(「Tickless Kernel」 を参照) を使用します。これにより、CPU はより長くより深くアイドル状態にいることができます。しかし、*タイマーティック* だけが CPU が過剰にウェイクアップする原因ではありません。アプリケーションからの関数呼び出しによっても、CPU がアイドル状態に入らない、またはその状態が続かないようにできます。50 以上のアプリケーションで不要な関数呼び出しの回数が減りました。

ストレージとネットワークの IO の削減

ストレージデバイスとネットワークインターフェースへの出入力 (IO) は、デバイスに電力を消費するよう強制します。アイドル時に節電状態になる機能 (例えば、ALPM や ASPM) を持つストレージとネットワークのデバイスでは、IO のトラフィックにより、デバイスがアイドル状態に入らない、またはその状態が続かないようにし、またハードドライブが使用されてない時に回転が停止しないようにできます。ストレージへの過度で不必要な要求は、数種のアプリケーションで最低限に抑制されています。ハードドライブの回転が停止しないようにする要求が特にそうです。

Initscript の監査

必要性に関係なく、自動的に開始するサービスではシステムリソースを無駄にする可能性が高くなります。そうではなく、サービスじゃ可能な限りデフォルトで「オフ」または「オンデマンド」にすべきです。例えば、Bluetooth サポートを有効にする **BlueZ** サービスは、以前は Bluetooth ハードウェアの有無に関係なくシステムの起動時に自動的に稼働していました。今では **BlueZ** initscript は、サービスを開始する前にシステム上に Bluetooth ハードウェアが存在することを確認します。

第4章 使用例

この章では、2 種類のユースケースを使ってこのガイドで説明している分析と設定方法を説明しています。最初は、標準的なサーバーを例にとり、その次は標準的なノート PC を例にして考えてみます。

4.1. 例 – サーバー

今日の一般的な標準サーバーは、Red Hat Enterprise Linux 6 でサポートされている必要なハードウェアの機能がすべて搭載されています。最初に考慮すべきなのは、サーバーの主要な使用目的となる作業負荷の種類です。この情報に基づき、節電のためにどのコンポーネントを最適化するか決定できます。

サーバーのタイプに関係なく、グラフィックス性能は一般的には必要ありません。そのため、GPU 節電はオンのままで結構です。

ウェブサーバー

ウェブサーバーにはネットワークとディスク I/O が必要です。外部の接続スピードによっては、100 Mbit/s で十分かも知れません。 マシンがほとんど静的なページを使用する場合は、CPU のパフォーマンスはあまり重要ではないでしょう。以下のような電力管理の選択肢があります。

- **tuned** にはディスクまたはネットワークのプラグインなし。
- ALPM をオンにする
- **ondemand** ガバナーをオンにする
- ネットワークカードは 100 Mbit/s に制限する

計算サーバー

計算サーバーには主に CPU が必要です。以下のような電力管理の選択肢があります。

- ジョブとデータストレージが発生する場所に応じて、**tuned** のディスク、又はネットワークプラグイン。または バッチモードシステムには、完全にアクティブな **tuned**。
- 使用量によっては、**performance** ガバナー。

メールサーバー

メールサーバーには、多くの場合ディスク I/O と CPU が必要です。以下のような電力管理の選択肢があります。

- **ondemand** ガバナーはオン。CPU パフォーマンスの最後の数パーセントは重要でないためです。
- **tuned** にはディスクまたはネットワークのプラグインなし。
- メールは内部で発生することが多く、1 Gbit/秒 か 10 Gbit/秒 のリンクから利用できるためネットワークスピードは制限しません。

ファイルサーバー

ファイルサーバーの要件はメールサーバーの要件に似ています。しかし使用するプロトコル次第では、さらなる CPU パフォーマンスが必要になる可能性があります。一般的に Samba ベースのサーバーは、NFS よりも CPU を要求して、NFS は一般的に iSCSI よりも CPU を要求します。それでも、**ondemand** ガバナーを使用できるはずです。

ディレクトリーサーバー

ディレクトリーサーバーのディスク I/O の要件は、一般的に低いものです。十分な RAM がある場合は特にそうです。ネットワーク遅延は重要ですが、 ネットワーク I/O はそれほどでもありません。リンクの速度が遅い遅延のネットワークのチューニングを考えられるかも知れませんが、これを特定のネットワークに注意深くテストするようにしてください。

4.2. 例 – ノート PC

電力管理と節電が実に効果をもたらすもうひとつの非常に一般的な対象は、ノート PC です。ノート PC はもともとワークステーションやサーバーよりも大幅に少ないエネルギーを使用するよう設計されているため、絶対的な節電ができる可能性は他のマシンよりも低くなります。ただし、バッテリーモードでは、どんな節電でもノートパソコンのバッテリー寿命を数分でも延長するのに役立ちます。このセクションでは、ノート PC のバッテリーモードにフォーカスしていますが、もちろん AC 電源での使用でもこうしたチューニングの一部、またはすべてを活用することができます。

1 つのコンポーネントの節電は、通常ワークステーションよりもノートパソコンで相対的に大きな効果をもたらします。例えば、100 Mbits/秒 で実行している 1 Gbit/秒 ネットワークインターフェースはおよそ 3–4 ワット節約します。約 400 ワットの合計消費電力を持つ標準的なサーバーには、この節約はおよそ 1 % です。約 40 ワットの合計消費電力を持つノートパソコンでは、この 1 つのコンポーネントの節電は合計でおよそ 10 % になります。

標準的なノート PC での特定の節電最適化としては以下のものがあります。

- システムの BIOS を使用しないすべてのハードウェアを無効にするように設定します。例えば、パラレルポートまたはシリアルポート、カードリーダー、Web カメラ、WiFi および Bluetooth などが可能です。
- スクリーンを見るために最高輝度が必要ない暗めの場所では、ディスプレイ輝度を低くします。そのためには、GNOME デスクトップでは、**システム+設定 → 電力管理** と進みます。KDE デスクトップでは、**アプリケーション起動キックオフ (Kickoff Application Launcher) +コンピュータ+システム設定+高度な設定 → 電力管理** と進みます。または、コマンドラインで **gnome-power-manager** か、**xbacklight** を実行するか、ノート PC でファンクションキーを使用します。
- **tuned-adm** の **laptop-battery-powersave** プロファイルを使用して、一連の節電メカニズムを有効にします。ハードドライブとネットワークインターフェースのパフォーマンスと遅延に影響があることに注意してください。

追加として (代替として) 各種システム設定に少し修正を加えることもできます。

- **ondemand** ガバナーを使用します (Red Hat Enterprise Linux 6 ではデフォルトで有効です) 。
- ノート PC モードを有効にします (**laptop-battery-powersave** プロファイルの一部) 。

```
echo 5 > /proc/sys/vm/laptop_mode
```

- ディスクへのフラッシュ時間を増加させます (**laptop-battery-powersave** プロファイルの一部) 。

```
echo 1500 > /proc/sys/vm/dirty_writeback_centisecs
```

- nmi watchdog を無効にします (**laptop-battery-powersave** プロファイルの一部) 。

```
echo 0 > /proc/sys/kernel/nmi_watchdog
```

- AC97 オーディオ節電機能を有効にします (Red Hat Enterprise Linux 6 ではデフォルトで有効です)。

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

- マルチコア節電を有効にします (**laptop-battery-powersave** プロファイルの一部)。

```
echo 1 > /sys/devices/system/cpu/sched_mc_power_savings
```

- USB 自動サスペンドを有効にします。

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

  USB 自動サスペンドはすべての USB デバイスで正常に機能するわけではありません。

- ALPM の最小電力設定を有効にします (**laptop-battery-powersave** プロファイルの一部)。

```
echo min_power >
/sys/class/scsi_host/host*/link_power_management_policy
```

- relatime を使用してファイルシステムをマウントします (Red Hat Enterprise Linux 6 ではデフォルトです)。

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

- ハードドライブに最善の節電モードをアクティベートします (**laptop-battery-powersave** プロファイルの一部)。

```
hdparm -B 1 -S 200 /dev/sd*
```

- CD-ROM ポーリングを無効にします (**laptop-battery-powersave** プロファイルの一部)。

```
hal-disable-polling --device /dev/scd*
```

- 画面の輝度を **50** かそれ以下に下げます。例えば以下のとおりです。

```
xbacklight -set 50
```

- スクリーンのアイドル状態に DPMS をアクティベートします。

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

- Wi-Fi の電力レベルを低くします (**laptop-battery-powersave** プロファイルの一部)。

```
for i in /sys/bus/pci/devices/*/power_level ; do echo 5 > $i ; done
```

- Wi-Fi を非アクティブ化します。

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

- 有線ネットワークを 100 Mbit/秒 に制限します (**laptop-battery-powersave** プロファイルの一部) 。

```
ethtool -s eth0 advertise 0x0F
```

付録A 開発者へのヒント

すべての優れたプログラミング教本では、メモリ割り当ての問題と特定機能のパフォーマンスを網羅しています。ご自身のソフトウェアを開発するにあたっては、ソフトウェアが実行するシステムで電力消費が増加する可能性があることに注意してください。こうした配慮は、コードのすべてのラインに影響するものではありませんが、頻繁にパフォーマンスのボトルネックになる領域ではコードを最適化することができます。

問題なることが多い手法は以下のとおりです。

- スレッドを使用する。
- 不必要な CPU のウェイクアップ、およびウェイクアップを効率良く使用しない状態。ウェイクアップする必要がある場合は、すべてを一度にできるだけ迅速に実行します (すぐにアイドル状態になるように迅速にすべてを実行します) 。
- **[f]sync()** を不必要に使用する。
- 不必要なアクティブポーリング、または短い通常のタイムアウトを使用する (代わりにイベントに反応する) 。
- ウェイクアップを効率的に使用していない。
- 低効率なディスクアクセス。頻繁なディスクアクセスを回避するために大きなバッファを使用してください。一度に大きなブロックを書き込みます。
- 低効率のタイマーを使用する。可能な限りアプリケーション群 (またはシステム群) にタイマーをグループ化します。
- 過度の I/O、電力消費、またはメモリ使用 (メモリリークを含む) 。
- 不必要に計算を実行する。

以下のセクションでは、これらの領域をさらに詳しく検証していきます。

A.1. スレッドの使用

スレッドを使用するとアプリケーションのパフォーマンスが改善し、より速くなると思われていますが、これはすべてのケースで当てはまるわけではありません。

Python

Python は Global Lock Interpreter [1] を使用するため、スレッドは大規模な I/O 運用でのみ効果的です。**Unladen-swallow** [2] は、コードを最適化できる可能性がある Python の高速実装です。

Perl

Perl のスレッドは、もともとはフォークがないシステム (32-bit Windows オペレーティングシステムのシステムなど) で実行するアプリケーション用に開発されました。Perl のスレッドでは、データはすべての単独スレッド用にコピー (コピーオンライト：Copy On Write) されます。ユーザーはデータ共有のレベルを定義できるため、データはデフォルトでは共有されません。データを共有するには、**threads::shared** モジュールを含める必要があります。しかし、データはコピー (コピーオンライト) されるだけでなく、モジュールはデータのタイ変数も作成します。これでさらに時間がかかり、遅くなります。[3]

C

C のスレッドは同じメモリを共有します。各スレッドはそれ自身のスタックを持ち、カーネルは新しいファイル記述子を作成したり、新しいメモリスペースの割り当てをする必要がありません。C はより多くのスレッドにより多くの CPU のサポートを実際に活用できます。そのため、スレッドのパフォーマンスを最大にするには、C か C++ などの低水準言語を使用すべきです。スクリプト言語を使用している場合は、C バインディングを考慮してください。プロファイラを使用すると、コードの正しく実行していない部分を特定できます。 [4]

A.2. ウェイクアップ (WAKE-UPS)

多くのアプリケーションは、設定ファイルの変更を確認するためにスキャンします。多くの場合、スキャンは例えば毎分など、決まった間隔で実行されます。このスキャンが問題になる理由は、スキャンにより回転が停止しているディスクをウェイクアップさせるためです。最善策は、適切な間隔、適切な確認方法を見つけるか、**inotify** で変更を確認して、イベントに対応することです。**inotify** はファイルまたはディレクトリで様々な変更を確認できます。

例を示します。

```
int fd;
fd = inotify_init();
int wd;
/* checking modification of a file - writing into */
wd = inotify_add_watch(fd, "./myConfig", IN_MODIFY);
if (wd < 0) {
  inotify_cant_be_used();
  switching_back_to_previous_checking();
}
...
fd_set rdfs;
struct timeval tv;
int retval;
FD_ZERO(&rdfs);
FD_SET(0, &rdfs);

tv.tv_sec = 5;
value = select(1, &rdfs, NULL, NULL, &tv);
if (value == -1)
  perror(select);
else {
  do_some_stuff();
}
...
```

このアプローチの優れている点は、確認できる方法が多岐に渡っている点です。

主な制限は、1 つのシステムでは利用できる監視の数が限られていることです。この数は **/proc/sys/fs/inotify/max_user_watches** から取得できます。この数値を変更することは可能ですが、推薦されません。さらに、**inotify** が失敗すると、コードは別の確認方法にフォールバックする必要がありますが、これは通常ソースコードの **#if #define** が多く発生することを意味しています。

inotify についての詳細情報は、inotify man ページを参照してください。

A.3. FSYNC

Fsync は I/O 負荷の高い動作として知られていますが、実際にはそうでない場合もあります。

ユーザーが新しいページに移動するためのリンクをクリックする度、 **Firefox** は **sqlite** ライブラリを呼び出していました。**sqlite** が **fsync** を呼び出すため、そのファイルシステム設定 (主に **data-ordered** モードの **ext3**) が原因で、何も起こらない場合は長い待ち時間が発生していました。同時に別のプロセスが大きなファイルをコピーしている場合には、最大で **30** 秒もの時間がかかっていました。

しかし、**fsync** がまったく使用されていない場合には、**ext4** ファイルシステムへの切り替えで問題が発生していました。**Ext3** は **data-ordered** モードに設定されていて、数秒毎にメモリを一掃してディスクに保存します。しかし、**ext4** の **laptop_mode** では保存の間隔が長いため、システムが不意にオフになった場合にはデータが消失する可能性がありました。現在、**ext4** は修正されていますが、それでもアプリケーションの設計をする際は慎重に検討し、適切に **fsync** を使用する必要があります。

設定ファイルからの読み込みと設定ファイルへの書き込みに関する簡単な例を使って、ファイルのバックアップが作成される流れと、データが消失してしまう流れを示します。

```
/* 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);
```

より適切な例は以下のようになります。

```
/* 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

付録B 改訂履歴

改訂 1.0-35.2 **Wed Jan 14 2015** **Kenzo Moriguchi**
翻訳、校閲完成

改訂 1.0-35.1 **Wed Jan 14 2015** **Chester Cheng**
翻訳ファイルを XML ソースバージョン 1.0-35 と同期

改訂 1.0-35 **Fri Oct 10 2014** **Yoana Ruseva**
6.6 GA リリース向けバージョン

改訂 1.0-34 **Fri Aug 8 2014** **Yoana Ruseva**
6.6 ベータリリース向けのバージョン

改訂 1.0-33 **Wed Sep 25 2013** **Yoana Ruseva**
6.5 GA リリース向けのバージョン

改訂 1.0-25 **Tue Feb 19 2013** **Jack Reed**
6.4 GA リリース向けバージョン 2

改訂 1.0-22 **Mon Feb 18 2013** **Jack Reed**
6.4 GA リリース向けバージョン

改訂 1.0-21 **Wed Nov 7 2012** **Jack Reed**
エラーの修正

改訂 1.0-20 **Wed Oct 31 2012** **Jack Reed**
警告を 2 つ削除

改訂 1.0-19 **Wed Oct 31 2012** **Jack Reed**
cpupower の代替に関する警告の間違いを訂正

改訂 1.0-18 **Tue Oct 30 2012** **Jack Reed**
cpupower 関連の警告を追加、 --policy オプションの機能に関する記載を更新

改訂 1.0-15 **Thu Oct 18 2012** **Jack Reed**
CPU モニターと CPU 節電ポリシーの各セクションを追加、 新しい cpupower ツールを反映するため数種類のコマンドを更新 - BZ#860874

改訂 1.0-14 **Fri Feb 10 2012** **Jack Reed**
ASPM パラメータのフォーマットに追加修正 - BZ#732859
tuned の新しいプロファイルを追加 - BZ#701924

改訂 1.0-12 **Fri Dec 16 2011** **Jack Reed**
ASPM パラメータのフォーマットに修正 - BZ#732859

改訂 1.0-11 **Thu Dec 15 2011** **Jack Reed**
fsync の付録より無効なリンクを削除 - BZ#706928

改訂 1.0-10 **Thu Dec 15 2011** **Jack Reed**
fsync 設定ファイルの例に変更 - BZ#706928

改訂 1.0-9 **Wed Dec 14 2011** **Jack Reed**
機能していない relatime のブートパラメータに関する記載を削除 - BZ#692859

改訂 1.0-8 **Mon Dec 12 2011** **Jack Reed**

タイプミスの訂正 - BZ#722798
fsync セクション内の設定ファイル例を編集 - BZ#706928
バッテリー使用時の設定に関する詳細を記載 - BZ#740794
ASPM ポリシーのパラメータフォーマットを訂正 - BZ#732859

改訂 1.0-7 **Thu Sep 29 2011** **Jack Reed**
GNOME 電源管理のセクションからバッテリー使用時のタブに関する記載を削除 - BZ#740794

改訂 1.0-6 **Tue May 24 2011** **Rüdiger Landmann**
再ビルド

改訂 1.0-2 **Fri Oct 22 2010** **Rüdiger Landmann**
テキスト内のマイナーエラー修正

改訂 1.0-1 **Thu Oct 7 2010** **Rüdiger Landmann**
「draft」タグの削除

改訂 1.0-0 **Thu Oct 7 2010** **Rüdiger Landmann**
GA リリース