**Monitoring Tools**

**TOP**

The top command shows actual process activity. By default, it displays the most

CPU-intensive tasks running on the server and updates the list every five seconds. You can sort the processes by PID (numerically), age (newest first), time (cumulative time), and resident memory usage and time (time the process has occupied the CPU since startup).

**# top**

**vmstat**

vmstat provides information about processes, memory, paging, block I/O, traps, and CPU activity. The vmstat command displays either average data or actual samples. The sampling mode is enabled by providing vmstat with a sampling frequency and a sampling duration

# vmstat 2

**free**

The command /bin/free displays information about the total amount of free and used memory (including swap) on the system. It also includes information about the buffers and cache used by the kernel.

# free

**iostat**

The iostat command shows average CPU times since the system was started (similar to uptime). It also creates a report of the activities of the disk subsystem of the server in two parts: CPU utilization and device (disk) utilization. To use iostat to perform detailed I/O bottleneck and performance tuning.

**sar**

The sar command is used to collect, report, and save system activity information. The sar command consists of three applications: sar, which displays the data, and sa1 and sa2, which

are used for collecting and storing the data.The sar tool features a wide range of options so be sure to check the man page for it. The sar utility is part of the sysstat package. If you have sar running on most if not all of your systems. In case of a performance problem, you will have very detailed information on hand at very small overhead and no additional cost.

root@server1 root]# sar -u 3 10

**mpstat**

The mpstat command is used to report the activities of each of the available CPUs on a multiprocessor server. Global average activities among all CPUs are also reported. The

mpstat utility is part of the sysstat package.

The mpstat utility enables you to display overall CPU statistics per system or per processor. mpstat also enables the creation of statistics when used in sampling mode analogous to the

vmstat command with a sampling frequency and a sampling count.

[root@linux ~]# mpstat -P ALL

**iptraf**

iptraf monitors TCP/IP traffic in a real time manner and generates real time reports. It shows TCP/IP traffic statistics by each session, by interface, and by protocol. The iptraf utility is provided by the iptraf package.

The iptraf give us reports like the following:

IP traffic monitor: Network traffic statistics by TCP connection

General interface statistics: IP traffic statistics by network interface

Detailed interface statistics: Network traffic statistics by protocol

Statistical breakdowns: Network traffic statistics by TCP/UDP port and by packet size

LAN station monitor: Network traffic statistics by Layer2 address

**tcpdump**

The tcpdump and ethereal are used to capture and analyze network traffic. Both tool use the libpcap library to capture packets. They monitor all the traffic on a network adapter with promiscuous mode and capture all the frames the adapter has received. To capture all the packets, these commands should be executed with super user privilege to make the interface

promiscuous mode.

You can use these tools to dig into the network related problems. You can find TCP/IP retransmission, windows size scaling, name resolution problem, network misconfiguration, and more. Just keep in mind that these tools can monitor only frames the network adapter has received, not entire network traffic.

# tcpdump

tcpdump is a simple but robust utility. It has basic protocol analyzing capability allowing you to get a rough picture of what is happening on the network. tcpdump supports many options and flexible expressions for filtering the frames to be captured (capture filter). We’ll take a look at this below.

Options:

-i <interface> Network interface

-e Print the link-level header

-s <snaplen> Capture <snaplen> bytes from each packet

-n Avoide DNS lookup

-w <file> Write to file

-r <file> Read from file

-v, -vv, -vvv Vervose output

tcpdump -i eth0 'udp port 53' ( DNS query packets)

tcpdump -i eth0 'dst 192.168.1.10 ( FTP control and FTP data session)

**ethereal**

ethereal has similar functionality to tcpdump but is more sophisticated and has advanced protocol analyzing and reporting capability. It also has a GUI interface and a command line interface that uses the ethereal command, which is part of an ethereal package. Like tcpdump, the capture filter can be used, and it also supports the display filter. It can be used to narrow down the frames.

**Syslog**

There are hundreds of Linux applications on the market, each with their own configuration files and help pages. This variety makes Linux vibrant, but it also makes Linux system administration daunting. Fortunately, in most cases, Linux applications use the syslog utility to export all their errors and status messages to files located in the /var/log directory.

Syslog, and the logrotate utility that cleans up log files, are both relatively easy to configure but they frequently doesn't get their fair share of coverage in most texts

# Syslog

Syslog is a utility for tracking and logging all manner of system messages from the merely informational to the extremely critical

The first describes the function (facility) of the application that generated it. For example, applications such as mail and cron generate messages with easily identifiable facilities named "mail" and "cron".

The files to which syslog will write each type of message received is set in the /etc/syslog.conf configuration file. This file consists of two columns, the first lists the facilities and severities of messages to expect and the second lists the files to which they should be logged. By default, RedHat/Fedora's

/etc/syslog.conf file is configured to put most of the messages the file /var/log/messages.

Here is a sample:

.info;mail.none;authpriv.none;cron.none /var/log/messages

all messages of severity "info" and above are logged, but none from the mail, cron or authentication facilities/subsystems. You can make this logging even more sensitive by replacing the line above with one that captures all messages from debug severity and above in the /var/log/messages file

\*.debug /var/log/messages

Certain applications will additionally log to their own application specific log files and directories independent of the syslog.conf file.

Activating Changes To The syslog Configuration File

Changes to /etc/syslog.conf will not take effect until you restart syslog. Issue this command to do so:

[root@server1 tmp]# /etc/init.d/syslog restart

**Logrotate**

Logrotate is a Linux utility that renames and reuses system error log files on a periodic basis so that they don't occupy excessive disk space.

The /etc/logrotate.conf File

This is logrotate's general configuration file in which you can specify the frequency with which the files are reused. You can specify either "weekly" or "daily" rotation parameter. In the case below the weekly option is "commented out" with a "#", allowing for daily updates Sample contents of /etc/logrotate.conf

# rotate log files weekly

#weekly

# rotate log files daily daily

# keep 4 weeks worth of backlogs

#rotate 4

# keep 7 days worth of backlogs rotate 7

# create new (empty) log files after rotating old ones create

The /etc/logrotate.d Directory

Most Linux applications that use syslog will put an additional configuration file in this directory to specify the names of the log files to be rotated. It is a good practice to verify that all new applications that you want to use the syslog log have configuration files in this directory /etc/logrotate.d/syslog

The /etc/logrotate.d/apache File (For Apache)

/var/log/httpd/access\_log /var/log/httpd/agent\_log

/var/log/httpd/error\_log /var/log/httpd/referer\_log {

missingok

sharedscripts

postrotate

/bin/kill -HUP `cat /var/run/httpd.pid 2>/dev/null` 2>

/dev/null || true

endscript

}

The above logrotate settings will not take effect until you issue the following command to do so:

[root@server1 tmp]# logrotate -f

If you want logrotate to reload only a specifc configuration file, and not all of them, then issue the logrotate command with just that filename as the argument like this:

[root@server1 tmp]# logrotate -f /etc/logrotate.d/syslog