Blueprints
Monitoring and controlling power consumption using Linux on System x
Blueprints

Monitoring and controlling power consumption using Linux on System x
Note
Before using this information and the product it supports, read the information in "Notices" on page 17.
Contents

Monitoring and controlling power consumption using Linux . . . . . . . . . 1
Scope, requirements, and support . . . . . . . . . . 2
Installing the pwrkap Software . . . . . . . . . . 3
Configuring the Server . . . . . . . . . . . . . . 3
Setting up Software Automatically . . . . . . . . 4
Setting up Software Manually . . . . . . . . . 4
Power Capping in Action . . . . . . . . . . . . . 5
Design Details . . . . . . . . . . . . . . . . . 9
Relating Power Use to Device Utilization and Power States . . . . . . . . 9
Processing a Snapshot . . . . . . . . . . . . . 10
Enforcing Caps . . . . . . . . . . . . . . . . . 10
Measuring the Quantity of Energy Consumed by a Task . . . . . . . . . . 10
Loading the Module . . . . . . . . . . . . . 10
Looking at the Sensors . . . . . . . . . . . . . 11
Programmatically Reading the Power Meter . . . . 11

Appendix A. Troubleshooting . . . . . . . . 13

Appendix B. Related Information and Downloads . . . . . . . . . . . . . . . 15

Notices . . . . . . . . . . . . . . . . . . . . . . . 17
Trademarks . . . . . . . . . . . . . . . . . . . . 18
Monitoring and controlling power consumption using Linux

In this blueprint, you will be shown how to monitor and control power use on System X servers that have the relevant hardware features--specifically, systems that support either Intel's Enhanced Speed Step (EIST) or AMD's PowerNow, and have power meter hardware installed. Key tools and technologies discussed in this demonstration include power meter, pwrkap, powercap, and Active Energy Manager (AEM).

Why Do This?

- **Avoid (over)loading power mains.** As of mid-2008, it is quite evident that high energy prices are here to stay. With the increasing density of datacenters, keeping those computers cool is also a problem. Combat this problem at the source by throttling power use, not raw performance, when electricity is expensive, and relaxing when it is cheap.
- **Measure energy consumption of tasks.** Measuring the energy footprint of a computer's workload is the first step to reducing that footprint. This enables one to build a database of best-power-practice configurations and measure progress against the baseline program. In addition, it allows one to determine a dollar cost for running a piece of code.
- **Look for consumption trends.** By analyzing the consumption trends of a computer, one can find the tasks that consume the greatest share of energy and optimize those tasks first. This is a powerful tool to target areas of power optimization that will bear the greatest reductions.

How Does it Work?

The details of pwrkap's operations can be found in the DESIGN document in the source code tarball. A brief discussion of the algorithmic details follows later in this blueprint.

- **Restricts available CPU frequency and voltage range to control power use.** pwrkap uses the CPU frequency regulation already present in modern-day Linux kernels. This mechanism, which alters both frequency and voltage, enables pwrkap to achieve impressive results--lower performance hits to correct a violation of the power cap, and better performance per watt while regulations are in effect.
- **Reports results to GUI client.** An optional GUI client can attach to the pwrkap daemon and show the current state of the datacenter or a graph of historical data.
- **Observes device use, power state, and power use.** pwrkap does not use a static table of values or theoretical calculations to determine what to do. Instead, it periodically profiles the system state so that its regulatory actions are backed by hard data. It will know that a certain change in power regulations produces a certain change in power consumption.
- **Remembers outcome of previous actions.** pwrkap measures and remembers the outcome of all regulatory actions it takes. By doing this, it adapts to changing environmental conditions, new hardware and differing workloads.

Scalability

- **Manages hundreds of systems.** pwrkap has been verified to work with several hundred systems. With optional aggregators, it has been shown to work with thousands of servers.
- **Flexible device control beyond CPU frequency.** Being an open source project, pwrkap can be extended by anybody to have the ability to regulate the power consumption of any piece of hardware. Network adapters, disks, video cards-- if it has more than one performance level, pwrkap can be made to modify it.
- **Doesn't require special firmware.** All one needs to run pwrkap is a power meter and a server containing devices that can be power-regulated.
Scope, requirements, and support

This blueprint applies to System x® running Linux. You can learn more about this blueprint, including the intended audience, the scope and purpose, the hardware and software requirements for the tasks detailed in this blueprint, and the types of support available to you.

Systems to which this information applies

System x running Linux

Intended audience

This paper is intended for expert users and administrators.

Scope and purpose

pwrkap is a software package attached to this blueprint that monitors and displays power utilization, and CPU load and frequency. While running, it stores periodic samples of both and uses the data to determine the maximum CPU speed allowable that will maintain the current power cap. This software is intended for use on a single system, though a GUI client can monitor several thousand systems.

Hardware and software requirements

The following equipment is required for this blueprint:

- An IBM® System X server with a power meter. These systems are: x3755, x3655, x3650, x3550, x3350, dx340, and selected HS21/LS21 blades.
- CPU options that support either Intel EIST or AMD PowerNow. For AMD this means a clock speed higher than 1.8GHz; for Intel, this means a clock speed higher than 2.8GHz (800MHz FSB P4), 3.2GHz (1.066GHz FSB P4), 2GHz (1.3GHz FSB Core2) or 2.4GHz (1.6GHz FSB Core2 Penryn).
- A recent Linux distribution such as Ubuntu 8.04, Red Hat Enterprise Linux 5, SUSE Linux Enterprise Server 10, or any distribution running the 2.6.24 kernel or newer.
- Software packages within the distro.
  - Debian/Ubuntu: python, pygtk, and python-matplotlib.
  - Red Hat: OpenIPMI-tools, python, and pygtk2.
  - SUSE: ipmitool, python, and python-gtk.

Relation to IBM Systems Director Active Energy Manager™ (AEM)

pwrkap is only intended to operate within a single system running a relatively static workload. It does not interface with power distribution units (PDUs), large-scale management frameworks such as IBM Director, workload schedulers such as IBM Tivoli®, or temperature control systems, nor is it intended for real-time management of large installations. For those features, it is more prudent to consider deploying AEM, as it is designed to handle these situations. Though pwrkap exists to handle simple configurations and allow advanced users to customize the software to fit their own systems, if both are used concurrently, AEM will not allow power use to exceed the AEM cap even if pwrkap tries to set a higher cap.

Author names

Darrick J Wong

Other contributors

Monza Lui
Kersten Richter
IBM services

Linux offers flexibility, options, and competitive total cost of ownership with a world class enterprise operating system. Community innovation integrates leading edge technologies and best practices into Linux.

IBM is a leader in the Linux community with over 600 developers in the IBM Linux Technology Center working on over 100 open source projects in the community. IBM supports Linux on all IBM servers, storage, and middleware, offering the broadest flexibility to match your business needs.

For more information about IBM and Linux, visit us at [ibm.com/linux](http://ibm.com/linux).

IBM support

Questions and comments regarding this documentation and the ‘pwrkap’ energy management tool may be posted on the DeveloperWorks Energy Management Blueprint Community Forum:


The IBM developerWorks® discussion forums let you ask questions, share knowledge, ideas, and opinions about technologies and programming techniques with other developerWorks users. Use the forum content at your own risk. While IBM will attempt to provide a timely response to all postings, the use of this DeveloperWorks forum does not guarantee a response to every question that is posted, nor do we validate the answers or the code that are offered.

Typographic conventions

The following typographic conventions are used in this blueprint:

<table>
<thead>
<tr>
<th><strong>Bold</strong></th>
<th>Identifies commands, subroutines, keywords, files, structures, directories, and other items whose names are predefined by the system. Also identifies graphical objects such as buttons, labels, and icons that the user selects.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Italics</strong></td>
<td>Identifies parameters whose actual names or values are to be supplied by the user.</td>
</tr>
<tr>
<td><strong>Monospace</strong></td>
<td>Identifies examples of specific data values, examples of text similar to what you might see displayed, examples of portions of program code similar to what you might write as a programmer, messages from the system, or information you should actually type.</td>
</tr>
</tbody>
</table>

Installing the pwrkap Software

pwrkap is a software package developed by the IBM Linux Technology Center (LTC) that monitors and displays power utilization, and CPU load and frequency. Before using pwrkap, you need to install and configure it.

Configuring the Server

In this section, the server’s firmware is configured to enable various power management features.

About this task

To configure the system’s firmware, follow these steps:
**Procedure**

1. Power on the System x server.
2. During boot, press F1 to enter Setup.
3. Enter the Advanced Features menu.
4. Enter either the Processor or CPU Options menu.
5. Enable CPU frequency changing. The option is named either Processor Performance States or Enhanced SpeedStep. If no such option exists, your server and/or CPU does not support CPU frequency switching and pwrkap will not work.
6. Exit setup and restart.

**Setting up Software Automatically**

In this section, the pwrkap software packages, library dependencies, and the operating system are properly installed and configured. This method is recommended for the smoothest installation experience possible.

**Procedure**

1. Install the distro. Be sure to install the following tools:
   - Debian/Ubuntu: `apt-get install ipmitool python pygtk python-matplotlib`.
   - Red Hat: `yum install OpenIPMI-tools python pygtk2`.
   - SUSE: `yast2 -i ipmitool python python-gtk`.
2. Download the pwrkap package. Link is available in [Appendix B, “Related Information and Downloads,” on page 15](#).
3. Install the software from the packages:
   - Debian/Ubuntu: `dpkg -i pwrkap-7.20_all.deb`
   - Red Hat: `rpm -i pwrkap-7.20-1.noarch-redhat.rpm`
   - SUSE: `rpm -i pwrkap-7.20-1.noarch-suse.rpm`
4. Proceed to the section “Power Capping in Action” on page 5.

**Setting up Software Manually**

This section presents an alternate method for installing the pwrkap software. This method is recommended for advanced users and developers only.

**Procedure**

1. Install the distro. Be sure to install the following tools:
   - Debian/Ubuntu: `apt-get install ipmitool python pygtk python-matplotlib`.
   - Red Hat: `yum install OpenIPMI-tools python pygtk2`.
   - SUSE: `yast2 -i ipmitool python python-gtk`.
2. Download the pwrkap tarball. See [Appendix B, “Related Information and Downloads,” on page 15](#) for link.
3. Unpack the tarball with `tar xzf pwrkap-7.20.tar.gz`
4. Enter `cd pwrkap`
5. Enter `make install`
6. To run pwrkap when the system is started up, create an init script:
   - Debian/Ubuntu:
     a. Copy `debian/init-script` from the source tarball to `/etc/init.d/pwrkap`
     b. Run `update-rc.d pwrkap defaults` to set up the init script
   - Red Hat/SUSE:
a. Copy redhat/init-script from the source tarball to /etc/init.d/pwrkap
b. Run chkconfig pwrkap on to set up the init script.

7. Proceed to "Power Capping in Action."

---

**Power Capping in Action**

In this section, pwrkap is used to monitor energy and power use, and demonstrate the power capping features.

**Procedure**

1. Start the pwrkap control software by running: `/etc/init.d/pwrkap start`
2. Wait a few minutes for pwrkap to train itself. The first time that pwrkap is run, it will generate artificial load for a few seconds to study the power consumption profile of the machine.
3. After a few minutes, check the status of pwrkap by running `/etc/init.d/pwrkap status`. If pwrkap is not running, pwrkap might not know how to configure your system. Double-check that your system meets the requirements and contact the support forum for help diagnosing the problem.
4. This demonstration uses the Apache web server to generate load. You can substitute your own load. If you want to use Apache, be sure to install:
   - Debian/Ubuntu: `apt-get install apache2 apache2-utils`
   - Red Hat: `yum install httpd`
   - SUSE: `yast2 -i apache2`
5. Start the pwrkap GUI program and attach it to the pwrkap controller by running: `pcap_gtk localhost`. Please note that the trending graphs are only supported on Ubuntu 8.04. They will not work on RHEL or SLES unless `matplotlib` is installed by hand. Installing `matplotlib` is beyond the scope of this blueprint. The main window looks like this initially:
6. After the system has run for a few minutes, notice the graphs of power consumption and CPU utilization begin to fill out with data.

7. Start either your workload or start ApacheBench with the command below. Notice the spike in system load and power consumption for the duration of the run.
   - Debian/Ubuntu
     a. 
     b. 
   - Red Hat:
     a. 
     b. 
   - SUSE:
     a. 
     b.

After a few minutes, the GUI monitor looks like this:
8. Set the power cap to something lower than the peak.

9. Rerun ApacheBench, but notice that the power use and performance levels are lower than before. If you set a cap that is lower than the power use of the system in its lowest performance state, the red “over cap” warning remains on the screen. The main window shows this:
10. Change the power cap midway through an ApacheBench run to watch it reprogram the CPU frequencies to allow higher throughput with higher power caps. The main window reflects this:
Design Details

The key ideas behind the algorithms presented in pwrkap are excerpted here.

For full details, see the DESIGN document in the source tarball.

Relating Power Use to Device Utilization and Power States

The heart of pwrkap is a four-dimensional table that enables pwrkap to guess what kind of impact a change in a device's power state has on the power domain's power use.

This large array is indexed like this:

\[
\text{transition_table}[\text{idomain}][\text{dev_use}][\text{p_state}][\text{p_state}] = \text{power_change}
\]

The \text{idomain} field specifies which devices in the power domain have identical power use profiles, which is a useful shortcut to reduce the amount of data that is needed to configure the devices correctly. The \text{dev_use} index cuts the rest of the table into % utilization buckets, because changes between power states of certain devices (CPUs in particular) affect power use differently depending on the device's utilization. The last two fields are used to index the current power state of the device and a proposed new power state. The value stored in the table is the average impact on power use given the four indexing factors.

This table can contain empty cells. By convention, the two \text{p-state} indices are always ordered with the lower of the two coming first, as it is assumed that transitions are commutative, for example a->b => c and b->a => -c.
Processing a Snapshot
When snapshots are taken, either during training or during normal operation of the pwrkap daemon, the software augments the transition database with the data that are being collected, thereby enabling the software to adapt to an environment that changes over time.

Each power domain remembers the past few snapshots that were taken of that domain. When a new snapshot is taken, it is compared to every snapshot currently in the retention buffer. The differences in states and in power use are noted in the transition table if it can be shown that the only difference between the new snapshot and the old snapshots is a change in power state one device in an idomain.

Enforcing Caps
Each power domain gets its own thread to run a control loop.

The operation of this control loop is as follows:
1. No more than once every MEASUREMENT_PERIOD seconds, take a snapshot of the power domain.
2. If less than ENFORCEMENT_INTERVAL seconds have passed since the last enforcement attempt, go to sleep and restart at step 1.
3. Compute the difference between the domain's power cap and power use.
4. Determine what to do:
   a. If use is less than cap, find all transitions that increase performance.
   b. If use is more than cap, find all transitions that decrease use.
   c. If they are equal, go back to step 1.
5. Find the transition with the most positive performance increase for the most negative increase in power use.
6. Implement the power state change specified by the transition and return to step 3.

Measuring the Quantity of Energy Consumed by a Task
It is possible to use pwrkap to measure the amount of energy that a server consumes in the time it takes to perform a task. By default, however, pwrkap only samples the energy meters every 15 seconds. This might not be sufficiently precise for certain kinds of measurements or a task may finish in less than that amount of time. For users that fall into this category, there is some technology that has been plumbed into a release of the Linux kernel that enables more precise measurement of energy use.

Beginning with the mainline kernel 2.6.26, there is a new driver named ibmaem that provides a direct interface to the energy counters that exist on certain System X servers.

Note: Be forewarned that this section is for advanced users who are comfortable with building their own kernels and installing their own software. These features will be included in future distributions, but for now they should be considered a technology preview.

Additional requirements
- All requirements laid out in “Scope, requirements, and support” on page 2
- Kernel 2.6.26 or newer. 2.6.26 was released in July 2008.
- (Optional) lm-sensors 3.0.2 or newer.

Loading the Module
Follow these instructions to load the AEM module.
Procedure
1. Configure a kernel with CONFIG_SENSORS_IBMAEM=m and CONFIG_IPMI_SI=m.
2. Build and install this kernel.
3. Load the IPMI driver with modprobe ipmi-si.
4. Load the AEM module with modprobe ibmaem.

Looking at the Sensors
The easiest method to read the power and energy sensors from the hardware is to use the sensors(1) program included in the lm-sensors package. Starting with version 3.0.2, lm-sensors knows how to display the output of power and energy sensors.

The output will look approximately like this:

```
# sensors
aem2-isa-0001
Adapter: ISA adapter
temp1: +35.0 C
temp2: +26.0 C
power1: 277.52 W (interval = 1.00 s)
power2: 357.21 W (interval = 1.00 s)
power3: 33.00 W
power4: 690.00 W
power5: 672.00 W
power6: 505.00 W
power7: 437.00 W
energy1: 277.90 MJ
energy2: 357.37 MJ
```

The driver presents two energy meters per system domain. On rack-mounted servers, energy2 gives a readout of the number of Joules consumed since the system was powered up. On blade servers, energy1 gives this readout.

Programmatically Reading the Power Meter
To read the energy meter from a program or a script, the current energy meter status is exposed via a sysfs interface.

The name of the `sysfs` file looks something like this:
```
/sys/devices/platform/aem.X/energyY_input
```
- X is the instance number of the meter hardware. These numbers increase from 0. Check that the sensors hardware is from a new enough version of the interface by examining `/sys/devices/platform/aem.X/version`; you will want v2.1 or newer.
- Y is 2 for a rack-mounted server and 1 for a blade server.

The energy consumption of a task can be measured with the sample script given below.
```
#!/bin/bash

BEFORE=`cat /sys/devices/platform/aem.1/energy2_input`
$*
AFTER=`cat /sys/devices/platform/aem.1/energy2_input`

echo "Energy: $(( (AFTER - BEFORE) / 100000 ))J"
```

If this script is saved as `energy.sh`, it can then be used to measure energy use in this manner:
```
$ energy.sh rsync -a /home backup_server:
Energy: 51238J
```
Appendix A. Troubleshooting

This topic discusses troubleshooting tips and caveats.

If the power use readouts do not appear in the GUI after a few minutes, check to make sure that the "ipmi-si" and "ipmi-devintf" modules are loaded. Ensure that the BMC is accessible by running "ipmitool sensor" to display all current sensor readings.
Appendix B. Related Information and Downloads

You can find additional information about the processes and tools described in these procedures.

Downloads

- Ubuntu 8.04:
- Red Hat Enterprise Linux 5:
- SUSE Linux Enterprise Server 10:
- Source:
- Project page:

Related information

- IBM Linux Energy Management on Linux Blueprint Support Forum
Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Dept. LRAS/Bldg. 903
11501 Burnet Road
Austin, TX 78758-3400
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.
For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

**Trademarks**

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® and ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at Copyright and trademark information at www.ibm.com/legal/copytrade.shtml

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Java™ and all Java-based trademarks and logos are registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.