11.1.8 logrotate

Click one of the buttons to take you to that part of the video.

logrotate 0:00-0:45

Log files become large very large. On a system that's used every day, they become huge in just a few months.

In this video, we're going to manage log files using the logrotate utility.

The logroate utility is included with most Linux distributions by default.

This utility helps you manage your log files by automatically rotating, compressing, removing, and mailing them.

In other words, logrotate looks at the log files and says, "This log file is getting really big. We need to back it up start a new file. And all these backup log files from last year should probably be deleted, since we don't need them anymore."

Because managing logs is such an important task, you should configure the cron daemon to run logrotate automatically every day.

logrotate Configuration 0:46-1:09

You can customize the way logrotate handles files using the /etc/logrotate.conf file.

This file contains global parameters that the logrotate program uses to determine how and when to rotate files.

But you can override the global settings for a specific log file by creating your own custom logrotate configuration file in the /etc/logrotate.d directory.

logrotate Syntax 1:10-1:54

To create a configuration file for the logrotate.d directory, all you need to do is create a text file named after the service the log will monitor.

For example, to customize how logrotate handles the log files for the Apache web server running on our system, we can create a file in the logrotate.d directory named apache2.

The first thing we need to do is specify the name of the log file we want to manage. For example, we could specify the path '/var/log/apache2' and then the name of the log file we want to rotate, access_log.

On this same line, we need to add an open curly bracket ({),to indicate that we're ready to list directives, which determine how the file is managed. And then, after we have added all of our directives, we close the list using the opposite curly bracket.

logrotate Directives - compress 1:55-2:22

So, what are the directives you can use with logrotate?

The first directive is compress, which compresses the access_log file.

Next, we have maxage.

This directive specifies that files over a certain age are removed from the system.

For example, if we wanted to delete versions of this log file that are more than a year old, we would add 365. Now, if a backup version of the log file is over 365 days old, logrotate will delete the file.

Use the dateext Directive 2:23-2:37

The next directive we can use is dateext.

This directive causes old versions of our log file to be saved with the saved date as an extension. This way, when you look at the old versions of the file, it's very easy to tell what date range that log file came from.

Use the rotate Directive 2:38-3:32

The next directive is rotate.

This directive specifies that when a new log file is created where one already exists, the original file is rotated.

This means that the original file's name is changed to indicate it's older than the first, and the new log file is given the original name.

For example, since we're working with the access_log file, when it's rotated, the access_log will become access_log.1, and the new file will become access_log. This process continues as often as needed.

The rotate directive specifies how many new or rotated files are created before the oldest file is deleted.

In other words, since I've specified ‘rotate 40', our access_log file will be rotated 40 times. And then, when the next log file is created, the fortieth file will be deleted, and the other files will rotate down, so we still have a maximin of 40 files.

This way, we maintain a minimal set of backup rotated logs on the system.

Use the size Directive 3:33-4:02

We also have a directive named size.

This directive is used to determine how large a log file can become before it's rotated.

To indicate the size, we add an equals sign (=), a plus sign (+), and then the maximum size of our file. In our example, that's 1024. And we'll add the letter k, for kilobytes, to indicate a metric. 1024 kilobytes equals one megabyte.

With this directive set to these parameters, as soon as our log file gets to be one megabyte in size, it's going to be rotated.

Use the notifempty Directive 4:03-4:21

Another directive we can use is notifempty.

This directive tells the system not to rotate the log file if it's empty.

For example, if a service just hasn't written anything to the log file because nothing's been going on, we're not going to worry about rotating the file. This protects you from having a whole bunch of rotated files that are all empty.

Use the missingok Directive 4:22-4:29

Our next directive is missingok.

This command tells the syslog daemon that it's okay not to write an error message if the log file is missing.

Use the create Directive 4:30-5:21

There's also a directive called create, which creates a new empty log file after rotation with the specified permissions, owner, and group.

For example, if we want the file owner to have read write permissions to the file, our first number will be 6. If we want the group owning the file to have just read access, our next number will be 4. And then, by adding the last 4, we indicate that all other authenticated users on our system will also have read only rights.

As another example, if we only want the owner to have rights, we'll use the number of 600. When you're granting permissions, you must always specify three numbers: one for the owner, one for the group, and one for all other users.

Next, we need to specify who will own this file. More than likely, you're just going to want to use root.

Last, we'll specify who the owning group will be. In our example, we're going to use the root group again.

Use the postrotate Directive 5:22-5:58

The last directive we're going to talk about here is postrotate.

postrotate is used to tell the logrotate service what to do after it's rotated the logs for this particular service.

For example, in Apache2, we may need to reload the service itself after the logs are rotated.

To do that, we would put 'postrotate', space, and then the name of the init or systemd script that we're going to use to restart Apache2, '/etc/init.d/apache2', followed by 'reload'.

This command causes the Apache2 service to reload its configuration file and apply any new changes without actually shutting down and restarting the Apache2 service itself.

Summary 5:59-6:16

That's it for this lesson.

In this lesson, we discussed how to manage Linux log files using the logrotate utility, and we saw how logrotate uses directives in the /etc/logrotate.conf file to determine what actions to take on all log files unless those settings are overwritten by directives found in the /etc/logrotate.d directory.