Adventures with single-drive backup to LTO tape using open source tools « Frederick's Timelog

Frederick ⋮ 6-7 minutes ⋮ 8/15/2021

I got a Tandberg LTO-6 drive off eBay recently as a way to have an offline, air-gapped third backup of data that normally lives on my NAS or backup storage server.

Although my NAS is already backed up daily to a ZFS pool on another server, all of these systems are networked—and therefore, vulnerable to ransomware, malware, sloppy sysadmin commands on the terminal, and even electric-surge-caused hardware malfunction. And although I do back up some data to cloud storage, not all data is worth the recurring monthly charges of S3/Glacier/Backblaze B2. Besides, playing with hardware is fun.

Magnetic tape, which can store as much as 2.5 TB uncompressed (in LTO-6, the generation I started with) or 12 TB uncompressed (in LTO-8, the current generation as of mid-2021), is a time-tested option that fits in perfectly.

Veeam Backup & Replication Community Edition works well with standalone tape drives. However, it's a proprietary system that uses Microsoft Tape Format for the on-tape format—a format that is very challenging to recover yourself without using proprietary tools. Moreover, the tape backup mechanism in Community Edition (i.e., without using licensed NAS backup features) is not meant for backing up large volumes of general purpose files—it's really designed for archiving VM backups from disk.

LTFS also works. However, my initial attempts to use it were foiled by a Microsemi HBA that doesn't support TLR. Also, if you don't use proprietary tape software, LTFS can actually perform more slowly for a bunch of reasons (e.g., multithreaded copying, large number of small files, etc.).

When using a Linux desktop, *way* more options are available using ***decades-old*** software that was designed for tape from the get-go.

Remember tar (tape archive)?

Turns out: tar is as usable in 2021 for tape as it was thirty years ago.

Piping a sorted list of filenames into tar

```
find 'Folder/' -type f | sort | tar -cvf /dev/nst0 --no-recursion -T -
```

Getting checksums before tar

```
find 'Folder/' -type f -print0 | sort -z | xargs -0 sha256sum | tee
Folder.20210812.sha256sum
```

You can use `sha256sum` directly with a glob spec of files, but `find` will recurse through directories.

Buffering tar with mbuffer

```
tar --label="archive-name" -b512 -cvf - | sudo mbuffer -m 8G -P 80 -s
262144 -o /dev/nst0
```

`tar -b512` sets the blocking factor to 512 so that each tar record matches the 262144-byte block size of the tape drive (512 × 512 = 262144).

`mbuffer -P 80` tries to fill the buffer to 80% before starting to write out.

`mbuffer -s 262144` matches the 262144-byte block size.

Verifying the contents of the tape archive

```
tar -tvf /dev/nst0
```

This only reads through the end of the file. You need to advance to the next file to read through another tape archive.

Advancing the tape

```
mt -f /dev/nst0 status
mt -f /dev/nst0 fsf 1
mt -f /dev/nst0 bsf
mt -f /dev/nst0 rewind
mt -f /dev/nst0 eject
```

Enabling hardware encryption (drive dependent)

This Tandberg drive seems to have the same guts as an HP LTO-6 drive. 256-bit encryption keys can be generated and loaded, but these drives require an extra flag (`-a 1`). The convenience advantage of enabling hardware encryption is that we can stream from tar directly to tape and back, and the encryption is all transparent to the applications.

```
stenc -g 256 -k keyfile.key -kd "optional key description"
stenc -f /dev/nst0 -e on -a 1 --ckod --protect -k keyfile.key
stenc -f /dev/nst0 --detail
stenc -f /dev/nst0 -e off -a 1
```

Bonus: Encoding a barcode into cartridge memory (aka LTO-CM

or MAM) using IBM ITDT

The barcode is set in the RFID memory chip and is assigned attribute number 0806. HPE's LTFS utilities can encode it as part of the LTFS format process, but I figured out how to do this when not using LTFS.

Every attribute is preceded by a 5-byte attribute header, which contains:

- 2 bytes: the attribute number itself (hex 08 06)
- 2 bytes: format—apparently ASCII (hex 01 00)
- 1 byte: length—this has to be 32 decimal (hex 20)

The remaining 32 bytes should be padded with spaces. An example 37-byte binary file, when dumped using xxd (hexadecimal representation on the left, ASCII on the right) should look like this:

```
$ xxd 0806.bin
00000000: 0806 0100 2046 4a4b 3637 304c 3620 2020  .... FJK670L6
00000010: 2020 2020 2020 2020 2020 2020 2020 2020
00000020: 2020 2020 20
```

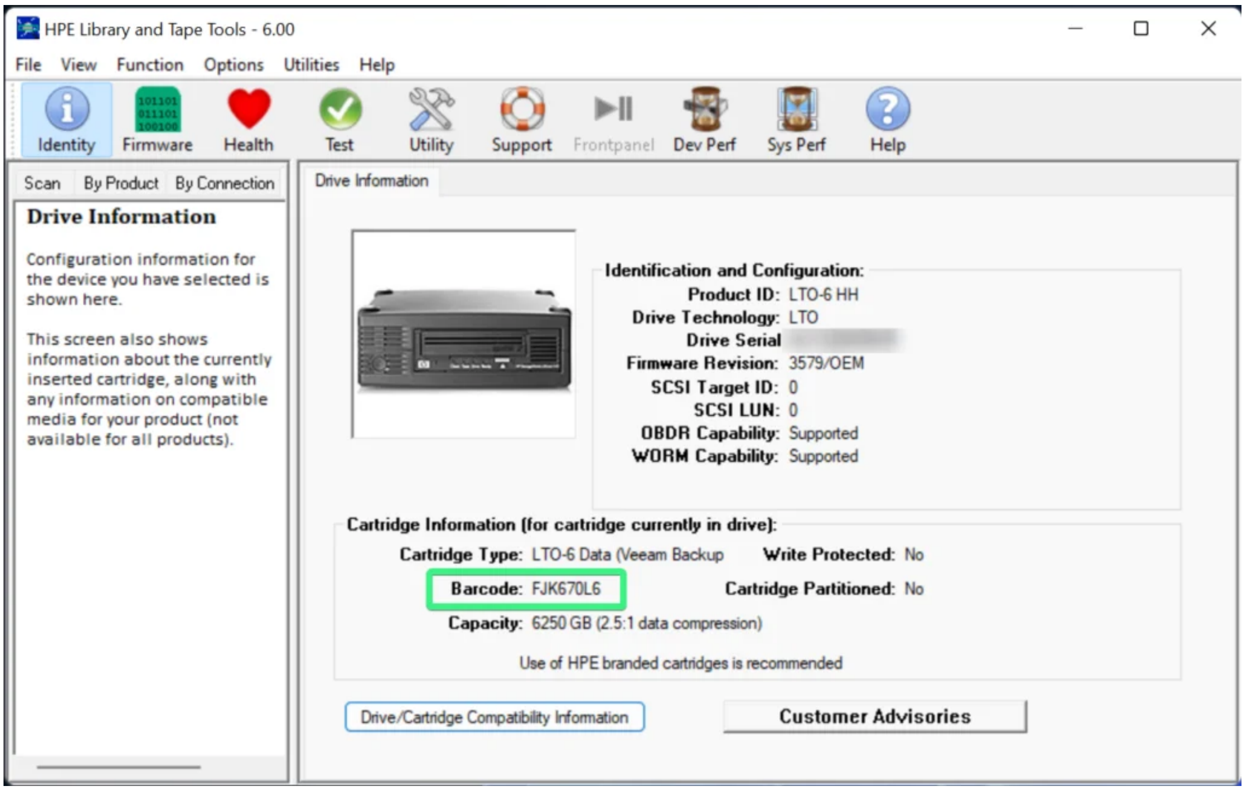
We can try to read the attribute from the cartridge using ITDT:

```
.\itdt.exe -f \\.\tape0 readattr -p 0 -a 0806 -d 0806.bin
```

And we can try to encode it to the cartridge using ITDT:

```
.\itdt.exe -f \\.\tape0 writeattr -p 0 -a 0806 -s 0806.bin
```

Here's the evidence that the barcode was properly encoded:


HPE Library and Tape Tools - 6.00
File View Function Options Utilities Help
Identity Firmware Health Test Utility Support Frontpanel Dev Perf Sys Perf Help
Scan By Product By Connection
Drive Information
Drive/Cartridge Compatibility Information
Customer Advisories

Drive Information

Configuration information for the device you have selected is shown here.

This screen also shows information about the currently inserted cartridge, along with any information on compatible media for your product (not available for all products).

Identification and Configuration:
Product ID: LTO-6 HH
Drive Technology: LTO
Drive Serial [redacted]
Firmware Revision: 3579/OEM
SCSI Target ID: 0
SCSI LUN: 0
OBDR Capability: Supported
WORM Capability: Supported

Cartridge Information (for cartridge currently in drive):
Cartridge Type: LTO-6 Data (Veeam Backup
Write Protected: No
Barcode: FJK670L6
Cartridge Partitioned: No
Capacity: 6250 GB (2.5:1 data compression)
Use of HPE branded cartridges is recommended

Screenshot of HPE Library and Tape Tools showing barcode field

Appendix: Source Code

These are backups of the open source programs used above, providing some assurance that even if these programs end up disappearing from Linux distributions' package repositories, I will still be able to access the data stored on these tapes. (There's probably nothing to worry about here; it's more likely LTO-6 drives will be EOL long before tar and mt-st disappear.)

- **stenc**
  - Upstream repository: https://github.com/scsitape/stenc
  - License: GPL 2
  - Tarball sourced from https://github.com/scsitape/stenc/releases/tag/1.1.1
- **mbuffer**
  - Upstream: https://www.maier-komor.de/mbuffer.html
  - License: GPL 3
- **mt-st**
  - Upstream repository: https://github.com/iustin/mt-st
  - License: GPL 2
  - Tarball sourced from https://github.com/iustin/mt-st/releases/tag/v1.4