Para comprimir una carpeta (y todo su contenido, obviamente) con la máxima compresión posible lo mejor es utilizar xz. Pero, como xz sólo comprime un único archivo, deberemos utilizar el truco de, primero comprimir la carpeta con tar (tal y como hacemos en este hack) para luego aplicar xz a esa compresión. Lo bueno es que podemos hacerlo todo desde un único comando indicando el parámetro -J a tar:

```
tar -cJvf /home/usuariox/CarpetaComprimida.tar.xz -C /home/usuariox/ CarpetaAComprimir/
```

Como podemos observar, la diferencia entre el comando del hack en el que comprimimos sólo con tar y este en el que usamos xz es que hemos agregado el parámetro -J, Pero cuidado, es importante que -J se agregue ANTES que el parámetro -f, porque -f espera inmediatamente el nombre del archivo final comprimido. Si prestamos atención vemos que, también a diferencia del hack original citado, en este que estamos haciendo ahora la extensión del archivo final no es .tar sino .tar.xz.

Ahora bien, la idea de este truco que estamos ejecutando no es comprimir sin más, sino aplicar la máxima compresión posible, tal y como hacemos en este otro hack, donde comprimimos al máximo posible un único archivo. La diferencia aquí es que al utilizar tar deberemos, antes de la ejecución del propio tar, pasarle los parámetros de máxima compresión a xz mediante la modificación de la variable XZ_OPT. Esto es:

```
XZ_OPT='-9 -e -T4 -k' tar -cJvf /home/usuariox/CarpetaComprimida.tar.xz -C /home/usuariox/ CarpetaAComprimir/
```

Y, al igual que se explica en ese otro hack, esto es lo que hace cada parámetro:

- -9: Utiliza el nivel de compresión más alto.
- -e: Activa el modo de compresión extrema, que puede llevar más tiempo pero logra una mejor compresión.
- -T4: Activa el multi-hilo, usando 4 hilos.
- -k: Conserva el archivo original. Si no se pone -k, borrará el original al acabar la compresión

Por ejemplo, podríamos comprimir un contenedor nspawn de systemd, así:

```
sudo env XZ_OPT='-9 -e -T4' tar --acls --xattrs --sparse --numeric-owner -cJvf /home/nico/Descargas/base-
debian-13.tar.xz -C /var/lib/machines/ base-debian-13/
```

Incluso podríamos hasta separar el archivo resultante, en archivos de 24MB, para poder subirlos a GitHub:

```
sudo env XZ_OPT='-9 -e -T4' tar --acls --xattrs --sparse --numeric-owner -cJvf - -C /var/lib/machines/ base-
debian-13/ | split -b 24MB -d -a 3 - /home/nico/Descargas/base-debian-13.tar.xz.part-
```

- **-b 24MB**: cada fragmento tendrá como máximo 24.000.000 bytes.
- **-d**: utiliza sufijos numéricos.
- **-a 3**: utiliza tres dígitos (000, 001, 002...).
- **–**: recibe los datos comprimidos desde la tubería.

Ese comando generará archivos como:

```
base-debian-13.tar.xz.part-000
base-debian-13.tar.xz.part-001
base-debian-13.tar.xz.part-002
base-debian-13.tar.xz.part-003
...
```

Luego, para reconstruir el archivo original, podríamos hacer:

```
cat /home/nico/Descargas/base-debian-13.tar.xz.part-* > /home/nico/Descargas/base-debian-13.tar.xz
```

Importante: los fragmentos no son archivos XZ independientes. Necesitas todos para reconstruir el archivo original.