Hewlett Packard Enterprise

# HPE StoreOpen and LTFS

Best practices

**Technical white paper**

# Contents

# Introduction

The Linear Tape File System, usually abbreviated to LTFS, has revolutionized the way that tape is used. LTFS offers a convenient way of transferring and archiving content, removing dependencies on proprietary software and tape formats. This white paper gives an overview of LTFS, describes its basic configuration and usage, and then explores some of the less well-known features.

The term LTFS refers to the format specification which describes the manner in which the tape contents are laid out, starting with two partitions (one for data, one for the index), setting out the format in terms of tape blocks and filemarks, and then defining the XML elements used to create the index. This open format specification is now owned and developed by the Storage Networking Industry Association (SNIA), and membership of the technical working group is open to anyone who joins SNIA. The URL is provided below in the section "References."

LTFS is also sometimes used to refer to the software application which creates the on-tape format, and which allows you to mount and use a tape volume. Hewlett Packard Enterprise provides a free software implementation called StoreOpen which comes in two varieties: HPE StoreOpen Standalone for use with single drives, and HPE StoreOpen Automation which extends usage to tape libraries. For the sake of clarity, in this document we will use **LTFS** to refer to the **format** written to the tape cartridge, and **HPE StoreOpen** to refer to the **standalone software application**. Many of the concepts described will also be applicable to the automation product, but may differ in some of the specific details.

A number of third-party software providers now also offer LTFS format support within their products, which complement the free StoreOpen software and extend the use of LTFS into further workflows.

## Suitability of HPE StoreOpen

LTFS employs a tape drive feature called partitioning, which was introduced in the fifth generation of LTO technology; hence LTFS requires LTO-5 and newer drives. HPE StoreOpen software helps you to format, mount and use LTO-5 and newer tape cartridges as if they were normal disk volumes. The large capacities and high sustained transfer rates mean that data can be stored quickly and reliably, then transported to a different location or stored offsite in secure storage. HPE StoreOpen with LTFS is applicable to any market segment where large amounts of data need to be archived or transferred between sites.

Some examples include:

- Media and entertainment, for video transfer or archiving
- Surveillance video storage
- Healthcare for storing high resolution images such as medical X-Rays or scans

Because the LTFS tape format is platform-independent, it can also be used to exchange data files between different platforms (for example between systems running Microsoft® Windows® and systems running Linux®). Simply write the files on your source system, unmount and transfer the tape cartridge to the target system, mount it, and access your files.

The HPE StoreOpen software is updated periodically to support updated versions of the format specification and to add new features, so be sure to check back for updates from time to time.

## Other LTFS solutions

In cases where sophisticated content and asset management is required, other commercially available LTFS solutions may be a better fit for your workflow. More and more vendors are including LTFS within their offering, and a list of known solutions is maintained by the LTO program (see "References" on page 25).

### Where is LTFS not applicable?

Although HPE StoreOpen with LTFS is a powerful and flexible tool, there may be some circumstances where it is not ideally suited. A configuration where you need to store many millions of small files will require a lot more memory in the host computer in order to manage the index, and may also lead to an extended delay in mounting the file system. As a very rough rule of thumb, each file stored in an LTFS volume will add about 1 kB to the index size, and so a volume containing 1,000,000 files is likely to require approximately 1 GB for the index.

Secondly, HPE StoreOpen is not intended to be a drop-in replacement for a traditional backup system. Whilst it can be used in that way as a light-weight solution, Hewlett Packard Enterprise recommends that it should be used as an archival or long-term storage system alongside a traditional backup application.

Thirdly, although HPE StoreOpen presents the tape volume content as if it were a disk-based file system, it is of course a linear tape in the back end. This means that use cases which perform frequent access and updates, such as a database or "live" editing sessions, will likely suffer from unacceptable delays as the tape has to be repositioned frequently. In such models, the recommendation would be to copy to disk for the editing activity, and then copy back to tape once the project is complete.

## Basic configuration and usage

This section briefly describes how to set up and use an LTFS volume using HPE StoreOpen Standalone software. It is intended only as an overview of the steps involved; for further details please refer to the User Guides mentioned in "References" on page 25.

### Hardware configuration

Apart from using an HPE LTO-5 and newer tape drive, the two most important hardware considerations are the amount of memory in the host system, and the host bus adapter (HBA) which will connect to the tape drive or tape library. As mentioned above the number of files stored on each cartridge volume will have a direct effect on the memory requirements; it is therefore prudent to budget for at least 1 GB per 1,000,000 files per cartridge. Actual usage will depend on many factors including file sizes, directory structure, and how many extended attributes are used.

The choice of HBA (host bus adapter) is important, as not all of the HBA products on the market work well with tape drives. In particular many hardware RAID cards do not expect or understand tape drive operation, and may cause errors or unexpected behavior. HPE StoreOpen tries to verify the capabilities of the HBA before going ahead with the mount, but some problems may not manifest until you try to access the volume. The best advice is to check the compatibility matrix on the HPE StoreOpen website and also the HPE Data Agile BURA Compatibility matrix (see "References" on page 25).

### Software configuration

The basic steps are: (1) identify which tape drive to use, (2) identify where you want it to appear within in the file system, and then (3) make a logical connection between the two. However the way in which these steps are accomplished depends on the operating system in use. Note that the following description assumes that the tape cartridge has already been formatted for LTFS; refer to the User Guide documentation for more information on how this is done for your platform.

#### For Windows

HPE StoreOpen provides a utility called the LTFS Configuration Tool to assist with these steps, shown in figure 1. The system is scanned for connected tape drives, and a drop-down list is populated with all suitable devices that are found. On Windows systems the LTFS volume will appear as a drive letter, so a list of unused letters is presented on another drop-down list.

The logical connection between the tape drive and the Windows drive letter is referred to as a "mapping." Clicking on the "Create mapping" button will check the drive, the HBA capabilities, and the state of the cartridge (if any) currently loaded in the tape drive, before initiating the operations needed to mount the LTFS volume.

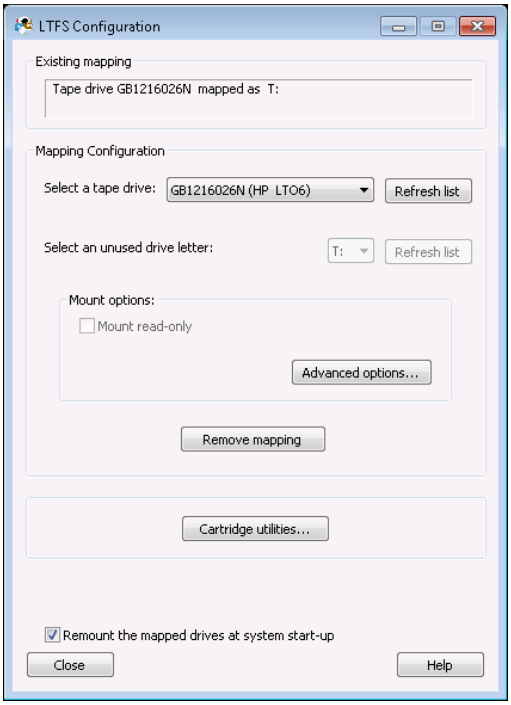



**Figure 1.** StoreOpen configuration tool for Windows

### For Mac

The HPE StoreOpen graphical user interface (GUI) on OS X takes the form of a wizard application, with successive tabs guiding you through the steps required. This is illustrated in figure 2. The Start tab provides a drop-down list of suitable devices connected to this system. The Prepare Cartridge tab helps load, eject, or format a cartridge ready for use. Once the drive and cartridge are ready, the Mount Volume tab lets you choose a name for the mounted volume and then initiates the mount process.

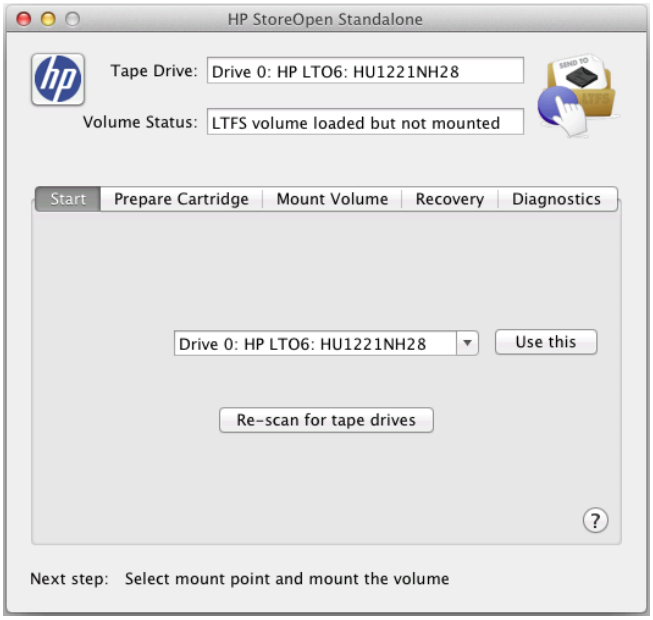



**Figure 2.** StoreOpen configuration tool for Mac

### For Linux

There is no GUI available on the Linux platform, so instead all the required steps are accomplished using shell commands at a terminal prompt as shown in figure 3.

```
moria:~ # cat /proc/scsi/sg/device_strs
HP              Ultrium 6-SCSI          J43V
hp              DVDROM DH20N            EB00
moria:~ #
moria:~ # mkdir /mnt/LTFSvolume
moria:~ #
moria:~ # ltfs /mnt/LTFSvolume -o devname=/dev/sg0
LTFS14000I LTFS starting, HP StoreOpen Standalone version 2.1.1, log level 2
LTFS14058I LTFS Format Specification version 2.1.0
LTFS14104I Launched by "ltfs /mnt/LTFSvolume -o devname=/dev/sg0"
LTFS14105I This binary is built for Linux (x86_64)
LTFS14106I GCC version is 4.3.2 [gcc-4_3-branch revision 141291]
LTFS17087I Kernel version: Linux version 2.6.27.19-5-default (geeko@buildhost) (gcc version 4.3.2 [gcc-
4_3-branch revision 141291] (SUSE Linux) ) #1 SMP 2009-02-28 04:40:21 +0100 x86_64
LTFS17089I Distribution: SUSE Linux Enterprise Server 11 (x86_64)
LTFS17089I Distribution: LSB_VERSION="core-2.0-noarch:core-3.2-noarch:core-4.0-noarch:core-2.0-x86_64:c
ore-3.2-x86_64:core-4.0-x86_64"
LTFS14063I Sync type is "time", Sync time is 300 sec
LTFS17085I Plugin: Loading "ltotape" driver
LTFS17085I Plugin: Loading "unified" iosched
LTFS20013I Drive type is HP LTO6, serial number is HU1221NH28
LTFS17160I Maximum device block size is 524288
LTFS11005I Mounting the volume
LTFS14111I Initial setup completed successfully
LTFS14112I Invoke 'mount' command to check the result of final setup
LTFS14113I Specified mount point is listed if succeeded
moria:~ #
```

**Figure 3.** LTFS configuration for Linux

The first task is to identify the tape drive; this is done by examining the contents of the special file **/proc/scsi/sg/device_strs**.

Second, create a directory to use as the mount point; in this example **/mnt/LTFSvolume**.

Finally run the **ltfs** command to initiate the mount process, providing both the mount point and the drive identification. When the prompt returns the volume will be accessible via the specified mount directory.

## Using the LTFS volume

Once the mount process has completed, the volume is ready for use and the content can be accessed. Because the LTFS volume now appears as part of the operating system, you can use any traditional file access tools to browse and copy files; however it is very important to be aware that GUI file managers (in particular the Finder application on Mac) make certain assumptions about file systems, and will try to locate some hidden system files every time you open a directory; furthermore they may try to build or refresh the thumbnails of the directory contents, which on a tape-based file system will result in considerable delays. Dragging files to the mounted volume icon without actually opening it will improve performance; and for some file managers it is possible to disable thumbnails which may also yield a better user experience. The optimal way of accessing the files is via a terminal window, using standard shell commands such as **ls** and **cp** to list and copy files to/from the LTFS volume. However this is a trade-off of performance against ease-of-use, and you are encouraged to determine what works best for you.

The Windows Explorer utility does not suffer from this problem to the same extent, because the Windows version of HPE StoreOpen reports each file and directory as being in the “offline” state; Explorer then does not try to examine each file. If accurate thumbnails are important to you, then this feature can be disabled via the advanced options window; however you are encouraged not to do so unless the resulting delays are acceptable in your workflows.

## Advanced usage

This section contains information on a number of less well-known features or opportunities for using HPE StoreOpen and LTFS. Typically these are accessed slightly differently for each operating system; so as well as a general description, specific information is provided for each platform.

### Sharing a volume across the network

The normal usage model for HPE StoreOpen is to have a tape drive directly connected to the computer system where it will be used; the LTFS volume is accessed by just one user who has sole control over the operation. In some circumstances it may be useful also to make it available to a different client across an Ethernet network, for example to copy files onto tape on a system which does not have the option of installing a host bus adapter. In this way the LTFS volume with its capacity and reliability can be shared between several different systems, perhaps to copy completed projects or to archive the day's work at the end of the day.

There are however a couple of caveats with doing this. Firstly the performance will be limited for all but the highest bandwidth (10GbE) Ethernet connections, because even Gigabit Ethernet (GbE) cannot transfer more than 100 MB/s, compared to the LTO-7 tape drive's native interface speed of 300 MB/s (160 MB/s for LTO-6, 140 MB/s for LTO-5). So even if the file transfer is able to take all available bandwidth on the Ethernet link, the best practical sustained transfer speed will be in the order of 40–60 MB/s.

The second issue is perhaps less apparent but may cause more confusion. Even though HPE StoreOpen makes the tape drive appear as a disk storage device, it is ultimately a linear tape which really works well with a single user. Imagine the situation where the tape drive is shared across a network with two other systems. A user on the first system may decide to start copying a 300 GB archive to tape, which will take some considerable time and effectively saturate both the Ethernet link and the tape drive. If a user on the second system then wishes to browse the contents of the tape, unaware that someone else is performing a large copy, their file browser (and potentially system) will block until the copy has completed. It is important to avoid this and to ensure that only one person accesses the tape at a time.

One further hidden effect of copying across a network connection is that depending on the network throughput, the data may arrive in a "bursty" fashion, resulting in fragmented writes to the LTFS volume. This will not be immediately obvious and may not have a direct impact on performance of either writes or reads, but may cause the size of the tape index to grow faster than expected as each burst of data may be recorded as a separate data extent.

It should also be mentioned that the client (remote) system will only see the LTFS volume as a mounted file system; it will not be aware that it is actually connected to a tape drive. This means that features such as formatting, checking, and rollback will not be available; they can only be executed on the server (local) system where the tape drive is physically connected. Other issues such as thumbnail generation and error handling may also be less clear when connecting from a remote system.

The following sections focus on how to share an LTFS volume within a homogeneous environment; it is also possible to share within a heterogeneous configuration by applying the same general steps, but the details are beyond the scope of this document. Issues such as appropriate network protocols, cross-platform user identities and permission mapping must be considered. The HPE StoreOpen webpages (see "References" on page 25) include a comprehensive technical white paper describing how to use Samba with HPE StoreOpen Automation to make a mount point from Linux visible on a Windows system; the same principles can be applied to HPE StoreOpen Standalone.

#### On Windows

The normal sharing mechanism (via Explorer) is not available because it does not work with tape drives. However sharing can be accomplished via the command prompt, following these steps:

1. Mount the volume on the local (server) system as usual using the StoreOpen Configuration Tool, and verify that the contents are now accessible.
2. Open a command prompt on the local system, and (assuming the LTFS volume is mapped as the **T:** drive) enter the following command:

   ```
   net share LTFSvolume=T:\ /users:1 /remark:"LTFS tape volume"
   ```

   This shares the **T:** drive with the name **LTFSvolume** for a maximum of one user. Note that it is not required to limit the number of users to 1, but as discussed above it is recommended in order to avoid resource contention. Also the "remark" parameter is not required but may be a helpful reminder. Further options are available, for example to restrict access to specific users; type net help share for further information.

3. Verify that the volume is now shared:

   ```
   net share LTFSvolume
   ```

4. On the remote client system, you can now connect to the LTFS tape volume on the server system **"servername"** either by using Explorer (map a network drive to the folder path \\servername\LTFSvolume) or by typing at a command prompt: net use T: \\servername\LTFSvolume.
5. The client system can now access the tape volume for reading and writing.
6. To terminate the connection, the client can just select the "Disconnect" option from within Explorer.
7. On the local server system, stop sharing the tape drive by typing at a command prompt:

   ```
   net share LTFSvolume /delete
   ```

8. Finally unmount the volume on the local server system using the Configuration tool.

---

**Important note**

The above was tested using Windows 7 and Windows Server® 2008. It may not work with later releases of Windows due to changes in the operating system, and so is not supported. For situations where a robust volume sharing solution is required, consider deploying a commercially available LTFS solution such as QStar Archive Manager. See "References" section for more information.

---

**On Linux**

Sharing between Linux-based computers is straightforward and is most easily accomplished using normal NFS configuration and commands. With care it is also possible to mount the volume via NFS on any other platform, subject to the caveats mentioned above regarding permissions and identities.

In general the volume should be mounted using the **hard** option to ensure that requests are retried indefinitely, rather than giving up after a timeout period. Along with this, specify the **intr** option to allow signals to interrupt file operations in a safe manner.

To improve performance, the remote system (performing the mount) should be configured to request reads and writes of 512 kB blocks rather than the typical 32 kB, if the system supports transfers of that size.

Combining these options will result in a command line option parameter set along the lines of:

```
-o hard,intr,rsize=524288,wsize=524288
```

along with the usual parameters (host/source, mountpoint, rw etc.) required for a mount operation. Note that the precise format of the options will depend on the platform performing the mount.

**On Mac**

Because of the way that the file system is implemented on OS X, security restrictions prevent sharing of the LTFS volume using the default sharing protocols afp and smb. One option is to enable and configure volume sharing using NFS; the manner in which this is accomplished is however not straightforward and is not describe here. Another workaround is to install and use the secure shell file system application (**sshfs**) on the remote (client) system; this package allows you to access a given resource such as the LTFS volume using secure sockets. This therefore preserves the integrity of the system whilst still enabling sharing of the volume. Assuming the **sshfs** package has already been downloaded and installed, the following steps will make enable sharing:

1. On the local (server) system, mount the LTFS volume using the HPE StoreOpen GUI as usual. For this example assume the volume is mounted as /Volumes/LTFStape.
2. On the remote (client) system, run the Terminal application (located in the Launchpad folder "Utilities" or "Other", depending on the version of OS X)
3. At the prompt, create a mount directory:

   ```
   mkdir /Volumes/ServerTape
   ```

4. Use the sshfs command to connect to the server LTFS volume. For this example, we're connecting to the server called **servername**, authenticating as the user called **username**. After entering this command you will be prompted for **username's** password.

   ```
   sshfs username@servername:/Volumes/LTFStape/Volumes/ServerTape
   ```

   You may wish to give the mounted volume a friendly name by adding the option

   ```
   -o volname="Friendly Name"
   ```

   It may also be necessary to increase the timeout value used, especially if sharing a StoreOpen Automation instance because the time taken to swap cartridges may cause issues. Add the option

   ```
   -o daemon_timeout=900
   ```

   The value is in seconds, so this example is for a timeout of 15 minutes.

5. The client system can now access the LTFS volume on the server as if it were locally mounted. Note that the caveat mentioned above in "Using the LTFS volume" is particularly relevant here because all traffic is across the network; best performance will be seen when copying files using the terminal window rather than using Finder.

## Accessing extended attributes

In addition to the normal metadata associated with files (name, location in directory structure, creation dates etc.)

HPE StoreOpen provides a wealth of further information which can be accessed by means of extended attributes. The LTFS format specification defines a number of values which can be read (and in some cases written); some examples of attributes available are:

- ltfs.indexTime—reports the date and time at which the last LTFS index was written
- ltfs.volumeName—the name of the volume provided when the volume was formatted
- ltfs.mediaLoads—the total number of times this cartridge has been loaded in a drive

Refer to the LTFS Format Specification available from the SNIA website (see "References" on page 25) for the full list of reserved extended attributes. Most of the LTFS reserved extended attributes are read-only as far as the user is concerned, because HPE StoreOpen maintains and updates them internally to reflect the current state of the volume or file.

Note that some of the LTFS reserved extended attributes relate to the entire LTFS volume (for example the index time) or to the media itself (for example the load count); others relate to specific files or directories. The following table summarizes the classes of attributes (metadata) and whether they are accessed by reading attributes of the specific file or the mount point itself:

**Table 1.** LTFS extended attribute classes

| ATTRIBUTE CLASS | EXAMPLE | HOW ACCESSED |
|---|---|---|
| **Object metadata** | ltfs.accessTime | File/directory |
| **Volume metadata** | ltfs.indexTime | Mount point |
| **Media metadata** | ltfs.mediaLoads | Mount point |
| **Software metadata** | ltfs.softwareVersion | Mount point file/directory |
| **Drive metadata** | ltfs.driveEncryptionMethod | Mount point |

Furthermore the specification also allows for any user-defined attributes to be stored within the LTFS XML index, so the mechanism is extensible for whatever purposes you may wish to achieve—for example keywords about the file content, image resolution, or geographic location information. HPE StoreOpen maintains all the LTFS-specific values, but it is up to the individual user (or higher-level software application) to define and use any custom values.

It would be prudent to use some constant-yet-unique identifier for any such user-defined attributes, so that all within a certain environment exist within their own "namespace" and hence are unlikely to clash or collide with attributes from another source (having the same name but different content). For example a product from company XYZ might wish to store the image resolution of JPEG files in an extended attribute, to avoid the need to open each file and read out the Exif information. The product designer may decide to use the attribute name **imageRes**. To avoid the risk of conflict, the full attribute name should be **XYZ.imageRes**. If the attribute is only relevant for a certain product "Alpha", then that can also be included in the attribute name: **XYZ.Alpha.imageRes**. The format of the content stored in this attribute is entirely up to the designer, but it may be of most use if a simple string were used—for example "1024x768."

**Accessing on Linux**

There are two standard system commands which interact with extended attributes. Use **getfattr** to read a current value, and **setfattr** to modify it (where applicable). When accessing the attributes on Linux it is necessary to prefix the name with "**user**." to indicate that they are in the user namespace. Note that the dump (**-d**) option of **getfattr** does not work with LTFS reserved extended attributes.

For example, to retrieve the date and time of the last index written to the volume:

```
getfattr -n user.ltfs.indexTime /mnt/ltfsVolume/

# file: /mnt/ltfsVolume/

user.ltfs.indexTime="2013-12-16T10:33:28.0010410002"
```

An example of modifying an extended attribute is to write the special metadata attribute **ltfs.sync** which will trigger a file system sync (index flush) operation. This attribute must be set for the mount point as it is classed as Volume Metadata. In this example we will also set a "commit message" to indicate the reason for the index flush; this is then saved in the XML index and can be retrieved later.

```
setfattr -n user.ltfs.commitMessage -v "Forced index flush" /mnt/ltfsVolume

setfattr -n user.ltfs.sync /mnt/ltfsVolume

getfattr -n user.ltfs.commitMessage /mnt/ltfsVolume

# file: /mnt/ltfsVolume user.ltfs.commitMessage="Forced index flush"
```

Another example would be to set a user-defined attribute to store the resolution of an image file, as described above:

```
setfattr -n user.XYZ.imageRes -v "1024x768" /mnt/ltfsVolume/MyPicture.jpg
```

**Accessing on Mac**

OS X provides the **xattr** command to read or modify extended attributes. Unlike the Linux examples above, you do not need (and must not include) the **user** namespace identifier. Using the same examples as already shown for Linux, firstly to retrieve the date and time of the last index written to the volume:

```
xattr -lpv ltfs.indexTime /Volumes/LTFStape

/Volumes/LTFStape: ltfs.indexTime: 2013-12-16T10:33:28.0010410002
```

The second example writes the special metadata attribute **ltfs.sync** to trigger a file system sync (index flush) operation. We also set a "commit message" to indicate the reason for the index flush; this is then saved in the XML index and can be retrieved later.

```
xattr -w ltfs.commitMessage "Forced index flush" /Volumes/LTFStape

xattr -w ltfs.sync 1 /Volumes/LTFStape

xattr -lpv ltfs.commitMessage /Volumes/LTFStape

/Volumes/LTFStape: ltfs.commitMessage: Forced index flush
```

Re-using the example described above to set a user-defined attribute storing the resolution of an image file:

```
xattr -w XYZ.imageRes "1024x768" /Volumes/LTFStape/MyPicture.jpg
```

**Accessing on Windows**

There is no standard way of accessing extended attributes on Windows, but HPE StoreOpen (from version 2.2.0 onwards) includes a command line utility called **ltfsattr**. This is installed in the same location as the rest of the HPE StoreOpen software (typically C:\Program Files\Hewlett-Packard\LTFS\) so you need to ensure that this folder is included in your executable path. One way of doing this is to open a command prompt window and type:

```
set PATH=%PATH%;C:\"Program Files\Hewlett-Packard\LTFS\"
```

However this setting is not persistent so must be repeated each time a new command prompt window is opened. To add this permanently it is necessary to open the "Advanced system settings" within the system control panel, and edit the PATH environment variable. Precise instructions may vary by system and are beyond the scope of this document.

**ltfsattr** supports several different modes of operation; these are listed by executing without any options or with the help (**-h**) option. Use the **-p** option to read (print out) a value. For example, the extended attribute holding the date and time of the last index written to the volume **T:** can be read using:

```
ltfsattr -p ltfs.indexTime T:\ 2014-06-12T13:25:28.000560000Z
```

The verbose (**-v**) option may also be included to show the name of the attribute as well as its value.

The examples shown above to trigger a file system sync (index flush) operation, with a specific commit message, can be executed as follows:

```
ltfsattr -w ltfs.commitMessage "Forced index flush" T:\ ltfsattr -w ltfs.sync 1 T:\

ltfsattr -v -p ltfs.commitMessage T:\

T:\: ltfs.commitMessage: Forced index flush
```

Re-using the example described above to set a user-defined attribute storing the resolution of an image file:

```
ltfsattr -w XYZ.imageRes "1024x768" T:\MyPicture.jpg
```

## Generating and storing a hash/checksum

One specific use for extended attributes is to store a secure hash or checksum of the file contents, to provide assurance that the file is still the same as when it was written to tape. A simple checksum may suffice, but added confidence will be gained by using a secure hash algorithm such as MD5 or SHA1, or a more modern alternative such as SHA256. Note that the word "secure" refers to the cryptographic derivation of these functions, but they are also of general use for verifying file contents since the likelihood of a hash match when the file contents do not match (known as a collision) is extremely small even for MD5. The better algorithms reduce further the risk of collisions but at the expense of increased computational time.

**Special note**

There is currently no standard attribute name for storing the hash value or checksum, so each solution will be vendor-specific; the next version of the LTFS Format standard will include a recommendation for this. In the interim it is suggested that the hash values should be stored as normal user extended attributes with the algorithm reflected in the name of the attribute, i.e., md5sum, sha1sum, or sha256sum.

In outline, the steps needed to protect a file with a hash are:

1. Write the file to the LTFS volume
2. Calculate the hash value for the file on disk
3. Calculate the hash value for the file now stored on tape
4. If the two values match then store the hash value as an extended attribute of the file. (In the unlikely event that they are different, the file needs to be rewritten or revalidated.)
5. To validate the file contents at some later date, calculate the hash value for the file stored on tape, and compare to the extended attribute value stored in the index

### Implementing on Windows

Microsoft provides an optional downloadable utility called File Checksum Integrity Verifier (**fciv**) which can generate hash values for the MD5 and SHA1 algorithms. By default it will calculate the MD5 value; supply the option—SHA1 if that is preferred. There is no support for calculating the SHA256 hash value. Note that the **fciv** utility also outputs the filename, so when checking the hash it is necessary to process the output to extract the desired string. The following example copies a file **sample.mp4** from the user's home directory to the LTFS volume mounted as the **T:** drive. The hash values are then calculated and compared, and if they match then the hash value is stored as an extended attribute of the target file:

```
copy C:\Users\name\sample.mp4 T:\
for /f "skip=3" %h in ('fciv C:\Users\name\sample.mp4') do set srchash=%h
for /f "skip=3" %h in ('fciv T:\sample.mp4') do set desthash=%h
if %srchash% neq %desthash% (
    echo ERROR - hash values do not match!
) else (
    ltfsattr -w md5sum %desthash% T:\sample.mp4
)
```

When desired, the hash value can be verified using the following steps:

```
for /f "skip=3" %h in ('fciv T:\sample.mp4') do set calchash=%h
for /f %h in ('ltfsattr -p md5sum T:\sample.mp4') do set savedhash=%h
if %calchash% neq %savedhash% (
    echo ERROR - hash values do not match!
) else (
    echo SUCCESS - hash values match
)
```

### Implementing on Linux

Linux provides a number of utilities to calculate and/or check hashes for files, including **md5sum**, **sha1sum**, and **sha256sum**. These utilities also output the filename, so when checking the hash it is necessary either to remove the filename, or to modify it to reflect the new path. The following example uses the first method. The file **/home/user/sample.mp4** is copied to the LTFS volume mounted as **/mnt/LTFSvolume/**. The hash values are then calculated and compared, and if they are the same then the hash value is stored as an extended attribute of the target file:

```
cp /home/user/sample.mp4 /mnt/LTFSvolume/
SRC_HASH=`md5sum /home/user/sample.mp4 | cut-c1-32`
DEST_HASH=`md5sum /mnt/LTFSvolume/sample.mp4 | cut -c1-32`
if [ $SRC_HASH != $DEST_HASH ]
then
    echo ERROR-hash values do not match!
else
    setfattr-n user.md5sum -v $DEST_HASH /mnt/LTFSvolume/sample.mp4
fi
```

When desired, the hash value can be verified using the following:

```
HASH1=`md5sum /mnt/LTFSvolume/sample.mp4 | cut -c1-32`
HASH2=`getfattr-n user.md5sum--only-values /mnt/LTFSvolume/sample.mp4`
if [ $HASH1 != $HASH2 ]
then
   echo ERROR-hash values do not match!
else
   echo SUCCESS-hash values match
fi
```

**Implementing on Mac**

Mac OS X has two different commands to calculate hash values: **md5** and **shasum**. The latter takes a parameter to indicate whether SHA1 (–a 1) or SHA256 (–a 256) should be used. The following example shows how to store and check an MD5 hash value for the file **/Users/user1/sample.mp4** which is to be copied to the LTFS volume mounted as **/Volumes/LTFSvolume**:

```
cp /Users/user1/sample.mp4 /Volumes/LTFSvolume/
SRC_HASH=`md5 -q /Users/user1/sample.mp4`
DEST_HASH=`md5 -q /Volumes/LTFSvolume/sample.mp4`
if [ $SRC_HASH != $DEST_HASH ]
then
   echo ERROR-hash values do not match!
else
   xattr -w md5sum $DEST_HASH /Volumes/LTFSvolume/sample.mp4
fi
```

When desired, the hash value can be verified using the following:

```
HASH1=`md5 -q /Volumes/LTFSvolume/sample.mp4`
HASH2=`xattr -p md5sum /Volumes/LTFSvolume/sample.mp4`
if [ $HASH1 != $HASH2 ]
then
   echo ERROR-hash values do not match!
else
   echo SUCCESS-hash values match
fi
```

## Saving and viewing index files

Every LTFS volume includes an index which describes the structure and content of the volume—filenames, directories, actual location on tape and so on. For everyday operation this is of limited interest to the user, because the file system can be inspected "live." However on occasion it may be helpful to be able to examine the index when the volume is not mounted, for example to locate the cartridge which contains a given file. Starting with version 2.1, HPE StoreOpen includes an option to save a copy of the index on your local disk when the LTFS volume is mounted and unmounted. This file can then be viewed or searched using any text editing tools, because it is a simple dump of the XML code which makes up the index. Although some familiarity with XML may be helpful, it is not required because the structure is fairly straightforward to understand.

The Windows version of HPE StoreOpen now includes a cartridge browser utility, which offers a straightforward way of viewing the available cartridge index files and displaying the content in an explorer-like view. Using this utility it is therefore possible to browse the content of offline cartridges. For more information refer to the User Guide documentation.

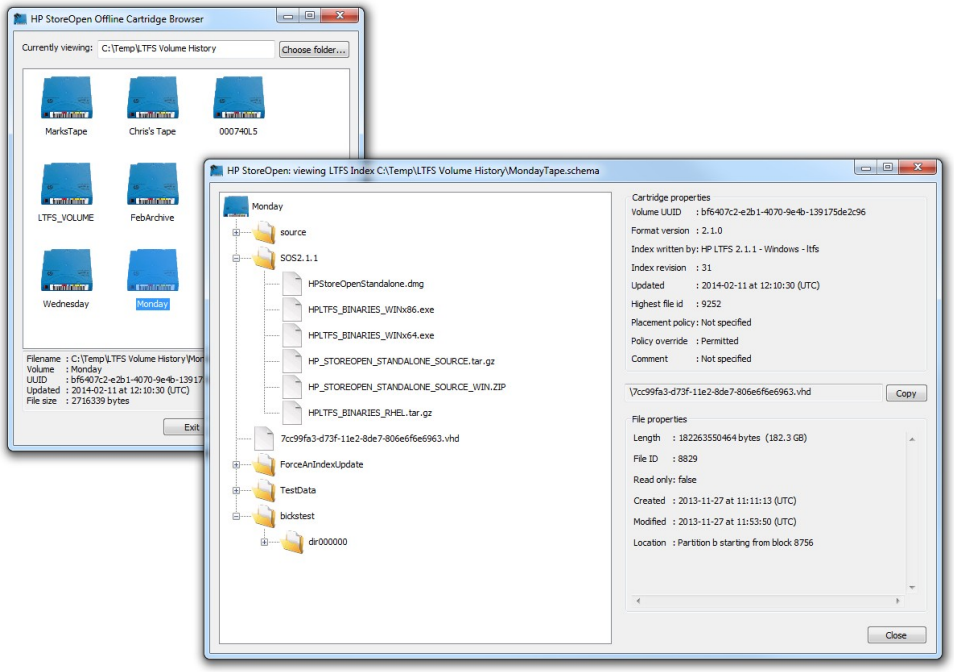



**Figure 4.** Offline cartridge browser utility

The location of the index files may be specified, but the actual name of each index file is determined by the HPE StoreOpen application. The name will depend on whether a serial number (bar code) was provided when the volume was formatted; if it was, then that serial number is used for the filename with .**schema** appended. For example the index file for a cartridge labeled as ASR674 would use

```
ASR674.schema
```

Otherwise the globally unique identifier (GUID) of the LTFS volume is used, again with .**schema** appended, resulting in a filename something like

```
eb3d2c8e-e47d-43b2-9535-fadf20e65df4.schema
```

The serial number method is easier for humans to manage and associate with a physical cartridge, but the GUID name is guaranteed to be unique to that LTFS volume.

Note that the application does not make any use of the file other than saving it for you, so you can move, modify, or delete it without affecting normal operation. Note also that the file will be overwritten on the next mount or unmount operation if the capture option is still selected, so be sure to move or copy the file to another location if you wish to preserve it for tracking or other purposes.

The following sections explain how to enable and locate the index files for each supported operating system.

### Enabling on Windows

From the Configuration dialog, click the "Advanced options" button.

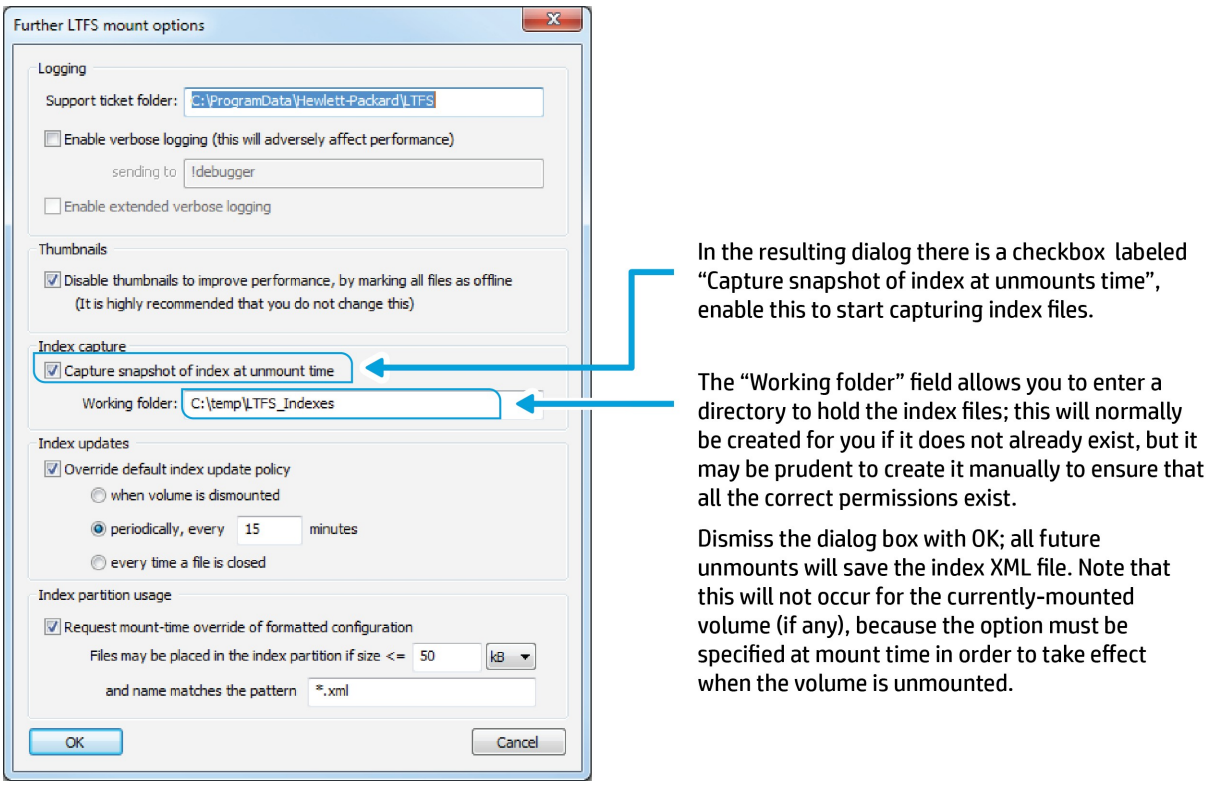



**Figure 5.** Configuring index capture on Windows

### Enabling on Linux

To create a snapshot of the index when the volume is unmounted, you need to add options to the mount (**ltfs**) command line. Enable this feature by including the option:

```
-o capture_index
```

By default the index files will be stored in **/tmp/ltfs/** but this can be changed by including the option:

```
-o work_directory=/home/user1/LTFScontent
```

to use the **LTFScontent** directory in the home directory of **user1**.

### Enabling on Mac

Index capture is managed via the Preferences dialog of the HPE StoreOpen application. Open using **Cmd-**, or from the HPE StoreOpen Standalone menu.

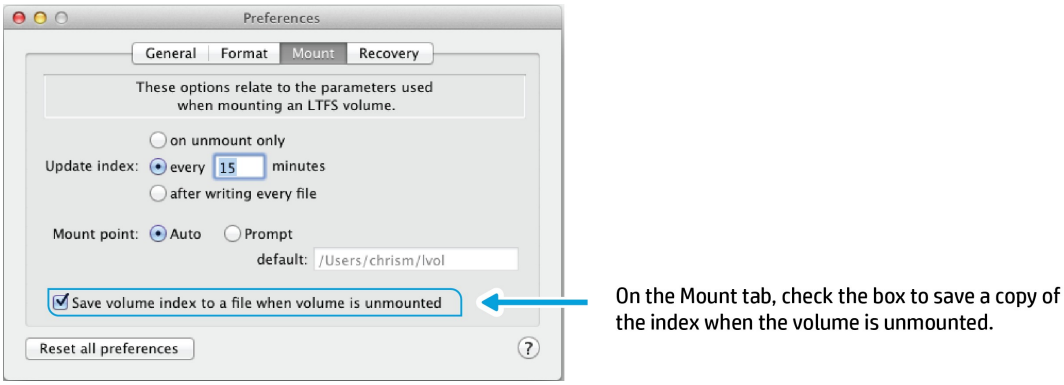



**Figure 6.** Configuring index capture on Mac

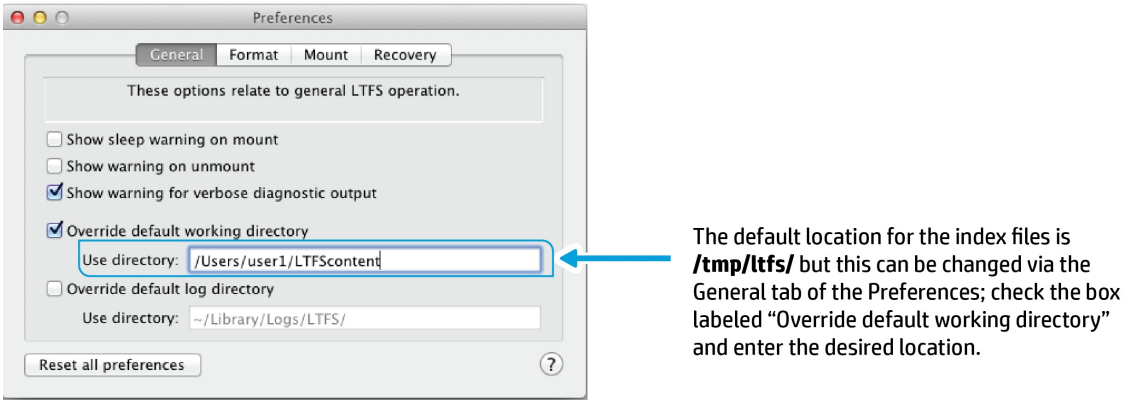



**Figure 7.** Changing working directory on Mac

## Optimizing file retrieval order

When browsing the contents of a directory, the files are typically presented sorted by some criteria—for example filenames in alphabetical order, or date of creation. Whilst this is logical for human interaction, it may lead to delays in retrieving the contents from tape because the perceived order may not match the order in which they are stored in the LTFS volume. And because tape is essentially a linear sequential access device, retrieving files in the "wrong" order may take many times longer than if they were accessed in their stored order. To address this, it is necessary to peek inside the LTFS index and pre-sort the files into the "right" order, rather than allowing the operating system to select its own order.

This can be accomplished by looking at the reserved extended attribute **ltfs.startblock** which records, in the XML index, the logical block number on tape at which the specified file starts. If the list of files to be retrieved is then sorted by logical block number, the operation can take place in a more efficient and timely manner. An alternative is to view/process the XML index file, if that has been captured as described above in "Saving and viewing index files."

### Implementing on Windows

Although HPE StoreOpen for Windows now includes the **ltfsattr** utility, which can read and report the **ltfs.startblock** value, the current implementation forces an open and read from a file in order to read the file's extended attributes. This means that the tape must be positioned to each file, which will actually take nearly as long as actually copying the file. Therefore the recommended solution for Windows at the time of writing is to examine the XML index rather than accessing the extended attribute.

**Implementing on Linux**

The following example defines a bash function which takes a list of files and outputs the LTFS start block and name of each file. Invoking this function with a list of files produces output can then be sorted by start block and the block number removed again, resulting in an ordered list of filenames:

```
function listsblk()
{
    for f in $*
    do
        sb=`getfattr -n user.ltfs.startblock--only-values --absolute-names $f`
        echo -e $sb\\t$f
    done
}
listsblk file1 file2.. filen | sort -n | cut -f2-
```

**Implementing on Mac**

The following example defines a bash function which takes a list of files and outputs the LTFS start block and name of each file. Invoking this function with a list of files produces output can then be sorted by start block and the block number removed again, resulting in an ordered list of filenames:

```
function listsblk()
{
    for f in $*
    do
        sb=`xattr -p ltfs.startblock $f`
        echo -e $sb\\t$f
    done
}
listsblk file1 file2.. filen | sort -n | cut -f2-
```

## Storing data in the index partition

The LTFS format defines a data partition and an index partition, and in normal usage all data is stored in the data partition. However there are some scenarios where it may be useful or required to store data in the index partition, and this is permissible within the format specification. The most likely usage is to store a manifest file of some form, containing details of all the files stored on the volume. By placing it at the start of the index partition it can quickly be accessed after mounting the volume, since the tape drive will always position to the start of the volume when mounting. If the file is stored in the data partition, then there may be a delay of several minutes while the correct position is located.

However whilst the format permits data files to be stored in the index partition, this is only really useful if the number and size of files is kept small; otherwise, the benefit outlined above is lost because the drive will have to seek to find the index itself (which is always written last in the index partition).

Data placement is controlled by two separate but related options, one when the cartridge is formatted and one when the cartridge is actually mounted. At format time you can specify a file pattern to be stored in the index partition, and also the maximum file size to be placed there. The final control is whether you want those values to be changeable at mount time. If they are set to be changeable, then when mounting a volume you can opt to override the values supplied at format time, and can provide a different file pattern/file size specifier. If at format time you decide that the values should not be overridden, then at mount time you will not be able to change them and the format time values will be used.

### Configuring on Windows

When running the HPE StoreOpen Format Wizard, the third screen "Cartridge options" includes an **Advanced** button. Click this to set up the index partition placement rules.

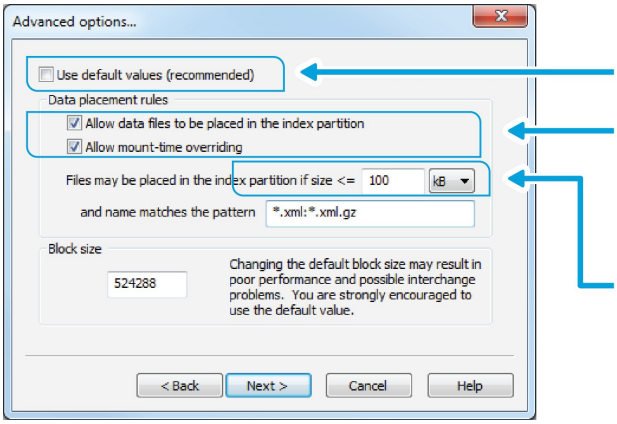



**Figure 8.** Configuring placement rules on Windows

For the example shown, files may be placed in the index partition if their name ends with .xml or .xml.gz and if their size is no more than 100 kB. This will be the default behavior for this volume, but it will be possible to change the rules at mount time since the "Allow mount-time overriding" box is checked.

---

**Note**

As indicated in the Advanced options dialog, it is also possible to change the Block size here; however it is strongly recommended that you do not do this.

---

When preparing to mount a volume using the Configuration utility, click on the Advanced options button. The following dialog appears which reveals further options:

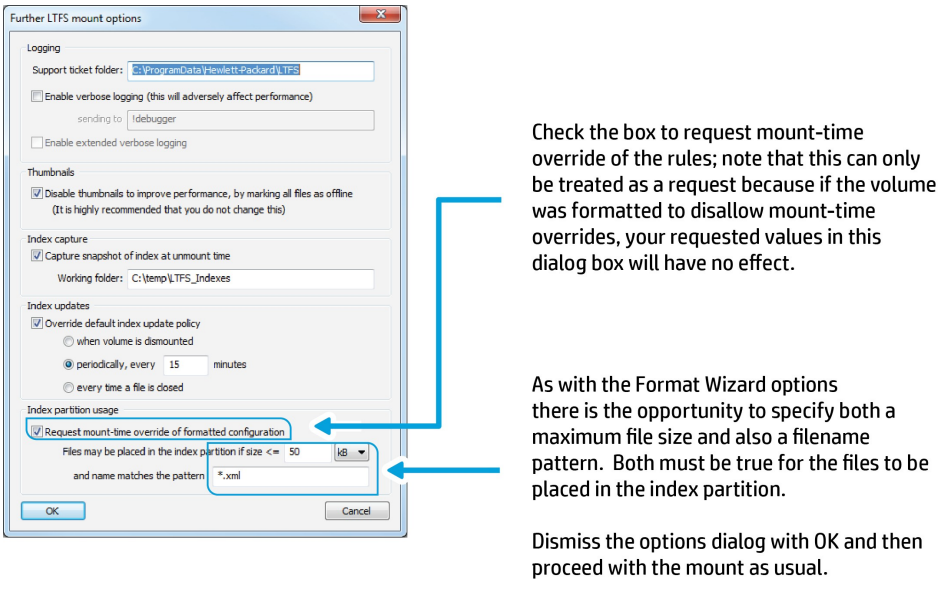



**Figure 9.** Configuring mount-time placement rules on Windows

### Configuring on Linux

For the Linux platform, the rules must be specified on the command line for the format (**mkltfs**) operation and/or the mount (**ltfs**) operation. For **mkltfs**, rules are specified with the **–r** (or **--rules**) option. For example, to specify that files may be placed in the index partition if their name ends with .xml or .xml.gz and if their size is no more than 100 kB, include the option:

```
--rules="size=100K/name=*.xml:*.xml.gz"
```

To format in such a way that mount-time placement cannot be changed, include the option

```
--no-override
```

on the **mkltfs** command line.

When mounting the volume with the **ltfs** command, include the option **–o rules** to request specific placement rules. Whether or not this is successful will depend on the options included at format time. For example, to request that only files ending with .xml up to 50 kB be placed, potentially overriding the format-time options, include the option

```
-o rules="size=50K/name=*.xml"
```

on the ltfs command line.

### Configuring on Mac

The index rules are managed as part of the HPE StoreOpen application preferences. Open the Preferences pane and select the Format tab.

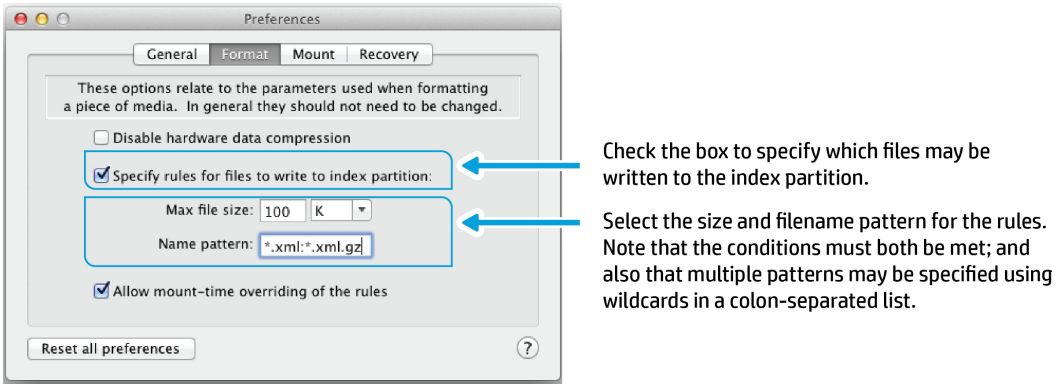
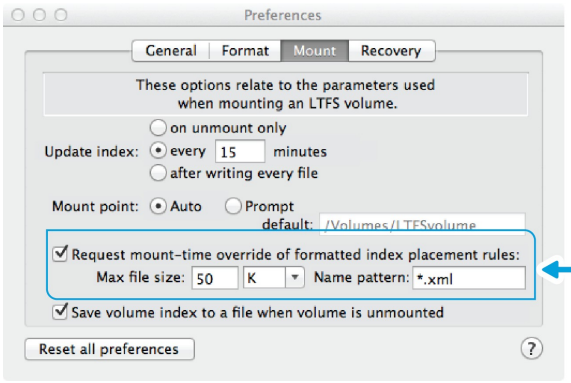



**Figure 10.** Configuring placement rules on Mac

For the example shown, files may be placed in the index partition if their name ends with .xml or .xml.gz and if their size is no more than 100 kB. This will be the default behavior for this volume, but it will be possible to change the rules at mount time since the "Allow mount-time overriding" option is checked.

Mount-time overriding of the placement rules can be requested via the Mount preferences tab.




Check the box to request mount-time override of the rules; note that this can only be treated as a request because if the volume was formatted to disallow mount-time overrides, your requested values in this dialog box will have no effect.

As with the Format preferences there is the opportunity to specify both a maximum file size and also a filename pattern. Both must be true for the files to be placed in the index partition.

**Figure 11.** Configuring mount-time placement rules on Mac

## Dealing with issues

In normal usage, HPE StoreOpen manages the tape contents automatically with no particular user intervention beyond configuring and using the volume. However when unexpected problems arise it may become necessary to take action in order to check the integrity of the volume and potentially to repair a damaged volume. Some of these problematic events include:

- Unexpected loss of power to the tape drive
- Interrupting communications with the drive (for example unplugging the SAS cable)
- Operating System crash or abnormal shutdown

The scale of damage to the volume will depend on what was happening at the time the problem occurred. If the volume had not been modified since mounting, then in most circumstances there will be no damage. If the volume had been modified and a current copy of the index updated on tape, then although the volume may be inconsistent, the HPE StoreOpen software will normally be able to restore consistency and so recover the volume when it is next mounted. The worst cases occur when there is no current copy of the index on tape, or when the drive is actually writing at the time of the unexpected event.

### Mitigation

Some issues can be avoided through careful planning; following these steps will minimize the risk of encountering problems:

- Ensure all connectors are fully seated and secured where possible
- Make sure all cables are out of the way as far as is possible to avoid accidental snagging
- Select the index update option of HPE StoreOpen which best suits your circumstances. Refer to the section “Index Updates” in the Windows User Guide or “Use of sync_type options” in the Linux/Mac User Guide
- Always unmount the volume before powering down the tape drive
- Always unmount the volume before shutting down the computer
- If power outages are not uncommon, consider investing in an uninterruptable power supply (UPS) and use it to power both the computer and the tape drive

### Recovery

Sometimes, despite all precautions, the LTFS volume may become corrupt or inconsistent due to circumstances beyond your control (such as a system crash). The first thing to do is take a deep breath and not panic! In almost every case the content on the volume can be recovered, usually fully, or sometimes up to the last index written to tape. The second step is to ensure that you do not make any further changes to the volume before running the recovery process.

**Important!**
Do not unformat or reformat the volume, as this will irretrievably destroy the existing content. Instead, HPE StoreOpen provides tools to help you recover your data from the corrupted LTFS volume.

To understand the need for the recovery process, consider the following three figures 12, 13, and 14.


**Figure 12.** Normal tape layout

As shown in figure 12, an LTFS volume is comprised of a number of data files interspersed with index records (depending on the selected style of index updates referred to above). HPE StoreOpen will write data files to the tape and periodically write a copy of the latest tape index as well, which describes the content of the tape up to that point. Thus Index **i** contains the information for all data files up to and including Data file **n+1**. Data file **n+2** is then appended, and Index **i+1** is written with information for all data files up to and including Data file **n+2**. At the end of the recorded area on each tape there is a special "End of Data" (EOD) marker, which represents the last accessible location on the media. The tape drive writes an EOD marker whenever it has finished writing, and also after a short timeout if the host computer stops sending write commands for a while. It is overwritten when further data is appended and is normally completely transparent to the user.

Figure 13 shows the tape layout that may result if the drive loses power or contact with the host computer after writing more data but before writing the updated index:


**Figure 13.** Missing index

In this instance, even though all of the data for Data file **n+2** has been written to tape successfully, none of the corresponding metadata information has been saved; consequently it is not possible to determine any information about the additional data blocks beyond Index **i**. The volume can be repaired but will erase Data file **n+2**, so that the volume ends with Index **i**. Note that it is sometimes possible to save the data blocks from this erased file, but because there is no information about it, manual identification will be required to identify and recover any data from the file(s) involved. This is not usually practical.

The third case is shown in figure 14, which illustrates what may happen if the drive loses power whilst actually writing data:


**Figure 14.** Missing EOD

This is the "worst-case" because not only is the LTFS format corrupted, but the underlying LTO tape format is also incomplete, which means that the tape drive will not be able to recognize the end of the recorded data. However it is still recoverable; a special "deep recovery" mode enables HPE StoreOpen to recognize the problem and in most cases it will be able to write an EOD immediately after Index **i**. The volume is then in the same state as after recovering from the situation shown in figure 13.

The same general approach to recovery applies to all supported operating systems, though the specific details depend on the platform. A brief outline is given here, but more information may be found in the User Guide documentation mentioned in References on page 25.

### Recovery on Windows

Volume checking and recovery is handled by the LTFS Check Wizard application, installed as part of the HPE StoreOpen package. This wizard will guide you through the process step-by-step; note the following:

1. Start by opening the Configuration utility and remove any existing drive mapping. Depending on what has happened in the past, it might be necessary to reboot before this can be done as Windows may have some stale handles to the tape drive which will prevent correct operation.
2. Run the check wizard, either from the "Cartridge utilities" found on the Configuration dialog window or from the Start menu.
3. Follow the prompts, making sure you read them carefully. Select the option for "Detailed progress information" if desired; then on the next screen choose the option "Check and repair volume" (as shown in figure 15). If you think that the scenario shown in figure 14 may apply, then check the box "Attempt deep recovery." (It is not harmful to select it regardless, but the recovery process will usually take longer.) You can also indicate here if you want HPE StoreOpen to try to save any unidentified data blocks.

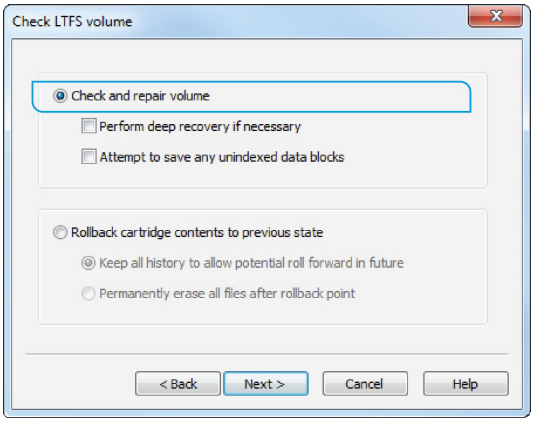



**Figure 15.** Windows CheckWizard options

4. After a summary screen, the Finish button will start the check/recovery operation which may take several minutes to complete.
5. The outcome will be displayed in the progress window; check this to ensure that the recovery was successful. Note that some error messages may be displayed as a result of the issues with the volume, but these can be ignored if the "success" messages are displayed as shown below in figure 16.

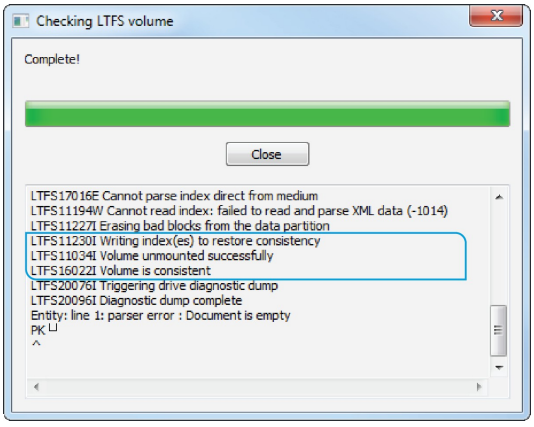



**Figure 16.** Windows CheckWizard outcome

6. Exit the wizard and re-start the Configuration utility to re-establish the mapping between tape drive and Windows drive letter. Verify that access to the content has been restored.

### Recovery on Linux

Volume checking and recovery is handled by the **ltfsck** command, installed as part of the HPE StoreOpen package. Provide appropriate command line parameters to configure the operation as described in the following:

1. Ensure the volume is not mounted and that no **ltfs** process is running.
2. Many of the options to **ltfsck** relate to the rollback of a volume and are not relevant for recovery. The minimum invocation (for example if the tape drive is located at **/dev/nst0**) would be:

   ```
   ltfsck /dev/nst0
   ```

3. If you think that the scenario shown above in figure 14 may apply, then include the option:

   ```
   -z  (or --deep-recovery)
   ```

   It is not harmful to include it regardless, but the recovery process will usually take longer.

4. If you want HPE StoreOpen to try to save any unidentified data blocks then include the option:

   ```
   -f  (or --full-recovery)
   ```

5. The command will start the check/recovery operation which may take several minutes to complete.
6. The outcome will be displayed in the terminal window; check this to ensure that the recovery was successful.
7. Re-mount the volume and verify that access to the content has been restored.

### Recovery on Mac

The HPE StoreOpen GUI, installed as part of the HPE StoreOpen package, includes a “Recovery” tab.

1. Ensure the volume is not mounted.
2. Open the application Preferences pane and select the Recovery tab (figure 17).

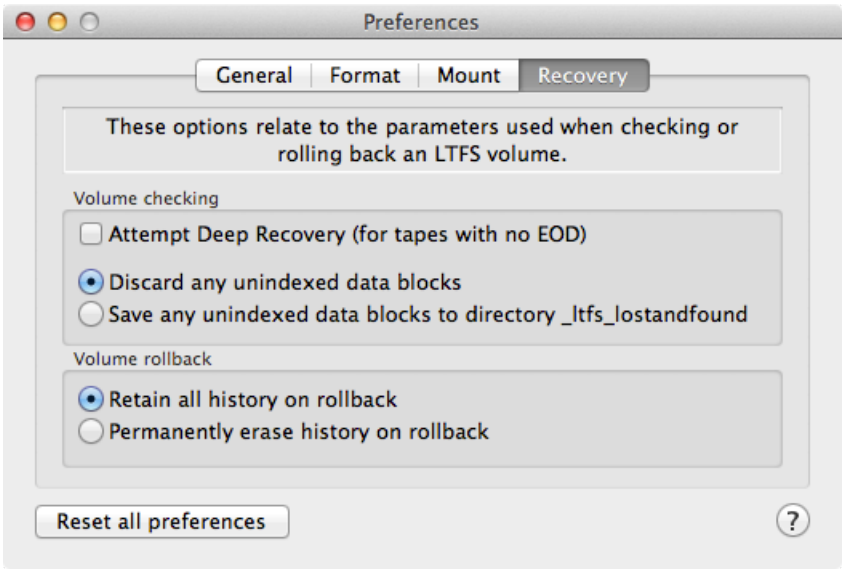



**Figure 17.** Mac Recovery preferences

3. If you think that the scenario shown above in figure 14 may apply, then check the box labeled “Attempt Deep Recovery.” It is not harmful to select it regardless, but the recovery process will usually take longer.
4. If you want HPE StoreOpen to try to save any unidentified data blocks then select the option labeled “Save any unindexed blocks to directory _ltfs_lostandfound.”
5. Close the Preferences pane and return to the main StoreOpen window. Click on the Recovery tab then on the button labeled “Check Volume.”

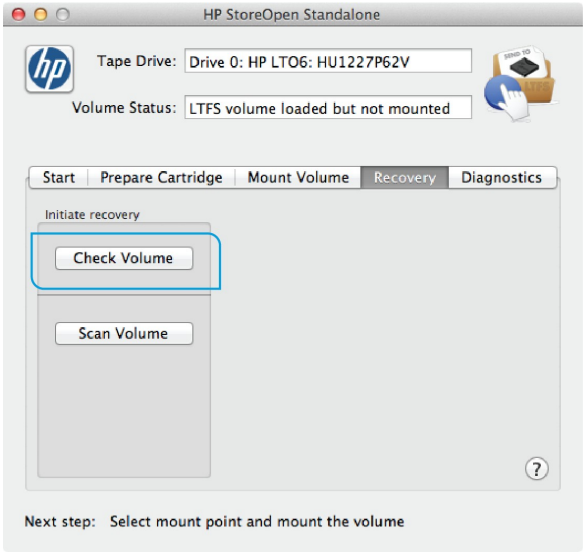



**Figure 18.** Mac Recovery

6. Carefully read the resulting popup dialog box which summarizes what is about to happen. If you are ready to proceed then answer “Yes” to that dialog and the recovery process will begin.
7. During the recovery process you can monitor progress by switching to the Diagnostics tab. Note that some error messages may be displayed in the diagnostic output as a result of the issues with the volume, but these can be ignored if the operation completes successfully. The Volume Status will be updated to indicate completion as shown below in figure 19.

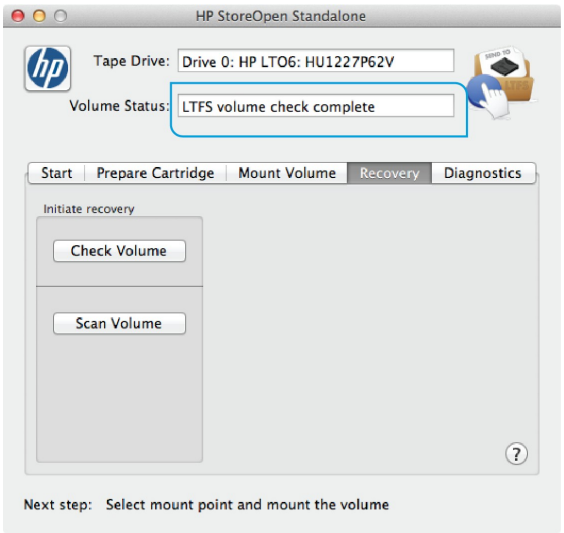



**Figure 19.** Completed Mac Recovery

## References

- The HPE StoreOpen home page can be found at hp.com/go/storeopen and includes links to download the freely available software
- The StoreOpen pages also include a set of User Guide documents for both Standalone and Automation versions; these describe the steps needed to download, install, and configure HPE StoreOpen
- For details of all the compatibility and interoperability of HPE's tape automation solutions, including StoreOpen and other LTFS implementations, refer to the HPE Data Agile BURA Compatibility Matrix
- See snia.org/ltfs for the latest news on the LTFS format specification
- The LTO program website has information about LTFS, including a list of vendors who are now supporting it, available at lto.org/technology/ltfs/
- The LTO program has also launched an LTFS Compliance Testing process. This process is intended for solution and hardware providers to verify that an LTFS offering produces a properly formatted volume that complies with the open LTFS format specification on an LTO cartridge. This gives customers confidence that the technology will integrate and interoperate within a heterogeneous environment. More details and a list of tested LTFS solutions can be found at lto.org/technology/ltfs/ltfs-compliance-verification/

## Learn more at

hp.com/go/storeopen



4AA5-1230ENW, December 2015, Rev. 3