# Hadoop Archive Guide

## Table of contents

## 1. What are Hadoop archives?

Hadoop archives are special format archives. The main use case of using archives is to reduce the namespace of the NameNode. Hadoop Archives collapses a set of files into a smaller number of files and provides a very efficient and easy interface to access these collapsed files. A Hadoop archive maps to a HDFS directory. A Hadoop archive always has a *.har extension.

## 2. How to create an archive?

```
Usage: hadoop archive -archiveName name -p <parent> <src>*
<dest>
```

-archiveName is the name of the archive you would like to create. An example would be foo.har. The name should have a *.har extension. The parent argument is to specify the relative path to which the files should be archived to. Example would be :

```
-p /foo/bar a/b/c e/f/g
```

Here /foo/bar is the parent path and a/b/c, e/f/g are relative paths to parent. Note that this is a Map/Reduce job that creates the archives. You would need a map reduce cluster to run this. For a detailed example the later sections.

If you just want to archive a single directory /foo/bar then you can just use

```
hadoop archive -archiveName zoo.har -p /foo/bar /outputdir
```

## 3. How to look up files in archives?

The archive exposes itself as a file system layer. So all the fs shell commands in the archives work but with a different URI. Also, note that archives are immutable. So, rename's, deletes and creates return an error. URI for Hadoop Archives is

```
har://scheme-hostname:port/archivepath/fileinarchive
```

If no scheme is provided it assumes the underlying filesystem. In that case the URI would look like

```
har:///archivepath/fileinarchive
```

## 4. Example on creating and looking up archives

```
hadoop archive -archiveName foo.har -p /user/hadoop dir1 dir2
```

`/user/zoo`

The above example is creating an archive using /user/hadoop as the relative archive directory. The directories /user/hadoop/dir1 and /user/hadoop/dir2 will be archived in the following file system directory -- /user/zoo/foo.har. Archiving does not delete the input files. If you want to delete the input files after creating the archives (to reduce namespace), you will have to do it on your own.

### 4.1. Looking up files and understanding the -p option

Looking up files in hadoop archives is as easy as doing an ls on the filesystem. After you have archived the directories /user/hadoop/dir1 and /user/hadoop/dir2 as in the exmaple above, to see all the files in the archives you can just run:

```
hadoop dfs -lsr har:///user/zoo/foo.har/
```

To understand the significance of the -p argument, lets go through the above example again. If you just do an ls (not lsr) on the hadoop archive using

```
hadoop dfs -ls har:///user/zoo/foo.har
```

The output should be:

```
har:///user/zoo/foo.har/dir1
har:///user/zoo/foo.har/dir2
```

As you can recall the archives were created with the following command

```
hadoop archive -archiveName foo.har -p /user/hadoop dir1 dir2
/user/zoo
```

If we were to change the command to:

```
hadoop archive -archiveName foo.har -p /user/ hadoop/dir1
hadoop/dir2 /user/zoo
```

then a ls on the hadoop archive using

```
hadoop dfs -ls har:///user/zoo/foo.har
```

would give you

```
har:///user/zoo/foo.har/hadoop/dir1
har:///user/zoo/foo.har/hadoop/dir2
```

Notice that the archived files have been archived relative to /user/ rather than /user/hadoop.

## 5. Using Hadoop Archive with Map Reduce

Using Hadoop Archive in Map Reduce is as easy as specifying a different input filesystem than the default file system. If you have a hadoop archive stored in HDFS in /user/zoo/foo.har then for using this archive for Map Reduce input, all you need to specify the input directory as har:///user/zoo/foo.har. Since Hadoop Archives is exposed as a file system Map Reduce will be able to use all the logical input files in Hadoop Archives as input.

## 6. File Replication and Permissions of Hadoop Archive

Hadoop Archive currently does not store the file information metadata that the files had before they were archived. The file permissions of Hadoop Archive created is the default permissions that a user creates file with. The file replication of data files in Hadoop Archive is set to 3 and the metadata information files have a replication factor of 5. You can increase this by increasing the replication factor of files under the har file directory. On restoration of hadoop archive files using something like:

```
distcp har:///path_to_har dest_path
```

the restored files will not have the permissions/replication of the original files that were archived.

## 7. Creating different block size and part size Hadoop Archive

You can create different hadoop block size and part size using the following options:

```
bin/hadoop archive -Dhar.block.size=512
-Dhar.partfile.size=1024 -archiveName ...
```

The above example allows you to set a block size of 512 bytes for part files and part file size of 1K. These numbers are only as examples. Using such low number for block size and part file size is not advisable at all!

## 8. Limitations of Hadoop Archive

Currently Hadoop archive do not support input paths with spaces in it. It throws out an exception in such a case. You can create archives with names in which space can be replaced by a valid character. Below is an example:

```
bin/hadoop archive -Dhar.space.replacement.enable=true
-Dhar.space.replacement="_" -archiveName ......
```

The above example replaces space with "_" in the archived file names.

## 9. Internals of Hadoop Archive

A Hadoop Archive directory contains metadata (in the form of _index and _masterindex) and data (part-*) files. The _index file contains the name of the files that are part of the archive and the location within the part files. The _masterindex file stores offsets into the _index file to make it easier to seek into the _index file for faster lookups.