11.1 Common Log Files

File/Directory	Contents
/var/log/boot.log **/var/log/boot.msg**	The system writes messages generated during the boot process to either the boot.log or boot.msg file depending on the distribution.
/var/log/faillog **/var/log/btmp**	Login failures for user accounts are listed in either the faillog or btmp file, depending on the distribution.
/var/log/firewall **/var/log/firewalld**	Depending on the distribution, the system stores log entries for the host firewall.
/var/log/lastlog	The lastlog file holds information about the last time each user logged in.
/var/log/maillog	The maillog file contains reports on mail server status and messages related to incoming and outgoing mail.
/var/log/messages	The messages file is the default file for storing system messages. This file may include copies of messages that appear on the console, internal kernel messages, and messages sent by networking programs. The messages file is based on the init daemon, used on older Linux distributions.
/var/log/warn	The warn file displays warning messages from many processes by default.
/var/log/wtmp	The wtmp file keeps track of all users who have logged into and out of the system, as well as listing every connection and runlevel change.
/var/log/dmesg	The dmesg file is often called the kernel ring buffer. It reports messages received in the process of configuring hardware devices as the system boots.
/var/log/secure	The secure file logs any attempts to log in as the root user or attempts to use the su command. This file also contains information on remote logins and failed root user login attempts.
/var/log/sa	The /var/log/sa directory stores /sa[n] files, which contain all performance information for the day of the month indicated by [n]. For example, /var/log/sa/sa15 contains performance information for the fifteenth day of the month, and it will be overwritten on the fifteenth day of the next month.
/var/log/cron	The cron file stores messages related to tasks scheduled with cron. It keeps track of which tasks are run and when they were started.
/var/log/rpmpkgs	On Red Hat systems, the rpmpkgs file tracks installed packages. It also records all kernel packages on the system.
/tmp/install.log **/root/install.log**	The install.log may or may not be present, depending on the distribution. This file records messages related to the installation and can be useful for installation records for a computer.
/etc/rsyslog.conf	The rsyslog.conf file is the main configuration file for the rsyslogd, which logs system messages on Linux systems. This file specifies rules for logging. For every log message received, rsyslog looks at its configuration file /etc/rsyslog.conf to determine how to handle that message. If no rule statement matches the message, Rsyslog discards it.
/var/log/kern.log	The var/log/kern.log file provides a detailed log of messages from the Linux kernel. These messages may prove useful for trouble-shooting a new or custom-built kernel.
/var/log/*[application]*	Many applications also create logs in the /var/log directory. If you list the contents of your /var/log subdirectory, you will see familiar names such as /var/log/apache2 representing the logs for the Apache 2 web server, or /var/log/samba, which contains the logs for the Samba server.

11.1 Centralized Logging (daemons)

Daemon	Description
rsyslog	A lightweight daemon installed on most common Linux distributions. It offers fast, high-performance, great security features and a modular design. It is able to accept inputs from a wide variety of sources, transform them, and output the result to diverse destinations. This is the most popular daemon.
syslog-ng	syslog-ng collects logs from any source, processes them in real-time, and delivers them to a wide variety of destinations. syslog-ng also allows you to flexibly collect, parse, classify, rewrite and correlate logs from across your infrastructure and store or route them to log analysis tools.
Fluentd	Lets you unify the data collection and consumption for better use and understanding of data. Fluentd tries to structure data as JSON as much as possible and has a flexible plug-in system that allows the community to extend its functionality.
logstash	A heavy-weight agent capable of performing more advanced processing and parsing. It is capable of obtaining data from a multitude of sources simultaneously, and after processing, it can then send it to your favorite “stash.”

11.1 journald Configuration (Parameters)

Parameter	Description
MaxFileSec	Specifies the maximum amount of time to store entries in the journal file before starting a new file.
MaxRetentionSec	Specifies the maximum amount of time to store journal entries. Any entries older than the specified time are automatically deleted from the journal file.
MaxLevelStore	Specifies the maximum log level of messages stored in the journal file. All messages equal to or less than the log level specified are stored. Any messages above the specified level are dropped. This parameter can be set to: • **emerg** (0) • **alert** (1) • **crit** (2) • **err** (3) • **warning** (4) • **notice** (5) • **info** (6) • **debug** (7)
ForwardToSyslog	Configures journald to forward log messages to the traditional syslogd daemon.

11.1 Commands to View the Journal

Command	Function
journalctl	Views the entire journal. The output begins at the beginning of the journal and pauses one page at a time. To exit out of journalctl, press q.
journalctl -b	Views system boot messages. The messages from the most recent system boot are displayed by default. To display messages from a specific boot, use the following options with this command: • Specify a positive number to display messages from the specified system boot, starting from the beginning of the journal. • Specify a negative number to display messages from the specified system boot starting from the end of the journal.
journalctl -u	Displays only log entries related to a specific service running on the system.
journalctl -f	Displays the last few entries in the journal. The journalctl command then monitors the journal and prints new entries as they are added.

11.1 View and Manage Binary Log Files

Command	Function
dmesg	Views the boot logs and troubleshoots hardware errors. The dmesg command shows information about all the hardware controlled by the kernel and displays error messages as they occur.
dmesg -n #	Controls which error messages are sent to the console. For example, dmesg -n 1 sends only the most critical errors (0 and 1) to the console. Other messages are still logged in the log files.
last	Shows all users who have logged in to and out of the system, as well as listing every connection and runlevel change (for example, the contents of the /var/log/wtmp file).
faillog **lastb**	Shows all failed login attempts on the system (for example, the contents of the /var/log/btmp file or /var/log/faillog file, depending on the distribution).
lastlog	Shows a list of the dates and times for the last login for each user.
logger	Changes the message severity and where logged messages are sent.
logrotate	Manages, compresses, renames, and deletes log files based on specific criteria (such as size or date).
sar	Views system statistics. sar is short for System Activity Report. It comes as part of the sysstat (System Statistics) package. When used alone, it returns CPU statistics. Common options include the following: **-A** displays all information. **-b** displays I/O statistics. **-B** displays swap statistics. **-f /var/log/sa *filename*** displays information from the specified file.

11.1 Commands Used with logrotate

Command	Function
compress	Compresses old log files using gzip.
maxage	Removes rotated logs that are older than the specified number of days.
dateext	Uses a daily extension on archived files using file.YYYYMMDD format.
rotate	Specifies the number of times to rotate the log before deleting it.
size	Rotates or removes log files based on file size as follows: • **size k** specifies the size in kilobytes. • **size M** specifies the size in megabytes. • **size G** specifies the size in gigabytes.
notifempty	Prohibits empty logs from being rotated.
missingok	Prevents errors from being displayed for missing log files.
create	Creates a log file with a name identical to the one just rotated. The command specifies the mode (permissions) of the file as well as the owner and group for the file.
postrotate	Indicates the start of script commands to be executed after log files are rotated. The term endscript must be used to indicate the end of the script.

11.2 Automatic Bug Reporting Tool (ABRT) Components

Component	Function
abrt	The ABRT utility consists of a daemon and a collection of tools for handling crashes and monitoring logs for errors.
gnome-abrt	GUI application for problem management and reporting.
libreport	A generic library that provides an API for reporting problems to different entities such as email, bugzilla, faf, scp upload, etc. By default, the notifications are sent to root at the local machine. You can use the conf file to change where notifications are sent.
faf	A crash-collecting server, also known as the ABRT server. It provides accurate statistics of incoming reports and acts as a proxy in front of issue tracking software, such as bugzilla. It's designed to receive anonymous µReports and to find similar information among them. For known issues, it generates responses with links to faf's problem page, an issue tracker, or knowledge base entry.
satyr	Algorithms for program failure processing, analysis, and reporting. More specifically, satyr: • Generates a description of the failure from various stack traces • Analyzes stack traces of failed processes. • Discovers the thread that caused the failure, in multi-threaded stack traces. • Generates a failure report in a specified format • Sends the report to a remote machine.
retrace server	Provides dump analysis and backtrace generation service over a network using HTTP protocol. It is currently being merged to faf.
µReports	Micro reports, µReports, are small, machine readable reports used for automated reporting. They identify the operating system, versions of software running when the system crashed, information about the call stack at the time of the crash, and a stack trace, or multiple stack traces in the case of multi-threaded programs. No private data is allowed in the report.

11.2 ABRT Tools and Commands

Command	Function
abrt-cli list	Lists all crashes on a machine. Before this command can be used, you may need to enable the autoreporting feature. To enable autoreporting, run: **abrt-auto-reporting enabled**
abrt-cli list -d *<ID_OR_PATH>*	Displays detailed report data about a particular problem.
abrt-cli report *<ID_OR_PATH>*	Reports a problem.
abrt-cli remove *<ID_OR_PATH>*	Deletes a problem.

11.2 CPU Monitoring and Configuration

Command	Function
sar	The sar (System Activity Report) command is part of the sysstat package. The sar command reports cumulative values in the count and interval parameters of activity counters in the operating system. Example of information you can use sar to view is as follows: • Collective CPU usage • Individual CPU statistics • Memory used and available • Swap space used and available • Overall I/O activities of the system • Individual device I/O activities • Network statistics **Be aware of the following about the sar command:** • Writes information the specified number of times, spaced at the specified intervals in seconds. If the interval parameter is set to zero, the sar command displays the average statistics for the time since the system was started. If the interval parameter is specified without the count parameter, then reports are generated continuously. The collected data can also be saved in the file specified by the -o filename flag, in addition to being displayed onto the screen. If filename is omitted, sar uses the standard system activity daily data file, the /var/log/sa/sadd file, where the dd parameter indicates the current day. By default, all the data available from the kernel is saved in the data file. • The sar command extracts and writes to standard output records previously saved in a file. This file can be either the one specified by the -f flag or, by default, the standard system activity daily data file. • Without the -P flag, the sar command reports system-wide (global among all processors) statistics, which are calculated as averages for values expressed as percentages, and as sums otherwise. If the -P flag is given, the sar command reports activity which relates to the specified processor or processors. If -P ALL is given, the sar command reports statistics for each individual processor and global statistics among all processors. • You can select information about specific system activities using flags. Not specifying any flags selects only CPU activity. Specifying the -A flag is equivalent to specifying **-bBdqrRSvwWy -I SUM -I XALL -n ALL -u ALL -P ALL**. • The default version of the sar command (CPU utilization report) might be one of the first facilities the user runs to begin system activity investigation, because it monitors major system resources. If CPU utilization is near 100 percent (user + nice + system), the workload sampled is CPU-bound. • If multiple samples and reports are desired, it is convenient to specify an output file for the sar command. Run the sar command as a background process. The syntax for this is: sar -o datafile interval count >/dev/null 2>&1 & • All data is captured in binary form and saved to a file (datafile). The data can then be selectively displayed with the sar command using the -f option. Set the interval and count parameters to select count records at interval second intervals. If the count parameter is not set, all the records saved in the file will be selected. Collection of data in this manner is useful to characterize system usage over a period of time and determine peak usage hours. *Note: The sar command only reports on local activities.*

/proc/cpuinfo	Displays details about individual CPU cores and outputs content with **less** or **cat**.
uptime	Displays of the following information. • Current time. • How long the system has been running. • How many users are currently logged on. • System load averages for the past 1, 5, and 15 minutes. Loads averages is a number that represents the average number of instructions waiting for CPU time. • A load average of 1 (for a single processor system CPU) means that the CPU is keeping up with the demand or is 100% utilized. • A load average of less than 1 indicates that the CPU is underutilized — in other words, it has perfect utilization. • Any number greater than 1 indicates that the CPU has been asked to do more than it can do in real time, and some tasks will have to wait for CPU time. In multiprocessor systems, you use the same metrics multiplied by the number of processes. For example, a load average of 2 for a dual processor system indicates that the processors are keeping up with demand.
sysctl	This command lets you configure the kernel parameters at runtime. To view all of the available parameters, as root run sysctl -a. If needed, you can change one of these parameters which takes effect immediately, and if needed, the change can be written permanently using the -w switch. Example: **sysctl -w kernel.sysrq="1"** Keep in mind that there may not be any parameters that directly correlate to CPU usage, but instead, changing some parameters may have a direct impact on the CPU’s performance. For example, tuning the network adapter by disabling the TCP timestamps option can result in better CPU utilization. This is really a powerful tool, so use it with care. There are a lot of kernel parameters that if changed could really mess your system up. Be sure to look at the man pages for sysctl before you start working with it, and be informed about any variable, and what it affects before changing the value.

11.2 Memory Monitoring and Configuration

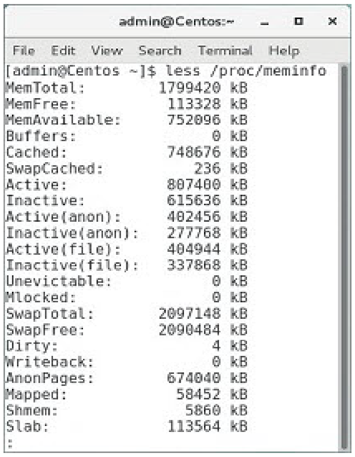
Command **Function**

vmstat

vmstat reports information about processes, memory, paging, block IO, traps, disks, and CPU activity. The first report produced gives averages since the last reboot. Additional reports give information on a sampling period of length delay. The process and memory reports are instantaneous in either case.

/proc/meminfo

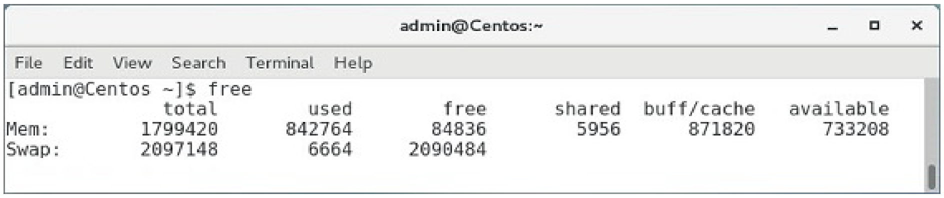
This file reports statistics about memory usage on the system. It is used by the free command (see below) to report the amount of free and used memory (both physical and swap) on the system, as well as the shared memory and buffers used by the kernel. Each line of the file consists of a parameter name, followed by a colon, the value of the parameter, and an optional unit of measurement (e.g., "kB").



```
[admin@Centos ~]$ less /proc/meminfo
MemTotal:        1799420 kB
MemFree:          113328 kB
MemAvailable:     752096 kB
Buffers:               0 kB
Cached:           748676 kB
SwapCached:          236 kB
Active:           807400 kB
Inactive:         615636 kB
Active(anon):     402456 kB
Inactive(anon):   277768 kB
Active(file):     404944 kB
Inactive(file):   337868 kB
Unevictable:           0 kB
Mlocked:               0 kB
SwapTotal:       2097148 kB
SwapFree:        2090484 kB
Dirty:                 4 kB
Writeback:             0 kB
AnonPages:        674040 kB
Mapped:            58452 kB
Shmem:              5860 kB
Slab:             113564 kB
:
```

free

The free command displays the total amount of free and used physical and swap memory in your computer, as well as the buffers and caches used by the kernel. The information is gathered by parsing /proc/meminfo.



```
[admin@Centos ~]$ free
              total        used        free      shared  buff/cache   available
Mem:        1799420      842764       84836        5956      871820      733208
Swap:       2097148        6664     2090484
```

Switches for the free command include the following:

-b displays the amount of memory in bytes
-k (default) displays it in kilobytes
-m displays it in megabytes
-t displays a line containing the totals.
-o disables the display of a "buffer adjusted" line. If the -o option is not specified, free subtracts buffer memory from the used memory and adds it to the free memory reported.
-s activates continuous polling delay seconds apart. You may actually specify any floating point number for delay, usleep is used for microsecond resolution delay times.
-l shows detailed low and high memory statistics.
-V displays version information.

Out-of-memory killer (OOM killer)	When a Linux computer is critically low on memory, the Linux kernel uses the out-of-memory killer (OOM killer) process to review all running processes and kill one or more of them in order to free up system memory and keep the system running. Some memory issues are related to memory leaks, which is a process that is consuming large amounts of memory, but not releasing it when finished using it. The out-of-memory killer works by reviewing all running processes and assigning them a badness score. The process that has the highest score is the one that is killed. The out-of-memory killer assigns a badness score based on a number of criteria, such as: • The process and all of its child processes are using a lot of memory. • The minimum number of processes that need to be killed (ideally one) in order to free up enough memory to resolve the situation. • Root, kernel, and important system processes are given much lower scores. Viewing the system log is the best method to see if OOM killer was the reason a program was killed. Another method is to run: dmesg | grep -i "killed process"
Buffer cache output	Buffers are used by programs with active I/O operations, i.e. data waiting to be written to disk. Cache is the result of completed I/O operations, i.e. buffers that have been flushed or data read from disk to satisfy a request.

11.2 Tools to Troubleshoot Hardware Issues

Tool	Function
dmidecode	The dmidecode tool is used for dumping a computer's desktop management interface (DMI) or SMBIOS table contents in a human-readable format. This table contains a description of the system's hardware components, as well as other useful pieces of information — such as serial numbers and BIOS revision. Using this table, you can retrieve this information without having to probe for the actual hardware. While this is a good point in terms of report speed and safeness, this also makes the presented information possibly unreliable.
lshw	The lshw is used to extract detailed information on the hardware configuration of the machine. It can report exact memory configuration, firmware version, mainboard configuration, CPU version and speed, cache configuration, bus speed, etc. on DMI-capable x86 or IA-64 systems and on some PowerPC machines (PowerMac G4 is known to work). It currently supports DMI (x86 and IA-64 only), OpenFirmware device tree (PowerPC only), PCI/AGP, CPUID (x86), IDE/ATA/ATAPI, PCMCIA (only tested on x86), SCSI and USB. -version displays the version of lshw and exits.