Rocky Enterprise Linux 9.2 Manual Pages on command 'ar.1'

$ man ar.1

AR(1) GNU Development Tools AR(1)

NAME

ar - create, modify, and extract from archives

SYNOPSIS

ar [-X32_64] [-]p[mod] [--plugin name] [--target bfdname] [--output dirname] [relpos] [count] archive [member...]

DESCRIPTION

The GNU ar program creates, modifies, and extracts from archives. An archive is a single file holding a collection of other files in a structure that makes it possible to retrieve the original individual files (called members of the archive).

The original files' contents, mode (permissions), timestamp, owner, and group are preserved in the archive, and can be restored on extraction.

GNU ar can maintain archives whose members have names of any length; however, depending on how ar is configured on your system, a limit on member-name length may be imposed for compatibility with archive formats maintained with other tools. If it exists, the limit is often 15 characters (typical of formats related to a.out) or 16 characters

(typical of formats related to coff).

ar is considered a binary utility because archives of this sort are most often used as libraries holding commonly needed subroutines.

ar creates an index to the symbols defined in relocatable object modules in the archive when you specify the modifier s. Once created, this index is updated in the archive whenever ar makes a change to its contents (save for the q update operation). An archive with such an index speeds up linking to the library, and allows routines in the library to call each other without regard to their placement in the archive.

You may use nm -s or nm --print-armap to list this index table. If an archive lacks the table, another form of ar called ranlib can be used to add just the table.

GNU ar can optionally create a thin archive, which contains a symbol index and references to the original copies of the member files of the archive. This is useful for building libraries for use within a local build tree, where the relocatable objects are expected to remain available, and copying the contents of each object would only waste time and space.

An archive can either be thin or it can be normal. It cannot be both at the same time. Once an archive is created its format cannot be changed without first deleting it and then creating a new archive in its place.

Thin archives are also flattened, so that adding one thin archive to another thin archive does not nest it, as would happen with a normal archive. Instead the elements of the first archive are added individually to the second archive.

The paths to the elements of the archive are stored relative to the archive itself.

GNU ar is designed to be compatible with two different facilities. You can control its activity using command-line options, like the different varieties of ar on Unix systems; or, if you specify the single command-line option -M, you can control it with a script supplied via standard

input, like the MRI "librarian" program.

OPTIONS

GNU ar allows you to mix the operation code p and modifier flags mod in any order, within the first command-line argument.

If you wish, you may begin the first command-line argument with a dash.

The p keyletter specifies what operation to execute; it may be any of the following, but you must specify only one of them:

d Delete modules from the archive. Specify the names of modules to be deleted as member...; the archive is untouched if you specify no files to delete.

If you specify the v modifier, ar lists each module as it is deleted.

m Use this operation to move members in an archive.

The ordering of members in an archive can make a difference in how programs are linked using the library, if a symbol is defined in more than one member.

If no modifiers are used with "m", any members you name in the member arguments are moved to the end of the archive; you can use the a, b, or i modifiers to move them to a specified place instead.

p Print the specified members of the archive, to the standard output file. If the v modifier is specified, show the member name before copying its contents to standard output.

If you specify no member arguments, all the files in the archive are printed.

q Quick append; Historically, add the files member... to the end of archive, without checking for replacement.

The modifiers a, b, and i do not affect this operation; new members are always placed at the end of the archive.

The modifier v makes ar list each file as it is appended.

Since the point of this operation is speed, implementations of ar have the option of not updating the archive's symbol table if one exists. Too many different systems however assume that symbol tables are always up-to-date, so GNU ar will rebuild the table even

with a quick append.

Note - GNU ar treats the command qs as a synonym for r - replacing already existing files in the archive and appending new ones at the end.

r Insert the files member... into archive (with replacement). This operation differs from q in that any previously existing members are deleted if their names match those being added.

If one of the files named in member... does not exist, ar displays an error message, and leaves undisturbed any existing members of the archive matching that name.

By default, new members are added at the end of the file; but you may use one of the modifiers a, b, or i to request placement relative to some existing member.

The modifier v used with this operation elicits a line of output for each file inserted, along with one of the letters a or r to indicate whether the file was appended (no old member deleted) or replaced.

s Add an index to the archive, or update it if it already exists.

Note this command is an exception to the rule that there can only be one command letter, as it is possible to use it as either a command or a modifier. In either case it does the same thing.

t Display a table listing the contents of archive, or those of the files listed in member... that are present in the archive.

Normally only the member name is shown, but if the modifier O is specified, then the corresponding offset of the member is also displayed. Finally, in order to see the modes (permissions), timestamp, owner, group, and size the v modifier should be included.

If you do not specify a member, all files in the archive are listed.

If there is more than one file with the same name (say, fie) in an archive (say b.a), ar t b.a fie lists only the first instance; to see them all, you must ask for a complete listing---in our example,

ar t b.a.

x Extract members (named member) from the archive. You can use the v modifier with this operation, to request that ar list each name as it extracts it.

If you do not specify a member, all files in the archive are extracted.

Files cannot be extracted from a thin archive, and there are restrictions on extracting from archives created with P: The paths must not be absolute, may not contain "..", and any subdirectories in the paths must exist. If it is desired to avoid these restrictions then used the --output option to specify an output directory.

A number of modifiers (mod) may immediately follow the p keyletter, to specify variations on an operation's behavior:

a Add new files after an existing member of the archive. If you use the modifier a, the name of an existing archive member must be present as the relpos argument, before the archive specification.

b Add new files before an existing member of the archive. If you use the modifier b, the name of an existing archive member must be present as the relpos argument, before the archive specification. (same as i).

c Create the archive. The specified archive is always created if it did not exist, when you request an update. But a warning is issued unless you specify in advance that you expect to create it, by using this modifier.

D Operate in deterministic mode. When adding files and the archive index use zero for UIDs, GIDs, timestamps, and use consistent file modes for all files. When this option is used, if ar is used with identical options and identical input files, multiple runs will create identical output files regardless of the input files' owners, groups, file modes, or modification times.

If binutils was configured with --enable-deterministic-archives, then this mode is on by default. It can be disabled with the U

modifier, below.

f Truncate names in the archive. GNU ar will normally permit file names of any length. This will cause it to create archives which are not compatible with the native ar program on some systems. If this is a concern, the f modifier may be used to truncate file names when putting them in the archive.

i Insert new files before an existing member of the archive. If you use the modifier i, the name of an existing archive member must be present as the relpos argument, before the archive specification. (same as b).

l This modifier is accepted but not used.

N Uses the count parameter. This is used if there are multiple entries in the archive with the same name. Extract or delete instance count of the given name from the archive.

o Preserve the original dates of members when extracting them. If you do not specify this modifier, files extracted from the archive are stamped with the time of extraction.

O Display member offsets inside the archive. Use together with the t option.

P Use the full path name when matching or storing names in the archive. Archives created with full path names are not POSIX compliant, and thus may not work with tools other than up to date GNU tools. Modifying such archives with GNU ar without using P will remove the full path names unless the archive is a thin archive. Note that P may be useful when adding files to a thin archive since r without P ignores the path when choosing which element to replace. Thus

```
ar rcST archive.a subdir/file1 subdir/file2 file1
```

will result in the first "subdir/file1" being replaced with "file1" from the current directory. Adding P will prevent this replacement.

s Write an object-file index into the archive, or update an existing one, even if no other change is made to the archive. You may use

this modifier flag either with any operation, or alone. Running ar s on an archive is equivalent to running ranlib on it.

S Do not generate an archive symbol table. This can speed up building a large library in several steps. The resulting archive can not be used with the linker. In order to build a symbol table, you must omit the S modifier on the last execution of ar, or you must run ranlib on the archive.

T Make the specified archive a thin archive. If it already exists and is a regular archive, the existing members must be present in the same directory as archive.

u Normally, ar r... inserts all files listed into the archive. If you would like to insert only those of the files you list that are newer than existing members of the same names, use this modifier. The u modifier is allowed only for the operation r (replace). In particular, the combination qu is not allowed, since checking the timestamps would lose any speed advantage from the operation q.

U Do not operate in deterministic mode. This is the inverse of the D modifier, above: added files and the archive index will get their actual UID, GID, timestamp, and file mode values.
This is the default unless binutils was configured with --enable-deterministic-archives.

v This modifier requests the verbose version of an operation. Many operations display additional information, such as filenames processed, when the modifier v is appended.

V This modifier shows the version number of ar.

The ar program also supports some command-line options which are neither modifiers nor actions, but which do change its behaviour in specific ways:

--help

Displays the list of command-line options supported by ar and then exits.

--version

Displays the version information of ar and then exits.

-X32_64

ar ignores an initial option spelled -X32_64, for compatibility with AIX. The behaviour produced by this option is the default for GNU ar. ar does not support any of the other -X options; in particular, it does not support -X32 which is the default for AIX ar.

--plugin name

The optional command-line switch --plugin name causes ar to load the plugin called name which adds support for more file formats, including object files with link-time optimization information.

This option is only available if the toolchain has been built with plugin support enabled.

If --plugin is not provided, but plugin support has been enabled then ar iterates over the files in ${libdir}/bfd-plugins in alphabetic order and the first plugin that claims the object in question is used.

Please note that this plugin search directory is not the one used by ld's -plugin option. In order to make ar use the linker plugin it must be copied into the ${libdir}/bfd-plugins directory. For GCC based compilations the linker plugin is called liblto_plugin.so.0.0.0. For Clang based compilations it is called LLVMgold.so. The GCC plugin is always backwards compatible with earlier versions, so it is sufficient to just copy the newest one.

--target target

The optional command-line switch --target bfdname specifies that the archive members are in an object code format different from your system's default format. See

--output dirname

The --output option can be used to specify a path to a directory into which archive members should be extracted. If this option is not specified then the current directory will be used.

Note - although the presence of this option does imply a x extraction operation that option must still be included on the

command line.

@file

Read command-line options from file. The options read are inserted in place of the original @file option. If file does not exist, or cannot be read, then the option will be treated literally, and not removed.

Options in file are separated by whitespace. A whitespace character may be included in an option by surrounding the entire option in either single or double quotes. Any character (including a backslash) may be included by prefixing the character to be included with a backslash. The file may itself contain additional @file options; any such options will be processed recursively.

SEE ALSO

nm(1), ranlib(1), and the Info entries for binutils.

COPYRIGHT