NAME

cp - copy files

SYNOPSIS

cp [**-R** [**-H** | **-L** | **-P**]] [**-f** | **-i** | **-n**] [**-alpsvx**] *source_file target_file*
cp [**-R** [**-H** | **-L** | **-P**]] [**-f** | **-i** | **-n**] [**-alpsvx**] *source_file ... target_directory*
cp [**-f** | **-i** | **-n**] [**-alPpsvx**] *source_file target_file*
cp [**-f** | **-i** | **-n**] [**-alPpsvx**] *source_file ... target_directory*

DESCRIPTION

In the first synopsis form, the **cp** utility copies the contents of the *source_file* to the *target_file*. In the second synopsis form, the contents of each named *source_file* is copied to the destination *target_directory*. The names of the files themselves are not changed. If **cp** detects an attempt to copy a file to itself, the copy will fail.

The following options are available:

-H If the **-R** option is specified, symbolic links on the command line are followed. (Symbolic links encountered in the tree traversal are not followed.)

-L If the **-R** option is specified, all symbolic links are followed.

-P No symbolic links are followed. This is the default if the **-R** option is specified.

-R If *source_file* designates a directory, **cp** copies the directory and the entire subtree connected at that point. If the *source_file* ends in a /, the contents of the directory are copied rather than the directory itself. This option also causes symbolic links to be copied, rather than indirected through, and for **cp** to create special files rather than copying them as normal files. Created directories have the same mode as the corresponding source directory, unmodified by the process' umask.

Note that **cp** copies hard linked files as separate files. If you need to preserve hard links, consider using tar(1), cpio(1), or pax(1) instead.

-a Archive mode. Same as **-RpP**.

-f For each existing destination pathname, remove it and create a new file, without prompting for confirmation regardless of its permissions. (The **-f** option overrides any previous **-i** or **-n** options.)

-i Cause **cp** to write a prompt to the standard error output before copying a file that would overwrite an existing file. If the response from the standard input begins with the character 'y' or 'Y', the file

copy is attempted. (The **-i** option overrides any previous **-f** or **-n** options.)

-l Create hard links to regular files in a hierarchy instead of copying.

-n Do not overwrite an existing file. (The **-n** option overrides any previous **-f** or **-i** options.)

-p Cause **cp** to preserve the following attributes of each source file in the copy: modification time, access time, file flags, file mode, ACL, user ID, and group ID, as allowed by permissions.

If the user ID and group ID cannot be preserved, no error message is displayed and the exit value is not altered.

If the source file has its set-user-ID bit on and the user ID cannot be preserved, the set-user-ID bit is not preserved in the copy's permissions. If the source file has its set-group-ID bit on and the group ID cannot be preserved, the set-group-ID bit is not preserved in the copy's permissions. If the source file has both its set-user-ID and set-group-ID bits on, and either the user ID or group ID cannot be preserved, neither the set-user-ID nor set-group-ID bits are preserved in the copy's permissions.

-s Create symbolic links to regular files in a hierarchy instead of copying.

-v Cause **cp** to be verbose, showing files as they are copied.

-x File system mount points are not traversed.

For each destination file that already exists, its contents are overwritten if permissions allow. Its mode, user ID, and group ID are unchanged unless the **-p** option was specified.

In the second synopsis form, *target_directory* must exist unless there is only one named *source_file* which is a directory and the **-R** flag is specified.

If the destination file does not exist, the mode of the source file is used as modified by the file mode creation mask (**umask**, see csh(1)). If the source file has its set-user-ID bit on, that bit is removed unless both the source file and the destination file are owned by the same user. If the source file has its set-group-ID bit on, that bit is removed unless both the source file and the destination file are in the same group and the user is a member of that group. If both the set-user-ID and set-group-ID bits are set, all of the above conditions must be fulfilled or both bits are removed.

Appropriate permissions are required for file creation or overwriting.

Symbolic links are always followed unless the **-R** flag is set, in which case symbolic links are not followed, by default. The **-H** or **-L** flags (in conjunction with the **-R** flag) cause symbolic links to be followed as described above. The **-H**, **-L** and **-P** options are ignored unless the **-R** option is specified. In addition, these options override each other and the command's actions are determined by the last one specified.

If **cp** receives a SIGINFO (see the **status** argument for stty(1)) signal, the current input and output file and the percentage complete will be written to the standard output.

EXIT STATUS

The **cp** utility exits 0 on success, and >0 if an error occurs.

EXAMPLES

Make a copy of file *foo* named *bar*:

```
$ cp foo bar
```

Copy a group of files to the */tmp* directory:

```
$ cp *.txt /tmp
```

Copy the directory *junk* and all of its contents (including any subdirectories) to the */tmp* directory:

```
$ cp -R junk /tmp
```

COMPATIBILITY

Historic versions of the **cp** utility had a **-r** option. This implementation supports that option, however, its behavior is different from historical FreeBSD behavior. Use of this option is strongly discouraged as the behavior is implementation-dependent. In FreeBSD, **-r** is a synonym for **-RL** and works the same unless modified by other flags. Historical implementations of **-r** differ as they copy special files as normal files while recreating a hierarchy.

The **-l**, **-s**, **-v**, **-x** and **-n** options are non-standard and their use in scripts is not recommended.

SEE ALSO

mv(1), rcp(1), umask(2), fts(3), symlink(7)

STANDARDS

The **cp** command is expected to be IEEE Std 1003.2 ("POSIX.2") compatible.

HISTORY

A **cp** command appeared in Version 1 AT&T UNIX.