10 Useful Sar (Sysstat) Examples for UNIX / Linux Performance Monitoring

by Ramesh Natarajan on March 29, 2011

Using sar you can monitor performance of various Linux subsystems (CPU, Memory, I/O..) in real time.

Using sar, you can also collect all performance data on an on-going basis, store them, and do historical analysis to identify bottlenecks.

Sar is part of the sysstat package.

This article explains how to install and configure sysstat package (which contains sar utility) and explains how to monitor the following Linux performance statistics using sar.

1. Collective CPU usage
2. Individual CPU statistics

3. Memory used and available
4. Swap space used and available
5. Overall I/O activities of the system
6. Individual device I/O activities
7. Context switch statistics
8. Run queue and load average data
9. Network statistics
10. Report sar data from a specific time

This is the only guide you'll need for sar utility. So, bookmark this for your future reference.

I. Install and Configure Sysstat

Install Sysstat Package

First, make sure the latest version of sar is available on your system. Install it using any one of the following methods depending on your distribution.

```
sudo apt-get install sysstat
(or)
yum install sysstat
(or)
rpm -ivh sysstat-10.0.0-1.i586.rpm
```

Install Sysstat from Source

Download the latest version from sysstat download page.

You can also use wget to download the


```
wget http://pagesperso-orange.fr/sebastien.godard/sysstat-10.0.0.tar.bz2

tar xvfj sysstat-10.0.0.tar.bz2

cd sysstat-10.0.0

./configure --enable-install-cron
```

Note: Make sure to pass the option –enable-install-cron. This does the following automatically for

you. If you don't configure sysstat with this option, you have to do this ugly job yourself manually.

- Creates /etc/rc.d/init.d/sysstat
- Creates appropriate links from /etc/rc.d/rc*.d/ directories to /etc/rc.d/init.d/sysstat to start the sysstat automatically during Linux boot process.
- For example, /etc/rc.d/rc3.d/S01sysstat is linked automatically to /etc/rc.d/init.d/sysstat

After the ./configure, install it as shown below.

```
make

make install
```

Note: This will install sar and other systat utilities under /usr/local/bin

Once installed, verify the sar version using "sar -V". Version 10 is the current stable version of sysstat.

```
$ sar -V
sysstat version 10.0.0
(C) Sebastien Godard (sysstat  orange.fr)
```

Finally, make sure sar works. For example, the following gives the system CPU statistics 3 times (with 1 second interval).

```
$ sar 1 3
Linux 2.6.18-194.el5PAE (dev-db)        03/26/2011      _i686_  (8 CPU)

01:27:32 PM       CPU     %user     %nice   %system   %iowait    %steal     %idle
01:27:33 PM       all      0.00      0.00      0.00      0.00      0.00    100.00
01:27:34 PM       all      0.25      0.00      0.25      0.00      0.00     99.50
01:27:35 PM       all      0.75      0.00      0.25      0.00      0.00     99.00
Average:          all      0.33      0.00      0.17      0.00      0.00     99.50
```

Utilities part of Sysstat

Following are the other sysstat utilities.

- **sar** collects and displays ALL system activities statistics.
- **sadc** stands for "system activity data collector". This is the sar backend tool that does the data collection.
- **sa1** stores system activities in binary data file. sa1 depends on sadc for this purpose. sa1 runs from cron.
- **sa2** creates daily summary of the collected statistics. sa2 runs from cron.
- **sadf** can generate sar report in CSV, XML, and various other formats. Use this to integrate sar data with other tools.
- **iostat** generates CPU, I/O statistics
- **mpstat** displays CPU statistics.
- **pidstat** reports statistics based on the process id (PID)
- **nfsiostat** displays NFS I/O statistics.
- **cifsiostat** generates CIFS statistics.

This article focuses on sysstat fundamentals and sar utility.

Collect the sar statistics using cron job – sa1 and sa2

Create sysstat file under /etc/cron.d directory that will collect the historical sar data.

```
# vi /etc/cron.d/sysstat
*/10 * * * * root /usr/local/lib/sa/sa1 1 1
53 23 * * * root /usr/local/lib/sa/sa2 -A
```

If you've installed sysstat from source, the default location of sa1 and sa2 is /usr/local/lib/sa. If you've installed using your distribution update method (for example: yum, up2date, or apt-get), this might be /usr/lib/sa/sa1 and /usr/lib/sa/sa2.

Note: To understand cron entries, read Linux Crontab: 15 Awesome Cron Job Examples.

/usr/local/lib/sa/sa1

- This runs every 10 minutes and collects sar data for historical reference.
- If you want to collect sar statistics every 5 minutes, change */10 to */5 in the above /etc/cron.d /sysstat file.
- This writes the data to /var/log/sa/saXX file. XX is the day of the month. saXX file is a binary file. You cannot view its content by opening it in a text editor.
- For example, If today is 26th day of the month, sa1 writes the sar data to /var/log/sa/sa26
- You can pass two parameters to sa1: interval (in seconds) and count.
- In the above crontab example: sa1 1 1 means that sa1 collects sar data 1 time with 1 second interval (for every 10 mins).

/usr/local/lib/sa/sa2

- This runs close to midnight (at 23:53) to create the daily summary report of the sar data.
- sa2 creates /var/log/sa/sarXX file (Note that this is different than saXX file that is created by sa1). This sarXX file created by sa2 is an ascii file that you can view it in a text editor.
- This will also remove saXX files that are older than a week. So, write a quick shell script that runs every week to copy the /var/log/sa/* files to some other directory to do historical sar data analysis.

II. 10 Practical Sar Usage Examples

There are two ways to invoke sar.

1. sar followed by an option (without specifying a saXX data file). This will look for the current day's saXX data file and report the performance data that was recorded until that point for the current day.
2. sar followed by an option, and additionally specifying a saXX data file using -f option. This will report the performance data for that particular day. i.e XX is the day of the month.

In all the examples below, we are going to explain how to view certain performance data for the current day. To look for a specific day, add "-f /var/log/sa/saXX" at the end of the sar command.

All the sar command will have the following as the 1st line in its output.

```
$ sar -u
Linux 2.6.18-194.el5PAE (dev-db)        03/26/2011      _i686_  (8 CPU)
```

- Linux 2.6.18-194.el5PAE – Linux kernel version of the system.

- (dev-db) – The hostname where the sar data was collected.
- 03/26/2011 – The date when the sar data was collected.
- _i686_ – The system architecture
- (8 CPU) – Number of CPUs available on this system. On multi core systems, this indicates the total number of cores.

1. CPU Usage of ALL CPUs (sar -u)

This gives the cumulative real-time CPU usage of all CPUs. "1 3″ reports for every 1 seconds a total of 3 times. Most likely you'll focus on the last field "%idle" to see the cpu load.

```
$ sar -u 1 3
Linux 2.6.18-194.el5PAE (dev-db)        03/26/2011      _i686_  (8 CPU)

01:27:32 PM       CPU     %user     %nice   %system   %iowait    %steal     %idle
01:27:33 PM       all      0.00      0.00      0.00      0.00      0.00    100.00
01:27:34 PM       all      0.25      0.00      0.25      0.00      0.00     99.50
01:27:35 PM       all      0.75      0.00      0.25      0.00      0.00     99.00
Average:          all      0.33      0.00      0.17      0.00      0.00     99.50
```

Following are few variations:

- **sar -u** Displays CPU usage for the current day that was collected until that point.
- **sar -u 1 3** Displays real time CPU usage every 1 second for 3 times.
- **sar -u ALL** Same as "sar -u" but displays additional fields.
- **sar -u ALL 1 3** Same as "sar -u 1 3″ but displays additional fields.
- **sar -u -f /var/log/sa/sa10** Displays CPU usage for the 10day of the month from the sa10 file.

2. CPU Usage of Individual CPU or Core (sar -P)

If you have 4 Cores on the machine and would like to see what the individual cores are doing, do the following.

"-P ALL" indicates that it should displays statistics for ALL the individual Cores.

In the following example under "CPU" column 0, 1, 2, and 3 indicates the corresponding CPU core numbers.

```
$ sar -P ALL 1 1
Linux 2.6.18-194.el5PAE (dev-db)        03/26/2011      _i686_  (8 CPU)

01:34:12 PM       CPU     %user     %nice   %system   %iowait    %steal     %idle
01:34:13 PM       all     11.69      0.00      4.71      0.69      0.00     82.90
01:34:13 PM         0     35.00      0.00      6.00      0.00      0.00     59.00
01:34:13 PM         1     22.00      0.00      5.00      0.00      0.00     73.00
01:34:13 PM         2      3.00      0.00      1.00      0.00      0.00     96.00
01:34:13 PM         3      0.00      0.00      0.00      0.00      0.00    100.00
```

"-P 1″ indicates that it should displays statistics only for the 2nd Core. (Note that Core number starts from 0).

```
$ sar -P 1 1 1
Linux 2.6.18-194.el5PAE (dev-db)        03/26/2011      _i686_  (8 CPU)

01:36:25 PM       CPU     %user     %nice   %system   %iowait    %steal     %idle
01:36:26 PM         1      8.08      0.00      2.02      1.01      0.00     88.89
```

Following are few variations:

- **sar -P ALL** Displays CPU usage broken down by all cores for the current day.
- **sar -P ALL 1 3** Displays real time CPU usage for ALL cores every 1 second for 3 times (broken down by all cores).
- **sar -P 1** Displays CPU usage for core number 1 for the current day.
- **sar -P 1 1 3** Displays real time CPU usage for core number 1, every 1 second for 3 times.
- **sar -P ALL -f /var/log/sa/sa10** Displays CPU usage broken down by all cores for the 10day day of the month from sa10 file.

3. Memory Free and Used (sar -r)

This reports the memory statistics. "1 3″ reports for every 1 seconds a total of 3 times. Most likely you'll focus on "kbmemfree" and "kbmemused" for free and used memory.

```
$ sar -r 1 3
Linux 2.6.18-194.el5PAE (dev-db)        03/26/2011      _i686_  (8 CPU)

07:28:06 AM kbmemfree kbmemused  %memused kbbuffers  kbcached  kbcommit   %commit  kl
07:28:07 AM   6209248   2097432     25.25    189024   1796544    141372      0.85
07:28:08 AM   6209248   2097432     25.25    189024   1796544    141372      0.85
07:28:09 AM   6209248   2097432     25.25    189024   1796544    141372      0.85
Average:      6209248   2097432     25.25    189024   1796544    141372      0.85
```

Following are few variations:

- sar -r
- sar -r 1 3
- sar -r -f /var/log/sa/sa10

4. Swap Space Used (sar -S)

This reports the swap statistics. "1 3″ reports for every 1 seconds a total of 3 times. If the "kbswpused" and "%swpused" are at 0, then your system is not swapping.

```
$ sar -S 1 3
Linux 2.6.18-194.el5PAE (dev-db)        03/26/2011      _i686_  (8 CPU)

07:31:06 AM kbswpfree kbswpused  %swpused  kbswpcad   %swpcad
07:31:07 AM   8385920         0      0.00         0      0.00
07:31:08 AM   8385920         0      0.00         0      0.00
07:31:09 AM   8385920         0      0.00         0      0.00
Average:      8385920         0      0.00         0      0.00
```

Following are few variations:

- sar -S
- sar -S 1 3
- sar -S -f /var/log/sa/sa10

Notes:

- Use "sar -R" to identify number of memory pages freed, used, and cached per second by the system.
- Use "sar -H" to identify the hugepages (in KB) that are used and available.

- Use “sar -B” to generate paging statistics. i.e Number of KB paged in (and out) from disk per second.
- Use “sar -W” to generate page swap statistics. i.e Page swap in (and out) per second.

5. Overall I/O Activities (sar -b)

This reports I/O statistics. “1 3″ reports for every 1 seconds a total of 3 times.

Following fields are displays in the example below.

- tps – Transactions per second (this includes both read and write)
- rtps – Read transactions per second
- wtps – Write transactions per second
- bread/s – Bytes read per second
- bwrtn/s – Bytes written per second

```
$ sar -b 1 3
Linux 2.6.18-194.el5PAE (dev-db)        03/26/2011      _i686_  (8 CPU)

01:56:28 PM       tps      rtps      wtps   bread/s   bwrtn/s
01:56:29 PM    346.00    264.00     82.00   2208.00    768.00
01:56:30 PM    100.00     36.00     64.00    304.00    816.00
01:56:31 PM    282.83     32.32    250.51    258.59   2537.37
Average:       242.81    111.04    131.77    925.75   1369.90
```

Following are few variations:

- sar -b
- sar -b 1 3
- sar -b -f /var/log/sa/sa10

Note: Use “sar -v” to display number of inode handlers, file handlers, and pseudo-terminals used by the system.

6. Individual Block Device I/O Activities (sar -d)

To identify the activities by the individual block devices (i.e a specific mount point, or LUN, or partition), use “sar -d”

```
$ sar -d 1 1
Linux 2.6.18-194.el5PAE (dev-db)        03/26/2011      _i686_  (8 CPU)

01:59:45 PM       DEV       tps  rd_sec/s  wr_sec/s  avgrq-sz  avgqu-sz     await
01:59:46 PM    dev8-0      1.01      0.00      0.00      0.00      0.00      4.00
01:59:46 PM    dev8-1      1.01      0.00      0.00      0.00      0.00      4.00
01:59:46 PM dev120-64      3.03     64.65      0.00     21.33      0.03      9.33
01:59:46 PM dev120-65      3.03     64.65      0.00     21.33      0.03      9.33
01:59:46 PM  dev120-0      8.08      0.00    105.05     13.00      0.00      0.38
01:59:46 PM  dev120-1      8.08      0.00    105.05     13.00      0.00      0.38
01:59:46 PM dev120-96      1.01      8.08      0.00      8.00      0.01      9.00
01:59:46 PM dev120-97      1.01      8.08      0.00      8.00      0.01      9.00
```

In the above example “DEV” indicates the specific block device.

For example: “dev53-1″ means a block device with 53 as major number, and 1 as minor number.

The device name (DEV column) can display the actual device name (for example: sda, sda1, sdb1 etc.,), if you use the -p option (pretty print) as shown below.

```
$ sar -p -d 1 1
Linux 2.6.18-194.el5PAE (dev-db)        03/26/2011      _i686_  (8 CPU)

01:59:45 PM       DEV       tps  rd_sec/s  wr_sec/s  avgrq-sz  avgqu-sz     await
01:59:46 PM       sda      1.01      0.00      0.00      0.00      0.00      4.00
01:59:46 PM      sda1      1.01      0.00      0.00      0.00      0.00      4.00
01:59:46 PM      sdb1      3.03     64.65      0.00     21.33      0.03      9.33
01:59:46 PM      sdc1      3.03     64.65      0.00     21.33      0.03      9.33
01:59:46 PM      sde1      8.08      0.00    105.05     13.00      0.00      0.38
01:59:46 PM      sdf1      8.08      0.00    105.05     13.00      0.00      0.38
01:59:46 PM      sda2      1.01      8.08      0.00      8.00      0.01      9.00
01:59:46 PM      sdb2      1.01      8.08      0.00      8.00      0.01      9.00
```

Following are few variations:

- sar -d
- sar -d 1 3
- sar -d -f /var/log/sa/sa10
- sar -p -d

7. Display context switch per second (sar -w)

This reports the total number of processes created per second, and total number of context switches per second. “1 3″ reports for every 1 seconds a total of 3 times.

```
$ sar -w 1 3
Linux 2.6.18-194.el5PAE (dev-db)        03/26/2011      _i686_  (8 CPU)

08:32:24 AM    proc/s   cswch/s
08:32:25 AM      3.00     53.00
08:32:26 AM      4.00     61.39
08:32:27 AM      2.00     57.00
```

Following are few variations:

- sar -w
- sar -w 1 3
- sar -w -f /var/log/sa/sa10

8. Reports run queue and load average (sar -q)

This reports the run queue size and load average of last 1 minute, 5 minutes, and 15 minutes. “1 3″ reports for every 1 seconds a total of 3 times.

```
$ sar -q 1 3
Linux 2.6.18-194.el5PAE (dev-db)        03/26/2011      _i686_  (8 CPU)

06:28:53 AM   runq-sz  plist-sz   ldavg-1   ldavg-5  ldavg-15   blocked
06:28:54 AM         0       230      2.00      3.00      5.00         0
06:28:55 AM         2       210      2.01      3.15      5.15         0
06:28:56 AM         2       230      2.12      3.12      5.12         0
Average:            3       230      3.12      3.12      5.12         0
```

Note: The “blocked” column displays the number of tasks that are currently blocked and waiting for I/O operation to complete.

Following are few variations:

- sar -q
- sar -q 1 3
- sar -q -f /var/log/sa/sa10

9. Report network statistics (sar -n)

This reports various network statistics. For example: number of packets received (transmitted) through the network card, statistics of packet failure etc.,. "1 3″ reports for every 1 seconds a total of 3 times.

```
sar -n KEYWORD
```

KEYWORD can be one of the following:

- DEV – Displays network devices vital statistics for eth0, eth1, etc.,
- EDEV – Display network device failure statistics
- NFS – Displays NFS client activities
- NFSD – Displays NFS server activities
- SOCK – Displays sockets in use for IPv4
- IP – Displays IPv4 network traffic
- EIP – Displays IPv4 network errors
- ICMP – Displays ICMPv4 network traffic
- EICMP – Displays ICMPv4 network errors
- TCP – Displays TCPv4 network traffic
- ETCP – Displays TCPv4 network errors
- UDP – Displays UDPv4 network traffic
- SOCK6, IP6, EIP6, ICMP6, UDP6 are for IPv6
- ALL – This displays all of the above information. The output will be very long.

```
$ sar -n DEV 1 1
Linux 2.6.18-194.el5PAE (dev-db)        03/26/2011      _i686_  (8 CPU)

01:11:13 PM     IFACE   rxpck/s   txpck/s   rxbyt/s   txbyt/s   rxcmp/s   txcmp/s  r
01:11:14 PM        lo      0.00      0.00      0.00      0.00      0.00      0.00
01:11:14 PM      eth0    342.57    342.57  93923.76 141773.27      0.00      0.00
01:11:14 PM      eth1      0.00      0.00      0.00      0.00      0.00      0.00
```

10. Report Sar Data Using Start Time (sar -s)

When you view historic sar data from the /var/log/sa/saXX file using "sar -f" option, it displays all the sar data for that specific day starting from 12:00 a.m for that day.

Using "-s hh:mi:ss" option, you can specify the start time. For example, if you specify "sar -s 10:00:00″, it will display the sar data starting from 10 a.m (instead of starting from midnight) as shown below.

You can combine -s option with other sar option.

For example, to report the load average on 26th of this month starting from 10 a.m in the morning, combine the -q and -s option as shown below.

```
$ sar -q -f /var/log/sa/sa23 -s 10:00:01
```

```
Linux 2.6.18-194.el5PAE (dev-db)        03/26/2011      _i686_  (8 CPU)

10:00:01 AM   runq-sz  plist-sz   ldavg-1   ldavg-5  ldavg-15   blocked
10:10:01 AM         0       127      2.00      3.00      5.00         0
10:20:01 AM         0       127      2.00      3.00      5.00         0
...
11:20:01 AM         0       127      5.00      3.00      3.00         0
12:00:01 PM         0       127      4.00      2.00      1.00         0
```

There is no option to limit the end-time. You just have to get creative and use head command as shown below.

For example, starting from 10 a.m, if you want to see 7 entries, you have to pipe the above output to "head -n 10".

```
$ sar -q -f /var/log/sa/sa23 -s 10:00:01 | head -n 10
Linux 2.6.18-194.el5PAE (dev-db)        03/26/2011      _i686_  (8 CPU)

10:00:01 AM   runq-sz  plist-sz   ldavg-1   ldavg-5  ldavg-15   blocked
10:10:01 AM         0       127      2.00      3.00      5.00         0
10:20:01 AM         0       127      2.00      3.00      5.00         0
10:30:01 AM         0       127      3.00      5.00      2.00         0
10:40:01 AM         0       127      4.00      2.00      1.00         2
10:50:01 AM         0       127      3.00      5.00      5.00         0
11:00:01 AM         0       127      2.00      1.00      6.00         0
11:10:01 AM         0       127      1.00      3.00      7.00         2
```

There is lot more to cover in Linux performance monitoring and tuning. We are only getting started. More articles to come in the performance series.

Previous articles in the Linux performance monitoring and tuning series:

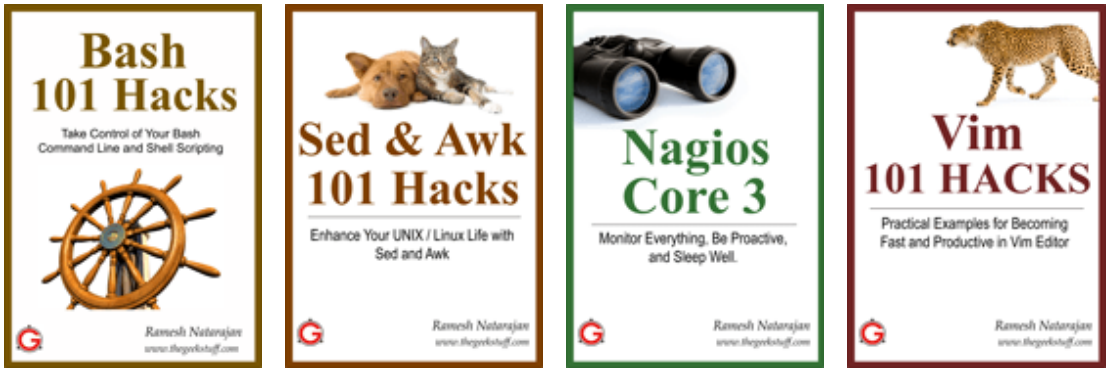
- Linux Performance Monitoring and Tuning Introduction
- 15 Practical Linux Top Command Examples

Linux provides several powerful administrative tools and utilities which will help you to manage your systems effectively. If you don't know what these tools are and how to use them, you could be spending lot of time trying to perform even the basic administrative tasks. The focus of this course is to help you understand system administration tools, which will help you to become an effective Linux system administrator.
Get the Linux Sysadmin Course Now!

If you enjoyed this article, you might also like..

1. 50 Linux Sysadmin Tutorials
2. 50 Most Frequently Used Linux Commands (With Examples)
3. Top 25 Best Linux Performance Monitoring and Debugging Tools
4. Mommy, I found it! – 15 Practical Linux Find Command Examples
5. Linux 101 Hacks 2nd Edition eBook Free

- Awk Introduction – 7 Awk Print Examples
- Advanced Sed Substitution Examples
- 8 Essential Vim Editor Navigation Fundamentals
- 25 Most Frequently Used Linux IPTables Rules Examples
- Turbocharge PuTTY with 12 Powerful Add-Ons



Tags: Linux Sar Report, Sar RedHat Examples, Sar Report Linux, Sar Ubuntu, Unix Sar Utility, Unix Sar wio

{ 28 comments… read them below or add one }

1 jaxxm March 29, 2011 at 3:08 am

Wow, really cool. keep up the good work. This is the most informative site on linux. Thanks again.

2 Felix Frank March 29, 2011 at 6:56 am

Thanks Ramesh for the nice overview of sar's options.

3 Alex March 29, 2011 at 11:26 am

Excelente información, he venido observando tu pagina lo que has publicado esta al día, felicitaciones por tu familia, salu2 desde Venezuela

———Spanish to English translation———-
Excellent information, I have been watching your site what you have posted this a day, congratulations on your family, salu2 from Venezuela

4 Patrick March 29, 2011 at 8:49 pm

In regards to section 10 there is an "end" option. Sar accepts times in 24 hour format. For instance:

sar -s 21:00:00 -e 21:30:00 -q

The above would show load statistics starting at 9:00PM and end at 9:30PM.

5 Phani March 31, 2011 at 7:41 am

Thnk you Ramesh Natarajan. nice post

6 Martin March 31, 2011 at 12:52 pm

Thanks Ramesh …

I've used sar for a while, and your thorugh description still adds value.
Now off to changing my standard script for collecting performance data to the management server.

Regards Martin Rønde

7 Rajinder Yadav April 17, 2011 at 11:47 pm

Ramesh,

you did an excellent job of covering the sar tools, this page is definitely going into my bookmark!

Kind Regards,
Rajinder Yadav

8 Kapil June 27, 2011 at 1:46 am

Ramesh,

Great document for learning sar utility for performance.

Regards
Kapil

9 Kirti Ranjan Nayak August 1, 2011 at 3:13 am

Awesome tutorial it is this… thanks a lot for helping others.

10 Kamal Kishore August 9, 2011 at 6:12 am

Hi Ramesh,

In the point no. 10, end option can be use as follows.
sar -q -f /var/log/sa/sa08 -s 09:04:00 -e 15:00:00
this will display between 09:04:00 a.m. to 15:00:00 (3 p.m.) output
Thanks to you for such a nice artical.

Regards,

Kamal

11 Ashok September 7, 2011 at 7:23 am

> "There is no option to limit the end-time. You just have to get creative and use head command as shown below."
>
> "sar" command has option for End-time. Something like this:
> sar -r -f /var/log/sa/sa07 -s 03:00:01 -e 05:00:01
>
> http://ashok-linux-tips.blogspot.com/2011/08/how-to-analyze-past-system-performance.html
>
> Thanks
> Ashok

12 Chaitanya November 11, 2011 at 4:33 pm

> very nice article helped me a lot

13 sid November 21, 2011 at 4:36 pm

> How can we generate graphs and reports from this data?

14 Greg March 12, 2012 at 1:37 pm

> I am trying to prove to a customer that they have their file systems set up wrong. They have multiple applications, SAS, ndm, and other running on a system and are using one large multi-terebyte file system for all of them. They keep wanting to blame disk I/O for poor performance. I realize that is probably the problem but should be addressed by chaning the file system layout to minimize issues first. If you could guide me in how to present either sar, iostat, vmstat, or some other information to them to prove this I would appreciate it.

15 daniel March 25, 2012 at 6:29 pm

> i am dying to know how to figure this out: I have download the latest version of sysstat (10.0.4) and already did the 2 initial steps for the installation into my Centos 5 (on a VMWare). The problem is when i type the command "make" after compiling with "./configure". I got many errors that don't let me install the sysstat.
> This are some of the errors i got:
> make: o: No se encontró el programa (english= "didn't find the program)
> make: [nls/sv.gmo] Error 127 (no tiene efecto)

16 Daniel March 27, 2012 at 9:08 am

> you should be able to run yum install sysstat and it will hopefully resolve dependacies for you

17 satish June 20, 2012 at 6:45 pm

> Nice one sir.

18 Tom George August 2, 2012 at 11:28 am

> Hello Ramesh,
> I read about a functionality for measuring CPU temperature using sysstat. I couldn't find the command for doing that. Do you have any idea regarding this? If yes, I think it would be a nice

addition to the blog

Thanks,
Tom

19 Bakkesh September 14, 2012 at 5:29 am

Hey i want to CPU performance monitoring for PTN equipment how can i monitor the graphical Views pls provide the related sites info.

20 Ole Laursen October 30, 2012 at 6:39 am

According to the man page, bread/s and bwrtn/s aren't bytes but 512 byte blocks.

Also, in your memory example, remember that Linux uses memory for caching purposes. So kbmemfree isn't really that interesting.

21 Gopu March 20, 2013 at 3:06 pm

Well done Ramesh!
Very useful blog which covers day to day issues……
Thanks again

22 sajid July 19, 2013 at 5:28 pm

good document……

23 vishal September 10, 2013 at 10:46 am

Hi,
I had installed the above mentioned step and it was successful.
I am getting error while executing without providing any number
vishal@vishal-Aspire-5920:~$ sar -u
Cannot open /var/log/sysstat/sa10: No such file or directory
Please check if data collecting is enabled in /etc/default/sysstat
vishal@vishal-Aspire-5920:~$ sar -r
Cannot open /var/log/sysstat/sa10: No such file or directory
Please check if data collecting is enabled in /etc/default/sysstat

Can you correct me where did i make mistake?

Thanks,
Vishal

24 JayJay October 8, 2013 at 6:45 am

Hi Ramesh,

Thank you for the informations provided on your blog.

I'm doing some personal training on Linux, and testing several bechmarks tools. There is something that I don't understand clearly about the "sar -w" option.
I tried to figure out what the "number of context switches per second" means. Could you give

me more details about it?

Regards,
JJ

25 Anonymous November 13, 2013 at 5:02 am

Hi.

How to setup an application in sar?
Ex: i have lima application, I have to monitor that application.

26 sangbang November 13, 2013 at 5:04 am

Hi Ramesh,
I have to setup an application in sar.
Ex: i have application called lima and monitor that application in every 15m?

27 Jonathan January 6, 2014 at 5:28 am

sar is really useful — but I didn't know how useful until I read this post. Thanks for the info on seeing historical data on network activity.

28 Khalid April 7, 2014 at 1:46 am

Is there any way to check disk usage history (df -kh) output for last 5 to 6 days ?

Leave a Comment

Name

E-mail

Website

Notify me of followup comments via e-mail

Submit

Previous post: Quick Info about the Upcoming eBook

Next post: Get Your Copy of Sed and Awk 101 Hacks eBook

- RSS | Email | Twitter | Facebook | Google+

Search

- **COURSE**
  - Linux Sysadmin CentOS 6 Course - Master the Tools, Configure it Right, and be Lazy
- **EBOOKS**
  - Free Linux 101 Hacks 2nd Edition eBook - Practical Examples to Build a Strong Foundation in Linux
  - Bash 101 Hacks eBook - Take Control of Your Bash Command Line and Shell Scripting
  - Sed and Awk 101 Hacks eBook - Enhance Your UNIX / Linux Life with Sed and Awk
  - Vim 101 Hacks eBook - Practical Examples for Becoming Fast and Productive in Vim Editor
  - Nagios Core 3 eBook - Monitor Everything, Be Proactive, and Sleep Well

- **POPULAR POSTS**
  - 12 Amazing and Essential Linux Books To Enrich Your Brain and Library
  - 50 UNIX / Linux Sysadmin Tutorials
  - 50 Most Frequently Used UNIX / Linux Commands (With Examples)
  - How To Be Productive and Get Things Done Using GTD

- 30 Things To Do When you are Bored and have a Computer
- Linux Directory Structure (File System Structure) Explained with Examples
- Linux Crontab: 15 Awesome Cron Job Examples
- Get a Grip on the Grep! – 15 Practical Grep Command Examples
- Unix LS Command: 15 Practical Examples
- 15 Examples To Master Linux Command Line History
- Top 10 Open Source Bug Tracking System
- Vi and Vim Macro Tutorial: How To Record and Play
- Mommy, I found it! -- 15 Practical Linux Find Command Examples
- 15 Awesome Gmail Tips and Tricks
- 15 Awesome Google Search Tips and Tricks
- RAID 0, RAID 1, RAID 5, RAID 10 Explained with Diagrams
- Can You Top This? 15 Practical Linux Top Command Examples
- Top 5 Best System Monitoring Tools
- Top 5 Best Linux OS Distributions
- How To Monitor Remote Linux Host using Nagios 3.0
- Awk Introduction Tutorial – 7 Awk Print Examples
- How to Backup Linux? 15 rsync Command Examples
- The Ultimate Wget Download Guide With 15 Awesome Examples
- Top 5 Best Linux Text Editors
- Packet Analyzer: 15 TCPDUMP Command Examples
- The Ultimate Bash Array Tutorial with 15 Examples
- 3 Steps to Perform SSH Login Without Password Using ssh-keygen & ssh-copy-id
- Unix Sed Tutorial: Advanced Sed Substitution Examples
- UNIX / Linux: 10 Netstat Command Examples
- The Ultimate Guide for Creating Strong Passwords
- 6 Steps to Secure Your Home Wireless Network
- Turbocharge PuTTY with 12 Powerful Add-Ons

- **CATEGORIES**

- Linux Tutorials
- Vim Editor
- Sed Scripting
- Awk Scripting
- Bash Shell Scripting
- Nagios Monitoring
- OpenSSH
- IPTables Firewall
- Apache Web Server
- MySQL Database
- Perl Programming
- Google Tutorials
- Ubuntu Tutorials
- PostgreSQL DB
- Hello World Examples
- C Programming
- C++ Programming
- DELL Server Tutorials
- Oracle Database

- VMware Tutorials

Ramesh Natarajan

Seguir

- ## About The Geek Stuff

 My name is **Ramesh Natarajan**. I will be posting instruction guides, how-to, troubleshooting tips and tricks on Linux, database, hardware, security and web. My focus is to write articles that will either teach you or help you resolve a problem. Read more about Ramesh Natarajan and the blog.

- ## Support Us

Support this blog by purchasing one of my ebooks.

Bash 101 Hacks eBook

Sed and Awk 101 Hacks eBook

Vim 101 Hacks eBook

Nagios Core 3 eBook

- ## Contact Us

Email Me : Use this Contact Form to get in touch me with your comments, questions or suggestions about this site. You can also simply drop me a line to say hello!.

Follow us on Google+

Follow us on Twitter

Become a fan on Facebook