



## Conceptos Clave: Respaldo, Empaquetado y Compresión

Antes de usar los comandos, es vital distinguir qué estamos haciendo:

1. Copia de Seguridad (Backup): Proceso de duplicar datos para recuperarlos en caso de pérdida.
2. Empaquetado (Archiving): Agrupar varios archivos y carpetas en un solo archivo (fichero "contenedor"), pero sin reducir su tamaño. El estándar en Linux es **tar**.
3. Compresión: Aplicar un algoritmo matemático para eliminar redundancias y reducir el espacio que ocupa un archivo.



## El Comando **tar** (Tape Archiver)

Es la herramienta universal para gestionar paquetes de archivos. Su sintaxis básica es:

```
tar [opciones] [nombre_del_paquete.tar]  
[ficheros_o_directorios_a_incluir]
```

### Tabla de Opciones Principales de **tar**

Opción	Función	Descripción
-C	Create	Crea un nuevo archivo de empaquetado.
-X	Extract	Extrae el contenido de un paquete.
-v	Verbose	Muestra en pantalla el progreso (qué archivos se procesan).
-f	File	Indica que el siguiente argumento es el nombre del archivo. Siempre va al final de las opciones.
-t	List	Lista el contenido de un paquete sin extraerlo.
-Z	Gzip	Filtra el archivo a través de <b>gzip</b> (compresión rápida).
-j	Bzip2	Filtra el archivo a través de <b>bzip2</b> (compresión media).
-J	Xz	Filtra el archivo a través de <b>xz</b> (compresión máxima).



## Herramientas de Compresión Individuales

Si no quieres empaquetar, sino comprimir un solo archivo de forma directa, usamos:

- **gzip / gunzip**: El más rápido y compatible. Extensión **.gz**.
- **bzip2 / bunzip2**: Más lento que **gzip** pero comprime mejor. Extensión **.bz2**.
- **xz / unxz**: El más moderno. Consume mucha CPU y tiempo, pero ofrece el menor tamaño posible. Extensión **.xz**.



# Prácticas Paso a Paso: Empaquetado y Compresión



## Ejercicio 1: Creación de un paquete simple (Sin compresión)

- Objetivo: Agrupar una carpeta en un solo archivo para transporte.
- Paso a paso: 1. Crea una carpeta con archivos: `mkdir fotos ; touch fotos/img1.jpg fotos/img2.jpg`  
2. Empaqueta: `tar -cvf mis_fotos.tar fotos/`
- Explicación: Hemos creado un "contenedor" llamado `mis_fotos.tar`. El tamaño del archivo `.tar` será la suma exacta de las fotos. 📦



## Ejercicio 2: Compresión estándar con Gzip

- Objetivo: Crear un backup comprimido de la configuración del sistema.
- Paso a paso: 1. Ejecuta: `tar -czvf config_backup.tar.gz /etc/*.conf` (Nota: puede dar errores de permiso en algunos archivos, es normal).
- Explicación: La opción `-z` indica a `tar` que, tras empaquetar, use `gzip` para comprimir. El resultado es un archivo mucho más pequeño. 🌐



## Ejercicio 3: Compresión de alta eficiencia con XZ

- Objetivo: Comprimir archivos grandes para ahorro máximo de espacio.
- Paso a paso:
  1. Ejecuta: `tar -cJvf logs_pesados.tar.xz /var/log`
- Explicación: Usamos `-J` (mayúscula). Es ideal para backups que se van a guardar a largo plazo porque el ahorro de disco es superior a `gzip`. 📊



## Ejercicio 4: Ver el contenido sin extraer

- Objetivo: Verificar qué hay dentro de un backup antes de restaurarlo.
- Paso a paso:
  1. Ejecuta: `tar -tvf config_backup.tar.gz`
- Explicación: La opción `-t` (list) nos permite ver la tabla de contenidos. Es una medida de seguridad para no sobrescribir archivos por error. 📋



## Ejercicio 5: Extracción selectiva

- Objetivo: Recuperar solo un archivo específico de un paquete grande.
- Paso a paso:
  1. Ejecuta: `tar -xzvf config_backup.tar.gz etc/adduser.conf`
- Explicación: Al añadir la ruta del archivo al final del comando, `tar` solo extraerá ese elemento y no todo el contenido. 🎯



## Ejercicio 6: Compresión directa de un fichero único

- Objetivo: Reducir un log sin usar `tar`.

- Paso a paso:
  1. Crea un archivo grande: `ls -R /etc > listado.txt`
  2. Comprime: `gzip listado.txt`
- Explicación: Al usar `gzip` directamente, el archivo original desaparece y es sustituido por `listado.txt.gz`. 😞

### **Ejercicio 7: Descompresión directa**

- Objetivo: Volver a tener el archivo original disponible.
- Paso a paso:
  1. Ejecuta: `gunzip listado.txt.gz`
- Explicación: Recuperamos `listado.txt` y el archivo comprimido se elimina. 🔓

### **Ejercicio 8: Uso de Bzip2 para archivos de texto**

- Objetivo: Probar el equilibrio entre velocidad y tamaño.
- Paso a paso:
  1. Crea un paquete: `tar -cjvf texto_mejorado.tar.bz2 /var/log/syslog` (si existe).
- Explicación: `-j` usa `bzip2`. Es muy eficiente con archivos de texto plano o código fuente.



### **Ejercicio 9: Comparación de tamaños**

- Objetivo: Analizar qué algoritmo es mejor.
- Paso a paso:

1. Crea tres versiones de una carpeta:
  - `tar -czf test.gz /etc`
  - `tar -cjf test.bz2 /etc`
  - `tar -cJf test.xz /etc`

2. Compara: `ls -lh test.*`

- Explicación: Observarás que el archivo `.xz` es generalmente el más pequeño de los tres. 📊

### **Ejercicio 10: Extracción en una carpeta diferente**

- Objetivo: Evitar "ensuciar" el directorio actual al descomprimir.
- Paso a paso:
  1. Crea una carpeta de destino: `mkdir restauracion`
  2. Extrae: `tar -xvzf config_backup.tar.gz -C restauracion/`
- Explicación: La opción `-C` (Change directory) le dice a `tar` que se mueva a esa carpeta antes de empezar a soltar los archivos. 📁

