8.12.2 Using tar

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

Use Tar 0:00-0:17

In this demonstration, we're going to look at using tar to manage archive files. An archive file is a single file in your file system that contains other files. And in order to save space, you can actually take that archive file and compress it down.

'tar' 0:18-0:42

In this demonstration, we're going to look at how to create a new archive file, and then we'll talk about how to extract files from an archive file.

Before we do that, let's take a look at the 'man' page for 'tar'. You need to understand that tar is a very extensive tool. It can do a lot of different things. We only have time and space in this demonstration to look at its basic functionality, but you should open up the man page for tar and become very familiar with all of its options.

'tar' Options 0:43-2:22

A couple of key ones that you need to be familiar with are listed right here. First of all, -c and -x. If you need to create a new archive file and then add files into it, you use the -c option. If, on the other hand, you already have an archive file and you want to pull files out of it, then you use the extract option.

Another useful option is -r. You use -r in situations where you already have an existing archive file and you want to just add a couple of files to it. Rather than recreating it completely, you can use -r to just drop those files in at the end of the archive.

A similar option is -u. -u is used if we have newer versions of files that already exist in archive and we want the newer versions to replace the older versions in the archive.

Basically, this tells tar to take a look at the timestamps on the files in the archive and the timestamps on the files that are being added to the archive, and if the ones being added are newer, go ahead and overwrite them. If they're not, it'll ignore them.

Another useful option with tar is -t, which we can use to see just what files exist within an archive. I like to use the -v option as well. Understand that if you don't use the -v option with the tar command, you have no idea what's going on.

And if you're managing a very large archive--either creating or extracting files from it--you won't know what's going on. And if that archive file is very big, you might assume that something is hung, when in fact it's just processing. I like to use -v so I can see exactly what's going on. It'll list each file as it's being added or removed from the archive.

Another option you will almost always use with a tar command is -f. We use -f to specify the name of the archive file that we're manipulating, whether we're creating, updating, or extracting files from an archive.

New Compressed Archive 2:23-2:52

Remember earlier I said that you can compress an archive file in order to save space on your hard disk drives? You have three different options you can choose from with the tar command.

You can use the -J option to compress the archive file with the bzip2 utility. You can use the -j option to use the xz compression utility. Or you can use the -z option down here to compress the archive using the gzip utility. Which one is better? They all work about the same, to be honest.

Create a New Archive File 2:53-3:46

With that in mind, let's get out of the man page and let's create a new compressed archive. On this system, we have a shared folder named shared. It resides in the '/mnt' directory.

These files are used by various users on the system. As the system administrator, I want to create a backup periodically of this directory so we don't lose any data from it.

Before we create the archive file, let's switch to the root user account. Let's go ahead and use the 'tar' command to create a new archive file. Because we're creating a new one, I need to use '-c' for create, and I'm going to throw the 'v' option in there for verbose, because I want to see what's happening as that archive file is being created.

With this small archive file, it won't be that big a deal, but if you're dealing with a huge file, you'll definitely want to use the v option so you can make sure that it's actually running and not hung.

Compress the Archive File 3:47-4:16

Let's also compress the archive after we create it. Remember, we can use either bzip2, gzip, or xz in order to compress the archive file. I'm going to go ahead and specify 'J' to use xz to compress the archive. Then I have to use the 'f' option to specify the name of the archive file that I am wanting to create.

Let's go ahead and just create it in my root user's /home directory, and let's name it 'backup.tar'. And then I'm going to add an extra extension on--'.xz'.

Extra Extensions 4:17-4:53

Is that required? No, but it's a good practice because it tells anybody who goes to extract this archive file what compression algorithm was used in order to compress it. By specifying an extension of .xz, I know that it was compressed with the xz utility. Therefore, anyone who goes to extract it knows which options to use with the tar command.

If I had compressed this with bzip2 instead of xz, then I would have used an extension of .bz2. Again, that indicates that bzip2 was used to compress the archive. Or if I had compressed it with gzip instead of xz, I would have used an extension of .gz.

Archive Content 4:54-5:16

So here's my backup file that I'm going to create--the tar archive that I'm going to create. I have to specify what I want to put in that archive. Let's just put the contents of '/mnt/shared'. Hit Enter, and here you can see all of the files that were added to the archive.

We're going to do the 'ls' command. We can see the new file is being created in my /home directory here.

Files from Backup 5:17-7:00

Let's suppose that an end user accidentally deleted all of the files in the /shared directory. Everybody is mad at that person, and we need to restore all of those files from backup. We could pull those files, again, out of the archive that we just created.

I'm actually going to copy the archive file to my /tmp directory. That way we don't end up with a whole bunch of extra files in my /home directory here. And let's switch to '/tmp', do an 'ls' command, and there's the backup file right there.

Let's go ahead and extract it here into my /tmp directory. To do this, I'd run the 'tar' command again, but this time instead of -c, I need to specify '-x', because I'm extracting from it rather than creating it now.

I'm going to use the 'v' option again to verbosely see what's going on as each file is extracted. I do need to also specify the 'J' to decompress the compressed archive file using the xz utility, and then I use 'f' to specify the filename "backup.tar.xz."

I'm not going to specify an extraction location in this command. Therefore, it's going to extract right here in /tmp. But if I wanted to, I could actually specify a path where I wanted the files to be extracted here. Let's not worry about that today; let's just go ahead and extract them in the local directory. Hit Enter. The files are extracted.

If I do an 'ls' command, you'll notice that there's a new directory created. Notice that, by default, the tar utility preserved the original directory structure of the files that were added, so we have an /mnt directory.

By switching to '/mnt', we should see a shared directory. By switching to 'shared', I now see all of the files that were extracted from the archive, and then I could copy whichever ones are necessary over to the original directory where my end user accidentally deleted the original files.

Summary 7:01-7:10

That's it for this demonstration. In this demo, we talked about using tar to manage archive files. We first talked about how to create a new archive file, and then we looked at how to extract files from an archive file.