

Compressing and Archiving

LinuXpert Systems

No.5, Dr. Ambedkar St, Rathinapuram,

Nanganallur, Chennai – 600 061.

Phone: 98841 65649, 90808 01382

E-mail: baskar@linuxpert.in

TABLE OF CONTENTS

1. tar 1

2. gzip 8

3. gunzip 15

4. bzip2 21

5. xz 27

6. zip 33

7. unzip 39

1. `tar` (Tape Archive)

The `tar` command is a standard Linux/Unix utility used to **combine multiple files and directories into a single archive file**. It can also **extract**, **list**, and **manage** archive files. Although originally designed for tape backups ("Tape ARchive"), it is now widely used for file packaging and backups.

> **Important:** By itself, `tar` only archives files. Compression is added using options like `-z` (gzip), `-j` (bzip2), or `-J` (xz).

Syntax

```
tar [OPTIONS] ARCHIVE_FILE FILES...
```

or

```
tar [OPTIONS] -f archive.tar
```

Common Archive Formats

Extension	Compression	Option
`.tar`	None	None
`.tar.gz` or `.tgz`	gzip	`-z`
`.tar.bz2`	bzip2	`-j`
`.tar.xz`	xz	`-J`

Frequently Used Options

Option	Meaning
`-c`	Create a new archive
`-x`	Extract an archive
`-t`	List archive contents
`-f`	Specify archive filename
`-v`	Verbose (show files being processed)
`-z`	Compress/decompress using gzip
`-j`	Use bzip2 compression
`-J`	Use xz compression
`-C`	Change to a directory before extracting
`-r`	Append files to an archive
`-u`	Update files in archive
`--delete`	Delete files from archive
`--exclude`	Exclude matching files/directories
`--wildcards`	Enable wildcard matching

Common Examples

1. Create a tar archive

```
tar -cvf backup.tar documents/
```

Explanation

- `-c` → Create archive
- `-v` → Show progress
- `-f` → Archive filename

Creates:

`backup.tar`

---

2. Archive multiple files

```
tar -cvf project.tar file1.txt file2.txt folder/
```

---

3. Extract an archive

```
tar -xvf backup.tar
```

Extracts all files into the current directory.

---

4. Extract to another directory

```
tar -xvf backup.tar -C /home/user/restore
```

---

5. List archive contents

```
tar -tvf backup.tar
```

Output:

```
drwxr-xr-x user user     0 documents/
-rw-r--r-- user user  1240 documents/file1.txt
-rw-r--r-- user user  4532 documents/file2.txt
```

---

6. Create a gzip-compressed archive

```
tar -czvf backup.tar.gz documents/
```

Creates:

```
backup.tar.gz
```

---

7. Extract a gzip archive

```
tar -xzvf backup.tar.gz
```

---

8. Create a bzip2 archive

```
tar -cjvf backup.tar.bz2 documents/
```

---

9. Extract a bzip2 archive

```
tar -xjvf backup.tar.bz2
```

---

10. Create an xz archive

```
tar -cJvf backup.tar.xz documents/
```

---

11. Extract an xz archive

```
tar -xJvf backup.tar.xz
```

---

12. Archive an entire home directory

```
tar -czvf home_backup.tar.gz /home/bharath
```

---

13. Exclude files while archiving

```
tar --exclude="*.log" -czvf backup.tar.gz project/
```

This skips all `.log` files.

14. Append files to an existing archive

```
tar -rvf backup.tar newfile.txt
```

Note: This works only with uncompressed `.tar` archives.

---

15. Update changed files

```
tar -uvf backup.tar documents/
```

Only files newer than those already in the archive are added.

---

16. Delete a file from an archive

```
tar --delete -f backup.tar oldfile.txt
```

Works only with uncompressed archives.

---

17. Extract a single file

```
tar -xvf backup.tar documents/file1.txt
```

---

18. Extract files matching a pattern

```
tar -xvf backup.tar --wildcards "*.txt"
```

---

19. View archive without extracting

```
tar -tf backup.tar
```

---

20. Create an archive from multiple directories

```
tar -cvf archive.tar dir1 dir2 dir3
```

Common Compression Comparison

Format	Command	Speed	Compression
tar	`tar -cvf`	Fastest	None
tar.gz	`tar -czvf`	Fast	Good
tar.bz2	`tar -cjvf`	Slower	Better
tar.xz	`tar -cJvf`	Slowest	Best

Practical Backup Example

Create a compressed backup:

```
tar -czvf website_backup.tar.gz /var/www/html
```

Restore it:

```
tar -xzvf website_backup.tar.gz -C /
```

Advantages

- Combines many files into one archive.
- Preserves file permissions, ownership, and timestamps.
- Supports multiple compression formats.
- Ideal for backups and software distribution.
- Available on nearly all Unix/Linux systems.

Limitations

- A plain `.tar` archive is **not compressed**.
- Updating or deleting files is only supported for uncompressed archives.
- Random access to individual files is slower than with formats like ZIP.

Summary

Task	Command
Create archive	`tar -cvf archive.tar files/`
Extract archive	`tar -xvf archive.tar`
List contents	`tar -tvf archive.tar`
Create gzip archive	`tar -czvf archive.tar.gz files/`
Extract gzip archive	`tar -xzvf archive.tar.gz`
Create bzip2 archive	`tar -cjvf archive.tar.bz2 files/`
Create xz archive	`tar -cJvf archive.tar.xz files/`
Append files	`tar -rvf archive.tar file.txt`
Update archive	`tar -uvf archive.tar files/`
Exclude files	`tar --exclude="*.log" -czvf archive.tar.gz dir/`
Extract to another directory	`tar -xvf archive.tar -C /destination/`

`tar` is one of the most important Linux commands for packaging files, creating backups, and distributing software. It is commonly used together with compression utilities like `gzip`, `bzip2`, and `xz` to create efficient, portable archive files.

2. gzip

The `gzip` command is used to **compress files** in Linux and Unix systems. It reduces the file size to save disk space and make file transfers faster.

- **Full Form:** GNU Zip
- **Default Extension:** `.gz`
- **Compression Algorithm:** DEFLATE
- **Works on:** Individual files (not directories directly)

Note: To compress an entire directory, you usually combine `tar` and `gzip`:

```
tar -czf archive.tar.gz directory/
```

Syntax

```
gzip [OPTIONS] FILE
```

Basic Examples

Compress a file

```
gzip file.txt
```

Before:

```
file.txt
```

After:

```
file.txt.gz
```

The original `file.txt` is removed after successful compression.

Compress multiple files

```
gzip file1.txt file2.txt file3.txt
```

Output:

```
file1.txt.gz
file2.txt.gz
file3.txt.gz
```

Keep the original file

```
gzip -k file.txt
```

Result:

```
file.txt
file.txt.gz
```

Decompress a file

```
gzip -d file.txt.gz
```

or

```
gunzip file.txt.gz
```

Result:

```
file.txt
```

Display compressed file contents

```
zcat file.txt.gz
```

This prints the contents without extracting the file.

Common Options

Option	Description
`-d`	Decompress a file
`-k`	Keep the original file
`-c`	Write output to standard output
`-r`	Compress files recursively in directories
`-f`	Force compression/overwrite existing files
`-v`	Verbose output
`-l`	List compressed file information
`-1`	Fastest compression (less compression)
`-9`	Best compression (slower)
`-t`	Test integrity of compressed file
`-n`	Do not save original filename and timestamp
`-N`	Restore original filename and timestamp when decompressing

---

Examples of Useful Options

Verbose mode

```
gzip -v report.txt
```

Example output:

```
report.txt: 75.4% -- replaced with report.txt.gz
```

Best compression

```
gzip -9 backup.sql
```

Uses maximum compression.

Fast compression

```
gzip -1 backup.sql
```

Compresses more quickly but produces a larger file.

Force overwrite

```
gzip -f data.log
```

Overwrites an existing `.gz` file if necessary.

Test a compressed file

```
gzip -t archive.gz
```

Checks whether the compressed file is valid without extracting it.

View compression information

```
gzip -l archive.gz
```

Example output:

```
compressed  uncompressed  ratio  uncompressed_name
1200        5400          78%    archive
```

Output to another file

```
gzip -c report.txt > report.txt.gz
```

The original file remains unchanged because the compressed data is written to standard output and redirected into a new file.

Recursive Compression

```
gzip -r logs/
```

Compresses all eligible files inside the `logs` directory and its subdirectories.

Compression Levels

Option	Speed	Compression
`-1`	Fastest	Lowest
`-2` to `-8`	Balanced	Medium
`-9`	Slowest	Highest

Checking File Type

```
file backup.gz
```

Example output:

```
backup.gz: gzip compressed data
```

Using `gzip` with Pipes

Compress command output:

```
cat largefile.txt | gzip > largefile.txt.gz
```

Decompress without creating a file:

```
gzip -dc largefile.txt.gz
```

or

```
zcat largefile.txt.gz
```

Difference Between `gzip` and `tar`

`gzip`	`tar`
Compresses files	Archives files/directories
Cannot combine multiple files into one archive	Combines many files into one archive
Creates `.gz` files	Creates `.tar` files
Often used together with `tar`	Often combined with `gzip` as `.tar.gz`

Example:

```
tar -czf backup.tar.gz Documents/
```

- `-c` → Create archive
- `-z` → Compress with `gzip`
- `-f` → Specify output filename

Practical Examples

Compress a log file:

```
gzip access.log
```

Compress with maximum compression:

```
gzip -9 database.sql
```

Keep the original file:

```
gzip -k notes.txt
```

Extract a compressed file:

```
gunzip backup.gz
```

Display contents without extracting:

```
zcat backup.gz
```

Compress an entire project directory:

```
tar -czf project.tar.gz project/
```

Advantages

- Fast and efficient compression
- Saves disk space
- Reduces file transfer time
- Available on nearly all Linux distributions

Limitations

- Compresses individual files only (directories require `tar`)
- Compression ratio is generally lower than modern formats like `xz` or `zstd`
- Random access inside compressed files is limited

Summary

`gzip` is one of the most widely used Linux compression utilities. It compresses individual files into the `.gz` format using the DEFLATE algorithm, making files smaller for storage and transfer. It is commonly paired with `tar` to create compressed archives (`.tar.gz`) for backups and software packages.

3.gunzip

The `gunzip` command is used to **decompress files that were compressed using the `gzip` command**. It restores a `.gz` compressed file back to its original form.

Syntax

```
gunzip [OPTIONS] FILE.gz
```

or

```
gzip -d FILE.gz
```

Both commands perform the same function.

What `gunzip` Does

- Decompresses `.gz` files.
- Removes the `.gz` extension after decompression.
- Deletes the compressed file by default after successfully extracting it.
- Works only with files compressed using the Gzip format.

Examples

Example 1: Decompress a file

```
gunzip file.txt.gz
```

Before:

```
file.txt.gz
```

After:

```
file.txt
```

The compressed file (`file.txt.gz`) is removed.

Example 2: Decompress multiple files

```
gunzip *.gz
```

Decompresses all `.gz` files in the current directory.

Example:

Before:

```
log1.gz
log2.gz
backup.gz
```

After:

```
log1
log2
backup
```

Example 3: Keep the compressed file

Normally, `gunzip` deletes the `.gz` file.

To keep it, use:

```
gzip -dk file.txt.gz
```

Options:

- `-d` → Decompress
- `-k` → Keep original compressed file

Result:

```
file.txt.gz
file.txt
```

Example 4: Display decompressed content without extracting

```
gunzip -c file.txt.gz
```

or

```
zcat file.txt.gz
```

Output:

```
Hello World
Linux Commands
...
```

The file remains compressed.

Example 5: Test a compressed file

```
gunzip -t backup.gz
```

Checks whether the compressed file is valid.

Possible output:

```
(no output)
```

Exit status:

- `0` → File is valid
- Non-zero → File is corrupted

Example 6: Show compression information

```
gunzip -l backup.gz
```

Example output:

```
compressed  uncompressed  ratio  file
2048        8192          75.0%  backup
```

Shows:

- Compressed size
- Original size
- Compression ratio
- Original filename

Common Options

Option	Description
`-c`	Write decompressed data to standard output
`-d`	Decompress (same as `gunzip`)
`-f`	Force decompression, even if prompted
`-k`	Keep the original compressed file (used with `gzip -d`)
`-l`	List information about compressed files
`-r`	Recursively decompress files in directories (with `gzip`)
`-t`	Test compressed file integrity
`-v`	Verbose mode
`-h`	Display help

Difference Between `gzip` and `gunzip`

Command	Purpose
`gzip file.txt`	Compresses `file.txt` into `file.txt.gz`
`gunzip file.txt.gz`	Decompresses `file.txt.gz` back to `file.txt`

Example:

```
gzip report.txt
```

Creates:

```
report.txt.gz
```

Then:

```
gunzip report.txt.gz
```

Restores:

```
report.txt
```

`gunzip` vs `unzip`

`gunzip`	`unzip`
Extracts `.gz` files	Extracts `.zip` archives
Usually contains a single compressed file	Can contain multiple files and directories
Uses the Gzip format	Uses the ZIP format

Example:

```
gunzip log.gz
```

vs.

```
unzip archive.zip
```

Practical Uses

- Decompress downloaded Linux source code (`.tar.gz` files, often using `tar -xzf`).
- Restore compressed log files.
- Recover backups stored as `.gz`.
- Read compressed files without fully extracting them using `gunzip -c` or `zcat`.

Exit Status

Exit Code	Meaning
`0`	Successful decompression
`1`	Error occurred
`2`	Warning (such as a non-fatal issue)

Summary

- **Purpose:** Decompress Gzip (`.gz`) files.
- **Basic syntax:** `gunzip file.gz`
- **Equivalent command:** `gzip -d file.gz`
- **Common options:** `-c`, `-f`, `-l`, `-t`, `-v`
- **Default behavior:** Restores the original file and removes the `.gz` file after successful decompression.

4.bzip2

The `bzip2` command is used to **compress files** in Linux and Unix-like operating systems. It reduces the size of files to save disk space or make them easier to transfer.

- **Compression format:** `.bz2`
- **Compression algorithm:** Burrows-Wheeler Transform (BWT) with Huffman coding
- **Typical compression:** Better than `gzip`, but usually slower.
- **Common use:** Compressing text files, log files, source code, backups, and archives.

Note: `bzip2` compresses **files only**, not directories. To compress directories, first use `tar` and then compress the archive.

Example:

```
tar -cf backup.tar Documents/
bzip2 backup.tar
```

Result:

```
backup.tar.bz2
```

Syntax

```
bzip2 [OPTIONS] FILE...
```

Basic Examples

1. Compress a file

```
bzip2 file.txt
```

Before:

```
file.txt
```

After:

```
file.txt.bz2
```

The original `file.txt` is removed by default.

---

2. Compress multiple files

```
bzip2 file1.txt file2.txt file3.txt
```

Result:

```
file1.txt.bz2
file2.txt.bz2
file3.txt.bz2
```

---

3. Keep the original file

```
bzip2 -k file.txt
```

Result:

```
file.txt
file.txt.bz2
```

---

4. Decompress a file

```
bzip2 -d file.txt.bz2
```

or

```
bunzip2 file.txt.bz2
```

Result:

```
file.txt
```

---

5. Display compressed output without creating a file

```
bzip2 -c file.txt > file.txt.bz2
```

The original file remains unchanged.

---

6. Force overwrite

```
bzip2 -f file.txt
```

Overwrites an existing compressed file if necessary.

---

7. Test a compressed file

```
bzip2 -t file.txt.bz2
```

Checks whether the compressed file is valid without extracting it.

Output (if valid):

```
(no output)
```

If corrupted:

```
Data integrity error
```

---

8. Show compression statistics

```
bzip2 -v file.txt
```

Example output:

```
file.txt: 2.514:1, 68.2% saved
```

Common Options

Option	Description
`-d`	Decompress
`-k`	Keep original file
`-f`	Force overwrite
`-c`	Write output to standard output
`-t`	Test compressed file
`-v`	Verbose mode
`-q`	Quiet mode
`-1`	Fastest compression (lower compression ratio)
`-9`	Best compression (slowest)

Compression Levels

Fastest:

```
bzip2 -1 file.txt
```

Best compression:

```
bzip2 -9 file.txt
```

Default compression:

```
bzip2 file.txt
```

Compress an Entire Directory

Since `bzip2` cannot compress directories directly:

```
tar -cf project.tar project/
bzip2 project.tar
```

Or in one command:

```
tar -cjf project.tar.bz2 project/
```

Extract:

```
tar -xjf project.tar.bz2
```

Compare with Other Compression Tools

Tool	Extension	Speed	Compression Ratio
`gzip`	`.gz`	Fast	Good
`bzip2`	`.bz2`	Medium	Better than `gzip`
`xz`	`.xz`	Slow	Excellent
`zstd`	`.zst`	Very Fast	Excellent

Exit Status

Exit Code	Meaning
`0`	Success
`1`	Environmental or I/O error
`2`	Corrupt compressed file
`3`	Internal consistency error

Real-World Examples

Compress a log file:

```
bzip2 system.log
```

Compress a backup archive:

```
tar -cf backup.tar /home/user
bzip2 backup.tar
```

Test a backup before sending it:

```
bzip2 -t backup.tar.bz2
```

Decompress a backup:

```
bunzip2 backup.tar.bz2
```

Extract it:

```
tar -xf backup.tar
```

Summary

- `bzip2` compresses files into the `.bz2` format.
- It typically achieves a **higher compression ratio than `gzip`**, though it is generally slower.
- It **does not compress directories directly**; use it together with `tar` for directory backups.
- Useful options include `-d` (decompress), `-k` (keep original), `-t` (test integrity), `-v` (verbose), and `-9` (maximum compression).

5. xz

The `xz` command is used to **compress and decompress files** using the **XZ compression format** (`.xz`). It provides a **very high compression ratio**, often better than `gzip`, although it usually takes longer to compress.

- **Command:** `xz`
- **Package:** xz-utils (on most Linux distributions)
- **File extension:** `.xz`

Syntax

```
xz [OPTIONS] FILE...
```

How it Works

- Compresses one or more files into `.xz` format.
- By default, the original file is removed after successful compression.
- To decompress, use `xz -d` or `unxz`.

Basic Examples

1. Compress a file

```
xz file.txt
```

Before:

```
file.txt
```

After:

```
file.txt.xz
```

The original `file.txt` is deleted.

2. Keep the original file

```
xz -k file.txt
```

Result:

```
file.txt
file.txt.xz
```

---

3. Decompress a file

```
xz -d file.txt.xz
```

or

```
unxz file.txt.xz
```

Result:

```
file.txt
```

---

4. Keep compressed file while decompressing

```
xz -dk file.txt.xz
```

Result:

```
file.txt
file.txt.xz
```

---

5. Compress multiple files

```
xz file1.txt file2.txt file3.txt
```

Result:

```
file1.txt.xz
file2.txt.xz
file3.txt.xz
```

Common Options

Option	Description
`-d`	Decompress
`-k`	Keep original file
`-f`	Force overwrite
`-l`	Show information about an `.xz` file
`-t`	Test compressed file integrity
`-v`	Verbose output
`-z`	Compress (default)
`-0` to `-9`	Compression level (`-9` = highest compression)
`-e`	Extreme compression mode
`-T0`	Use all available CPU cores
`-c`	Write output to standard output

Compression Levels

```
xz -1 file.txt
```

Fast compression.

```
xz -6 file.txt
```

Default compression level.

```
xz -9 file.txt
```

Maximum compression.

Extreme Compression

```
xz -9e file.iso
```

- Highest compression ratio
- Much slower
- Uses more memory

Multi-core Compression

Use all CPU cores:

```
xz -T0 largefile.img
```

Use four CPU cores:

```
xz -T4 largefile.img
```

Display File Information

```
xz -l backup.tar.xz
```

Example output:

```
Strms  Blocks   Compressed  Uncompressed  Ratio
1       1       120 MiB     520 MiB       0.231
```

Test File Integrity

```
xz -t backup.tar.xz
```

No output means the archive is valid.

Compress to Standard Output

```
xz -c file.txt > file.txt.xz
```

The original file remains untouched.

Using with `tar`

Compress a directory:

```
tar -cJf backup.tar.xz Documents/
```

Options:

- `-c` → Create archive
- `-J` → Use `xz` compression
- `-f` → Output filename

Extract:

```
tar -xJf backup.tar.xz
```

Compare with Other Compression Tools

Tool	Extension	Compression	Speed
`gzip`	`.gz`	Good	Fast
`bzip2`	`.bz2`	Better	Medium
`xz`	`.xz`	Excellent	Slower
`zstd`	`.zst`	Excellent	Very Fast

Common Use Cases

- Compressing backups
- Distributing software packages
- Compressing large log files
- Archiving source code
- Saving disk space

Practical Examples

Compress a log file:

```
xz server.log
```

Compress using maximum compression:

```
xz -9 database.sql
```

Compress while keeping the original:

```
xz -k report.pdf
```

Extract an archive:

```
xz -d report.pdf.xz
```

Test archive integrity:

```
xz -t report.pdf.xz
```

Compress using all CPU cores:

```
xz -T0 ubuntu.iso
```

Exit Status

- `0` – Success
- `1` – Error occurred
- `2` – Warning (such as a non-fatal issue)

Summary

- `xz` compresses files into the **`.xz`** format.
- It generally provides a **higher compression ratio** than `gzip` and `bzip2`, at the cost of slower compression.
- Use `-d` to decompress, `-k` to keep the original file, `-9` for maximum compression, `-e` for extreme compression, and `-T0` to utilise all available CPU cores.
- `tar` integrates with `xz` using the `-J` option to create or extract compressed archives.

6. zip

The `zip` command is used to **compress one or more files and directories into a ZIP archive**. ZIP files reduce storage space and make it easier to transfer multiple files as a single file.

Syntax

```
zip [OPTIONS] archive_name.zip files...
```

or

```
zip [OPTIONS] archive_name.zip directory/
```

- **archive_name.zip** – Name of the ZIP archive to create.
- **files/directories** – Files or folders to compress.

Common Options

Option	Description
`-r`	Compress directories recursively
`-9`	Maximum compression
`-1`	Fastest compression (least compression)
`-u`	Update existing ZIP archive
`-d`	Delete files from an archive
`-m`	Move files into archive (delete originals after zipping)
`-e`	Encrypt archive with a password
`-q`	Quiet mode
`-v`	Verbose output
`-x`	Exclude files
`-j`	Junk directory names (store only filenames)

Examples

1. Compress a single file

```
zip backup.zip file.txt
```

Creates:

`backup.zip`

containing:

`file.txt`

---

2. Compress multiple files

```
zip documents.zip file1.txt file2.txt report.pdf
```

Creates one archive containing all three files.

---

3. Compress an entire directory

```
zip -r project.zip project/
```

The `-r` option recursively includes:

```
project/
├── src/
├── images/
└── README.md
```

into `project.zip`.

---

4. Maximum compression

```
zip -9 archive.zip largefile.iso
```

Uses the highest compression level.

---

5. Fast compression

```
zip -1 archive.zip video.mp4
```

Compresses quickly with lower compression.

---

6. Password-protect a ZIP file

```
zip -e secret.zip confidential.txt
```

You'll be prompted for a password.

---

7. Update an existing ZIP archive

```
zip -u backup.zip report.txt
```

Adds `report.txt` if it's new or updates it if it has changed.

---

8. Remove a file from a ZIP archive

```
zip -d backup.zip oldfile.txt
```

Deletes `oldfile.txt` from the archive.

---

9. Exclude files

```
zip -r project.zip project -x "*.log"
```

Compresses the directory while excluding all `.log` files.

---

10. Store files without directory paths

```
zip -j images.zip photos/*.jpg
```

Instead of:

```
photos/
    image1.jpg
    image2.jpg
```

the archive contains:

```
image1.jpg
image2.jpg
```

---

11. Move files into the archive

```
zip -m archive.zip *.txt
```

Creates the archive and deletes the original `.txt` files after they have been successfully added.

Viewing ZIP Contents

To list files inside a ZIP archive:

```
unzip -l archive.zip
```

Example output:

```
Archive: archive.zip

Length   Name
------   ----------------
1024     file1.txt
2048     report.pdf
4096     images/logo.png
```

Extracting ZIP Files

Use the `unzip` command:

```
unzip archive.zip
```

Extract to a specific directory:

```
unzip archive.zip -d /home/user/extracted
```

Compression Levels

Level	Meaning
`-0`	No compression
`-1`	Fastest compression
`-6`	Default compression
`-9`	Best compression

Practical Examples

Backup source code

```
zip -r source_backup.zip src/
```

Compress log files

```
zip logs.zip *.log
```

Create a password-protected backup

```
zip -er secure_backup.zip Documents/
```

Archive an entire home directory

```
zip -r home_backup.zip /home/bharath/
```

Advantages

- Compresses files to save storage space.
- Combines multiple files into a single archive.
- Cross-platform (Linux, Windows, macOS).
- Supports password protection.
- Widely supported by archive utilities.

Limitations

- Compression is less effective on already compressed files (e.g. `.mp4`, `.jpg`, `.png`, `.zip`).
- ZIP encryption (`-e`) provides basic protection; for highly sensitive data, stronger encryption methods (such as AES-enabled ZIP tools or encrypted archives) are preferable.
- Large archives may take noticeable time to create, depending on the compression level.

Summary

Feature	Description
Command	`zip`
Purpose	Compress files and directories into ZIP archives
Recursive	`zip -r archive.zip directory/`
Password Protection	`zip -e archive.zip file`
Maximum Compression	`zip -9 archive.zip file`
Update Archive	`zip -u archive.zip file`
Delete from Archive	`zip -d archive.zip file`
Extract	`unzip archive.zip`
List Contents	`unzip -l archive.zip`

The `zip` command is one of the most commonly used tools for creating compressed archives that are compatible across Linux, Windows, and macOS.

7.unzip

The `unzip` command is used to **extract files and directories from ZIP archive files** (`.zip`). It is the standard utility for decompressing ZIP files on Linux and Unix-like systems.

Syntax

```
unzip [OPTIONS] archive.zip
```

- `archive.zip` – The ZIP file to extract.
- `OPTIONS` – Modify how the extraction is performed.

Basic Examples

1. Extract a ZIP file

```
unzip files.zip
```

What it does:

- Extracts all files from `files.zip` into the current directory.

Example:

```
unzip project.zip
```

Result:

```
inflating: project/main.c
inflating: project/readme.txt
```

---

2. Extract to a specific directory

```
unzip archive.zip -d /path/to/directory
```

Example:

```
unzip backup.zip -d /home/user/restore
```

The contents will be extracted into:

```
/home/user/restore
```

---

3. List the contents without extracting

```
unzip -l archive.zip
```

Example:

```
unzip -l project.zip
```

Output:

```
Length      Date    Time    Name
---------  ---------- -----   ----------
  2048  07-20-2026 09:00   main.c
  1050  07-20-2026 09:00   README.md
```

---

4. View detailed information

```
unzip -v archive.zip
```

Shows:

- Compression method
- Compressed size
- Original size
- Compression ratio
- CRC values

---

5. Overwrite existing files automatically

```
unzip -o archive.zip
```

Without `-o`, `unzip` asks before replacing existing files.

Example:

```
unzip -o update.zip
```

6. Never overwrite existing files

```
unzip -n archive.zip
```

Existing files remain untouched.

---

7. Extract only specific files

```
unzip archive.zip filename
```

Example:

```
unzip project.zip README.md
```

Extracts only:

```
README.md
```

You can also use wildcards:

```
unzip project.zip "*.txt"
```

---

8. Exclude specific files

```
unzip archive.zip -x pattern
```

Example:

```
unzip project.zip -x "*.log"
```

Extracts everything except `.log` files.

---

9. Test the archive

```
unzip -t archive.zip
```

Checks the integrity of the ZIP file without extracting it.

Example:

```
unzip -t backup.zip
```

Output:

```
testing: data.txt OK
No errors detected
```

---

10. Quiet mode

```
unzip -q archive.zip
```

Suppresses most output during extraction.

Common Options

Option	Description
`-l`	List archive contents
`-v`	Show verbose information
`-d DIR`	Extract into directory
`-o`	Overwrite files without prompting
`-n`	Never overwrite existing files
`-t`	Test archive integrity
`-q`	Quiet mode
`-x`	Exclude files
`-j`	Junk directory names (extract all files into one directory)
`-P PASSWORD`	Use the specified password for encrypted ZIP files (note: passing passwords on the command line can expose them to other users on the system)

Real-World Examples

Extract downloaded software

```
unzip vscode.zip
```

Restore a backup

```
unzip backup.zip -d /restore
```

Extract documentation only

```
unzip project.zip "*.pdf"
```

Skip existing files

```
unzip -n archive.zip
```

Test before extraction

```
unzip -t backup.zip
```

Installing `unzip`

If `unzip` is not installed:

Ubuntu/Debian

```
sudo apt install unzip
```

RHEL/CentOS

```
sudo dnf install unzip
```

or

```
sudo yum install unzip
```

openSUSE

```
sudo zypper install unzip
```

Arch Linux

```
sudo pacman -S unzip
```

Exit Status

Exit Code	Meaning
0	Extraction completed successfully
1	Warnings (for example, some files were skipped)
2	Generic ZIP format error
3	Severe ZIP format error
9	ZIP file not found
11	Incorrect password for an encrypted archive

Related Commands

Command	Purpose
`zip`	Create ZIP archives
`zipinfo`	Display information about ZIP archives
`tar`	Create and extract TAR archives
`gzip`	Compress `.gz` files
`gunzip`	Decompress `.gz` files
`7z`	Create and extract many archive formats
`bsdtar`	Extract various archive formats, including ZIP

Summary

- **Purpose:** Extract files from ZIP archives.
- **Common usage:** `unzip archive.zip`
- **Extract elsewhere:** `unzip archive.zip -d directory`
- **List contents:** `unzip -l archive.zip`
- **Overwrite automatically:** `unzip -o archive.zip`
- **Test archive:** `unzip -t archive.zip`
- **Extract specific files:** `unzip archive.zip "*.txt"`

`unzip` is one of the most commonly used archive utilities on Linux for handling ZIP files created on Linux, Windows, and macOS.