

Utilities

CISL provides the tools described on this page to support users' code development efforts and to facilitate good programming and file management practices. To identify the versions that are available on Cheyenne, use the **which** and **module av** commands.

See [Software for Cheyenne users](#) for more information.

Page contents

- [Compressing and archiving files](#)
 - [Editing files](#)
 - [Making executables](#)
 - [Version control](#)
-

Compressing and archiving files

Because files often contain large amounts of redundant data and empty space, you can reduce their size significantly with tools such as **gzip** and **bzip2**.

Bzip2 provides greater compression, but it is also slower. Gzip offers a good balance of compression rate and speed. Note that **gzip** and **bzip2** are able to compress *only individual files*. See the tools' man pages for details.

To both package and compress *a group of files*, use the **tar** command to create an archive file – also called a tar file or tarball – as shown in the following examples. Also use the **tar** command to *extract* files when needed.

Create an archive file with **tar**:

```
tar -cvf archive_name.tar file1 file2 file3
```

Extract files from the archive file:

```
tar -xvf archive_name.tar
```

Create a **bzip2**-compressed archive file:

```
tar cjf archive_name.tar.bz2 filename
```

Extract files from the archive:

```
tar xf archive_name.tar.bz2
```

Create a **gzip**-compressed archive file:

```
tar czf archive_name.tgz filename
```

Extract files from the gzipped archive:

```
tar xf archive_name.tgz
```

The "j" and "z" options that are included when *creating* files with bzip2 or gzip are not needed for *extracting* files; they are used automatically. (If they are included in a script you use, there is no harm in leaving them there.)

Editing files

Among the most widely used text editors are **Vim**, **Vi** and **Emacs**.

Several excellent tutorials are available for text editors, including the following:

- [GNU Emacs Manual – gnu.org](#)
- [The Emacs Tutorial](#) – Free Software Foundation
- [A Tutorial Introduction to GNU Emacs](#) – University of Chicago Library

Making executables

Using simple makefiles with the appropriate compiler is usually sufficient for making your own executables on Cheyenne.

If you are tasked with developing cross-platform software, however, consider using the **GNU Build System**, also known as [GNU Autotools](#). This is a suite of programming tools designed to assist in making source code packages portable to different UNIX-like systems. It consists of the GNU Utility programs `autoconf`, `automake`, and `libtool`. Man pages are available for each.

Generating the executable usually is a two-step process using the [GNU Build System](#):

1. Invoke the **configure** script, which creates a **makefile** with some settings that are appropriate to the machine.
2. Invoke **make**, which will use the makefile to compile the executable, often using a complex set of rules.

GNU make (`gmake`) is the GNU incarnation of **POSIX make**, and as such it is more user friendly and full-featured. Usually it is used as a tool for managing compilation and builds of software packages, but it can do more. It functions consistently across platforms. See the [GNU Make Manual](#).

Version control

Git (`git`) is the most popular distributed version control system in use and is available on Cheyenne.

Mercurial (`hg`) is another good distributed version control system. Its functionality is similar to that of Git.