Open Tape for Open Compute

Recent open-source software for easier use of tape

Slavisa Sarafijanovic, IBM Research Zurich

Open Compute Project (OCP) 2019 Symposium, 26-27. Sep. 2019, Amsterdam

Goal

Make tape easy to use!

- Develop software that deals with the complexities of tape so that the user doesn’t need to
  - Ideally, make tape transparent to the user
  - But, without giving up flexibility
- Provide software components at multiple levels of the stack
  - Can be used as components of a custom solution
  - Main intended uses: **file-on-tape and object-on-tape**
- Contribute the software components to the community as **open-source** projects

Content

Tape HW and SW intro/overview

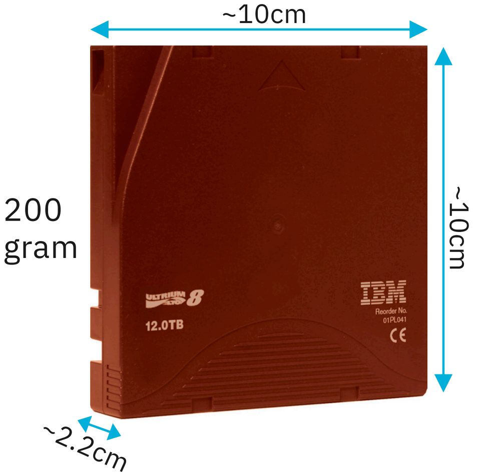
LTFS DM: open-source software for file on tape

SwiftHLM: open-source software for object on tape

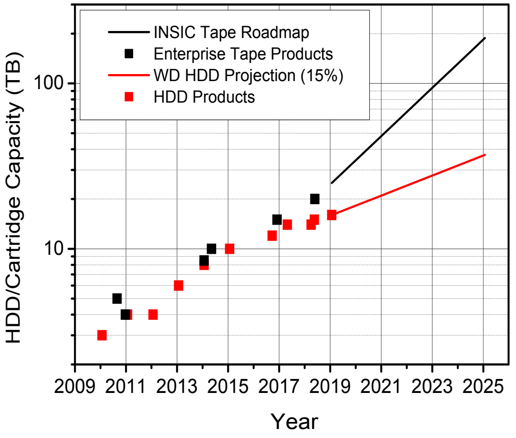
Summary and Outlook

Tape Cartridge

- **12TB LTO 8 cartridge** available since Nov. 2017
  - 30TB @ 2.5x (typical) drive compression
- **Cost** Tape vs HDD: 1/3.7 in 2015, 1/7 2020 est. [1]
- **40% CAGR to 2029** per INSIC Tape Roadmap [2]
- Write area: 960m long x ~2cm wide (~5.6µm thick)
- 2017 lab demo: 201 Gb/in^2 ➔ 330 TB cartridge
- Need a tape drive to write/read tape



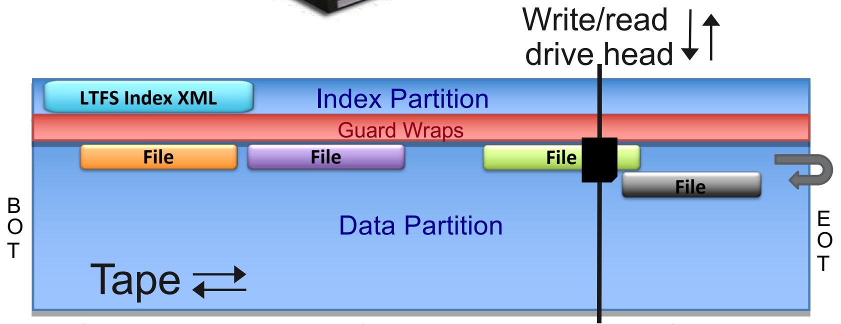
Tape and HDD Capacity Roadmaps [2,3]



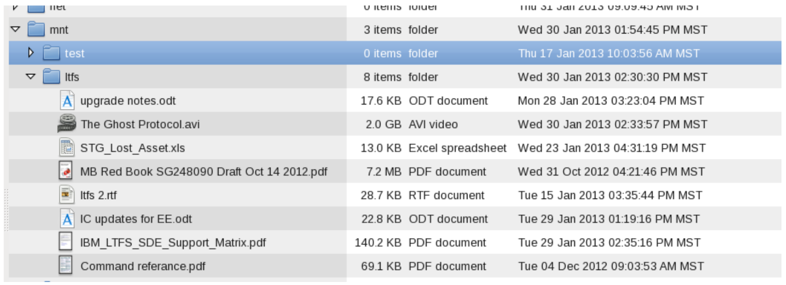
References:
[1] Tape Cost vs HDD at large scale (source MS Azure): https://tapepower.fujifilmrmd.com/LA2015/video/id/presentation.5
[2] Tape Scaling: INSIC Roadmap: http://www.insic.org
[3] HDD Scaling:
https://www.wdc.com/about-wd/newsroom/press-room/2017-10-11-western-digital-unveils-next-generation-technology-to-preserve-and-access-the-next-decade-of-big-data.html
https://www.extremetech.com/computing/283200-western-digital-plans-first-16gb-mamr-hdds-in-2019

Tape Drive & LTFS SDE* (Single Drive Edition)

- **Tape drive**
  - Mount/seek/write/read/unmount tape
  - Serpentine recording of data blocks to tape wraps
  - Exposed externally as a block storage device
- **File system on tape**
  - LTFS (Linear Tape File System): the open standard format for a file system on tape
    - LTFS Index stored on the tape
  - IBM LTFS SDE* (Single Drive Addition): the first LTFS implementation
    - Available since 2010, open-source and free
    - POSIX file interface to user/application, but with tape access latency
    - Works on Linux, Windows, MacOS



LTFS Index makes the content self-describing and the cartridge self-contained



net	0 items	folder	Thu 31 Jan 2013 09:09:45 AM MST
mnt	3 items	folder	Wed 30 Jan 2013 01:54:45 PM MST
test	0 items	folder	Thu 17 Jan 2013 10:03:56 AM MST
ltfs	8 items	folder	Wed 30 Jan 2013 02:30:30 PM MST
upgrade notes.odt	17.6 KB	ODT document	Mon 28 Jan 2013 03:23:04 PM MST
The Ghost Protocol.avi	2.0 GB	AVI video	Wed 30 Jan 2013 02:33:57 PM MST
STG_Lost_Asset.xls	13.0 KB	Excel spreadsheet	Wed 23 Jan 2013 04:31:19 PM MST
MB Red Book SG248090 Draft Oct 14 2012.pdf	7.2 MB	PDF document	Wed 31 Oct 2012 04:21:46 PM MST
ltfs 2.rtf	28.7 KB	RTF document	Tue 15 Jan 2013 03:35:44 PM MST
IC updates for EE.odt	22.8 KB	ODT document	Tue 29 Jan 2013 01:19:16 PM MST
IBM_LTFS_SDE_Support_Matrix.pdf	140.2 KB	PDF document	Tue 29 Jan 2013 02:35:16 PM MST
Command referance.pdf	69.1 KB	PDF document	Tue 04 Dec 2012 09:03:53 AM MST

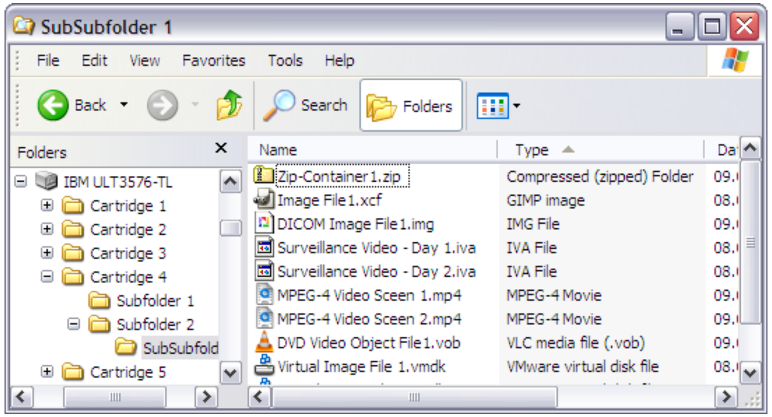
With LTFS SDE* a tape content can be mounted as a file system and accessed using the standard POSIX file interface, e.g. via an OS CLI or from a file browser

* IBM LTFS SDE was lately renamed to IBM Spectrum Archive SDE

Tape Library & LTFS LE* (Library Edition)

- **Tape library**
  - – Multiple connected storage frames, each with up to 16 shared tape drives and up to 100s of shared tape cartridges
  - – Up to 351 PB of raw storage capacity**
- **IBM LTFS LE***
  - – Mounts the resources of a tape library (or its partition) under a file system mount point
  - – Each cartridge as a subdirectory
  - – Available since 2011, currently a free software with a programming API
    - POSIX file system interface
    - API to mount/unmount/format tapes

IBM TS4500 Tape library



- IBM ULT3576-TL
  - Cartridge 1
  - Cartridge 2
  - Cartridge 3
  - Cartridge 4
    - Subfolder 1
    - Subfolder 2
      - SubSubfold
  - Cartridge 5

Name	Type	Da
Zip-Container1.zip	Compressed (zipped) Folder	09.(
Image File1.xcf	GIMP image	08.(
DICOM Image File1.img	IMG File	09.(
Surveillance Video - Day 1.iva	IVA File	08.(
Surveillance Video - Day 2.iva	IVA File	08.(
MPEG-4 Video Sceen 1.mp4	MPEG-4 Movie	09.(
MPEG-4 Video Sceen 2.mp4	MPEG-4 Movie	09.(
DVD Video Object File1.vob	VLC media file (.vob)	09.(
Virtual Image File 1.vmdk	VMware virtual disk file	08.(

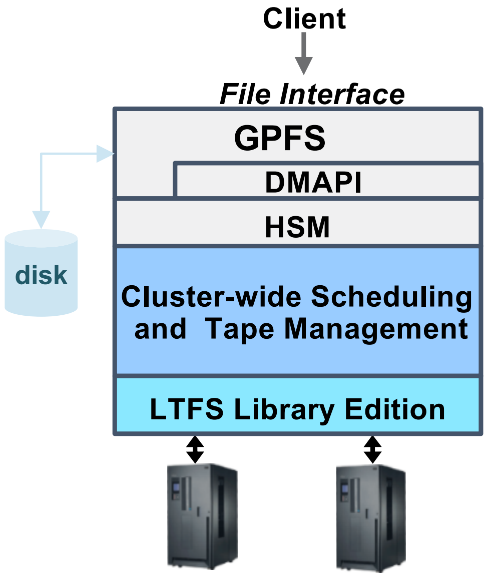
With IBM LTFS LE* each cartridge of a tape library becomes accessible as a folder under the filesystem mountpoint used for the tape library

* IBM LTFS LE was lately renamed to IBM Spectrum Archive LE

** Assuming IBM TS1160 drive and 20TB cartridge (available since Nov 2018)

LTFS EE* (Enterprise Edition)

- Integrates tape with IBM's GPFS** distributed disk file system, available since 2013
- Organizes tape storage into tape pools
- User/application access files via the original GPFS namespace
- Main function: Transparent or explicit migration/recall of disk file data to/from the tape polls
- Additional tape functions: reconcile/export/import/reclaim
- Key new component: queuing and scheduling tape access requests and managing tape resources



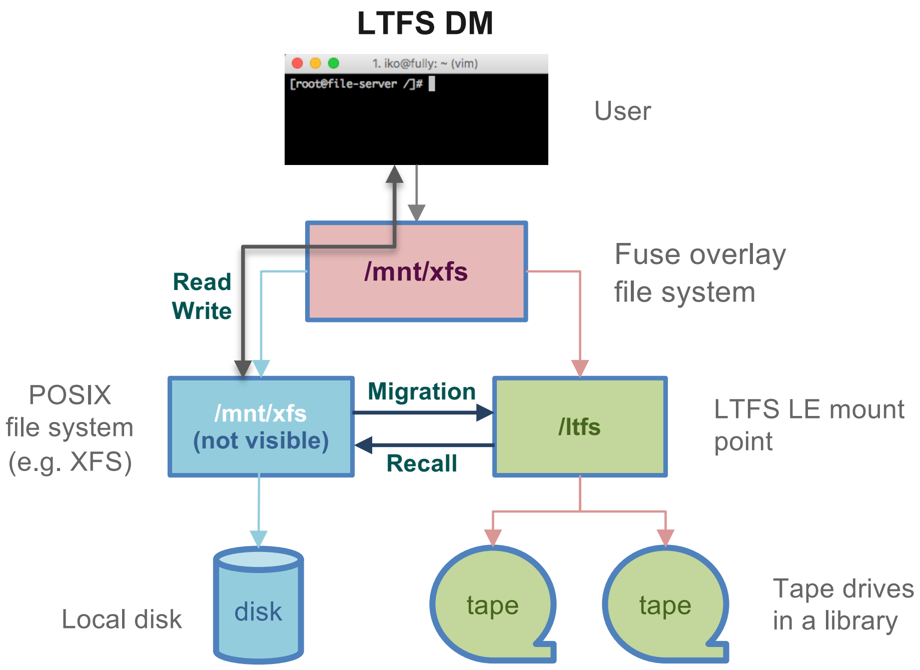
IBM LTFS EE* preserves the global GPFS cluster namespace and moves disk file data to and from tape pools in the tape libraries.
DMAPI = Data Management API
HSM = IBM's implementation of DM

* IBM LTFS EE was lately renamed to IBM Spectrum Archive EE
** IBM's GPFS (General Parallel File System) was lately renamed to IBM Spectrum Scale

LTFS Data Management (LTFS DM)

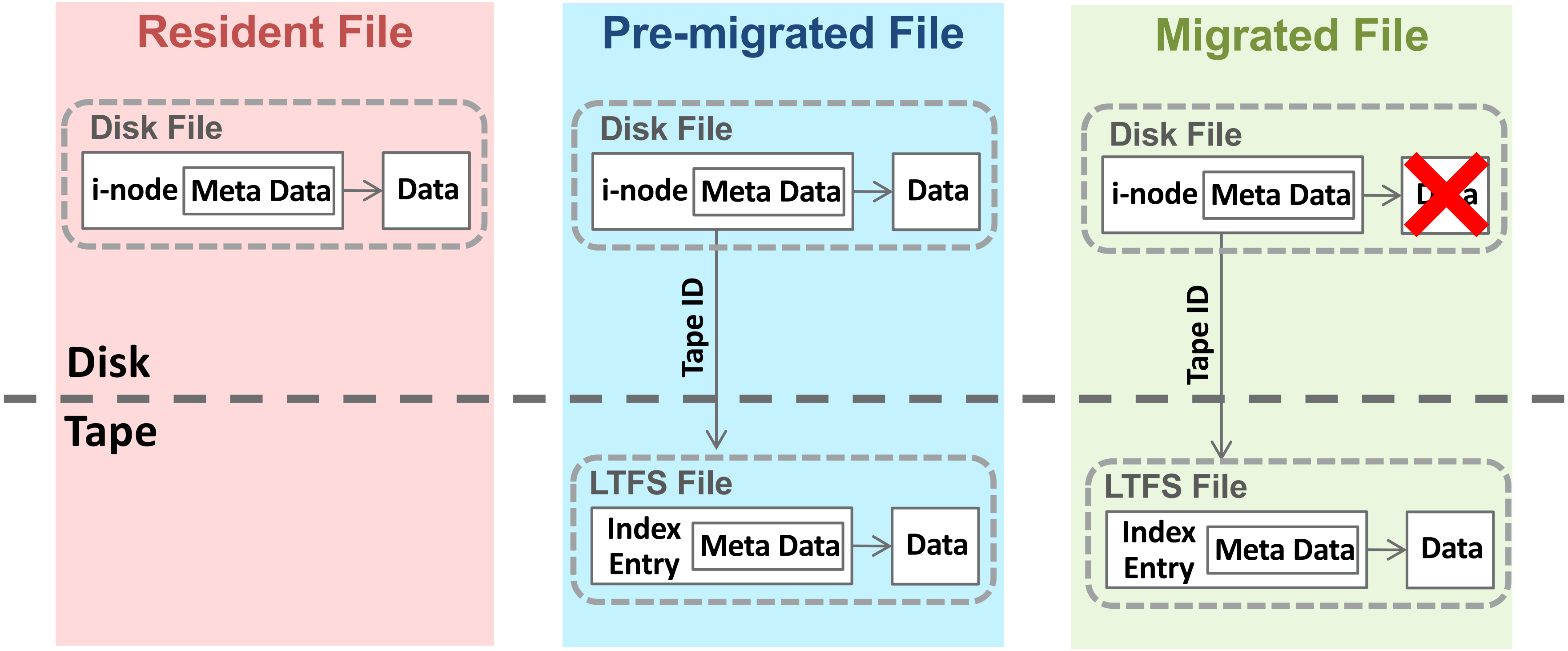
- LTFS DM converts a single-node disk file system into a disk and tape file system
  - Seamless integration of tape into open Linux filesystems such as xfs, ext3, ext4
  - Hides tape complexity from the user
- The original disk name space is exposed to the users and applications
- LTFS DM main function:
  - (Pre)migrate file data from disk to tape
  - Explicit data recall from tape to disk
  - Transparent data recall from tape to disk upon user access of a migrated file
- LTFS-DM is open-source since end April 2018

Beta https://github.com/ibm-research/LTFS-Data-Management



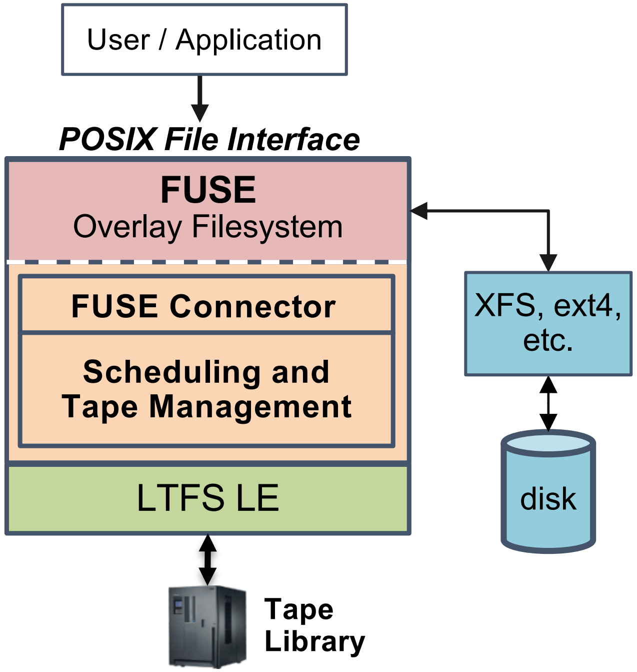
```
[root@file-server /]#
```

File Migration States

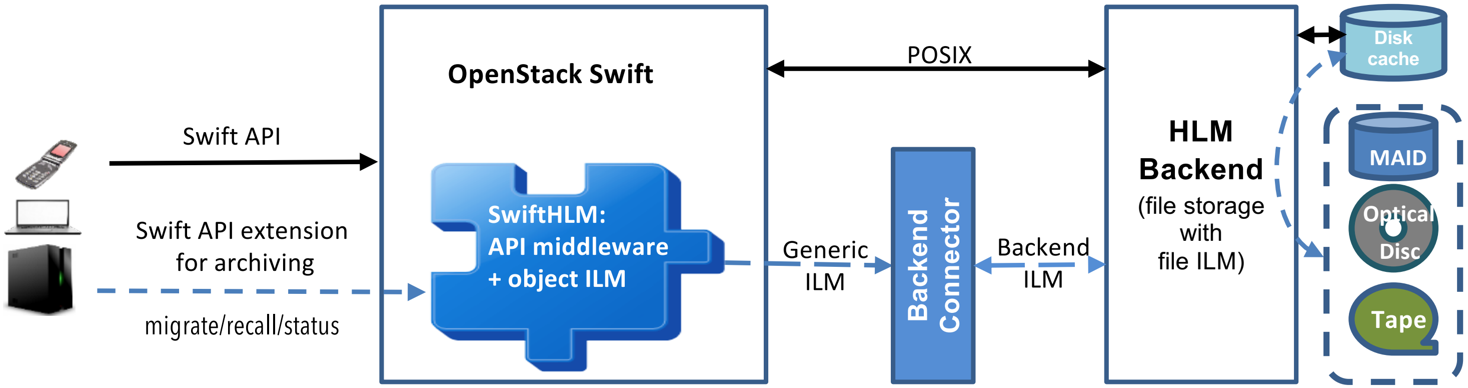


LTFS DM Architecture and Additional Function

- LTFS DM uses FUSE to manage file access and data movement during migration and recall
- Scheduling and Tape Management is the core component of LTFS DM:
  - Queuing and optimized managing of migration and recall operations
  - Managing tape resources
- Additional functions:
  - Tape Storage Virtualization: create tape pools consisting of multiple cartridges
  - Replication: one or multiple copies of a file data can be stored in different tape pools
  - Collocation: store files with logical similarity on the same cartridges / pools



Swift High Latency Media (Swift HLM)

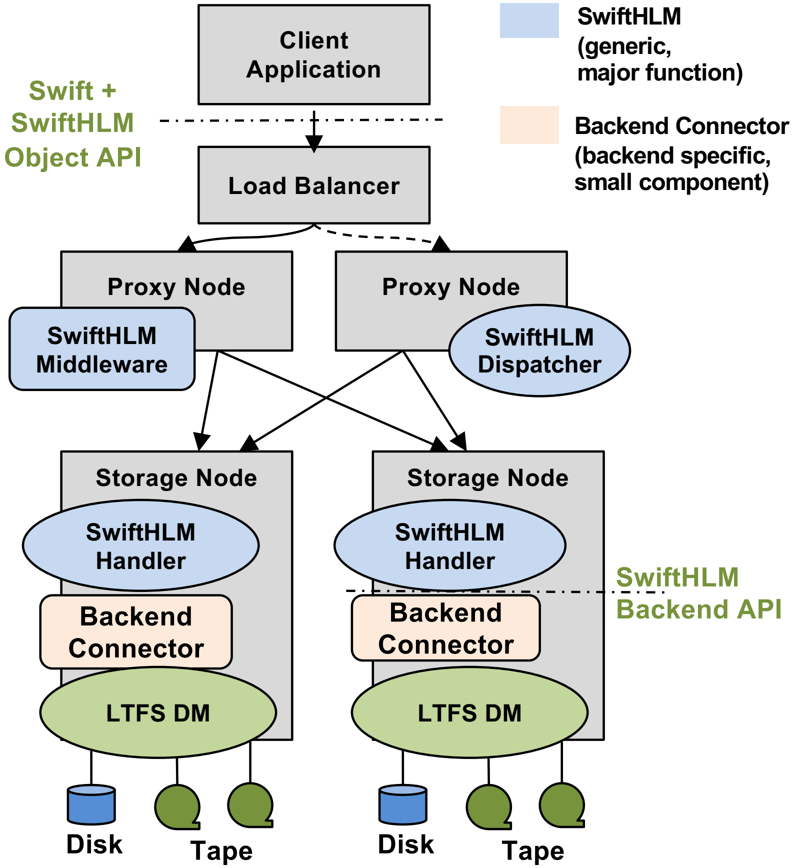


- Swift HLM:
  - Extends Swift API with HLM operations*: MIGRATE, RECALL, STATUS
  - Maps/distributes Swift API requests to backend requests across storage nodes and replicas
  - Supports Replicated or Erasure Coded object on tape
  - Open source since May 2017: https://github.com/ibm-research/swifthlm *
- Backend Connector:
  - Maps Swift HLM generic backend interface (GBI) operations to specific backend ILM operations
  - Connectors are available for: LTFS DM*, Spectrum Archive**, Spectrum Protect**

* Swift HLM is an Open Stack Swift associated project: https://docs.openstack.org/swift/latest/associated_projects.html#alternative-api
** Connector for LTFS DM is available within the main Swift HLM repository https://github.com/ibm-research/swifthlm
*** IBM Redbook for use with Spectrum Archive or Spectrum Protect, as a trial version, available at: http://www.redbooks.ibm.com/abstracts/redp5430.html

Swift HLM Architecture

- SwiftHLM middleware:
  - Implements the archiving API operations
  - Queues object ILM requests by storing them in a special container
- Dispatcher:
  - Distributes Object ILM requests across relevant Swift storage nodes
- Handler:
  - Determines the file that stores a full copy or an erasure coded (EC) part of the object
  - Invokes the backend ILM operations on the file
- Backend Connector:
  - Translates generic backend ILM request to backend-specific ILM requests



Summary and Outlook

- We provided open-source software for tape integration with file and object storage
- For Swift HLM we consider creating a variant that supports invoking object HLM operations via extended attributes (EAs), aimed at:
  - Use with backends that can intercept and act upon EAs, e.g. LTFS DM can be enhanced for that
  - Supporting archiving operations for both the Swift and S3 APIs of Swift
  - Simplify internal object HLM processing