Understanding logrotate utility

Logs are useful when you want to track usage or troubleshoot an application. As more information gets logged, however, log files use more disk space. Over time a log file can grow to unwieldy size. Running out of disk space because of a large log file is a problem, but a large log file can also slow down the process of resizing or backing up your virtual server. Additionally, it's hard to look for a particular event if you have a million log entries to skim through. So it's a good idea to keep log files down to a manageable size, and to prune them when they get too old to be of much use.

Fortunately, the logrotate utility makes log rotation easy. "Log rotation" refers to the practice of archiving an application's current log, starting a fresh log, and deleting older logs. The system usually runs logrotate once a day, and when it runs it checks rules that can be customized on a per-directory or per-log basis.

How logrotate works

The system runs logrotate on a schedule, usually daily. On most distributions, the script that runs logrotate daily is located at `/etc/cron.daily/logrotate`.

Some distributions use a variation. For example, on Gentoo the logrotate script is located at `/etc/cron.daily/logrotate.cron`.

If you want logrotate to run more often (for hourly log rotation, for example) you need to use cron to run logrotate through a script in `/etc/cron.hourly`.

When logrotate runs, it reads its configuration files to determine where to

find the log files that it needs to rotate, how often the files should be rotated, and how many archived logs to keep.

logrotate.conf

The main logrotate configuration file is located at `/etc/logrotate.conf`.

The file contains the default parameters that logrotate uses when it rotates logs. The file is commented, so you can skim it to see how the configuration is set up. Several of the specific commands in that file are described later in this article.

Note that one line in the file reads:

```
include /etc/logrotate.d
```

That directory contains most of the application-specific configuration files.

logrotate.d

Use the following command to list contents of the directory that stores application-specific log settings:

```
ls /etc/logrotate.d
```

Depending on how much is installed on your server, this directory might contain no files or several. In general, applications that are installed through your package manager will also create a config file in `/etc/logrotate.d`.

Usually the directory contains a configuration file for your syslog service, which logrotate reads when it rotates the system logs. This file contains an

entry for various system logs, along with some commands similar to those contained in `logrotate.conf`.

NOTE: On versions of Ubuntu operating systems earlier than Karmic Koala (9.10) there is no entry for a syslog service. Before that release, the system logs were rotated by a `savelog` command run from the `/etc/cron.daily/sysklogd` script.

Inside an application file

As an example, consider the contents of a logrotate configuration file that might be put in place when you install Apache on a Fedora system:

```
/var/log/httpd/*log {
    missingok
    notifempty
    sharedscripts
    postrotate
    /sbin/service httpd reload > /dev/null 2>/dev/null || true
    endscript
  }
```

When logrotate runs, it checks for any files in `/var/log/httpd` that end in `log` and rotates them, if they aren't empty. If it checks the httpd directory and doesn't find any log files, it doesn't generate an error. Then it runs the command in the `postrotate/endscript` block (in this case, a command that tells Apache to restart), but only after it has processed all the specified logs.

This example file does not contain some settings that are included in the `logrotate.conf` file. The commands in `logrotate.conf` act as defaults for

log rotation. You can specify different settings for any application when you want to override the defaults. For example, if you run a busy web server, you might want to include a `daily` command in Apache's configuration block so that Apache's logs will rotate daily instead of the default weekly rotation.

The next section describes some of the more commonly-used commands actually do in a logrotate configuration file.

Configuration commands

You can get a full list of commands used in logrotate configuration files by checking the man page:

```
man logrotate
```

This section describes the more commonly-used commands.

Remember, the configuration files for applications in `/etc/logrotate.d` inherit their defaults from the main `/etc/logrotate.conf` file.

Log files

A log file and its rotation behavior are defined by listing the log file (or files) followed by a set of commands enclosed in curly brackets. Most application configuration files will contain just one of these blocks, but it's possible to put more than one in a file, or to add log file blocks to the main `logrotate.conf` file.

You can list more than one log file for a block by using a wildcard in the name or by separating log files in the list with spaces. For example, to specify all files in the directory /var/foo that end in `.log`, and the file

`/var/bar/log.txt`, you would set up the block as follows:

```
 /var/foo/*.log /var/bar/log.txt {
        rotate 14
        daily
        compress
        delaycompress
        sharedscripts
        postrotate
                /usr/sbin/apachectl graceful > /dev/null
        Endscript
}
```

Rotate count

The `rotate` command determines how many archived logs are returned before logrotate starts deleting the older ones. For example:

```
rotate 4
```

This command tells logrotate to keep four archived logs at a time. If four archived logs exist when the log is rotated again, the oldest one is deleted to make room for the new archive.

Rotation interval

You can specify a command that tells logrotate how often to rotate a particular log. The possible commands include:

```
daily
weekly
```

```
monthly
yearly
```

If a rotation interval is not specified the log will be rotated whenever logrotate runs (unless another condition like `size` has been set).

If you want to use a time interval other than the defined ones, you need to use cron to create a separate configuration file. For example, if you want to rotate a particular log file hourly, you could create a file in `/etc/cron.hourly` (you might need to create that directory too) that would contain a line like the following:

```
/usr/sbin/logrotate /etc/logrotate.hourly.conf
```

Then you would put the configuration for that hourly run of logrotate (the log file location, whether or not to compress old files, and so on) into `/etc/logrotate.hourly.conf`.

Size

You can use the `size` command to specify a file size for logrotate to check when determining whether to perform a rotation. The format of the command tells logrotate what units you're using to specify the size:

```
size 100k
size 100M
size 100G
```

The first example would rotate the log if it gets larger than 100 kilobytes, and the second if it's larger than 100 megabytes, and the third if it's over

100 gigabytes. I don't recommend using a limit of 100G, mind you, the example just got a little out of hand there.

The size command takes priority over and replaces a rotation interval if both are set.

Compression

If you want archived log files to be compressed (in gzip format), you can include the following command, usually in `/etc/logrotate.conf`:

```
compress
```

Compression is normally a good idea, because log files are usually all text and text compresses well. If, however, you have some archived logs that you don't want to compress, but you still want compression to be on by default, you can include the following command in an application-specific configuration:

```
nocompress
```

Another command of note in regard to compression is as follows:

```
delaycompress
```

This command is useful if you want to compress the archived logs, but want to delay the compression. When `delaycompress` is active, an archived log is compressed the next time that the log is rotated. This can be important when you have a program that might still write to its old log file for a time after a fresh one is rotated in. Note that `delaycompress` works only if you

have `compress` in your configuration.

An example of a good time to use `delaycompress` would be when logrotate is told to restart Apache with the "graceful" or "reload" directive. Because old Apache processes do not end until their connections are finished, they could potentially try to log more items to the old file for some time after the restart. Delaying the compression ensures that you won't lose those extra log entries when the logs are rotated.

Postrotate

Logrotate runs the `postrotate` script each time it rotates a log specified in a configuration block. You usually want to use this script to restart an application after the log rotation so that the app can switch to a new log.

```
postrotate
    /usr/sbin/apachectl restart > /dev/null
endscript
```

`>/dev/null` tells logrotate to pipe the command's output to nowhere. In this case, you don't need to view the output if the application restarted correctly.

The `postrotate` command tells logrotate that the script to run, starts on the next line, and the `endscript` command says that the script is done.

Sharedscripts

Normally logrotate runs the `postrotate` script every time it rotates a log. This is also true for multiple logs that use the same configuration block. For example, a web server configuration block that refers to both the access log

and the error log will, if it rotates both, run the `postrotate` script twice (once for each file rotated). If both files are rotated, the web server is restarted twice.

To keep logrotate from running that script for every log, you can include the following command:

```
sharedscripts
```

This command tells logrotate to check all the logs for that configuration block before running the `postrotate` script. If one or both of the logs is rotated, the `postrotate` script runs only once. If none of the logs is rotated, the `postrotate` script doesn't run.

Where to go next

This article provides an overview of what logrotate does and what kind of configuration options are available to you. You should now be able to explore the existing configurations and adapt them to your needs. To learn how to create an example configuration (to rotate the logs for custom virtual hosts), see Sample logrotate configurations and troubleshooting.