Linux Administration

File compression

Xavier Belanger

Compressing files

- Various utilities and files formats are available on Linux.
- *zip* is an easy solution when you need compatibility with other systems.
- *tar* (historically “tape archiver”) is usually the go-to tool to manipulate archives files along with *gzip* or *xz*, used for compression.

Manipulating zip archives

- Permissions and ownership are not supported.
- Create: *zip <archive>.zip <file>*
- Extract: *unzip <archive>.zip <file>*

tar archives

- *tar* will aggregate multiples files into one, keeping the directory structure, the file permissions, ownership and other information.
- By itself *tar* doesn’t compress the data. Additional libraries are used for that purpose.
- A tar file is sometimes referred as a *tarball*.

Manipulating tgz archives

The *gzip* compressor can be used with *tar* to produce .tgz or .tar.gz files.

- Create: *tar cvfz <archive>.tgz <file>*
- Extract: *tar xvfz <archive>.tgz <file>*

Manipulating txz archives

The xz compressor can be used with tar to produce .txz or .tar.xz files.

- Create: *tar cvfJ <archive>.txz <file>*
- Extract: *tar xvfJ <archive>.txz <file>*

Managing backups

- Since most pieces of data are stored in various files, you can use regular archive commands to manage some simple backups.
- Depending on your environment, more advanced backup solutions may be available.
- When restoring a file from backup, make sure to not overwrite existing copies, if any. You may also want to validate permissions and ownership.
- Rules for keeping backups are usually based on available storage space, the ability to regenerate some files, and policies and legal requirements.

Backup tips

- Include the archive creation date in the file name:

  tar cfz myfiles-$(/usr/bin/date +"%F").tgz myfiles

- Create an index of archived files:

  tar tvfz myfiles-2025-10-02.tgz > myfiles-2025-10-02-index.txt

- Keep a checksum of your backup files:

  sha256sum myfiles-2025-10-02.tgz > myfiles-2025-10-02.tgz.sha256

  sha256sum -c myfiles-2025-10-02.tgz.sha256

Encrypting files

- Backup files may need to be encrypted depending where they are created and stored.
- A reliable process need to be used to make sure that files could be decrypted in the future.

Encrypting files with GPG

GPG (GNU Privacy Guard, open-source version of PGP – Pretty Good Privacy) is a tool commonly available on most Linux distributions.

- Encrypting a file:
  gpg --symmetric --cipher-algo AES256 <file>
- Decrypting a file:
  gpg -o <file> --decrypt <file>.gpg

Synchronizing files

- Archive/compressed files can be used to maintain files consistency between systems, but with an additional delay.
- *rsync* can be used to maintain a more real-time synchronization.

The rsync command

- This tool can be used to maintain files and directories synchronized across different locations (on the same system, or different systems).
- *rsync* can use its own network protocol, or be used with SSH for additional security.
- Basic usage:
  rsync -a <source> <destination>

Email encoding

- When attaching files to an email, the MIME standard is required to convert files to ASCII characters. This is performed by transforming the files using base64 encoding.
- Base64 is an encoding algorithm using letters (uppercase and lowercase), numbers, characters plus and slash. The equal sign is used for padding if needed.
- Encoded files are larger (by 33%), not smaller.

MIME: Multipurpose Internet Mail Extensions
ASCII: American Standard Code for Information Interchange

base64 encoding

- Encoding a file:
  openssl enc -base64 -in <input file>
  -out <output file>
- Decoding a file:
  openssl enc -base64 -d -in <input file>
  -out <output file>
- You can also use the command named “base64”.