

SOLARWINDS TECHNICAL REFERENCE

APM Linux Universal Template Pack

Linux CPU Monitoring Universal
Linux Disk Monitoring Universal
Linux Memory Monitoring Universal
Linux Sendmail Monitoring Universal

1 | This document describes the templates included
3 | in the APM Linux Universal Template Pack.
5 |
8 | **Note:** These templates are provided as-is and
support is only available through the APM
thwack forums on <http://thwack.com>.

Copyright© 1995-2011 SolarWinds. All rights reserved worldwide. No part of this document may be reproduced by any means nor modified, decompiled, disassembled, published or distributed, in whole or in part, or translated to any electronic medium or other means without the written consent of SolarWinds. All right, title and interest in and to the software and documentation are and shall remain the exclusive property of SolarWinds and its licensors. SolarWinds Orion™, SolarWinds Cirrus™, and SolarWinds Toolset™ are trademarks of SolarWinds and SolarWinds.net® and the SolarWinds logo are registered trademarks of SolarWinds All other trademarks contained in this document and in the Software are the property of their respective owners.

SOLARWINDS DISCLAIMS ALL WARRANTIES, CONDITIONS OR OTHER TERMS, EXPRESS OR IMPLIED, STATUTORY OR OTHERWISE, ON SOFTWARE AND DOCUMENTATION FURNISHED HEREUNDER INCLUDING WITHOUT LIMITATION THE WARRANTIES OF DESIGN, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL SOLARWINDS, ITS SUPPLIERS OR ITS LICENSORS BE LIABLE FOR ANY DAMAGES, WHETHER ARISING IN TORT, CONTRACT OR ANY OTHER LEGAL THEORY EVEN IF SOLARWINDS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Microsoft® and Windows 2000® are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Graph Layout Toolkit and Graph Editor Toolkit © 1992 - 2001 Tom Sawyer Software, Oakland, California. All Rights Reserved.

Portions Copyright © ComponentOne, LLC 1991-2002. All Rights Reserved.

Document Revised: 03/16/2011

Linux CPU Monitoring Universal

This template assesses the CPU performance of a Linux computer. It uses universal scripts for monitoring performance.

This template works with the following shells and languages: `perl`, `sh`, `bash`, `csh`, `tcsh`. The script logic works as follows:

- If it finds `perl` installed – it uses `perl` scripts only
- If it finds `sh` or `bash` – it uses `sh` or `bash`
- If it finds `csh` or `tcsh` – it uses `csh` or `tcsh`

Note: This template has currently been tested on CentOS 5.5.

Prerequisites: SSH and Perl installed on the target server.

Credentials: Root credentials on the target server.

Monitored Components:

CPU User Time

Time, in percentages, spent running non-kernel code (user time, including nice time). This represents the time spent executing user code. It depends on the programs that the user uses. This should be as low as possible at all times.

CPU System Time

Time, in percentages, spent running system kernel code (system time).

This should be as low as possible at all times.

Wait IO

Time, in percentages, spent waiting for input/output (IO) operations.

This should be as low as possible at all times.

If CPU waits IO is high, there may be problems with hard disk or problems with accessing NFS shares (if you use NFS).

CPU Idle Time

Time, in percentages, spent idle (this includes IO-wait time).

This should be as high as possible at all times.

Run queue

The number of processes waiting for run time.

This should be as low as possible, but not more than 4 per processor. If the run queue is constantly growing, it may indicate the need for a more powerful CPU or more CPUs.

Note: Set the thresholds appropriately for your environment.

Interrupts per second

The number of interrupts per second, including the clock.

This depends on the processor. For current CPUs, use a threshold of 1500 interrupts per second. A dramatic increase in this counter value without a corresponding increase in system activity indicates a hardware problem. Identify the network adapter or disk controller card causing the interrupts. You may need to install an additional adapter or controller card.

Note: Set the thresholds appropriately for your environment.

Context switches per second

The number of context switches per second.

High activity rates can result from inefficient hardware or poorly designed applications. The normal amount of Context Switches/Sec depends on your servers and applications. To set the threshold, you really need to baseline the server. The threshold for Context Switches/sec is cumulative for all processors, so you need a minimum of 14000 per processor (single=14000, dual=28000, quad=56000 and so forth).

Note: Set the thresholds appropriately for your environment.

Total amount of interrupts after boot

The total number of interrupts after boot.

Total amount of CPU context switches after boot

The total number of CPU context switches after boot.

Linux Disk Monitoring Universal

This template assesses the disk performance of a Linux computer. This template uses universal scripts for monitoring performance.

This template works with the following shells and languages: `perl`, `sh`, `bash`, `csh`, `tcsh`. The script logic works as follows:

- If it finds `perl` installed – it uses `perl` scripts only
- If it finds `sh` or `bash` – it uses `sh` or `bash`
- If it finds `csh` or `tcsh` – it uses `csh` or `tcsh`

Note: This template has currently been tested on CentOS 5.5.

Prerequisites: SSH and Perl installed on the target server.

Credentials: Root credentials on the target server.

Monitored Components:

Blocks received from block device (blocks/sec)

This shows the number of blocks read from the disk in the previous interval. All Linux blocks are currently 1024 bytes. Old kernels may report blocks as 512 bytes, 2048 bytes, or 4096 bytes.

Blocks sent to a block device (blocks/sec)

This indicates the total number of blocks written to disk in the previous interval. All Linux blocks are currently 1024 bytes. Old kernels may report blocks as 512 bytes, 2048 bytes, or 4096 bytes.

Available space on / partition (Mb)

This shows the available space on the root (`/`) partition in Mb.

You should set this threshold according to your Linux installation and your requirements. In the worst case, it should be more than 512 Mb.

Timing cached reads of first IDE hard drive (Mb/sec)

Perform timings of cache reads for benchmark and comparison purposes. This displays the speed of reading directly from the Linux buffer cache without disk access. This measurement is essentially an indication of the throughput of the processor, cache, and memory of the system under test.

This should be as high as possible.

Note: If you do not have an IDE hard drive on the target server, just ignore this counter. You should monitor this counter for some time and then set thresholds appropriately for your environment.

Note: This counter monitors only the first IDE hard drive. If you need to monitor the second IDE hard drive, you should find all lines with `"/dev/hda"` and replace with `"/dev/hdb"` in the entire script (it is recommended to use Windows notepad for the replacement). If you need to monitor the third IDE hard drive – replace with `"/dev/hdc"`.

Timing buffered disk reads of first IDE hard drive (Mb/sec)

Perform timings of device reads for benchmark and comparison purposes. This displays the speed of reading through the buffer cache to the disk without any prior caching of data. This measurement is an indication of how fast the drive can sustain sequential data reads under Linux, without any file system overhead.

This should be as high as possible.

Note: If you do not have an IDE hard drive on the target server, just ignore this counter. You should monitor this counter for some time and then set thresholds appropriately for your environment.

Note: This counter monitors only the first IDE hard drive. If you need to monitor the second IDE hard drive, you should find all lines with `"/dev/hda"` and replace with `"/dev/hdb"` in the entire script (it is recommended to use Windows notepad for the replacement). If you need to monitor the third IDE hard drive – replace with `"/dev/hdc"`.

Timing cached reads of first SATA/SCSI hard drive (Mb/sec)

Perform timings of cache reads for benchmark and comparison purposes. This displays the speed of reading directly from the Linux buffer cache without disk access. This measurement is essentially an indication of the throughput of the processor, cache, and memory of the system under test.

This should be as high as possible.

Note: If you do not have a SATA/SCSI hard drive on the target server, just ignore this counter. You should monitor this counter for some time and then appropriately for your environment.

Note: This counter monitors only the first SATA/SCSI hard drive. If you need to monitor the second SATA/SCSI hard drive, you should find all lines with `"/dev/sda"` and replace with `"/dev/sdb"` in the entire script (it is recommended to use Windows notepad for the replacement). If you need to monitor the third SATA/SCSI hard drive – replace with `"/dev/sdc"`.

Timing buffered disk reads of first SATA/SCSI hard drive (Mb/sec)

Perform timings of device reads for benchmark and comparison purposes. This displays the speed of reading through the buffer cache to the disk without any prior caching of data. This measurement is an indication of how fast the drive can sustain sequential data reads under Linux, without any file system overhead.

This should be as high as possible.

Note: If you do not have a SATA/SCSI hard drive on the target server, just ignore this counter. You should monitor this counter for some time and then appropriately for your environment.

Note: This counter monitors only the first SATA/SCSI hard drive. If you need to monitor the second SATA/SCSI hard drive, you should find all lines with `"/dev/sda"` and replace with `"/dev/sdb"` in the entire script (it is recommended to use Windows notepad for the replacement). If you need to monitor the third SATA/SCSI hard drive – replace with `"/dev/sdc"`.

Linux Memory Monitoring Universal

This template assesses the memory performance of a Linux computer. This template uses universal scripts for monitoring performance.

This template works with the following shells and languages: `perl`, `sh`, `bash`, `csh`, `tcsh`. The script logic works as follows:

- If it finds `perl` installed – it uses `perl` scripts only
- If it finds `sh` or `bash` – it uses `sh` or `bash`
- If it finds `csh` or `tcsh` – it uses `csh` or `tcsh`

Note: This template has currently been tested on CentOS 5.5.

Prerequisites: SSH and Perl installed on the target server.

Credentials: Root credentials on the target server.

Monitored Components:

Total memory (kb)

This shows the amount of total usable RAM in kb.

Used memory (kb)

This shows the amount of used memory in kb.

This should be as low as possible.

Free memory (kb)

This shows the amount of available memory in kb.

This should be more than 100000 kb at all times or paging will occur.

Total swap (kb)

This shows the amount of total swap space in kb.

Used swap (kb)

This shows the amount of used swap space in kb. Paging is one of the memory-management schemes by which a computer can store and retrieve data from secondary storage for use in main memory. Paging is an important part of virtual memory implementation in most contemporary general-purpose operating systems, allowing them to use disk storage for data that does not fit into physical random-access memory (RAM).

This should be as low as possible. Ideally it should be near zero. If the value is large, it may indicate that there is no free memory left.

Free swap (kb)

This shows amount of available swap space in kb.

It should be near the “total swap” value. If paging occurs in the system, the value should be as high as possible.

Buffers (kb)

RAM that is allocated to disk write operations.

Cache (kb)

RAM that is allocated to disk read operations.

Amount of zombie processes

This show the number of “zombie” processes. A “zombie” or defunct process is a process that has completed execution, but still has an entry in the process table. This entry is still needed to allow the process that started the (now zombie) process to read its exit status. When a process finishes execution, it will have an exit status to report to its parent process. Because of this last little bit of information, the process will remain in the operating system’s process table as a zombie process, indicating that it is not to be scheduled for further execution, but that it cannot be completely removed (and its process ID cannot be reused) until it has been determined that the exit status is no longer needed.

This should always be zero (0). If it is not zero, you should manually kill zombie processes. Use the following command to show these zombie processes (and look for a `z` in the STAT column):

```
ps aux
```

To kill zombie processes:

The first option is to wait. It is possible that the parent process is intentionally leaving the process in a zombie state to ensure that future children it may create will not receive the same pid. Or perhaps the parent is occupied, and will reap the child process momentarily.

The second option is to send a SIGCHLD signal to the parent (`kill -s SIGCHLD <ppid>`). This will cause well-behaving parents to reap their zombie children.

The third option is to kill the parent process of the zombie. At that point, all of the parent’s children will be adopted by the init process (pid 1), which periodically runs `wait()` to reap any zombie children.

Dirty Pages (kb)

The total amount of memory, in kilobytes, waiting to be written back to the disk.

This should be as low as possible.

Note: You should monitor this counter for some time and then set thresholds appropriately for your environment.

Anonymous Pages (kb)

Anonymous memory is memory that is managed by `segvn` but is not really directly associated with a file. It is used for things like process stacks, heap, or COW (copy on write) pages. A good example of this is if you fork a process. All the addresses in the second process actually map back to the same bits of physical memory (the same pages). However if your child process was then to do something different with the memory (for example the child went off and manipulated an array in memory), the VM subsystem would copy those pages and change the mappings in the child process to point to the new pages. This new memory would be anonymous memory, and the child process would merrily make the changes to the array, unaware it now had new "physical" memory it was addressing.

This should be as low as possible.

Note: You should monitor this counter for some time and then set thresholds appropriately for your environment.

Linux Sendmail Monitoring Universal

This template assesses the Linux sendmail status and performance. This template uses universal scripts for monitoring performance.

This template works with the following shells and languages: `perl`, `sh`, `bash`, `csh`, `tcsh`. The script logic works as follows:

- If it finds `perl` installed – it uses `perl` scripts only
- If it finds `sh` or `bash` – it uses `sh` or `bash`
- If it finds `csh` or `tcsh` – it uses `csh` or `tcsh`

Note: This template has currently been tested on CentOS 5.5.

Prerequisites: SSH and Perl installed on the target server.

Credentials: Root credentials on the target server.

The counters and services in this template are based on the following information:

- *Monitoring sendmail:*
http://opensmart.sourceforge.net/docs/documentation/userguide/appmon_sendmail.html

Monitored Components:

SMTP Port Monitor

Check availability of TCP port 25 (smtp).

Amount of running Sendmail instances

This shows how many instances of sendmail daemon are currently started.

0 – Sendmail is stopped. Run sendmail manually.

>1 – Sendmail is up and running.

Daemon: syslogd

This shows the status of syslogd daemon (Syslog is a standard for logging program messages).

0 – Syslog daemon is stopped. Run syslogd manually.

1 – Syslog daemon is up and running.

Disk usage: /var/spool/mail (kb)

This shows how much disk space in kb that mail accounts use.

Note: Set thresholds according to your requirements.

Disk usage: /var/spool/mqueue (kb)

This shows how much disk space in kb that the mail queue uses.

Note: Set thresholds according to your requirements.

Mail queue

This shows the mail queue length (how many items are in the queue for delivery).

This should be as low as possible.

If the mail queue value is constantly rising, it may indicate problems with delivering messages.

Note: Set thresholds according to your requirements.

Available space on partition with `/var/spool` (Mb)

This shows the available space on the partition with `/var/spool` folder in Mb.

Note: You should set this threshold according to your requirements.

Note: By default it checks available space on the root (`/`) partition. If you have created a separate partition `/var` or `/var/spool` you need change it in the script (`grep "/\ $"`). You can investigate what partitions you have by using the following command: `df`

For the `/var` partition, you should make the following change: **change** `grep "/\ $"` **to** `grep "/var\ $"`

For the `/var/spool` partition, you should make the following change: **change** `grep "/\ $"` **to** `grep "/var/spool\ $"`