gzip, xz and zstd Compress files

CONCEPTS

gzip and xz delete the input file by default; zstd keeps it. Add -k to gzip/xz, or --rm to zstd to delete.

gzip fast, universal
↓ smaller
zstd fast, modern
↓ smaller, slower
xz smallest, slow

These tools compress single streams, not folders. Use tar to bundle a directory first (tar -caf picks the compressor from the suffix).

Rule of thumb: gzip for compatibility, zstd for speed and everyday use, xz for archives that are written once and downloaded often.

Higher levels cost far more time than they save space. xz -7 to -9 also need much more memory to decompress.

GZIP

gzip *<file>*
Compress to <file>.gz, remove original

gzip -k *<file>*
Compress, keep the original

gzip -d *<file.gz>*
Decompress (same as gunzip)

gzip -c *<file>* **>** *<out.gz>*
Write to stdout, keep original

gzip -9 *<file>*
Best compression (slowest)

XZ

xz *<file>*
Compress to <file>.xz, remove original

xz -k *<file>*
Compress, keep the original

xz -d *<file.xz>*
Decompress (same as unxz)

xz -c *<file>* **>** *<out.xz>*
Write to stdout, keep original

xz -9 *<file>*
Strong compression, high memory

XZ (cont.)

xz -T0 *<file>*
Use all CPU cores

ZSTD

zstd *<file>*
Compress to <file>.zst, keep original

zstd -d *<file.zst>*
Decompress (same as unzstd)

zstd -c *<file>* **>** *<out.zst>*
Write to stdout

zstd --rm *<file>*
Delete input after success

zstd -19 *<file>*
High compression, slower

zstd -T0 *<file>*
Use all CPU cores

zstd --long=27 *<file>*
Long-distance matching, 128 MB window

READ AND INSPECT

zcat *<file.gz>*
Print decompressed content

xzcat *<file.xz>*
Print decompressed content

zstdcat *<file.zst>*
Print decompressed content

zcat *<file.gz>* **| less**
Page through a compressed file

zgrep *<pat>* *<file.gz>*
Search without unpacking

zless *<file.gz>*
Browse a compressed file

gzip -l *<file.gz>*
Show sizes and ratio

xz -l *<file.xz>*
Show sizes, check and blocks

zstd -l *<file.zst>*
Show frames, sizes, ratio

TEST AND CHECK

gzip -t *<file.gz>*
Test integrity, silent if ok

xz -t *<file.xz>*
Test integrity

zstd -t *<file.zst>*
Test integrity

PIPES AND ARCHIVES

<cmd> **| gzip >** *<out.gz>*
Compress a stream

gzip -dc *<file.gz>* **|** *<cmd>*
Decompress into a pipe

tar -caf *<out.tar.zst>* *<dir>*
Archive, compressor from suffix

tar -xf *<out.tar.zst>*
Extract any tar compression

ZIP AND UNZIP

zip -r *<out.zip>* *<dir>*
Zip a directory

unzip *<file.zip>*
Extract here

unzip *<file.zip>* **-d** *<dir>*
Extract into a directory

unzip -l *<file.zip>*
List contents

WHICH FORMAT

Compared

Tool	Speed	Size	Use for
`gzip`	fast	medium	compatibility, logs
`zstd`	very fast	good	everyday, backups
`xz`	slow	smallest	release tarballs
`zip`	fast	medium	sharing with Windows

RECIPES

Max-compress a big file using all cores

```
zstd -19 -T0 --long=27 big.img
# decompress: zstd -d --long=27
big.img.zst
```

Compress a database dump on the fly

```
pg_dump mydb | zstd -T0 >
mydb.sql.zst
zstdcat mydb.sql.zst | psql mydb
```

Compress every log older than a day

```
find /var/log/app -name '*.log' -
mtime +1 \
  -exec gzip {} +
```

Compare formats on your data

```
for c in 'gzip -9' 'zstd -19' 'xz
-9'; do
  printf '%s: ' "$c"; $c -c data
| wc -c
done
```