LinuxLogFiles

Needs Expansion
This article is incomplete, and needs to be expanded.
More info...

Introduction

Contents

One of the things which makes GNU/Linux a great operating system is that virtually anything and everything happening on and to the system may be logged in some manner. This information is invaluable for using the system in an informed manner, and should be one of the first resources you use to trouble-shoot system and application issues. The logs can tell you almost anything you need to know, as long as you have an idea where to look first.

Your Ubuntu system provides vital information using various system log files. These log files are typically plain ASCII text in a standard log file format, and most of them sit in the traditional system log subdirectory `/var/log`. Many are generated by the system log daemon, `syslogd` on behalf of the system and certain applications, while some applications generate their own logs by writing directly to files in `/var/log`.

This guide talks about how to read and use several of these system log files, how to use and configure the system logging daemon, `syslogd`, and how log rotation works. See the **Resources** section for additional information.

Target Audience

This guide will be simple enough to use if you have any experience using the console and editing text files using a text editor. See the end of this document for some essential commands that may help you find your way around these files if you're relatively new to the command line.

System Logs

System logs deal primarily with the functioning of the Ubuntu system, not necessarily with additional applications added by users. Examples include authorization mechanisms, system daemons, system messages, and the all-encompassing system log itself, *syslog*.

Authorization Log

The Authorization Log tracks usage of authorization systems, the mechanisms for authorizing users which prompt for user passwords, such as the Pluggable Authentication Module (PAM) system, the `sudo` command, remote logins to `sshd` and so on. The Authorization Log file may be accessed at `/var/log/auth.log`. This log is useful for learning about user logins and usage of the sudo command.

Use `grep` to cut down on the volume. For example, to see only information in the Authorization Log pertaining to `sshd` logins, use this:

```
grep sshd /var/log/auth.log | less
```

Daemon Log

A daemon is a program that runs in the background, generally without human intervention, performing some operation important to the proper running of your system. The daemon log at `/var/log/daemon.log` and contains information about running system and application daemons such as the Gnome Display Manager daemon `gdm`, the Bluetooth HCI daemon `hcid`, or the MySQL database daemon `mysqld`. This can help you trouble-shoot problems with a particular daemon.

Again, use grep to find specific information, plugging in the name of the daemon you're interested in.

Debug Log

The debug log at `/var/log/debug` and provides detailed debug messages from the Ubuntu system and applications which log to `syslogd` at the DEBUG level.

Kernel Log

The kernel log at `/var/log/kern.log` provides a detailed log of messages from the Ubuntu Linux kernel. These messages may prove useful for trouble-shooting a new or

custom-built kernel, for example.

Kernel Ring Buffer

The kernel ring buffer is not really a log file per se, but rather an area in the running kernel you can query for kernel bootup messages via the `dmesg` utility. To see the messages, use this:

```
dmesg | less
```

Or to search for lines that mention the Plug & Play system, for example, use `grep` like this:

```
dmesg | grep pnp | less
```

By default, the system initialization script `/etc/init.d/bootmisc.sh` sends all bootup messages to the file `/var/log/dmesg` as well. You can view and search this file the usual way.

System Log

The system log typically contains the greatest deal of information by default about your Ubuntu system. It is located at `/var/log/syslog`, and may contain information other logs do not. Consult the System Log when you can't locate the desired log information in another log. It also contains everything that used to be in `/var/log/messages`.

Application Logs

Many applications also create logs in `/var/log`. 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. This section of the guide introduces some specific examples of application logs, and information contained within them.

Apache HTTP Server Logs

The default installation for Apache2 on Ubuntu creates a log subdirectory: `/var/log/apache2`. Within this subdirectory are two log files with two distinct purposes:

- `/var/log/apache2/access.log` - records of every page served and every file loaded by the web server.
- `/var/log/apache2/error.log` - records of all error conditions reported by the HTTP server

By default, every time Apache accesses a file or page, the access logs record the IP address, time and date, browser identification string, HTTP result code and the text of the actual query, which will generally be a GET for a page view. Look at the Apache documentation for a complete rundown; quite a lot can be gleaned from this file, and indeed many statistical packages exist that perform analyses of these logs.

Also, every time any error occurs, Apache adds a line to the error log. If you run PHP with error and warning messages disabled, this can be your only way to identify bugs.

CUPS Print System Logs

The Common Unix Printing System (CUPS) uses the default log file `/var/log/cups/error_log` to store informational and error messages. If you need to solve a printing issue in Ubuntu, this log may be a good place to start.

Rootkit Hunter Log

The Rootkit Hunter utility (`rkhunter`) checks your Ubuntu system for backdoors, sniffers and rootkits, which are all signs of compromise of your system. The log rkhunter uses is located at `/var/log/rkhunter.log`.

Samba SMB Server Logs

The Server Message Block Protocol (SMB) server, Samba is popularly used for sharing files between your Ubuntu computer and other computers which support the SMB protocol. Samba keeps three distinct types of logs in the subdirectory `/var/log/samba`:

- `log.nmbd` - messages related to Samba's NETBIOS over IP functionality (the network stuff)
- `log.smbd` - messages related to Samba's SMB/CIFS functionality (the file and print sharing stuff)
- `log.[IP_ADDRESS]` - messages related to requests for services from the IP address contained in the log file name, for example, `log.192.168.1.1`.

X11 Server Log

The default X11 Windowing Server in use with Ubuntu is the Xorg X11 server, and assuming your computer has only one display defined, it stores log messages in the file `/var/log/Xorg.0.log`. This log is helpful for diagnosing issues with your X11 environment.

Non-Human-Readable Logs

Some log files found in the `/var/log` subdirectory are designed to be readable by applications, not necessarily by humans. Some examples of such log files which appear in `/var/log` follow.

Login Failures Log

The login failures log located at `/var/log/faillog` is actually designed to be parsed and displayed by the `faillog` command. For example, to print recent login failures, use this:

```
faillog
```

Last Logins Log

The last logins log at `/var/log/lastlog` should not typically be parsed and examined by humans, but rather should be used in conjunction with the `lastlog` command. For example to see a listing of logins with the `lastlog` command, displayed one page per screen with the `less` command, use the following command:

```
lastlog | less
```

Login Records Log

The file `/var/log/wtmp` contains login records, but unlike `/var/log/lastlog` above, `/var/log/wtmp` is not used to show a list of recent logins, but is instead used by other utilities such as the `who` command to present a listed of currently logged in users. This command will show the users currently logged in to your machine:

```
who
```

System Logging Daemon (syslogd)

The system logging daemon `syslogd`, also known as `sysklogd`, awaits logging messages from numerous sources and routes the messages to the appropriate file or network destination. Messages logged to `syslogd` usually contain common elements like system hostnames and time-stamps in addition to the specific log information.

Configuration of syslogd

The `syslogd` daemon's configuration file is `/etc/syslog.conf`. Each entry in this file consists of two fields, the selector and the action. The selector field specifies a facility to be logged, such as for example the **auth** facility which deals with authorization, and a priority level to log such information at, such as **info**, or **warning**. The action field consists of a target for the log information, such as a standard log file (i.e. `/var/log/syslog`), or the hostname of a remote computer to send the log information to.

Echoing Messages to syslogd With Logger

A neat utility exists in the `logger` tool, which allows one to place messages into the System Log (i.e. `/var/log/syslog`) arbitrarily. For example, assume your user name is `buddha`, and you would like to enter a message into the syslog about a particularly delicious pizza you're eating, you could use a command such as the following at a terminal prompt:

```
logger This Pizza from Vinnys Gourmet Rocks
```

and you would end up with a line in the `/var/log/syslog` file like this:

```
Jan 12 23:34:45 localhost buddha: This Pizza from Vinnys Gourmet Rocks
```

You can even specify a tag the messages come from, and redirect the output standard error too.

```
#
# sample logger error jive
#
logmsg="/usr/bin/logger -s -t MyScript "

# announce what this script is, even to the log
$logmsg "Directory Checker FooScript Jive 1.0"

# test for the existence of Fred's home dir on this machine
if [ -d /home/fred ]; then
   $logmsg "I. Fred's Home Directory Found"
else
   $logmsg "E. Fred's Home Directory was NOT Found. Boo Hoo."
   exit 1
fi
```

Executing this script as `chkdir.sh` on the machine `butters` where Fred does not have a home directory, `/home/fred`, gives the following results:

```
bumpy@butters:~$./chkdir.sh
MyScript: Directory Checker FooScript Jive 1.0
MyScript: E. Fred's Home Directory was NOT Found. Boo Hoo.
bumpy@butters:~$tail -n 2 /var/log/syslog
Jan 12 23:23:11 localhost MyScript: Directory Checker FooScript Jive 1.0
Jan 12 23:23:11 localhost MyScript: E. Fred's Home Directory was NOT Found. Boo Hoo.
```

So, as you can see, we received the messages both via standard error, at the terminal prompt, and they also appear in our syslog.

Log Rotation

When viewing directory listings in `/var/log` or any of its subdirectories, you may encounter log files with names such as `daemon.log.0`, `daemon.log.1.gz`, and so on. What are these log files? They are 'rotated' log files. That is, they have automatically been renamed after a predefined time-frame, and a new original log started. After even more time the log files are compressed with the `gzip` utility as in the case of the example `daemon.log.1.gz`. The purpose of log rotation is to archive and compress old logs so that they consume less disk space, but are still available for inspection as needed. What handles this functionality? Why, the `logrotate` command of course! Typically, logrotate is called from the system-wide cron script `/etc/cron.daily/logrotate`, and further defined by the configuration file `/etc/logrotate.conf`. Individual configuration files can be added into `/etc/logrotate.d` (where the apache2 and mysql configurations are stored for example).

This guide will not cover the myriad of ways `logrotate` may be configured to handle the automatic rotation of any log file on your Ubuntu system. For more detail, check the **Resources** section of this guide.

 **NOTE:** You may also rotate system log files via the `cron.daily` script `/etc/cron.daily/sysklogd` instead of using logrotate. Actually, the utility `savelog` may produce unexpected results on log rotation which configuring `logrotate` seems to have no effect on. In those cases, you should check the cron.daily `sysklogd` script in `/etc/cron.daily/sysklogd` and read the `savelog` manual page to see if `savelog` is not in fact doing the rotation in a way that is not what you are specifying with `logrotate`.

Essential Commands

If you're new to the console and the Linux command line, these commands will get you up and running to the point where you can work with log files at a basic level.

Getting Started

To change to the log directory, where most of these files sit, use the `cd` command. This saves having to type out a full path name for every subsequent command:

```
cd /var/log
```

Editing Files

You can view and edit files in GEdit or Kate, the simple text editors that come with Ubuntu and Kubuntu respectively, but these can be overkill when all you want to do is look at a file or make simple changes. The easiest editor to use from the console is nano, which is less powerful but also less complicated than vim or emacs. The command to edit a particular logfile `/var/log/example.log` using nano is:

```
nano example.log
```

Press Ctrl+X to exit. It will ask if you want to save your changes when you exit, but unless you run it with the `sudo` command the files won't be writable. In general, you won't want to save your changes to log files, of course.

Viewing Files

To simply look at a file, an editor is overkill. Use the `less` command, which pages through a file one screen at a time:

```
less example.log
```

You don't need `sudo` to look at a file. Press h for help, or q to quit. The cursor keys and page up/down keys will work as expected, and the slash key ("/") will do a case-sensitive search; the n key repeats the last search.

Viewing the Beginning of Files

To see the first ten lines of a file, use the `head` command:

```
head example.log
```

To see some other number of lines from the beginning of the file, add the -n switch, thus:

```
head -n 20 example.log
```

Viewing the End of Files

To see the final ten lines of a file, the analogous command is `tail`:

```
tail example.log
```

Again, the -n switch gives you control over how many lines it displays:

```
tail -n 20 example.log
```

Watching a Changing File

Also, the -f ("follow") switch puts tail into a loop, constantly waiting for new additions to the file it's displaying. This is useful for monitoring files that are being updated in real time:

```
tail -f example.log
```

Press Ctrl+C to quit the loop.

Searching Files

Because log files can be large and unwieldy, it helps to be able to focus. The `grep` command helps you strip out only the content you care about. To find all the lines in a file containing the word "system", for example, use this:

```
grep "system" example.log
```

To find all the lines containing "system" at the beginning of the line, use this:

```
grep "^system" example.log
```

Note the caret symbol, a regular expression that matches only the start of a line. This is less useful for standard log files, which always start with a date and time, but it can be handy otherwise. Not all files have a standard format.

Any time the result of a `grep` is still too long, you can pipe it through `less`:

```
grep "system" example.log | less
```

Resources

Additional information on system and application logs and `syslogd` is available via the following resources:

Local System Resources

`man dmesg`	System manual page for the `dmesg` kernel ring buffer utility
`man faillog`	System manual page for the `faillog` command (and also the faillog configuration file via `man 5 faillog`)
`man grep`	System manual page for the `grep` pattern searching utility

`man head`	System manual page for the `head` utility
`man klogd`	System manual page for the kernel log daemon (`klogd`)
`man last`	System manual for the `last` command which shows last logged in users
`man less`	System manual page for the `less` paging utility
`man logger`	System manual page for the `logger` command-line interface to syslog utility
`man logrotate`	System manual page for the the `logrotate` utility
`man savelog`	System manual page for the `savelog` log file saving utility
`man syslogd`	System manual page for the system log daemon (`syslogd`)
`man syslog.conf`	System manual page for the `syslogd` configuration file
`man tail`	System manual page for the `tail` utility

WWW Resources

Checking Your System Logs with awk

Syslog - Watching Your Logs

http://www.ibm.com/developerworks/linux/library/l-roadmap5/-Linux Logging

Sawing Linux Logs With Simple Tools

CategorySystem

LinuxLogFiles (last edited 2015-01-23 18:17:29 by jackbravo @ 189-212-110-221.static.axtel.net[189.212.110.221]:jackbravo)