8.12.1 File Archives

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

File Archives 0:00-1:03

In this lesson we're going to review how you can use the tar command to create file archives.

The tar utility has been around for a very long time, and it's a very commonly used Linux backup tool. The acronym tar stands for tape archiver, because in the old days it was used--and still can be used for that matter--to back up data to a tape drive.

Today, what we usually do is use the tar utility to take a list of specified files and copy them into a single archive file called a .tar file. That tar file can then be compressed using a compression utility on your Linux system and the resulting file is a compressed tar archive that we call a tarball. The tar utility can be used to send backup jobs to a variety of different backup media.

You can use it to back up data to a tape drive still, if you have one, and you can also use it to back up data to, say, your removable hard disk drive. With that in mind, let's take a look at how you use tar to create a new archive file.

Create New Archive 1:04-4:16

The syntax for using tar to create a new archive file is shown here. We first run tar, and then we use the -c option, which tells tar to create a new archive file.

The -v option can be used, it's optional. It tells tar to work in verbose mode. What this will do is display each file that gets added to the archive on the screen. You don't have to use -v, but I like to use it because I like to see what all is being added to the archive. If you don't care, you could leave the -v option off.

Then we use the -f option to specify the name of the new archive file to be created. Then, of course, we specify the directory or file that we want to add to the archive.

For example, let's suppose we wanted to back up the home directory on my Linux system, and we want to save it in a directory named homeback.tar. And we want to put that archive file on an external USB hard drive that's mounted in the /media/usb directory.

To do this, we would run tar and then I'd use the -cvf options to create a new archive, verbosely, and we use f to specify the name of the backup file we want to create. It's going to be in /media/usb, which is where our USB drive is mounted, and the name of the file is homeback.tar, and we want to back up the /home directory.

Notice when we run this, that there's a message displayed right here. It states removing leading forward slashes (/) from member names.

This is important because when this tar archive is created, the absolute paths to each file and directory will be converted to relative paths by default. Essentially, what that's going to do is get rid of this character right here from the paths. It'll get rid of the leading forward slash (/).

There may be times when you want to do that, and maybe there's times when you don't. By removing the leading forward slash (/), it allows me to restore these files into any directory that I want to. For example, I might want to do a test restore of these files into the /tmp directory on my system.

Because we have removed the leading forward slashes from the filenames, it's not a problem. It'll create a directory named home within the /tmp directory and then create all the subdirectories and all these files within /tmp/home. Not a problem.

There are options you can use with tar that will disable this functionality, and leave the absolute path in the archive file, in which case these files and directories will be restored to their original paths. If the original files that were backed up are still there, they'll be overwritten by the new files. You have to decide which way you want to go.

The default is to remove the leading forward slash (/). And I like that because it gives me a lot of flexibility when I'm doing a restore. If I still have an old tape drive kicking around instead of a USB drive, I can still write my archive to the tape drive instead of to a file in the file system. You do this by replacing the filename parameter that we just saw with the device filename for your tape drive.

Extract an Archive 4:17-6:25

On most distributions, the first SCSI tape drive in the system is referenced using the /dev/st0 device file. Therefore, if I wanted to back up my /home directory to a tape drive, I'd run tar -cvf /dev/st0 and then I specify the directory that I want to back up, /home.

Once I've created an archive, I can then extract files from it. To do this, I use the tar command again. But this time, notice that instead of using a -c parameter, I use an -x parameter. -x means extract. Let's suppose I want to extract all the files from the backup file we just created /homeback.tar.

To do this, I would run tar -xvf and then I'd specify the name of the backup file /media/usb/homeback.tar. Then all of the files within the archive are restored in the current directory.

Notice, as we talked about earlier, that the leading forward slash (/) has already been removed from each of these files, and I'm extracting the archive file in the /tmp directory. So when I'm done, I should see a directory in /tmp named home. And within home I will see all of these subdirectories and all of these files within the subdirectories.

What I've shown you is just a very basic example of how to use tar. There are many different options you can use with tar to customize the way it's going to work. To see all of the different options, you need to look at the tar man page.

Some of the more useful options, in my opinion, are show here. First of all, as we saw, you can use the -c option to create a new archive file.

Another useful option is the -d option. When you use -d, it's going to compare the files inside of the archive file against the files that exist in the file system. It's going to identify any differences that it finds. Basically, you're looking for files that have changed--that don't match up between the file system and the archive file.

Compress File Archives 6:26-10:43

Another useful option is the -J option. This option will do one of two things, depending on whether you're creating an archive or whether you're extracting from an archive. If you're creating a new archive, the -J option will compress that new tar archive file by running it through the xz utility.

If, on the other hand, you're extracting files from the archive and that file has already been compressed with the xz utility, you need to specify -J to tell tar to first decompress that tar archive using xz, and then go ahead and extract the files out of it. -j does a similar thing, except that it uses bzip compression instead of xz compression.

You can also use the -P option to tell tar to not strip that leading forward slash (/) from the file name. This essentially preserves each file's and directory's full file path. As we said earlier there may be times when that's useful. It has to be used with caution, however.

You can use the -r option to add files to the end of an existing tar archive. You can use the -t option to list the contents that are inside of an archive file.

You can use the -u option to append files to an existing tape archive, only if they have a modification date that is newer than the existing files in the archive. This is a great option to use if you're backing up data in the file system and you want to make sure that you back up only files that have changed, instead of backing up all the files in the file system.

The -x option, as we saw earlier, extracts files from the archive. -z is similar to the two -j options.

And then finally we have the -X option, which tells tar to not include certain files when it's adding files to the archive. What you do is create a text file that lists all the files that you do not want added to the archive, and then you use the -X option followed by the name of the exclusion file. Then tar will read it and say, "Okay, I'm not going to add these certain files to the archive that I'm creating."

With this in mind, let's revisit the example we worked with earlier where we created a simple archive file. We can shrink down the size of the archive file using compression. For example, we could run the archive through the xz compression utility by adding the -J option to the command: tar cJvf, and then the same parameters we used last time.

Here's the name of the backup file, and here's the name of the directory to be backed up. Notice that we changed the name of the backup file slightly. Notice that we added a dot xz (.xz) extension to the end of the archive file name.

Do you have to do this? No, but it's a good practice. Why? Because it reminds us what compression utility was used to originally compress this archive file when it was created.

Remember, you can use three different compression utilities with tar. You can use bzip2, gzip, or xz. By putting the .xz at the end of the file, I know what compression utility was used to create this archive file, because I'm going to have to use the appropriate option when I go to extract files from it.

Other extensions are used to identify different compression utilities. If you see a .xz or .lzma extension on a tar archive file, you know it was compressed with xz. If you see .bz2 as an extension on the archive filename, you know it was compressed with bzip. And if you see a .gz, you know that that archive file was compressed with gzip.

As I said a minute ago, this is important. Because if you want to pull files out of this archive, you have to tell tar which compression scheme was used to originally create the archive file.

In this case, we have a .xz extension, so I know that it used xz to compress the archive file. So when I do an extract from this file with the -x option, I need to also include the -J option to tell tar to first decompress the archive file with the xz utility; and once that's done, go ahead and pull all the files out of it.

Summary 10:43-10:54

That's it for this lesson. In this lesson, we discussed how to manage file archives on Linux using the tar command. We looked at using tar to create an archive, to extract files from an archive, and also to compress archive files.

Copyright © 2022 TestOut Corporation All rights reserved.