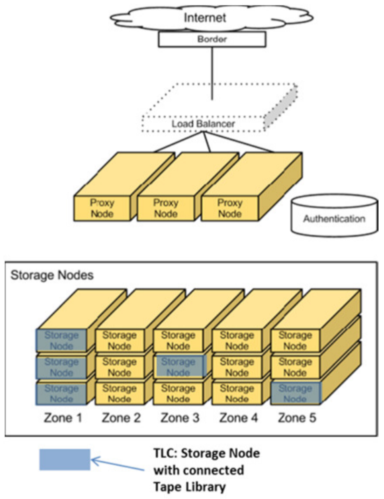
OpenStack SWIFT – Tape Library Connector (TLC)

Object Storage with SWIFT TLC

/// Why do we need SWIFT TLC?

- Long term data storage requirements continue to grow rapidly
- OpenStack is quickly becoming the standard open-source cloud computing solution
- SWIFT is OpenStack’s highly scalable object storage system to store large amounts of unstructured data
- Tape Libraries provide the best value for long-term data storage
- SWIFT TLC makes tape look like a standard disk storage node

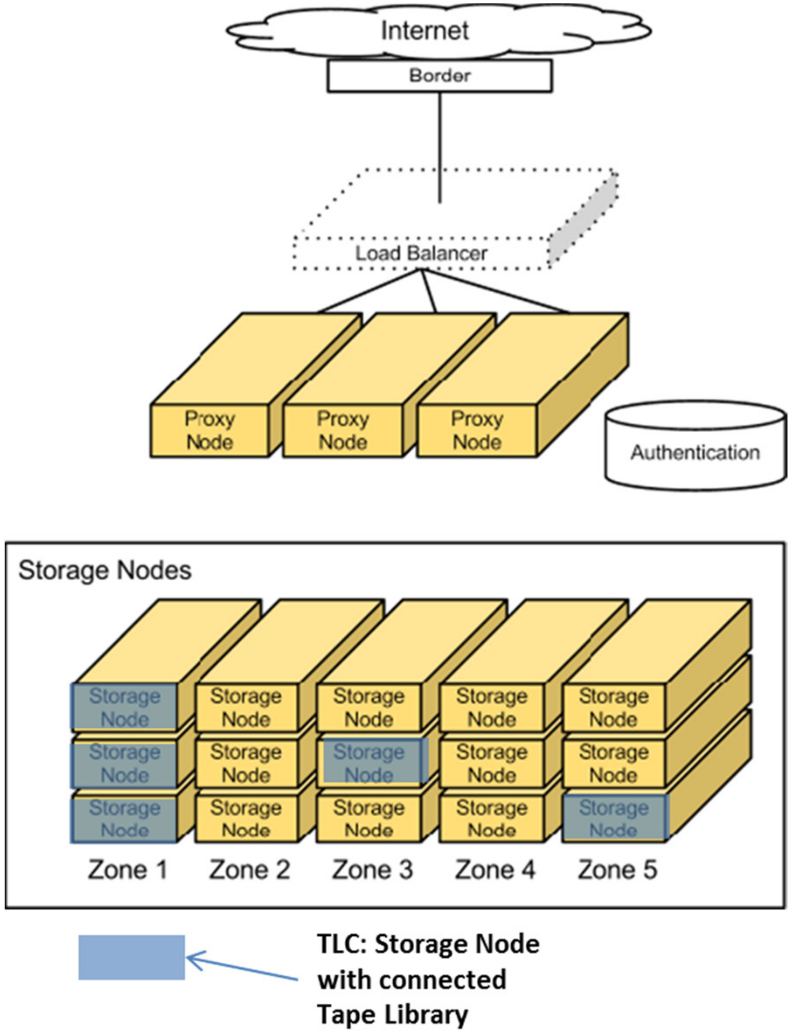


➢SWIFT TLC enables simple, seamless, transparent tape library integration into an OpenStack SWIFT environment

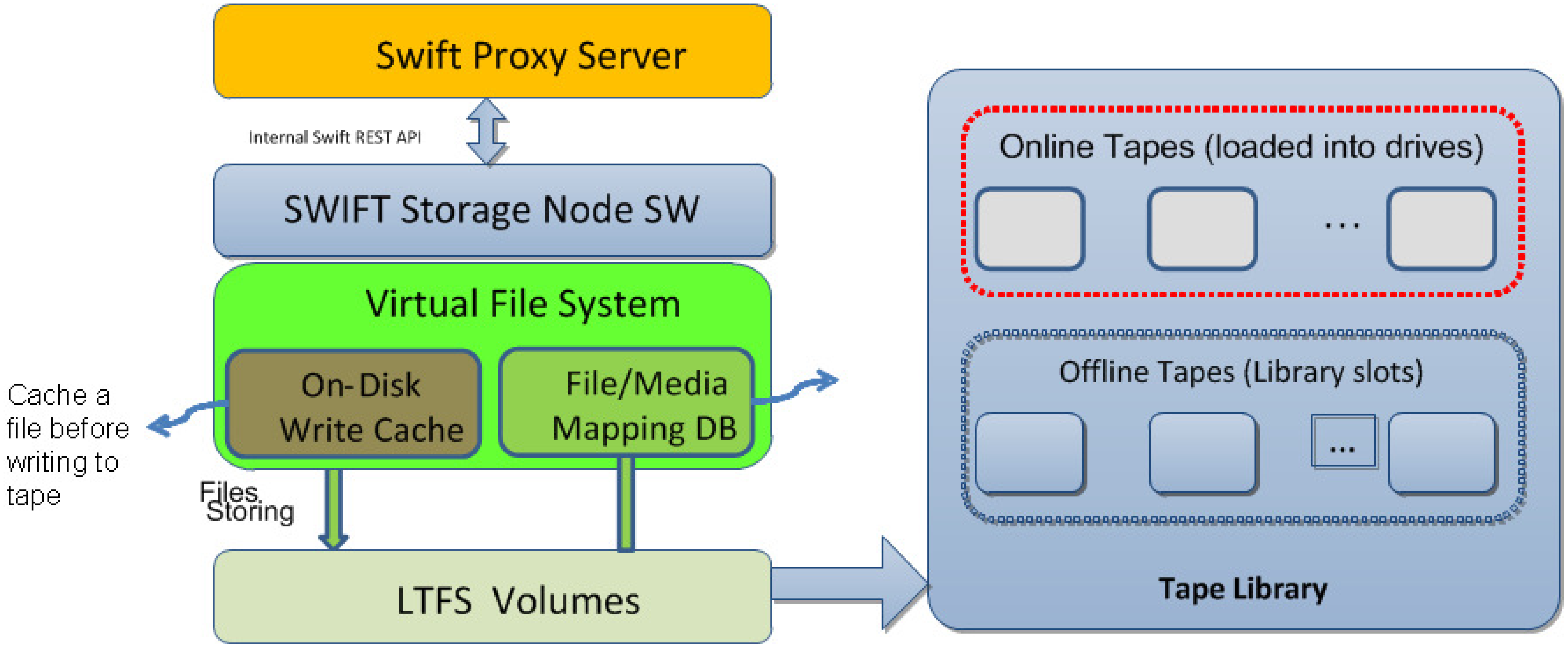
/// Overview – SWIFT Tape Library Connector (TLC)

SWIFT Object Storage TLC:

- Connects a Tape Library as a standard Storage Node into an OpenStack SWIFT Object Storage environment
- Integrates the library seamlessly into a standard SWIFT environment with no modification to core SWIFT code
- Supports standard SWIFT Object Storage functionality and operations (Upload, Download, Delete, Replication, Auditing, Recovery, Versioning, Multi Nodes, Zoning)
- Powerful Tape-specific Auditing Handler to support SWIFT auditing for Tape
- Uses LTFS to store objects on tape



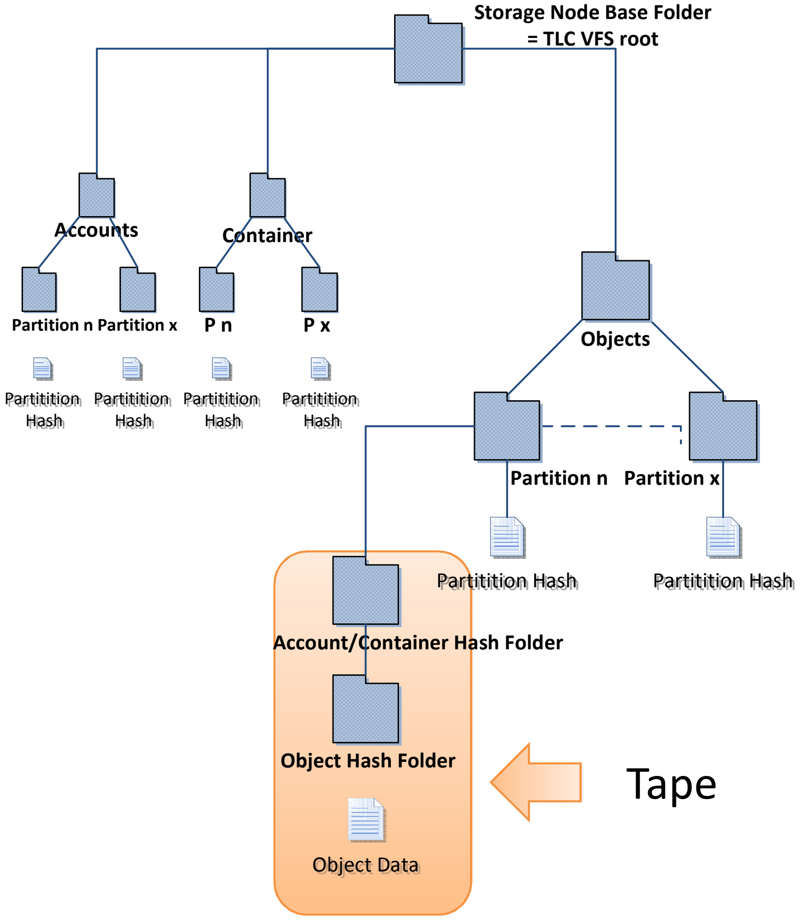
/// Overview – TLC Architecture



/// Overview – TLC Virtual File System (VFS)

TLC VFS:

- Based on FUSE to span the storage capacity of tapes to one large connected file system
- The VFS root folder becomes the mount point for a standard SWIFT Storage Node (exactly like a disk based Storage Node)
- Uses LTFS to store Object data (files) directly on tape in a open format
- Has a transparent, disk based, fast and scalable Data Cache to buffer files before they are written to tape and supporting fast read access
- Stores all VFS file metadata on a disk based Metadata Cache
- Keeps all SWIFT hashes, accounts and containers on disk



/// Cofigure SWIFT to use TLC

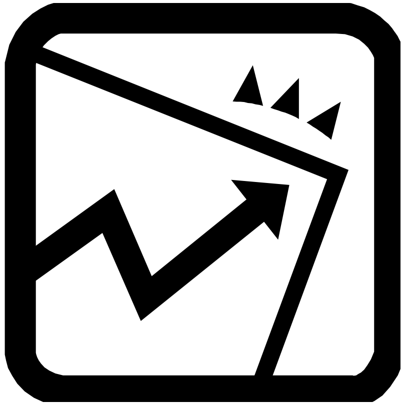
Modification	File	Comment(Purpose)
increase time out for proxy server [app:proxy-server] node_timeout = 300	/etc/swift/proxy-server.conf	Allow longer read response times for the SWIFT nodes
Add line "vs_cache_dir = /srv/vs_cache"	TLC node: object-server.conf	Set the data cache folder for TLC customized SWIFT auditor handler.

Update entry in
[app:object-server]
from
use = egg:swift#object
to
use = egg:vs_auditor#vs_object
	TLC node: object-server.conf	Register the TLC Auditor to be used for SWIFT, Python egg vs_auditor will be installed by TLC auditor installer

Patch standard SWIFT object	/usr/lib64/python2.6/site-packages/swift/obj/auditor.py	Patch this SWIFT auditor source file to use TLC Tape Auditor for TLC nodes. Done by TLC Auditor installer script
Add TLC node to SWIFT ring e.g. "swift-ring-builder object.builder add r1z3-172.16.56.5:6000/vsnode 7000"	Proxy Server: /etc/swift/object.builder /etc/swift/container.builder /etc/swift/account.builder	Connect the TLC VFS (TLC:/srv/node/vsnode) to SWIFT

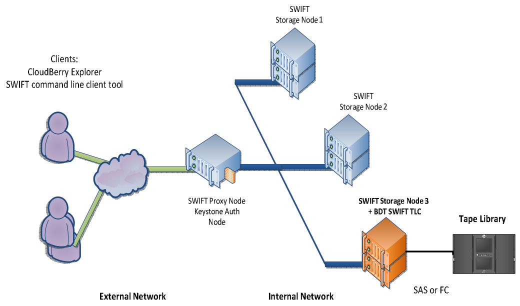
/// SWIFT TLC Value

- Extremely simple integration of Tape Libraries into SWIFT environments
  - Simple installer
  - Very lean architecture
  - no knowledge about SCSI and Tape libraries necessary
  - No modification on OpenStack SWIFT necessary
  - No proprietary functionality, uses all default functionality from SWIFT (auditing, replication, load balancing, recovery, zoning, …)
- Good vertical scale out support – just expand tape library to extend capacity or add tape drives to tape library to increase performance
- Good horizontal scale out support – just add more TLC storage nodes to SWIFT environment
- Fits well into existing mixed environments based on disk based Storage Nodes
- Support for all types of tape libraries, vendor independent, no limitation in supported capacity or drive quantity, support for SAS and FC as well



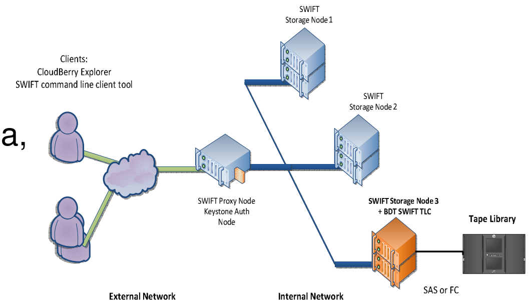
/// SWIFT TLC Evaluation Package

- BDT SWIFT TLC Installer Package
  - Installs TLC software on existing SWIFT Storage node
  - Configures SWIFT to use TLC
- BDT Test Scripts
  - Scripts which could be used for intensive testing
- Tool to demonstrate tape specific SWIFT file auditing and recovery
- SWIFT TLC Installation Guide
- SWIFT TLC Overview Presentation



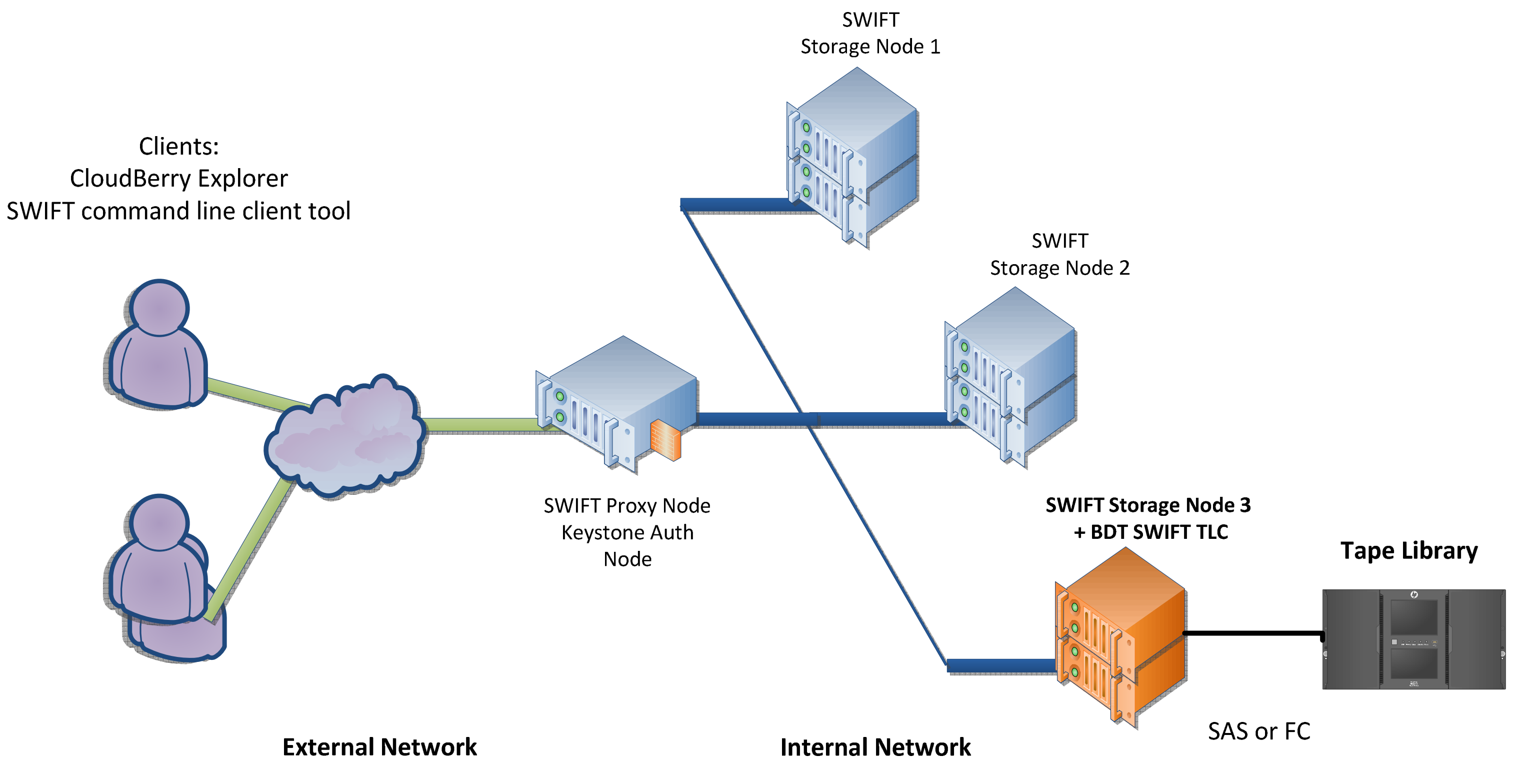
/// SWIFT TLC Evaluation Setup

- Setup default OpenStack SWIFT environment with 1 Proxy Server and 3 Storage Nodes, configure SWIFT and verify functionality, use CentOS 7 as Server OS
- Add SAS HBA (e.g. LSI 6Gb SAS 9205-8e) with LSI chip and Tape Support (no RAID disk controller) to Storage Node 3 and attach Tape Library, verify that Tape Library and Tape Drives will be detected from OS (lsscsi –g)
- Possible Hardware Configuration for TLC server:
  - Data Cache: RAID0 1-2TB (SSD recommended)
  - Metadata: RAID5 50-100GB (SSD recommended)
  - Memory: 16GB
  - CPU: Quad-Core (e.g. XEON X5)
  - Tape library: tape library from HP, IBM, DELL, Fujitsu, Overland, Spectra Logic, actidata, BDT
  - Tape drives: 1-2
- Create Storage Volumes for Data Cache and Metadata Cache and mount to this folder
  - Data Cache: /opt/VS/vsCache/diskCache
  - Metadata: /opt/VS/vsCache/meta
- Install BDT TLC software and SWIFT service using installer (refer SWIFT_TLC_Quickstart document)
  - Installer will install everything automatically and configures the SWIFT Storage Node configuration accordingly
- Add TLC VFS root folder (/srv/node/vsnode) to SWIFT ring configuration on Proxy Server and distribute modified SWIFT Ring configuration files to all Storage Nodes



Detailed Information available in BDT SWIFT_TLC _QuickStart.pdf

/// SWIFT TLC Evaluation Setup Overview



- Default OpenStack SWIFT environment (ideally based on CentOS 7)
  - 1 OS Controller Node with Keystone and SWIFT Proxy Server running (VM possible)
  - 2 default SWIFT Storage Nodes (disk based) (VM possible)
- Default SWIFT Storage Node + BDT SWIFT Tape Library Connector (TLC) with attached Tape Library (MultiStak or FlexStor) (needs to be a physical server), OS: CentOS7