

# Active Archive Implementation Guide with IBM Spectrum Scale Object and IBM Spectrum Archive

Larry Coyne

Joe Dain

Khanh Ngo

Aaron Palazzolo



**Storage**





International Technical Support Organization

**Active Archive Implementation Guide with IBM  
Spectrum Scale Object and IBM Spectrum Archive**

March 2016

**Note:** Before using this information and the product it supports, read the information in “Notices” on page v.

**First Edition (March 2016)**

This edition applies to Version 4, Release 2 of IBM Spectrum Scale and Version 1 of IBM Spectrum Archive Enterprise Edition

© Copyright International Business Machines Corporation 2016. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

# Contents

<b>Notices</b> .....	v
Trademarks .....	vi
<b>IBM Redbooks promotions</b> .....	vii
<b>Preface</b> .....	ix
Authors .....	ix
Now you can become a published author, too! .....	x
Comments welcome .....	x
Stay connected to IBM Redbooks .....	xi
<b>Chapter 1. IBM Spectrum Scale active archive</b> .....	1
1.1 Introduction .....	2
1.2 IBM Spectrum Scale .....	3
1.3 IBM Spectrum Archive .....	8
1.4 IBM Spectrum Scale active archive architecture .....	9
<b>Chapter 2. Active archive lab environment</b> .....	13
<b>Chapter 3. Adding IBM Spectrum Archive EE node to existing IBM Spectrum Scale protocols cluster</b> .....	17
3.1 Installing and configuring IBM Spectrum Scale on an IBM Spectrum Archive node ..	18
3.2 Installing IBM Spectrum Archive prerequisite software on a Linux system .....	23
3.3 Installing the IBM tape device driver .....	24
3.4 Installing the IBM Spectrum Archive EE software .....	24
3.5 Preparing the IBM Spectrum Scale file systems for IBM Spectrum Archive .....	25
3.6 Configuring the IBM Spectrum Archive EE software .....	27
3.7 Starting the IBM Spectrum Archive EE software .....	28
3.8 Preparing IBM Spectrum Archive EE for migration .....	28
<b>Chapter 4. Swift account, container, and object storage pool placement</b> .....	31
4.1 IBM Spectrum Scale storage pool configuration .....	32
4.2 Container and account placement policies .....	33
4.3 Object placement and migration policies .....	34
<b>Chapter 5. Object heatmap data tiering</b> .....	39
5.1 Enabling heatmap .....	40
5.2 Object heatmap policy .....	40
<b>Chapter 6. Storing objects in the IBM Spectrum Archive tape tier</b> .....	43
6.1 Container-based migration of data to tape .....	44
6.2 Sweeping migration of data to tape .....	47
<b>Chapter 7. Retrieving objects from the IBM Spectrum Archive tape tier</b> .....	49
7.1 Application-transparent object recall from tape .....	50
7.2 Prefetched recall of object data from tape .....	50
<b>Chapter 8. Runtime considerations</b> .....	55
8.1 Verifying IBM Spectrum Scale OpenStack services with Boto .....	56
8.2 Determining the location of objects in the active archive .....	58

**Chapter 9. Conclusion** ..... 63

**Related publications** ..... 65

IBM Redbooks ..... 65

Other publications ..... 65

Online resources ..... 65

Help from IBM ..... 66

# Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing, IBM Corporation, North Castle Drive, MD-NC119, Armonk, NY 10504-1785, US*

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

## COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.


# Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at “Copyright and trademark information” at <http://www.ibm.com/legal/copytrade.shtml>

The following terms are trademarks or registered trademarks of International Business Machines Corporation, and might also be trademarks or registered trademarks in other countries.

developerWorks®  
GPFS™  
IBM®  
IBM Elastic Storage™  
IBM Spectrum™

IBM Spectrum Archive™  
IBM Spectrum Control™  
IBM Spectrum Scale™  
Linear Tape File System™  
POWER8®

Redbooks®  
Redpaper™  
Redbooks (logo) ®  
Tivoli®

The following terms are trademarks of other companies:

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Linear Tape-Open, LTO, Ultrium, the LTO Logo and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and other countries.

Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.



## Find and read thousands of IBM Redbooks publications

- ▶ Search, bookmark, save and organize favorites
- ▶ Get personalized notifications of new content
- ▶ Link to the latest Redbooks blogs and videos

Get the latest version of the Redbooks Mobile App



Download  
Now

Android



## Promote your business in an IBM Redbooks publication

Place a Sponsorship Promotion in an IBM® Redbooks® publication, featuring your business or solution with a link to your web site.

Qualified IBM Business Partners may place a full page promotion in the most popular Redbooks publications. Imagine the power of being seen by users who download millions of Redbooks publications each year!



[ibm.com/Redbooks](http://ibm.com/Redbooks)

About Redbooks → Business Partner Programs

THIS PAGE INTENTIONALLY LEFT BLANK

# Preface

Enterprises are struggling to provide the right storage infrastructure to keep up with the explosion of unstructured data in addition to facing increased pressure to retain this data for an extended period of time. Object storage is rapidly emerging as a viable method for building scalable big data archiving solutions to address these unstructured data growth challenges.

OpenStack Swift is an emerging open source object storage platform that is widely used for cloud storage. IBM® Spectrum Scale V4.2 delivers a fast, highly available, highly scalable shared file system that enables transparent access to files and objects spanning different storage tiers such as flash, disk, and tape. IBM Spectrum™ Archive Enterprise Edition is designed to enable the use of IBM Linear Tape File System™ (LTFS) for the policy management of tape as a storage tier in IBM Spectrum Scale™ to significantly reduce cost.

This IBM Redpaper™ publication describes how to create an Enterprise class, low-cost, highly scalable object storage infrastructure with IBM Spectrum Scale 4.2, leveraging OpenStack Swift and IBM Spectrum Archive™. It describes benefits of the solution and provides reference architectures, preferred practices, and runtime considerations. It is suitable for IBM clients, IBM Business Partners, IBM specialist sales representatives, and technical specialists.

## Authors

This paper was produced by a team of specialists from around the world working at the International Technical Support Organization, Poughkeepsie Center.

**Larry Coyne** is a Project Leader at the International Technical Support Organization, Tucson Arizona center in the US. He has 33 years of IBM experience with 23 in IBM storage software management. He holds degrees in Software Engineering from the University of Texas at El Paso and Project Management from George Washington University. His areas of expertise include client relationship management, quality assurance, development management, and support management for IBM Tivoli® Storage Software.

**Joe Dain** is a Senior Engineer and Master Inventor in Tucson, Arizona and works in the Storage and Software Defined Infrastructure CTO Office. He joined IBM in 2003 with a BS in Electrical Engineering. His areas of expertise include backup, restore, disaster recovery, object storage, data reduction techniques such as data deduplication and compression, and emerging storage technology trends. He is on his fourteenth IBM invention plateau with over 60 patents issued and pending worldwide, including 22 high-value patents.

**Khanh Ngo** is an IBM Senior Engineer and Master Inventor in Tucson, Arizona. He serves as the Tape FVT Test Architect, designing tests and test solutions for the tape drive, tape library, and IBM Spectrum Archive product lines. He joined IBM in 2000 with a BS in Electrical Engineering and a BS in Computer Science. Later, he received an MS in Engineering Management. As a developer and tester, he has designed and created several software products, including the IBM Tape System Reporter (IBM TSR), TS3500CLI, TS3310CLI, and various IBM Spectrum Scale Enterprise Edition scripts. He is on his fifth IBM patent plateau.

**Aaron Palazzolo** is a Senior Engineer specializing in technical leadership of geographically dispersed test teams for cloud storage products. He is responsible for deployment, infrastructure implementation, and virtualization within the IBM Spectrum Scale test and

development teams. During his 16 years at IBM, he has led test teams on cloud products such as Scale Out Network Attached Storage and V7000 Unified. Prior experience includes technical and execution leadership for the TS7700 enterprise level virtual tape products and IBM zSeries Mainframe tape attachment. Aaron holds a degree in Computer Engineering and Music from the University of Arizona.

Thanks to the following people for their contributions to this project:

Dean Hildebrand  
**IBM Research**

David Anizar Ibarra  
Sean Lee  
Bill Owen  
**IBM Systems**

## Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Learn more about the residency program, browse the residency index, and apply online at:

[ibm.com/redbooks/residencies.html](http://ibm.com/redbooks/residencies.html)

## Comments welcome

Your comments are important to us!

We want our papers to be as helpful as possible. Send us your comments about this paper or other IBM Redbooks® publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

[ibm.com/redbooks](http://ibm.com/redbooks)

- Send your comments in an email to:

[redbooks@us.ibm.com](mailto:redbooks@us.ibm.com)

- Mail your comments to:

IBM Corporation, International Technical Support Organization  
Dept. HYTD Mail Station P099  
2455 South Road  
Poughkeepsie, NY 12601-5400

## Stay connected to IBM Redbooks

- ▶ Find us on Facebook:  
<http://www.facebook.com/IBMRedbooks>
- ▶ Follow us on Twitter:  
<http://twitter.com/ibmredbooks>
- ▶ Look for us on LinkedIn:  
<http://www.linkedin.com/groups?home=&gid=2130806>
- ▶ Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:  
<https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm>
- ▶ Stay current on recent Redbooks publications with RSS Feeds:  
<http://www.redbooks.ibm.com/rss.html>





# IBM Spectrum Scale active archive

This chapter introduces the active archive use case and solution provided by IBM Spectrum Scale and IBM Spectrum Archive.

## 1.1 Introduction

The enormous amount and rapid pace of unstructured data generated in today's world poses significant cost and complexity issues to traditional enterprise storage infrastructure. In addition, enterprises are facing increasingly stringent requirements to retain this vast amount of data for extended periods of time, often for many years, in a secure manner.

Object storage provides a simple, flexible, cost-efficient, and highly scalable solution to address these burdens through its ability to store data in a flat namespace that scales to trillions of objects and ability to use low-cost commodity hardware. Object storage provides global data distribution and also simple REST-based access that makes data globally accessible and easy to integrate with mobile applications, and also provides the ability to tag objects with custom metadata that can be indexed and searched. However, at large scale, enterprises still cannot afford to keep all of their unstructured data on disk-based object storage for extended periods of time.

Object storage-based active archives incorporate multiple tiers of storage, such as flash, disk, and tape, and automatically move data to the optimal storage tier based on customer-defined policies. The movement of data between storage pools is transparent to applications, resulting in a simple, massively scalable archive infrastructure. Costs are minimized as large amounts of cold data are automatically migrated to low-cost tape, which is recalled transparently to applications upon request.

This paper describes how to create a secure, low-cost, highly scalable, and highly reliable, active archive by using IBM Spectrum Scale, OpenStack Swift-based object storage, and IBM Spectrum Archive to address the burdens of storing vast amounts of unstructured data for extended periods of time. See the following chapters:

- ▶ Chapter 1, “IBM Spectrum Scale active archive” on page 1 (current chapter) introduces the active archive benefits, concepts, components, and reference architecture.
- ▶ Chapter 2, “Active archive lab environment” on page 13 describes the lab environment used in for this paper.
- ▶ Chapter 3, “Adding IBM Spectrum Archive EE node to existing IBM Spectrum Scale protocols cluster” on page 17 describes the IBM Spectrum Archive node installation and configuration.
- ▶ Chapter 4, “Swift account, container, and object storage pool placement” on page 31 describes how to place a Swift account and container-related structures on a high-performance solid-state drive (SSD) pool for optimal performance.
- ▶ Chapter 5, “Object heatmap data tiering” on page 39 describes how to configure information lifecycle management (ILM) policies that automatically move frequently accessed objects to high performance storage tiers and rarely accessed objects to lower cost storage tiers.
- ▶ Chapter 6, “Storing objects in the IBM Spectrum Archive tape tier” on page 43 describes two methods for migrating objects to the IBM Spectrum Archive tape tier.
- ▶ Chapter 7, “Retrieving objects from the IBM Spectrum Archive tape tier” on page 49 describes two methods for recalling objects from the IBM Spectrum Archive tape tier.
- ▶ Chapter 8, “Runtime considerations” on page 55 describes how to get and put objects in the active archive and how to determine the location of objects within the active archive.
- ▶ Chapter 9, “Conclusion” on page 63 summarizes the benefits and advantages of the active archive.



## 1.2 IBM Spectrum Scale

IBM Spectrum Scale is a proven, scalable, high-performance data and file management solution (based on IBM General Parallel File System or GPFS™ technology) that is being used extensively in multiple industries worldwide. IBM Spectrum Scale provides simplified data management and integrated information lifecycle tools that are capable of managing petabytes of data and billions of files and objects in order to manage the growing cost of handling ever increasing amounts of data.

IBM Spectrum Scale removes data-related bottlenecks by providing parallel access to data, eliminating single filer choke points or hot spots. IBM Spectrum Scale also simplifies data management at scale by providing a single namespace that can be scaled simply, quickly, and infinitely by simply adding more scale-out resources, eliminating “filer sprawl” and its associated headaches.

### IBM Spectrum Scale 4.2

IBM Spectrum Scale 4.2 introduces *protocol nodes*, which provide data access to a shared storage infrastructure through enhanced support for Network File System (NFS), Server Message Block (SMB), Swift Object, and S3 Object protocols. The nodes provide 10 Gb or 40 Gb Ethernet connectivity and also transparent node failover to client applications for the following protocols:

- ▶ NFS v3 and v4
- ▶ SMB 2 and SMB 3.0 mandatory features providing Common Internet File System (CIFS) for Windows support
- ▶ OpenStack Swift and S3 API support for object storage

**GPFS:** The terms IBM Spectrum Scale and GPFS are interchangeable in this paper.

IBM Spectrum Scale protocol nodes are described in the IBM Knowledge Center:

<https://ibm.biz/BdH9r5>

IBM Spectrum Scale 4.2 also offers the following benefits:

- ▶ Unified file and object access, enabling users to access the same data as an object using Swift and S3 protocols and as a file using NFS, SMB, and POSIX protocols.

An additional benefit of unified file and object access is the naming convention of objects, which are stored in the file system. In this manner, determining the location and managing with ILM policies become easier.

For more details pertaining to unified file and object access, see the following web page:

<https://ibm.biz/BdH9sd>

- ▶ Improved client experience with configuration guidance, an installation toolkit, and integration with IBM Spectrum Control™ for health monitoring.
- ▶ Advanced Edition support for asynchronous multisite disaster recovery (DR).
- ▶ Native file system compression and encryption (encryption of data at rest and secure-erase is included in the Advanced Edition).

## IBM Spectrum Scale 4.2 object storage

IBM Spectrum Scale 4.2 provides Swift and S3 object connectivity by leveraging the Openstack Swift services running in the IBM Spectrum Scale 4.2 protocol nodes.

IBM Spectrum Scale 4.2 object storage offers the following benefits:

- ▶ Simplified data scaling and management through a flat namespace that scales to trillions of objects:
  - Seamless infrastructure expansion with no disruption
  - Linear capacity and performance scale-out
- ▶ Cost-efficient highly scalable infrastructure with software-defined storage flexibility:
  - You can use low-cost commodity hardware including high-density storage.
- ▶ Global, secure multitenant access:
  - Geographically distributed synchronous and asynchronous replication provide multisite data protection and the ability to ingest and access data from any data center.
  - Multitenant management and data access with role-based authentication.
  - Enterprise class encryption of data in flight and at rest, immutable objects, and secure data erase.
- ▶ Single global namespace that supports multiple protocol access and multiple storage tiers
  - Supports NFS, SMB, Swift object, S3 object, Hadoop, and POSIX access in the same namespace.
  - Supports SSD or flash, disk, and tape storage pools transparently under the same namespace.

Figure 1-1 illustrates the IBM Spectrum Scale 4.2 high-level architecture.

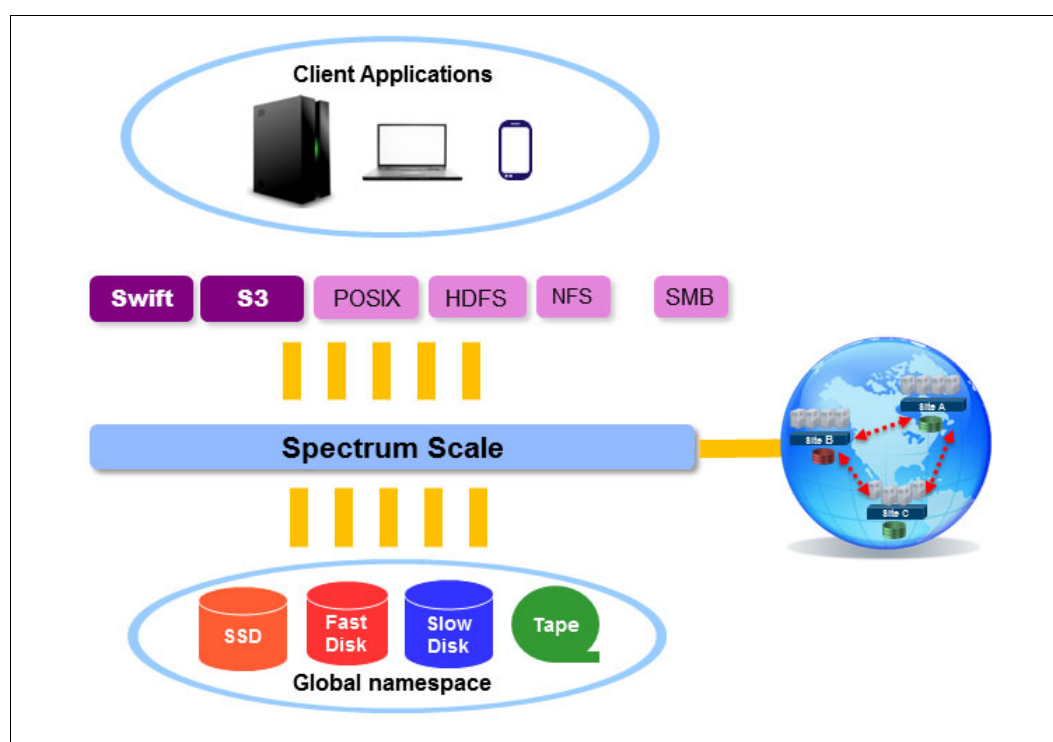


Figure 1-1 IBM Spectrum Scale object high-level architecture

## **IBM Elastic Storage Server**

The IBM Elastic Storage™ Server is a modern implementation of software defined storage built on the IBM General Parallel File System (GPFS). This technology combines the CPU and I/O capability of the IBM POWER8® architecture and matches it with 2U and 4U storage enclosures to provide high-capacity storage for analytics and cloud-serving. This architecture permits the IBM Spectrum Scale RAID software capability to actively manage all RAID functionality formerly accomplished by a hardware disk controller. Newly developed RAID techniques from IBM use this CPU and I/O power to help overcome the limitations of current disk drive technology and simplify your transition to a multitier storage architecture employing solid state flash technology and robotic tape libraries.

In addition to the technological advancements available with the Elastic Storage Server, it can also address other data issues found in many businesses. For example, as each department or division in your organization evolves its own storage needs, the result can be a costly duplication of hardware resources. The resulting islands of information can hold valuable insights that might not be accessible in such a disparate environment. Consolidate storage requirements across your organization onto the Elastic Storage Server, so you can reduce inefficiency and acquisition costs, simplify management, and improve data protection.

The Elastic Storage Server includes these capabilities:

### **Data redundancy**

IBM Spectrum Scale RAID supports highly reliable 2-fault-tolerant and 3-fault-tolerant Reed-Solomon based parity codes and also 3-way and 4-way replication.

### **Large cache**

Using a combination of internal and external flash devices along with large memory cache in the Power Server, the Elastic Storage Server is better able to mask the inefficiencies and long latency times of nearline (NL) SAS drives, while still using the high-density of the drives themselves.

### **Superior streaming performance**

The system can deliver over 12 GB per second of sustained performance from a storage server. These can be clustered to meet any performance requirement.

### **Scalability**

As server configurations are added to an installed configuration, the capacity, bandwidth, performance and the single name space all grow. This means installations can start small, and grow as data needs expand.

### **IBM Spectrum Scale RAID**

IBM Spectrum Scale RAID runs IBM disks in a dual-ported storage enclosure, which does not require external RAID storage controllers or other custom hardware RAID acceleration. It distributes client data, redundancy information, and spare space uniformly across all disks of a JBOD. This distribution reduces the rebuild or disk failure recovery process overhead compared to conventional RAID. Critical rebuilds of failed full multi-terabyte drives can be accomplished in minutes, rather than hours or even days when using traditional RAID technology.

### **Designed for rapid deployment**

The Elastic Storage Server can be easy to deploy with a complete infrastructure, including preloaded software that is assembled and tested prior to being delivered. After the system arrives on site, services are available from IBM for fast configuration and integration into your data center's Ethernet or InfiniBand network.

### **Developed to manage data**

The Elastic Storage Server is equipped to manage the complete data lifecycle. Installation-specific data tier rules can be created to define what kinds of data are created, who has access to each tier of storage, and how data is migrated to lower tiers of storage after it.

### **Graphical user interface (GUI)**

The intuitive GUI allows management and monitoring of the system, both locally and remotely.

## **IBM Spectrum Scale RAID is an enabling technology**

IBM Spectrum Scale RAID is a software implementation of storage RAID technologies within IBM Spectrum Scale, which creates the foundation for the IBM Elastic Storage Server described previously. Using conventional dual-ported disks in a JBOD (just a bunch of disks) configuration, IBM Spectrum Scale RAID implements sophisticated data placement and error correction algorithms to deliver high levels of storage reliability, availability, and performance. Standard IBM Spectrum Scale file systems are created from the Network Shared Disks (NSDs) defined through IBM Spectrum Scale RAID.

IBM Spectrum Scale RAID integrates the functionality of an advanced storage controller into the IBM Spectrum Scale NSD server. Unlike an external storage controller, where configuration, logical unit number (LUN) definition, and maintenance are beyond the control of IBM Spectrum Scale, IBM Spectrum Scale RAID takes ownership of a JBOD array to directly match LUN definition, caching, and disk behavior to IBM Spectrum Scale file system requirements.

Sophisticated data placement and error correction algorithms deliver high levels of storage reliability, availability, serviceability, and performance. IBM Spectrum Scale RAID provides a variation of the IBM Spectrum Scale NSD called a virtual disk (VDisk). Standard NSD clients transparently access the VDisk NSDs of a file system by using the conventional NSD protocol.

The IBM Spectrum Scale RAID includes these features:

### **Software RAID**

IBM Spectrum Scale RAID runs on standard Linux disks in a dual-ported JBOD array, which does not require external RAID storage controllers or other custom hardware RAID acceleration.

### **Declustering**

IBM Spectrum Scale RAID distributes client data, redundancy information, and spare space uniformly across all disks of a JBOD. This distribution reduces the rebuild (disk failure recovery process) overhead compared to conventional RAID.

### **Checksum**

An end-to-end data integrity check, using checksums and version numbers, is maintained between the disk surface and NSD clients. The checksum algorithm uses version numbers to detect silent data corruption and lost disk writes.

### **Data redundancy**

IBM Spectrum Scale RAID supports highly reliable 2-fault-tolerant and 3-fault-tolerant Reed-Solomon based parity codes and 3-way and 4-way replication.

### **Large cache**

A large cache improves read and write performance, particularly for small I/O operations.

### **Arbitrarily sized disk arrays**

The number of disks is not restricted to a multiple of the RAID redundancy code width, which allows flexibility in the number of disks in the RAID array.

### **Multiple redundancy schemes**

One disk array can support VDIs with different redundancy schemes, for example Reed-Solomon and replication codes.

### **Disk Hospital**

A disk hospital asynchronously diagnoses faulty disks and paths, and requests replacement of disks by using past health records.

### **Automatic recovery**

Seamlessly and automatically recovers from primary server failure.

### Disk scrubbing

A disk scrubber automatically detects and repairs latent sector errors in the background.

### Familiar interface

Standard IBM Spectrum Scale command syntax is used for all configuration commands; including, maintaining and replacing failed disks.

### Flexible hardware configuration

Supports JBOD enclosures with multiple disks physically mounted together on removable carriers.

### Configuration and data logging

Internal configuration and small-write data are automatically logged to solid-state drives for improved performance.

For an in-depth outline of IBM Spectrum Scale RAID and its capabilities, see the video:

[https://www.youtube.com/watch?v=VvIgjVYPc\\_U](https://www.youtube.com/watch?v=VvIgjVYPc_U)

The Elastic Storage Server (ESS) GL models use 4U 60 drive storage enclosures with 2 TB, 4 TB, or 6 TB drives, and offers a client-ready petabyte in a single rack. Figure 1-2 highlights the Elastic Storage Server GL models.

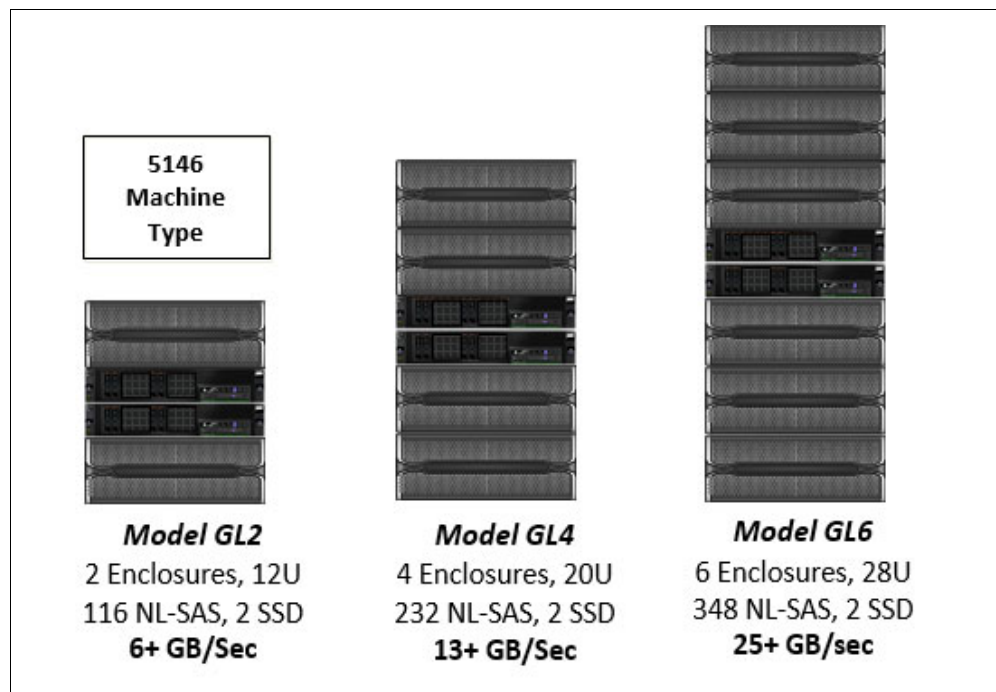


Figure 1-2 Elastic Storage Server GL models

The Elastic Storage Server GS models offer smaller capacity points with the highest performance per U delivered to clients using 2u, 24 drive enclosures with 400 GB, 800 GB SSDs, or 1.2 TB SAS drives. Figure 1-3 highlights the Elastic Storage Server GS models.

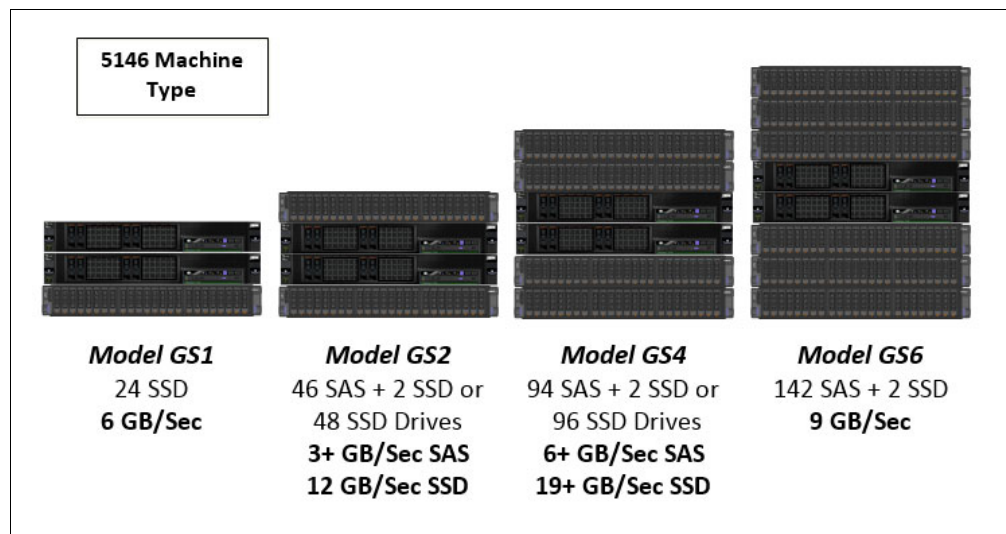


Figure 1-3 Elastic Storage Server GS models

## 1.3 IBM Spectrum Archive

IBM Spectrum Archive Enterprise Edition (EE) seamlessly integrates a low-cost tape tier into the IBM Spectrum Scale storage pool hierarchy to significantly reduce cost. With IBM Spectrum Archive EE, accessing data stored on an IBM tape cartridge, instead of on disk, is transparent. IBM Spectrum Scale information lifecycle management policies are used to transparently migrate data from disk to tape. A small pointer to the data on tape is retained in disk, enabling the data to be automatically retrieved from tape without user intervention upon request by applications. IBM Spectrum Archive EE allows IBM Spectrum Scale installations to add extensive capacity with lower media, floor space, and power costs without impacting data availability.

For more information about IBM Spectrum Archive, see the following web page:

<https://ibm.biz/Bd4BT4>

### IBM Spectrum Archive supported tape libraries, drives, and cartridges

Using the IBM Linear Tape File System format, IBM Spectrum Archive EE supports IBM Linear Tape-Open (LTO) Ultrium 7, 6, and 5 tape drives in IBM TS3310, TS3500, and TS4500 tape libraries. IBM TS1140 and TS1150 tape drives are supported in TS3500 and TS4500 tape libraries.

### IBM TS4500 Tape Library

The IBM TS4500 tape library is a next-generation storage solution designed to address high data volumes and the growth in data centers, increasing cost of data center storage footprints, the difficulty of migrating data across vendor platforms, and increased complexity of IT training and management as staff resources shrink. The TS4500 delivers a low cost per terabyte and a high terabyte density per square foot. The TS4500 can store up to 5.5 PB of data in a single 10-square-foot library frame.

The TS4500 provides the following maximum capacities based on the drive and cartridge type being used:

- ▶ 3592 advanced cartridges: Up To 35.5 PB per library (106.5 PB with 3:1 compression)
- ▶ LTO Ultrium 7 cartridges: Up to 139 PB per library (up to 347.5 PB with 2.5:1 compression)

For more information about IBM tape libraries, see the following web page:

<https://ibm.biz/Bd4BTs>

## 1.4 IBM Spectrum Scale active archive architecture

Information has a different value to enterprises at different times, and data access requirements change depending on where the information resides throughout the information lifecycle. Data ingestion, creation, and data processing requires higher performance and throughput. After initial processing, data can be made available for other uses but high performance is not critical, thus it can be put on slower, more cost-effective storage tiers. In object storage deployments, often the custom metadata of the objects is accessed more often than the objects themselves; therefore, placing object metadata on high-performance storage such as SSD, while keeping the actual object data on lower-cost storage tiers such as physical tape, provides an optimal storage infrastructure.

IBM Spectrum Scale provides a rich information lifecycle policy management (ILM) engine that offers intelligent automated tiering with extreme scalability. IBM Spectrum Scale also supports the ability to have separate storage pools with different storage characteristics such as flash, SSD, disk, and IBM Spectrum Archive enabled tape. The policy engine provides seamless migration and also file and object placement between storage pools under a single global namespace.

The active archive high-level reference architecture consists of a platinum pool using SSD storage, a gold pool using 10K RPM SAS based HDDs, a silver pool containing 7.2K RPM drives can be provided with SATA interface or NL-SAS drives, and a bronze tier providing a physical tape-based IBM Spectrum Archive pool. To avoid unnecessary recall of data from tape, and to optimize performance, object metadata is stored in IBM Spectrum Scale file system extended attributes, which are placed in the SSD pool.

Additionally, Swift container and account information is placed in the SSD-based system pool to allow fast searching and listing of objects and containers. Newly ingested objects are placed in the silver pool by default; frequently accessed “hot” objects are automatically migrated to the higher performing system and gold storage pools. Less frequently accessed older objects are automatically migrated to the silver SATA or NL-SAS based tier and also to the bronze tape archive tier.

Figure 1-4 shows details of the active archive.

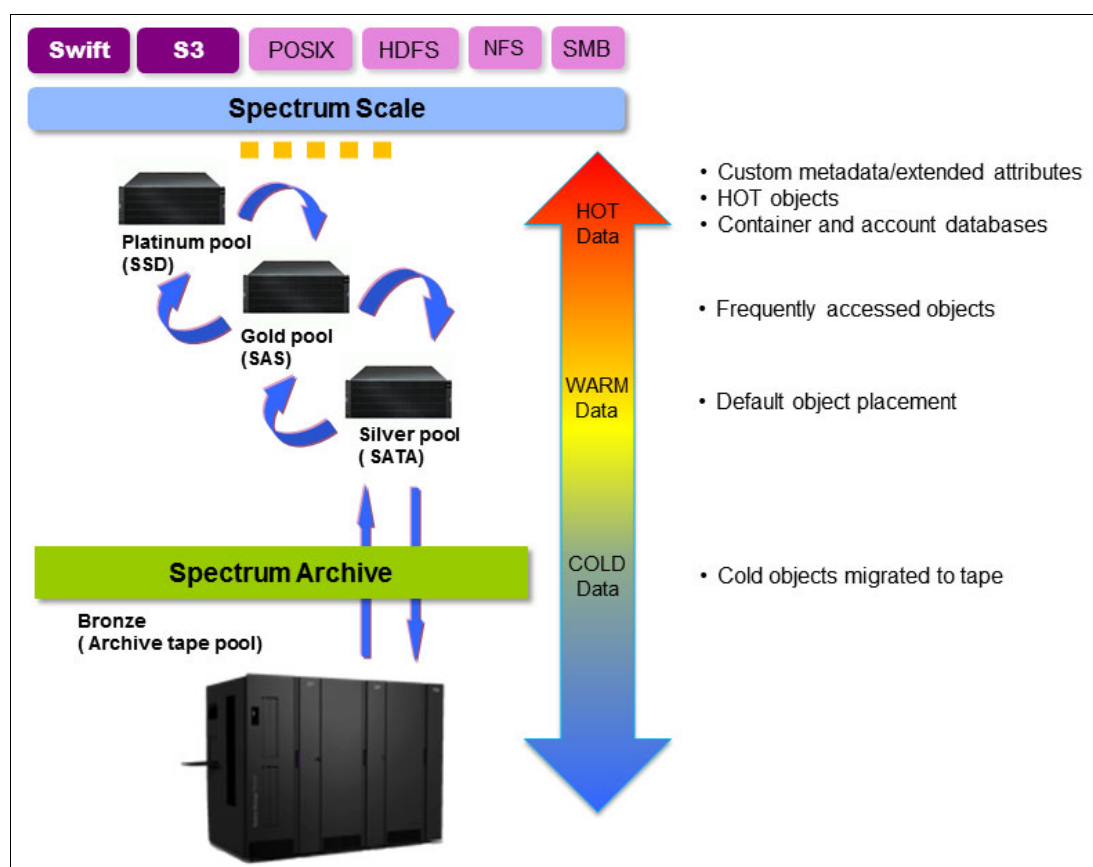


Figure 1-4 Active Archive Tiering

### Elastic Storage Server based reference architecture

The Elastic Storage Server based active archive reference architecture combines the benefits of Elastic Storage Server with the benefits of IBM Spectrum Archive. This reference architecture uses the following building blocks:

- ▶ Model GS2 Elastic Storage Server with 48 SSDs for the system pool.
- ▶ Model GS4 Elastic Storage Server with 94 SAS and 2 SSD drives for the gold pool.
- ▶ Model GL6 Elastic Storage Server with 348 NL-SAS HDDs and 2 SSDs for the silver pool.
- ▶ Two or more IBM Spectrum Archive nodes attached through 10 GbE to the Elastic Storage Server tiered storage with a Fibre Channel attached TS4500 tape library for the bronze tier.
- ▶ One or more IBM Spectrum Scale protocol nodes attached with 10 GbE to the Elastic Storage Server tiered storage. For scaling considerations for the protocol nodes, see the following web page:

<https://ibm.biz/BdH99G>

**Note:** Running a combined IBM Spectrum Scale protocol node and IBM Spectrum Archive node is currently not supported.



Figure 1-5 illustrates the Elastic Storage Server based reference architecture.

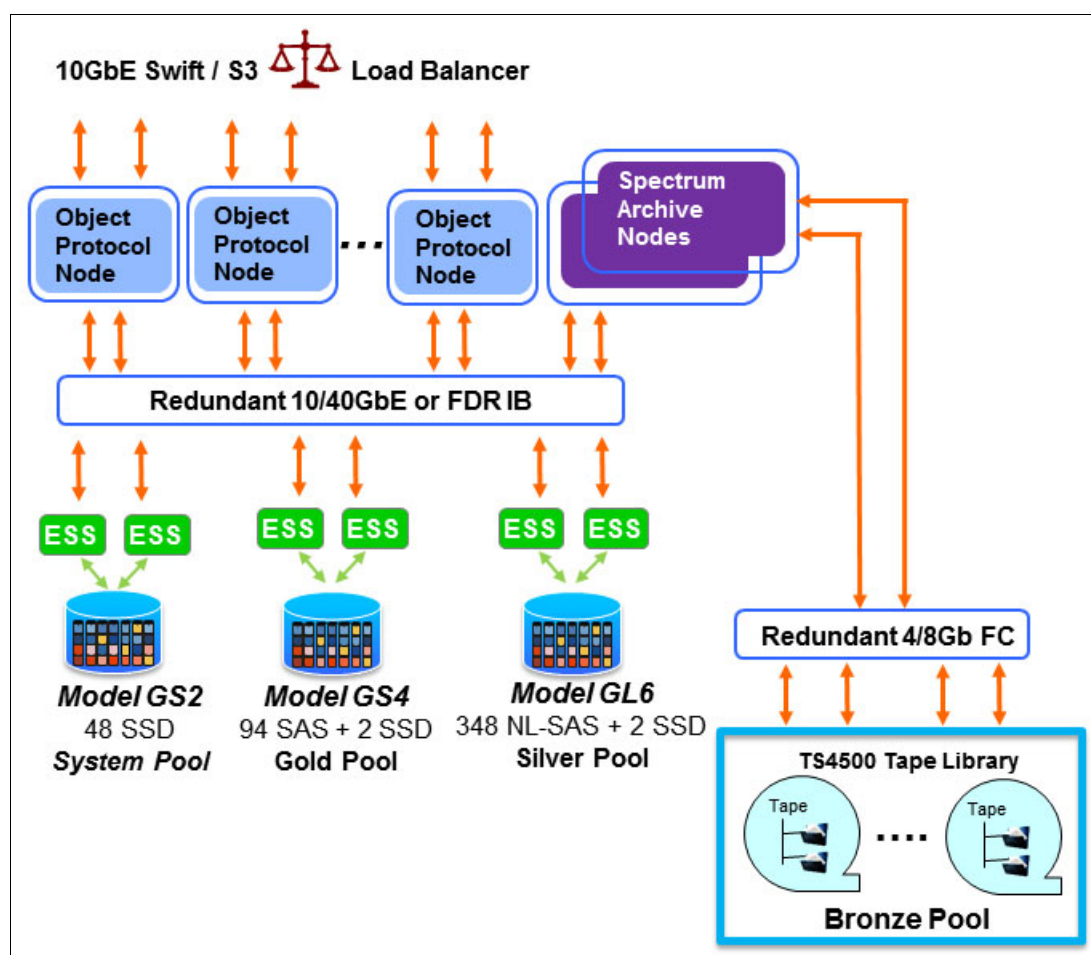


Figure 1-5 Elastic Storage Server based reference architecture

Another possibility is to use IBM Spectrum Scale NSD servers connected to SAN-attached storage to provide the shared storage infrastructure for protocol nodes, providing Swift and S3 connectivity. This reference architecture uses the following building blocks:

- ▶ Two or more NSD servers.
- ▶ Two or more IBM Spectrum Archive nodes attached through 10 GbE to the NSD servers with a Fibre Channel attached TS4500 tape library for the bronze tier.
- ▶ One or more IBM Spectrum Scale protocol nodes attached through 10 GbE to the Elastic Storage Server tiered storage. Scaling considerations for the protocol nodes are described at the following web page:  
<https://ibm.biz/BdH99G>
- ▶ Dual RAID protected controller with three storage tiers:
  - SSD based platinum tier
  - 10K RPM SAS based gold tier
  - 7.2K RPM NL-SAS based silver tier

Figure 1-6 illustrates the NSD server based reference architecture.

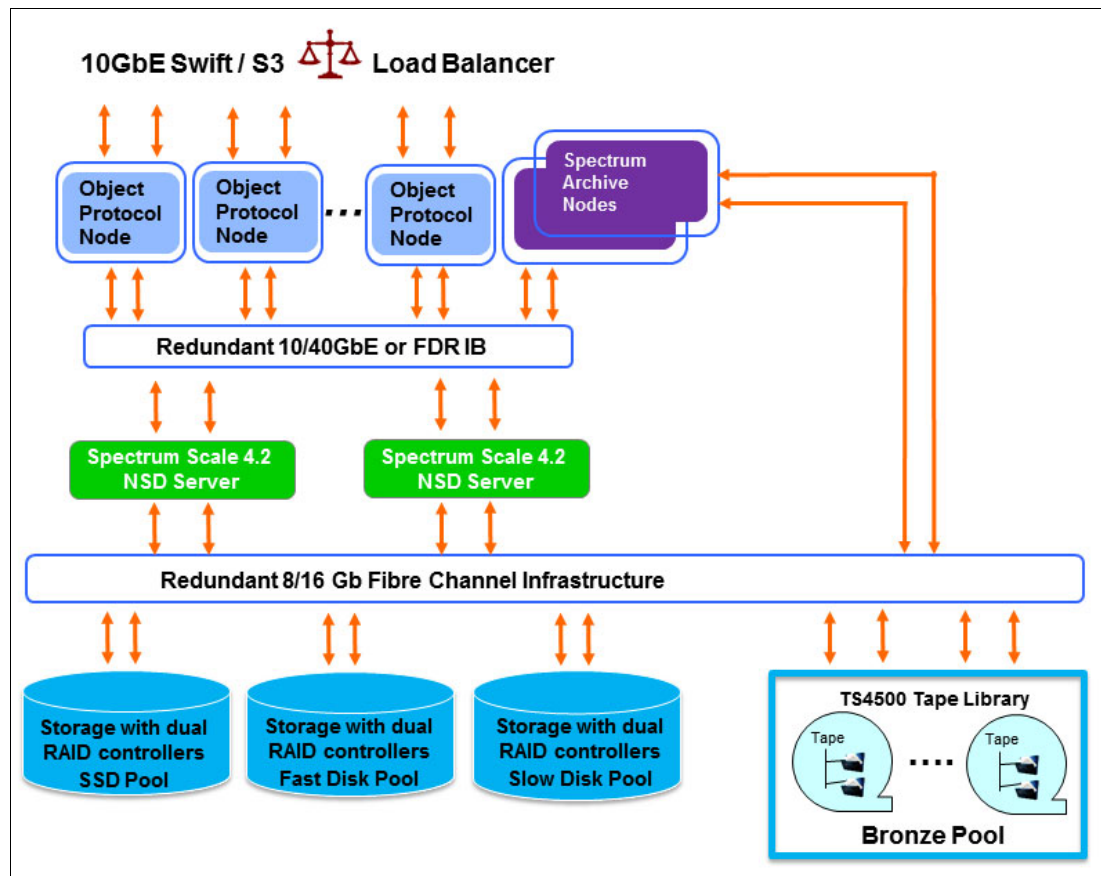


Figure 1-6 NSD based active archive reference architecture

## Active archive lab environment

The assumption in this document is that an existing IBM Spectrum Scale 4.2 protocols cluster is available. For details about how to build and configure an IBM Spectrum Scale 4.2 protocols cluster, see the protocols quick overview:

<https://ibm.biz/Bd4BTS>

Figure 2-1 illustrates the lab environment used for this implementation.

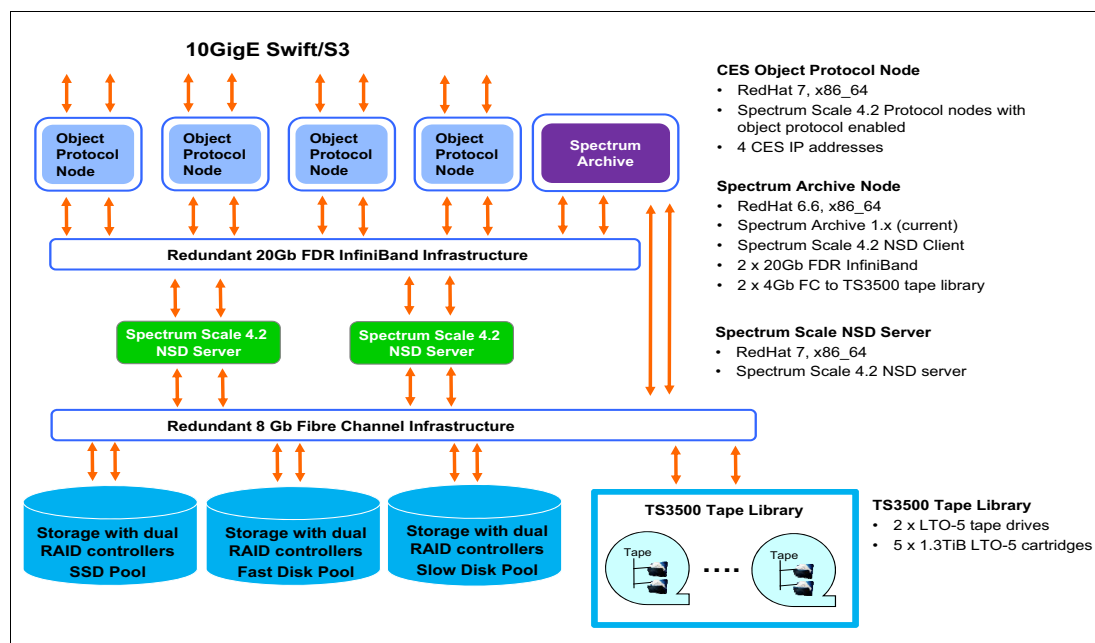


Figure 2-1 IBM Spectrum Scale object and IBM Spectrum Archive lab environment

**Note:** The IBM Spectrum Archive node accesses the IBM Spectrum Scale file system as an IBM Spectrum Scale client over 20 Gb FDR InfiniBand and accesses the tape infrastructure over 4 Gb FC. In addition, the IBM Spectrum Archive node must run Red Hat 6.6 while the other IBM Spectrum Scale nodes run Red Hat 7.x. This temporary restriction will be lifted in the future.

## Additional lab environment details

This section describes the commands used to show the cluster and protocol node details used in the lab environment.

### Cluster configuration

Example 2-1 shows use of the `mm1scluster` command to describe GPFS cluster information.

*Example 2-1 GPFS cluster information from the mm1scluster command*

---

```
[root@Boglin.ltfs001st001 ~]$ mm1scluster
```

GPFS cluster information  
=====

```
GPFS cluster name:      boglin.tuc.stglabs.ibm.com
GPFS cluster id:       11377005508221052921
GPFS UID domain:      boglin.tuc.stglabs.ibm.com
Remote shell command:  /usr/bin/ssh
Remote file copy command: /usr/bin/scp
Repository type:       CCR
```

Node	Daemon node name	IP address	Admin node name	Designation
1	prt001st001	172.31.132.1	prt001st001	quorum
2	nsd002st001	172.31.134.2	nsd002st001	quorum-manager
3	nsd001st001	172.31.134.1	nsd001st001	quorum-manager
4	prt003st001	172.31.132.3	prt003st001	
5	prt002st001	172.31.132.2	prt002st001	
6	prt004st001	172.31.132.4	prt004st001	
7	ltfs001st001	172.31.132.5	ltfs001st001	

```
[root@Boglin.ltfs001st001 ~]$
```

---

### Protocol node Information

Example 2-2 shows use of the `mmces node list` command to see the protocol nodes used in the lab environment demonstration.

*Example 2-2 Protocol node names using the mmces node list command*

---

```
[root@Boglin.ltfs001st001 ~]$ mmces node list
```

Node name	Node Flags
prt001st001	none
prt002st001	none
prt003st001	none
prt004st001	none

```
[root@Boglin.ltfs001st001 ~]$
```

---

### Protocol node IP address information

Example 2-3 shows use of the **MMCES address list** command to see IP address information of the protocol nodes.

**Note:** The 172.31.132.x network is used for the internal protocol node to NSD server InfiniBand connectivity; the 9.11.213.xx IP addresses are exported for object protocol access by clients.

*Example 2-3 Protocol Node IP address information using the mmces address list command*

```
[root@Boglin.ltfs001st001 ~]$ mmces address list
```

Node	Daemon node name	IP address	CES IP address list
1	prt001st001	172.31.132.1	9.11.213.58
4	prt003st001	172.31.132.3	9.11.213.59
5	prt002st001	172.31.132.2	
9.11.213.56(object_database_node,object_singleton_node)			
6	prt004st001	172.31.132.4	9.11.213.57

```
[root@Boglin.ltfs001st001 ~]$
```





## **Adding IBM Spectrum Archive EE node to existing IBM Spectrum Scale protocols cluster**

This chapter describes the prerequisites, installation, and configuration of IBM Spectrum Archive Enterprise Edition (EE; formerly LTFS EE). Adding an IBM Spectrum Archive EE node to the IBM Spectrum Scale protocols cluster enables the use of low-cost tape as a tier in the object storage hierarchy.

## 3.1 Installing and configuring IBM Spectrum Scale on an IBM Spectrum Archive node

This section describes the installation of the prerequisite software on a Linux system.

### Assumptions

Assumptions are as follows:

- ▶ IBM Spectrum Archive EE node is at Red Hat Enterprise Linux 6.6.
- ▶ IBM Spectrum Archive EE node has access to Red Hat Enterprise Linux repositories or manually installed RPMs required for a basic IBM Spectrum Scale installation and **mmbuildgpl** command.
- ▶ IBM Spectrum Scale protocols are installed and deployed successfully on Red Hat Enterprise Linux 7.x nodes.
- ▶ All cluster nodes will use IBM Spectrum Scale 4.2, specifically from the Protocols code package.

**Note:** If cluster nodes cannot use IBM Spectrum Scale 4.2, stop here because protocols necessitate all quorum and manager nodes to be at 4.2, CCR be set, and the release be set to LATEST.

### Steps

Complete the following steps:

1. Load the `Spectrum_Scale_install-4.2.0.0_xxxx` standard or advanced IBM Spectrum Scale 4.2 installation toolkit onto the IBM Spectrum Archive EE node. To do this, run the IBM Spectrum Scale installer on the IBM Spectrum Archive EE node as follows:

```
#./Spectrum_Scale_install-4.2.0.0_x86_64-linux_advanced_gpfs --silent
```

2. On the IBM Spectrum Archive EE node, use the **rpm -ivh** command to install all IBM Spectrum Scale RPMs located in the `/usr/lpp/mmfs/4.2.0.0/` directory (shown in Example 3-1) except the `gpfs.protocols-support` RPM. This RPM is not meant to be installed manually and is not necessary for the IBM Spectrum Archive EE node.

*Example 3-1 Install IBM Spectrum Scale rpms*

---

```
rpm -ivh gpfs.base-4.2.0-0.x86_64.rpm
rpm -ivh gpfs.gskit-8.0.50-47.x86_64.rpm
rpm -ivh gpfs.gpl-4.2.0-0.noarch.rpm
rpm -ivh gpfs.docs-4.2.0-0.noarch.rpm
rpm -ivh gpfs.ext-4.2.0-0.x86_64.rpm
rpm -ivh gpfs.crypto-4.2.0-0.x86_64.rpm
rpm -ivh gpfs.msg.en_US-4.2.0-0.noarch.rpm
```

---

3. On the IBM Spectrum Archive EE node, run **mmbuildgpl** to build the portability layer for IBM Spectrum Scale as shown in Example 3-2.

*Example 3-2 Build portability layer*

---

```
#/usr/lpp/mmfs/bin/mmbuildgpl
-----
mmbuildgpl: Building GPL module begins at Mon Oct 26 01:52:53 MST 2015.
-----
Verifying Kernel Header...
```



```

kernel version = 20632431 (2.6.32-431.el6.x86_64, 2.6.32-431)
module include dir = /lib/modules/2.6.32-431.el6.x86_64/build/include
module build dir   = /lib/modules/2.6.32-431.el6.x86_64/build
kernel source dir  = /usr/src/linux-2.6.32-431.el6.x86_64/include
Found valid kernel header file under
/lib/modules/2.6.32-431.el6.x86_64/build/include
Verifying Compiler...
make is present at /usr/bin/make
cpp is present at /usr/bin/cpp
gcc is present at /usr/bin/gcc
g++ is present at /usr/bin/g++
ld is present at /usr/bin/ld
Verifying Additional System Headers...
Verifying kernel-headers is installed ...
Command: /bin/rpm -q kernel-headers
The required package kernel-headers is installed
make World ...
make InstallImages ...

-----
mmbuildgpl: Building GPL module completed successfully at Mon Oct 26 01:53:14
MST 2015.

```

---

4. On the IBM Spectrum Archive EE node and on a protocol node, verify that the IBM Spectrum Scale version installed on the IBM Spectrum Archive EE node matches the protocol nodes by comparing the Build Date fields between nodes: `rpm -qi | gpfs.base` (see Example 3-3). The dates should match, signifying that GPFS versions are the exact same.

*Example 3-3 Verify IBM Spectrum Scale version installed on the IBM Spectrum Archive EE node matches the protocol nodes*

---

```

# rpm -qi gpfs.base
Name       : gpfs.base
Version    : 4.2.0
Release    : 0
Architecture: x86_64
Install Date: Fri 23 Oct 2015 11:16:43 PM MST
Group      : System Environment/Base
Size       : 48861712
License    : (C) COPYRIGHT International Business Machines Corp. 2001

Signature  : (none)
Source RPM : gpfs.base-4.2.0-0.src.rpm
Build Date : Fri 23 Oct 2015 01:04:58 PM MST
Build Host : bldlnx83.pok.stglabs.ibm.com
Relocations : (not relocatable)
Packager   : IBM Corp. <gpfs@us.ibm.com>
Vendor     : IBM Corp.
URL        : http://www.ibm.com/systems/software/gpfs/index.html
Summary    : GPFS File Manager
Description :
General Parallel File System File Manager

```

---

5. Verify that the IBM Spectrum Archive EE node is set up with a matching network configuration, firewall rules, and SSH configuration to the rest of the cluster that you will joining it to:
  - a. Networking should be set up on the IBM Spectrum Archive EE node in such a way that it can communicate with all other nodes of the IBM Spectrum Scale cluster. This requires the IBM Spectrum Archive EE node to be on the same subnet as the other IBM Spectrum Scale nodes or to have routing set up in such a way that all nodes can see each other. Running `mm1sc1uster` on an existing node will give a listing of all current node IPs. The IBM Spectrum Archive EE node should be able to ping all of these IPs.
  - b. On the IBM Spectrum Archive EE node, the firewall ports used by IBM Spectrum Scale should be opened. A typical TCP port for base IBM Spectrum Scale communication among nodes is 1191. Be sure this port is opened on the IBM Spectrum Archive EE node by using the Red Hat, `iptables` command. In addition, verify that any ephemeral ports, used by your current IBM Spectrum Scale cluster, are also opened on your IBM Spectrum Archive EE node. The currently configured ephemeral ports can be viewed by running the following command on any node within your existing cluster:

```
mm1sconfig | grep tscCmdPortRange
```

Use the Red Hat `iptables` command to open all ephemeral ports listed.

More firewall information and a listing of all ports used by IBM Spectrum Scale is available in the IBM Knowledge Center:

[https://www.ibm.com/support/knowledgecenter/#!/STXKQY\\_4.2.0/com.ibm.spectrum.scale.v4r2.adv.doc/b11adv\\_firewall.htm](https://www.ibm.com/support/knowledgecenter/#!/STXKQY_4.2.0/com.ibm.spectrum.scale.v4r2.adv.doc/b11adv_firewall.htm)

- c. Every administration node in an IBM Spectrum Scale cluster must be able to run IBM Spectrum Scale administration commands, generally known as `mm` commands, on any other nodes of the cluster. The same is true with the LTFS EE node.

One secure method of doing so is to use `sudo` wrapper scripts:

[https://www.ibm.com/support/knowledgecenter/#!/STXKQY\\_4.2.0/com.ibm.spectrum.scale.v4r2.adm.doc/b11adm\\_sudowrapper.htm](https://www.ibm.com/support/knowledgecenter/#!/STXKQY_4.2.0/com.ibm.spectrum.scale.v4r2.adm.doc/b11adm_sudowrapper.htm)

Another possible method is to configure promptless SSH for root. If choosing this method, promptless SSH should be established between the IBM Spectrum Archive EE node and the admin node, which is the node that is used for running all admin commands of your cluster.

Many ways exist to establish promptless SSH, but one method is to run the `ssh-keygen` command and follow up by running the `ssh-copy-id node_to_copy_to` command. Run these commands on the admin node of your cluster and on the LTFS-EE node, thus allowing key exchange to occur.

Promptless SSH can be verified as working by using SSH from the admin node to the IBM Spectrum Archive EE node and from the IBM Spectrum Archive EE node back to the admin node. Promptless SSH is working so long as these SSH logins occur without password prompts and without yes and no (Y/N) prompts.

6. Add the IBM Spectrum Archive EE node to the existing GPFS cluster; from a node in the existing GPFS cluster run the **mmaddnode <ltfsnode\_hostname>** command (Example 3-4).

*Example 3-4 Add IBM Spectrum Archive EE node to the existing GPFS cluster*

---

```
# mmaddnode ltfs001st001
=====
| Warning:
|   This cluster contains nodes that do not have a proper GPFS license
|   designation. This violates the terms of the GPFS licensing agreement.
|   Use the mmchlicense command and assign the appropriate GPFS licenses
|   to each of the nodes in the cluster. For more information about GPFS
|   license designation, see the Concepts, Planning, and Installation Guide.
|=====
Tue May 19 20:54:45 MST 2015: mmaddnode: Processing node
ltfs001st001.tuc.stglabs.ibm.com
mmaddnode: Command successfully completed
mmaddnode: Warning: Not all nodes have proper GPFS license designations.
    Use the mmchlicense command to designate licenses as needed.
mmaddnode: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

---

7. Accept licenses for the new IBM Spectrum Archive EE node from a node within the existing GPFS cluster and run the **mmchlicense client --accept -N <ltfs\_hostname>** command (Example 3-5).

*Example 3-5 The mmchlicense -N command to verify proper license setup*

---

```
# mmchlicense client --accept -N ltfs001st001
```

---

Output from the **mmchlicense** command shows that the following nodes will be designated as possessing GPFS client licenses:

```
ltfs001st001.tuc.stglabs.ibm.com
mmchlicense: Command successfully completed
mmchlicense: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

From a node within the existing GPFS cluster run **mmllslicense -L** to verify that the proper licenses are set up, as shown in Example 3-6.

*Example 3-6 The mmllslicense -L command to verify proper license setup*

---

```
mmllslicense -L
```

Node name	Required license	Designated license
prt001st001	server	server
nsd002st001	server	server
nsd001st001	server	server
prt002st001	server	server
prt003st001	server	server
prt004st001	server	server
ltfs001st001	client	client

```
Summary information
-----
Number of nodes defined in the cluster:      7
Number of nodes with server license designation: 6
Number of nodes with client license designation: 1
Number of nodes still requiring server license designation: 0
Number of nodes still requiring client license designation: 0
This node runs IBM Spectrum Scale Standard Edition
```

---

8. Start IBM Spectrum Scale on the IBM Spectrum Archive EE node (this might need to be done before step 4, if license acceptance gives any issues).

From the IBM Spectrum Archive EE node, run **mmstartup** (Example 3-7).

*Example 3-7 The mmstartup command on IBM Spectrum Archive EE node*

---

```
# mmstartup
Mon Nov  2 10:21:05 MST 2015: mmstartup: Starting GPFS ...
```

---

9. From the IBM Spectrum Archive EE node, verify all nodes are up and “talking” by running the **mmgetstate -a** command (Example 3-8). Also do a spot check from another node to verify every node sees every node correctly.

*Example 3-8 The mmgetstate -a command*

---

```
# mmgetstate -a
```

Node number	Node name	GPFS state
1	prt001st001	active
2	nsd002st001	active
3	nsd001st001	active
4	prt003st001	active
5	prt002st001	active
6	prt004st001	active
7	ltfs001st001	active

---

From the IBM Spectrum Archive EE node, run the **mmclscluster** command (Example 3-9).

*Example 3-9 The mmclscluster command*

---

```
# mmclscluster
```

GPFS cluster information  
=====

```
GPFS cluster name:      boglin.tuc.stglabs.ibm.com
GPFS cluster id:       11377005508221052921
GPFS UID domain:       boglin.tuc.stglabs.ibm.com
Remote shell command:  /usr/bin/ssh
Remote file copy command: /usr/bin/scp
Repository type:       CCR
```

Node	Daemon node name	IP address	Admin node name	Designation
1	prt001st001	172.31.132.1	prt001st001	quorum
2	nsd002st001	172.31.134.2	nsd002st001	quorum-manager
3	nsd001st001	172.31.134.1	nsd001st001	quorum-manager
4	prt003st001	172.31.132.3	prt003st001	
5	prt002st001	172.31.132.2	prt002st001	
6	prt004st001	172.31.132.4	prt004st001	
7	ltfs001st001	172.31.132.5	ltfs001st001	

---

From the IBM Spectrum Archive EE node, run the `mm1scluster --ces` command (Example 3-10), which demonstrates that the IBM Spectrum Scale version on the IBM Spectrum Archive EE node is aware of CES functionality on the protocol nodes.

*Example 3-10 The `mm1scluster --ces` command*

---

```
# mm1scluster --ces

GPFS cluster information
=====
GPFS cluster name:      boglin.tuc.stglabs.ibm.com
GPFS cluster id:       11377005508221052921

Cluster Export Services global parameters
-----
Shared root directory:  /ibm/fs1/ces
Enabled Services:       OBJ SMB NFS
Log level:              0
Address distribution policy: node-affinity

Node  Daemon node name      IP address      CES IP address list
-----
1    prt001st001             172.31.132.1    9.11.213.58
4    prt003st001             172.31.132.3    9.11.213.57
5    prt002st001             172.31.132.2    9.11.213.59
6    prt004st001             172.31.132.4    9.11.213.56
(object_database_node,object_singleton_node)
```

---

## 3.2 Installing IBM Spectrum Archive prerequisite software on a Linux system

Various prerequisite RPM packages must be installed before the IBM Spectrum Archive Enterprise Edition installation. This section lists the prerequisite RPM packages for Red Hat Enterprise Linux.

Here are the required software packages for Red Hat Enterprise Linux systems:

- ▶ openssl 0.9.8e or later
- ▶ fuse 2.7.4 or later
- ▶ fuse-libs 2.7.4 or later
- ▶ libxml2-2.6.26 or later
- ▶ libuuid 1.39 or later
- ▶ libicu 3.6 or later
- ▶ glibc 2.12-1.47.el6.i686 or later (prerequisite for 64-bit BA client)
- ▶ nss-softokn-freebl 1.0.0 or later
- ▶ Python 2.4 or later, but earlier than 3.0
- ▶ IBM tape device driver for Linux (lin\_tape) 2.7.0 or later
- ▶ net-snmp 5.5-37 or later
- ▶ net-snmp-libs 5.5-37 or later
- ▶ lm\_sensors-libs 3.1.1-10 or later

### 3.3 Installing the IBM tape device driver

Install the IBM Linux tape device driver (`lin_tape`) on Red Hat Linux Release 6 (64 bit) as follows:

1. Download the appropriate level of the `lin_tape` source and `lin_taped` binary RPM package to a directory of your choice on the Linux server system.
2. Run the `rpmbuild --rebuild <filename>` command where `<filename>` is the name of the source RPM file. This creates an installable binary RPM package for your specific kernel from the source RPM package as shown in the following example:

```
rpmbuild --rebuild lin_tape-2.9.6-1.src.rpm
```

3. The output from the binary build is shown. Near the end of the output, a line indicates the file name and location of the binary RPM file as shown in the following example:

```
Wrote: /root/rpmbuild/RPMS/x86_64/lin_tape-2.9.6-1.x86_64.rpm
```

4. To install the `lin_tape` driver from the binary package as the next step, run the `rpm -ivh <filename>` command against the `.rpm` file that was created in the previous step, as shown in the following example:

```
rpm -ivh /root/rpmbuild/RPMS/x86_64/lin_tape-2.9.6-1.x86_64.rpm
```

5. The final package is to install the `lin_taped` daemon. Run the `rpm -ivh <filename>` command where `<filename>` is the `lin_taped` binary RPM package as shown in the following example:

```
rpm -ivh lin_taped-2.9.6-rhel6.x86_64.rpm
```

6. Finally, you can confirm the successful installation and list the found and configure IBM tape libraries and tape drives by running the following Linux command (shown here with the output):

```
cat /proc/scsi/IBM*
lin_tape version: 2.9.6
lin_tape major number: 247
Attached Changer Devices:
Number model      SN                      HBA          SCSI          FO Path
0       03584L32     0000013400190412      lpfc          2:0:0:1        NA
lin_tape version: 2.9.6
lin_tape major number: 247
Attached Tape Devices:
Number model      SN                      HBA          SCSI          FO Path
0       ULT3580-TD5  10WT013484            lpfc          2:0:0:0        NA
1       ULT3580-TD5  1068045923            lpfc          3:0:0:0        NA
[root@boglin.ltf001st001 ~]#
```

### 3.4 Installing the IBM Spectrum Archive EE software

Install the IBM Spectrum Archive Enterprise Edition (LTFS EE) package on Red Hat Linux Release 6 64-bit as follows:

1. Download the appropriate LTFS EE package to a directory of your choice on the Linux server system.
2. Ensure the LTFS EE binary package has execute permissions:

```
chmod +x ltfsee-1.1.2.0-9800.x86_64.bin
```

3. Extract the LTFS EE binary RPM packages from the LTFS EE installation package and accept the license when prompted:

```
./ltfsee-1.1.2.0-9800.x86_64.bin -i console
```

4. Run the `cd /root/rpm` command to change directories to the installation path.
5. Run the `./ltfsee_install -p` command to verify that all prerequisite packages are installed. A successful installation is shown in Example 3-11.

---

*Example 3-11 Results of IBM Spectrum Archive Enterprise Edition package installation*

---

```
Checking rpm installation and version...
The version 1.0.1e 15.el6 of package openssl is installed.
The version 2.8.3 4.el6 of package fuse is installed.
The version 2.8.3 4.el6 of package fuse-libs is installed.
The version 5.5 49.el6 of package net-snmp-libs is installed.
The version 2.7.6 14.el6 of package libxml2 is installed.
The version 2.17.2 12.14.el6 of package libuuid is installed.
The version 4.2.1 9.1.el6_2 of package libicu is installed.
The version 2.6.6 51.el6 of package python is installed.
The version 4.2.0 0 of package gpfs.base is installed.
The version 4.2.0 0 of package gpfs.docs is installed.
The version 4.2.0 0 of package gpfs.gpl is installed.
The version 4.2.0 0 of package gpfs.msg.en_US is installed.
The version 2.9.6 1 of package lin_tape is installed.
IBM Spectrum Scale (GPFS) portability layer is installed.
IBM Spectrum Scale (GPFS) daemons are running on the local host.
All prerequisite packages are installed.
```

---

6. Run the `./ltfsee_install -i` command to perform the LTFS EE rpms installation. A successful installation shows the following message:

```
All rpm packages are installed successfully
```

## 3.5 Preparing the IBM Spectrum Scale file systems for IBM Spectrum Archive

Enable an IBM Spectrum Scale file system for IBM Spectrum Archive attachment as follows:

1. Shut down IBM Spectrum Scale on all protocol nodes that use the `cesNodes` node class:

```
mmshutdown -N cesNodes
```

2. Verify IBM Spectrum Scale is down on all protocol nodes, by using the `mmgetstate -a` command, as shown in Example 3-12.

---

*Example 3-12 Verify IBM Spectrum Scale is down on protocol nodes*

---

```
[root@boglin.ltf001st001 ~]# mmgetstate -a
```

Node number	Node name	GPFS state
1	nsd002st001	active
2	prt001st001	down
3	nsd001st001	active
4	prt002st001	down
5	prt003st001	down
6	prt004st001	down
7	ltfs001st001	active

---

3. Unmount, on all nodes, the file system (or file systems) you want DMAPI enabled on:

```
mmunmount fs1 -a
```

```
Tue Oct 27 07:58:04 MST 2015: mmunmount: Unmounting file systems ...
```

4. Enable DMAPI on the file system (or file systems) to be used with LTFS EE:

```
mmchfs fs1 -z yes
```

```
mmchfs: Propagating the cluster configuration data to all  
affected nodes. This is an asynchronous process.
```

5. Verify the DMAPI flag was set to yes on each file system by using the **mm1sfs** command, as shown in Example 3-13.

*Example 3-13 Verify that the DMAPI flag is set to yes on each file system*

---

```
[root@boglin.ltfs001st001 ~]# mm1sfs fs1 -z
```

flag	value	description
-z	yes	Is DMAPI enabled?

---

6. Start IBM Spectrum Scale on all protocol nodes:

```
mmstartup -N cesNodes
```

7. Verify IBM Spectrum Scale is active on all protocol nodes by using the **mmgetstate -a** command, as shown in Example 3-14.

*Example 3-14 Verify IBM Spectrum Scale is active on all protocol nodes*

---

```
[root@boglin.ltfs001st001 ~]# mmgetstate -a
```

Node number	Node name	GPFS state
1	nsd002st001	active
2	prt001st001	active
3	nsd001st001	active
4	prt002st001	active
5	prt003st001	active
6	prt004st001	active
7	ltfs001st001	active

---

8. Mount the DMAPI enabled file system on all nodes:

```
mmmout fs1 -a
```

9. Verify protocols are running on all protocol nodes, as shown in Example 3-15.

*Example 3-15 Verify protocols are running on all protocol nodes*

---

```
[root@boglin.ltfs001st001 ~]# mmces service list -a
```

```
Enabled services: OBJ SMB NFS
```

```
prt004st001: OBJ is running, SMB is running, NFS is running
```

```
prt002st001: OBJ is running, SMB is running, NFS is running
```

```
prt003st001: OBJ is running, SMB is running, NFS is running
```

```
prt001st001: OBJ is running, SMB is running, NFS is running
```

---



## 3.6 Configuring the IBM Spectrum Archive EE software

Configure the IBM Spectrum Archive Enterprise Edition on a single IBM Spectrum Archive node as follows:

1. Run the `/opt/ibm/ltfsee/bin/ltfsee_config -m CLUSTER` command. Follow the wizard to answer the various prompted questions.

**Note:** For the lab setup that was used to write this paper, only one IBM Spectrum Scale mount point (/fs1) was used. This single mount point for the file system was used to configure for IBM Spectrum Archive EE metadata and as the IBM Spectrum Scale shared file system. In a true production environment, you should have two distinct file systems in place to separate the IBM Spectrum Archive EE metadata content from the user data in the IBM Spectrum Scale shared file system.

For more information, see the following page in the IBM Knowledge Center:

<https://ibm.biz/BdHtsa>

2. Changing SSH daemon settings are suggested. The default values for MaxSessions and MaxStartups are too low and must be increased to allow for successful operations with IBM Spectrum Archive EE.
  - MaxSessions specifies the maximum number of open sessions permitted per network connection. The default is 10.
  - MaxStartups specifies the maximum number of concurrent unauthenticated connections to the SSH daemon. Additional connections will be dropped until authentication succeeds or the LoginGraceTime expires for a connection. The default is 10:30:100.

To change MaxSessions and MaxStartups values to 1024, use these steps:

- a. Edit the `/etc/ssh/sshd_config` file to set the values to 1024:

```
MaxSessions = 1024
```

```
MaxStartups = 1024
```

- b. Restart the **sshd** service.

**Note:** More IBM Spectrum Archive nodes may be added to the system. To add IBM Spectrum Archive nodes use the `/opt/ibm/ltfsee/bin/ltfsee_config -m ADD_NODE` on the node that is to be added.

For more details about IBM Spectrum Archive, see *IBM Linear Tape File System Enterprise Edition V1.1.1.2: Installation and Configuration Guide*, SG24-8143.

For the latest IBM Spectrum Archive product version support for IBM Spectrum Scale 4.2, contact your IBM sales representative.

## 3.7 Starting the IBM Spectrum Archive EE software

The start up and verification procedures are as follows:

1. Start the embedded LTFS system by using the following command:

```
ltfs /ltfs -o changer_devname=/dev/IBMchangerX
```

In this command, X is the tape library changer that you want to use for IBM Spectrum Archive EE. You can query all available IBM changers by using the following command:

```
cat /proc/scsi/IBMchanger
```

2. Verify the embedded LTFS system started successfully by using the following command:

```
df
```

If successfully started, the following message is displayed:

```
ltfs:/dev/IBMchanger0          2199023255040          0 2199023255040    0%  
/ltfs
```

3. Start IBM Spectrum Archive EE by using the following command:

```
/opt/ibm/ltfsee/bin/ltfsee start <Your GPFS file system>
```

## 3.8 Preparing IBM Spectrum Archive EE for migration

Tape cartridge pools are logical groupings of tape cartridges. The groupings might be based on their intended function (for example, OnsitePool and OffsitePool) or based on their content (for example, MPEGpool and JPEGpool) or simply for their redundancy (for example, primaryPool and copyPool).

To perform file migrations, you must first create and define tape cartridge pools, which are the targets for migrations. Then adding tape cartridges to or removing them from the tape cartridge pools is possible.

Consider the following guidelines for tape cartridge pools:

- ▶ Multiple jobs can be performed in parallel when more than one tape cartridge is defined in a tape cartridge pool. It is recommended to have multiple tape cartridges in each tape cartridge pool to increase performance.
- ▶ When tape cartridges are removed from a tape cartridge pool but not exported from LTFS EE, they are no longer targets for migrations. However, files on tape cartridges are still accessible for recalls.

Set up and perform object migrations to tape cartridge pools as follows:

1. You create tape cartridge pools by using the **create** option of the **ltfsee pool** command. For example, the following commands create tape cartridge pools named Primary and another one named Copy:

```
/opt/ibm/ltfsee/bin/ltfsee pool create Primary  
/opt/ibm/ltfsee/bin/ltfsee pool create Copy
```

2. You can confirm that the pools are created successfully by running the following command:

```
/opt/ibm/ltfsee/bin/ltfsee info pools
```

3. After the pools are created, tape cartridges must be added to the tape cartridge pools. You add tape cartridges by using the **add** option of the **ltfsee pool** command. To view the list of available tapes, use the **ltfsee info tapes** command.

If any of the tapes have a status of `Unavailable`, use the **ltfsee pool add** command to make them available to IBM Spectrum Archive EE. For example, the following commands add tape cartridges to the Primary and Copy pools:

```
/opt/ibm/ltfsee/bin/ltfsee pool add Primary PRI000JD PRI001JD PRI002JD
/opt/ibm/ltfsee/bin/ltfsee pool add Copy COP000JD COP001JD COP002JD
```

If these tapes are new and are not LTFS-formatted tapes, append the **-f** option to the end of the line as follows:

```
/opt/ibm/ltfsee/bin/ltfsee pool add Primary PRI000JD PRI001JD PRI002JD -f
/opt/ibm/ltfsee/bin/ltfsee pool add Copy COP000JD COP001JD COP002JD -f
```

4. You can confirm that tape cartridges were added to the appropriate pools by using the following command:

```
/opt/ibm/ltfsee/bin/ltfsee info tapes
```


If you plan to have multiple redundant copies on tape, be sure to modify the drive attributes by splitting them as follows:

- Half of the drives to perform migration jobs
- The other half of the drives to perform copy jobs

Use the **ltfsee drive add** and **ltfsee drive remove** commands. Drive attributes is a logical OR of the attributes: `migrate(8)`, `copy(4)`, `recall(2)`, and `generic(1)`. If the attribute is set, the corresponding job can be executed on that drive. Drive attributes can be specified after the colon character (:), which follows the drive serial number, and must be a decimal number. For example, 10 is the logical OR of `migrate(8)` and `recall(2)` so migration jobs and recall jobs are allowed to be executed on the corresponding drive. The following example is a command to set drive 1168001187 to perform *only* migration and recall jobs, and to add the drive to node 1:

```
/opt/ibm/ltfsee/bin/ltfsee drive add 1168001187:10 1
```





## Swift account, container, and object storage pool placement

This chapter provides instructions for ensuring that Swift account and container databases and related files are placed in a high-performance IBM Spectrum Scale tier.

Because Swift account and container databases are frequently accessed, placing them in a high-performance SSD based pool can improve overall performance.

This chapter also provides instructions for placing all newly created objects in an NL-SAS based silver tier while ensuring the custom metadata of objects is always placed in high-performance SSD based pool.

## 4.1 IBM Spectrum Scale storage pool configuration

This chapter uses the storage pool configuration listed in Example 4-1.

*Example 4-1 Storage pool configuration*

```
[root@Boglin.prt002st001 ~]$ mmdf /dev/fs1
disk          disk size  failure holds   holds          free KB          free KB
name          in KB     group metadata data          in full blocks  in fragments
-----
Disks in storage pool: system (Maximum disk size allowed is 51 TB)
nsd1          6836715520      1 Yes      Yes      6825921536 (100%)      13088 ( 0%)
-----
(pool total)   6836715520                                6825921536 (100%)      13088 ( 0%)

Disks in storage pool: gold (Maximum disk size allowed is 51 TB)
nsd6          6836715520      0 No      Yes      6836646912 (100%)      2624 ( 0%)
-----
(pool total)   6836715520                                6836646912 (100%)      2624 ( 0%)

Disks in storage pool: silver (Maximum disk size allowed is 51 TB)
nsd2          6836715520      0 No      Yes      6836647936 (100%)      1888 ( 0%)
nsd4          6836715520      0 No      Yes      6836647936 (100%)      1888 ( 0%)
nsd5          6836715520      0 No      Yes      6836647936 (100%)      1888 ( 0%)
nsd3          6836715520      0 No      Yes      6836647936 (100%)      1888 ( 0%)
-----
(pool total)   27346862080                                27346591744 (100%)      7552 ( 0%)

=====
(data)         41020293120                                41009160192 (100%)      23264 ( 0%)
(metadata)     6836715520                                6825921536 (100%)      13088 ( 0%)
=====
(total)        41020293120                                41009160192 (100%)      23264 ( 0%)

Inode Information
-----
Total number of used inodes in all Inode spaces:      63932
Total number of free inodes in all Inode spaces:      938564
Total number of allocated inodes in all Inode spaces: 1002496
Total of Maximum number of inodes in all Inode spaces: 34706176
[root@Boglin.prt002st001 ~]$
```

**Note:** The custom metadata of objects is stored as extended attributes in the inodes of the IBM Spectrum Scale file system. IBM Spectrum Scale places inodes in pools that are enabled to store metadata. In this example, the system pool backed by SSD is the only pool that is enabled to store metadata; therefore, all custom metadata of objects will be stored in the system pool.

## 4.2 Container and account placement policies

To implement the placement policies, a new *dependent fileset* must be created to hold all account and container data. The system pool is then used that is backed by SSD. A file placement policy must then be written and deployed so that all files that are created in the new dependent fileset automatically are placed into the SSD backed system storage pool.

The IBM Spectrum Scale 4.2 protocols installer sets the Swift **devices** parameter in both account server and container server configuration files to use a directory named *ac* under the object fileset. The Swift **devices** parameter in the object server configuration file is set to use a directory that is named *o* under the object fileset. This way allows the account and container-dependent fileset to be easily linked to the *ac* directory. Example 4-2 shows the fileset configuration.

*Example 4-2 Fileset configuration*

---

```
[root@Boglin.ltf001st001 ~]$ mmlsfileset /dev/fs1
```

Filesets in file system 'fs1':

Name	Status	Path
root	Linked	/ibm/fs1
Object_Fileset	Linked	/ibm/fs1/Object_Fileset

```
[root@Boglin.ltf001st001 ~]$
```

---

To configure the account and container filesets, complete the following steps:

1. Create the `object_acfs` dependent fileset for account and container data by running the command shown in Example 4-3.

*Example 4-3 Create dependent fileset*

---

```
mmlcrfileset fs1 obj_acfs --inode-space Object_Fileset
Fileset object_acfs created with id 3 root inode 1085441.
```

---

2. Link the new dependent fileset as *ac* under the `Object_Fileset` independent fileset as indicated in Example 4-4.

*Example 4-4 Link dependent fileset*

---

```
mmlinkfileset fs1 obj_acfs -J /ibm/fs1/Object_Fileset/ac
Fileset obj_acfs linked at /ibm/fs1/Object_fileset/ac
```

---

3. Create a `placement.policy` file as indicated in Example 4-5.

*Example 4-5 Create placement policy*

---

```
/* policy rules to place account and container files into the system pool */
/* PLACEMENT policy 1 - put account and container files in system system is
previously created storage pool and obj_acfs is previously created dependent
fileset */
RULE 'db files' SET POOL 'system' FOR FILESET('obj_acfs')
/* Default placement policy rule */
RULE 'default' SET POOL 'silver'
```

---

4. To test the policy, run the command shown in Example 4-6.

*Example 4-6 Test Placement policy*

---

```
mmchpolicy gpfs placement.policy -I test
Validated policy 'placement.policy': Parsed 2 policy rules.
```

---

5. If no errors are detected, apply the policy by running the command listed in Example 4-7.

*Example 4-7 Apply policy*

---

```
mmchpolicy gpfs placement.policy -I yes
Validated policy 'placement.policy': Parsed 2 policy rules.
Policy 'placement.policy' installed and broadcast to all nodes.
```

---

## 4.3 Object placement and migration policies

This section describes configuring ILM policies that configure the default storage pool placement of objects during creation and also migrating existing objects to specific storage pools.

The IBM Spectrum Scale 4.2 release introduces object storage policies that allow you to control the use of features such as compression and replication. By using the **mmobj** command, you can define policies and also define which features to enable for the policy. By default, a policy is created with no advanced features. The policies are then applied to Swift containers and S3 buckets upon creation. For more details about IBM Spectrum Scale object storage policies, see the following page in the IBM Knowledge Center:

<https://ibm.biz/BdH9jB>

This lab environment uses a default Swift storage policy that maps to Object\_Fileset and also a custom policy, named tapeTier, that maps to the obj\_tapeTier fileset. The Object\_Fileset and obj\_tapeTier filesets are then used in the IBM Spectrum Scale object placement and object migration policies. Example 4-8 illustrates this configuration.

*Example 4-8 List object storage policies and filesets*

---

```
mmobj policy list --verbose
```

Index Details	Name	Default	Deprecated	Fileset	Fileset Path	Functions	Function
0	SwiftDefault	yes		Object_Fileset	/ibm/fs1/Object_Fileset		
44831511210	tapeTier			obj_tapeTier	/ibm/fs1/obj_tapeTier	default	regions="1"

---

**Note:** To create custom Swift storage policies, see Example 6-1 on page 44.

### Placement and migration ILM policies for default Swift storage policies

This section provides instructions for placement of objects upon creation and also migration to the silver pool of existing objects, by using the default Swift storage policy.



**Note:** When registering automated policies such as file placement and capacity threshold-based migration, all rules must be combined into a single policy file followed by issuing the **mmapplypolicy** to register the policies in the policy file with the IBM Spectrum Scale file system. In Example 4-9, the object placement rule is inserted into the `placement.policy` file after the account and container policy but before the default policy.

The policy engine will scan the file system based on the order of the rules in the `placement.policy` file so that anything that falls out of the first two rules will be run through the default placement rule.

The steps are as follows:

1. Example 4-9 shows the object placement policy, highlighted in **bold**, for the base Object\_Fileset, which is inserted between the account and container placement policy and the default placement policy.

*Example 4-9 Object placement policy*

---

```
/* PLACEMENT policy 1 - put account and container files in system pool upon
creation. System is previously created storage pool and obj_acfs is previously
created dependent fileset */
RULE 'db files' SET POOL 'system' FOR FILESET('obj_acfs')

/* PLACEMENT policy 2 - put objects in the silver pool upon creation. Silver
is previously created storage pool and Object_Fileset is previously created
independent fileset */

RULE 'obj data files' SET
POOL silver FOR
FILESET('Object_Fileset')
WHERE NAME LIKE 'tmp%' OR NAME
LIKE '%.data'

/* Default placement policy rule */
RULE 'default' SET POOL 'silver'
```

---

2. To test the policy, run the command shown in Example 4-10.

*Example 4-10 Test placement policy*

---

```
mmchpolicy fs1 placement.policy -I test
Validated policy 'placement.policy': Parsed 3 policy rules
```

---

3. If no errors are detected, apply the policy by running the command listed in Example 4-11.

*Example 4-11 Run placement policy*

---

```
mmchpolicy fs1 placement.policy -I yes
Validated policy 'placement.policy': Parsed 3 policy rules.
Policy 'placement.policy' installed and broadcast to all nodes.
```

---

Migrate existing objects to the silver pool as follows:

1. Create a file named `object_migration` and add the rule listed in Example 4-12.

*Example 4-12 Object migration policy*

---

```
/*policy rule to migrate existing objects from the system pool to the silver
pool*/
RULE 'obj_data_migration_policy'
MIGRATE FROM POOL system TO POOL silver FOR FILESET('Object_Fileset')
WHERE NAME LIKE 'tmp%' OR NAME LIKE '%.data'
```

---

2. Test the policy by running the command shown in Example 4-13.

*Example 4-13 Test migration policy*

---

```
mmapplypolicy fsl -P object_migration -I test
[I] GPFS Current Data Pool Utilization in KB and %
Pool_Name          KB_Occupied      KB_Total    Percent_Occupied
gold               169462784      6836715520   2.478716330%
silver             137216         13673431040  0.001003523%
system            6985728        13673431040  0.051089796%
[I] 5665 of 34706176 inodes used: 0.016323%.
[I] Loaded policy rules from object_migration.
Evaluating policy rules with CURRENT_TIMESTAMP = 2015-11-21@02:32:10 UTC
Parsed 1 policy rules.
RULE 'obj_data_migration_policy'
MIGRATE FROM POOL system TO POOL silver FOR FILESET('Object_Fileset')
WHERE NAME LIKE 'tmp%' OR NAME LIKE '%.data'
[I] 2015-11-21@02:32:10.252 Directory entries scanned: 1560.
[I] Directories scan: 1223 files, 337 directories, 0 other objects, 0 'skipped'
files and/or errors.
[I] 2015-11-21@02:32:10.255 Sorting 1560 file list records.
[I] Inodes scan: 1222 files, 337 directories, 1 other objects, 1 'skipped'
files and/or errors.
[I] 2015-11-21@02:32:10.479 Policy evaluation. 1560 files scanned.
[I] 2015-11-21@02:32:10.482 Sorting 1 candidate file list records.
[I] 2015-11-21@02:32:10.482 Choosing candidate files. 1 records scanned
[I] Summary of Rule Applicability and File Choices:
Rule#      Hit_Cnt      KB_Hit      Chosen      KB_Chosen
KB_
  0          1          0          1          0

[I] Filesystem objects with no applicable rules: 1559.

[I] GPFS Policy Decisions and File Choice Totals:
Chose to migrate OKB: 1 of 1 candidates;
Predicted Data Pool Utilization in KB and %:
Pool_Name          KB_Occupied      KB_Total    Percent_Occupied
gold               169462784      6836715520   2.478716330%
silver             136192         13673431040  0.000996034%
system            6985728        13673431040  0.051089796%
```

---

3. If no errors are detected, apply the policy by running the command listed in Example 4-14.

*Example 4-14 Run migration policy*

---

```
mmapplypolicy fs1 -P object_migration -I yes
```

---

## Placement and migration ILM policies for custom Swift storage policies

This section provides instructions for initial placement of objects and migration to the silver pool of existing objects, by using the tapeTier custom Swift storage policy.

**Note:** All rules must be combined into a single policy file followed by issuing the **mmapplypolicy** command to register the policies in the policy file with the IBM Spectrum Scale file system. In Example 4-15 the object placement rule is inserted into the `placement.policy` file after placement policy 2 but before the default policy.

1. Example 4-15 shows the object placement policy, highlighted in **bold**, for the base Object\_Fileset, which is inserted between the account and container placement policy and the default placement policy.

*Example 4-15 Object placement*

---

```
/* PLACEMENT policy 1 - put account and container files in system pool upon
creation. System is previously created storage pool and obj_acfs is previously
created dependent fileset */
RULE 'db files' SET POOL 'system' FOR FILESET('obj_acfs')
```

```
/* PLACEMENT policy 2 - put objects in the silver pool upon creation.
Silver is previously created storage pool and Object_Fileset is previously
created independent fileset */
```

```
RULE 'obj data files' SET POOL silver FOR FILESET('Object_Fileset')
WHERE NAME LIKE 'tmp%' OR NAME LIKE '%.data'
```

```
/* PLACEMENT policy 3 - put objects in the silver pool upon creation.
Silver is previously created storage pool and obj_tapeTier is previously
created independent fileset */
```

```
RULE 'obj data files' SET
POOL silver FOR
FILESET('obj_tapeTier')
WHERE NAME LIKE 'tmp%' OR NAME
LIKE '%.data'
```

```
/* Default placement policy rule */
RULE 'default' SET POOL 'silver'
```

---

2. To test the policy, run the command shown in Example 4-16.

*Example 4-16 Test placement policy*

---

```
mmchpolicy fs1 placement.policy -I test
Validated policy 'placement.policy': Parsed 4 policy rules.
```

---

3. If no errors are detected, apply the policy by running the command listed in Example 4-17.

*Example 4-17 Run placement policy*

---

```
mmchpolicy fs1 placement.policy -I yes
Validated policy 'placement.policy': Parsed 4 policy rules. Policy
'placement.policy' installed and broadcast to all nodes
```

---

Migrate existing objects to the silver pool as follows:

1. Create a file named `object_migration` and add the rule listed in Example 4-18.

*Example 4-18 Object migration policy*

---

```
/*policy rule to migrate existing objects from the system pool to the silver
pool*/
RULE 'obj_data_migration_policy'
MIGRATE FROM POOL system TO POOL silver FOR FILESET('Object_Fileset')
WHERE NAME LIKE 'tmp%' OR NAME LIKE '%.data'
```

---

2. To test the policy, run the command shown in Example 4-19.

*Example 4-19 Test migration policy*

---

```
mmapplypolicy fs1 -P object_migration -I test
```

---

3. If no errors are detected, apply the policy by running the command listed in Example 4-20.

*Example 4-20 Run migration policy*

---

```
mmapplypolicy fs1 -P object_migration -I yes
```

---

## Additional information

When OpenStack Swift creates new objects, it first creates a temporary file similar to the following format:

```
/ibm/fs1/object_fileset/o/z1device42/tmp/tmp2abp6j
```

Swift then renames the temporary file to a file that ends with `.data` similar to this example:

```
/ibm/fs1/object_fileset/o/z1device42/objects/6736/ed3/69434535e56fe8ca540853059
cddaed3/1440445383.27850.data
```

Similarly, when creating new containers, Openstack Swift first creates temporary files similar to the following format:

```
/ibm/fs1/object_fileset/ac/z1device8/containers/8060/c2d/7df38a9844b513122cbc2
a380387fc2d/tmpEYE21r.tmp
```

Swift then renames the `*.tmp` file to a file that ends with `.db` similar to this example:

```
/ibm/fs1/object_fileset/ac/z1device8/containers/8060/c2d/7df38a9844b513122cbc2
a380387fc2d/7df38a9844b513122cbc2a380387fc2d.db
```

The policies in this chapter are designed to ensure that temporary object and permanent object files are placed in the silver pool, and that temporary and permanent database files are placed in the system pool.



## Object heatmap data tiering

Most data that you access on a daily basis in the IBM Spectrum Scale and IBM Spectrum Archive enabled object store is sitting on disk. This chapter describes how to use IBM Spectrum Scale heatmap tiering policies for data that is accessed on a daily basis and that will automatically place frequently accessed objects on higher performing storage pools such as the SSD-based pool. The heatmap policy also automatically migrates less frequently accessed objects to slower disk-based storage pools.

As data ages and is no longer accessed daily, it can be migrated from the slow disk-based storage pool to tape, as described in Chapter 6, “Storing objects in the IBM Spectrum Archive tape tier” on page 43.

## 5.1 Enabling heatmap

Example 5-1 shows the enabling of heatmap tracking on the IBM Spectrum Scale file system.

*Example 5-1 Enable file system heatmap tracking*

```
[root@Boglin.nsd001st001 ~]$ mmchconfig
fileheatperiodminutes=1440,fileheatlosspercent=10
mmchconfig: Command successfully completed
mmchconfig: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
[root@Boglin.nsd001st001 ~]$
```

## 5.2 Object heatmap policy

Objects in the IBM Spectrum Scale file system with the \*.data file type are good candidates for the heatmap migration policy. This policy will manage the placement of objects denoted with the \*.data file type between the system, gold, and silver pools and specifies a maximum capacity threshold to ensure capacity is not over utilized in these storage tiers.

Complete the following steps:

1. Create a file named `file_heat_policy` and add the policy listed in Example 5-2. The policy will place the most frequently accessed objects in the SSD-backed system pool until the system pool reaches 70% capacity utilization. Hot objects will also be placed in the gold pool until it reaches 75% capacity utilization.

*Example 5-2 Object migration policy*

```
/* Define pool group using three on-line pools*/
RULE 'DefineTiers' GROUP POOL 'TIERS'
    IS 'system' LIMIT(70)
    THEN 'gold' LIMIT(75)
    THEN 'silver'
RULE 'Rebalance' MIGRATE FROM POOL 'TIERS' TO POOL 'TIERS' WEIGHT(FILE_HEAT)
FOR FILESET('Object_Fileset')
WHERE NAME LIKE '%.data'
```

**Unnecessary data movement:** The heatmap policy does not attempt to migrate temporary files generated by Swift between storage tiers because the temporary files are ultimately replaced by permanent files with the .data extension. Attempting to migrate temporary files might result in unnecessary data movement.

2. To test the policy, run the command shown in Example 5-3.

*Example 5-3 Test migration policy*

```
mmapplypolicy fsl -P object_heat_policy -I test
[I] GPFS Current Data Pool Utilization in KB and %
Pool_Name           KB_Occupied      KB_Total  Percent_Occupied
gold                169462784       6836715520  2.478716330%
silver              136192          13673431040  0.000996034%
system              8990720         13673431040  0.065753211%
[I] 6050 of 42706176 inodes used: 0.014167%.
```

```

[I] Loaded policy rules from object_heat_policy.
Evaluating policy rules with CURRENT_TIMESTAMP = 2015-11-22@02:30:19 UTC
Parsed 2 policy rules.
RULE 'DefineTiers' GROUP POOL 'TIERS'
      IS 'system' LIMIT(70)
      THEN 'gold' LIMIT(75)
      THEN 'silver'

RULE 'Rebalance' MIGRATE FROM POOL 'TIERS' TO POOL 'TIERS'
WEIGHT(computeFileHeat(CURRENT_TIMESTAMP-ACCESS_TIME,xattr('gpfs.FileHeat'),KB_
ALLOCATED))
FOR FILESET('Object_Fileset') WHERE NAME LIKE '%.data'
[I] 2015-11-22@02:30:20.045 Directory entries scanned: 1945.
[I] Directories scan: 1223 files, 594 directories, 128 other objects, 0
'skipped' files and/or errors.
[I] 2015-11-22@02:30:20.050 Sorting 1945 file list records.
[I] Inodes scan: 1223 files, 594 directories, 128 other objects, 0 'skipped'
files and/or errors.
[I] 2015-11-22@02:30:20.345 Policy evaluation. 1945 files scanned.
[I] 2015-11-22@02:30:20.350 Sorting 1 candidate file list records.
[I] 2015-11-22@02:30:20.437 Choosing candidate files. 1 records scanned.
[I] Summary of Rule Applicability and File Choices:
  Rule#      Hit_Cnt      KB_Hit      Chosen      KB_Chosen
KB_I11      Rule
      0          1          0          1          0
0      RULE 'Rebalance' MIGRATE FROM POOL 'TIERS' WEIGHT(.) TO POOL 'TIERS' FOR
FILESET(.) WHERE(.)

[I] Filesystem objects with no applicable rules: 1944.

[I] GPFS Policy Decisions and File Choice Totals:
  Chose to migrate OKB: 1 of 1 candidates;
Predicted Data Pool Utilization in KB and %:
Pool_Name      KB_Occupied      KB_Total      Percent_Occupied
gold            169462784      6836715520      2.478716330%
silver          136192      13673431040      0.000996034%
system          8990720      13673431040      0.065753211%

```

3. If no errors are detected, apply the policy by running the command shown in Example 5-4.

*Example 5-4 Run migration policy*

---

```
mmapplypolicy fsl -P object_file_heat -I yes
```

---







## Storing objects in the IBM Spectrum Archive tape tier

This chapter describes the two main methods for using the IBM Spectrum Archive tape tier in the IBM Spectrum Scale object store:

- One method is to have the application or user explicitly copy data to specific S3 buckets and containers where the S3 buckets and Swift containers use IBM Spectrum Scale object storage policies that immediately migrate the entire content of the buckets and containers to tape. This method might be useful in creating a global archive for an entire data center, where the IBM Spectrum Scale and IBM Spectrum Archive object store is primarily targeted toward cold data that will be immediately put on tape. The data put into the S3 buckets and Swift containers is not actively accessed and is not treated as an active resource.

The main advantage of this method is that fine grain control of the migration of data on S3 bucket and Swift container boundaries is provided. The main disadvantage is that data must be explicitly copied into the S3 buckets and Swift containers that immediately move the content to tape, which requires user or application awareness of the tape tier.

- Another method creates an online active archive that provides a single namespace to contain warm and cold data. Data can be migrated to tape in a selective, automated, and sweeping manner across all buckets and containers without application awareness. In this scenario, some of the data S3 buckets and Swift containers might reside on disk and some of the older data in the S3 buckets and Swift containers might reside on tape.

The main advantage to this method is that no data copy or movement by the user or application is needed. The main disadvantage is that there is no S3 bucket or Swift container boundary to control the movement of data to the tape tier, and as a result, determining which objects reside on tape and must first be recalled in order to avoid application time-outs becomes more difficult.

## 6.1 Container-based migration of data to tape

This example creates a special archive S3 bucket that puts all the data objects in the bucket onto IBM Spectrum Archive tape. This is accomplished by creating a Swift storage policy that maps to a Swift tape device ring. The tape device ring maps to a specific region on the IBM Spectrum Scale file system so that the IBM Spectrum Scale information lifecycle policy engine can be used to identify these objects on the file system, and IBM Spectrum Archive can be used to move them to physical tape. Figure 6-1 illustrates the migration path from the archive S3 bucket to IBM Spectrum Archive tape.

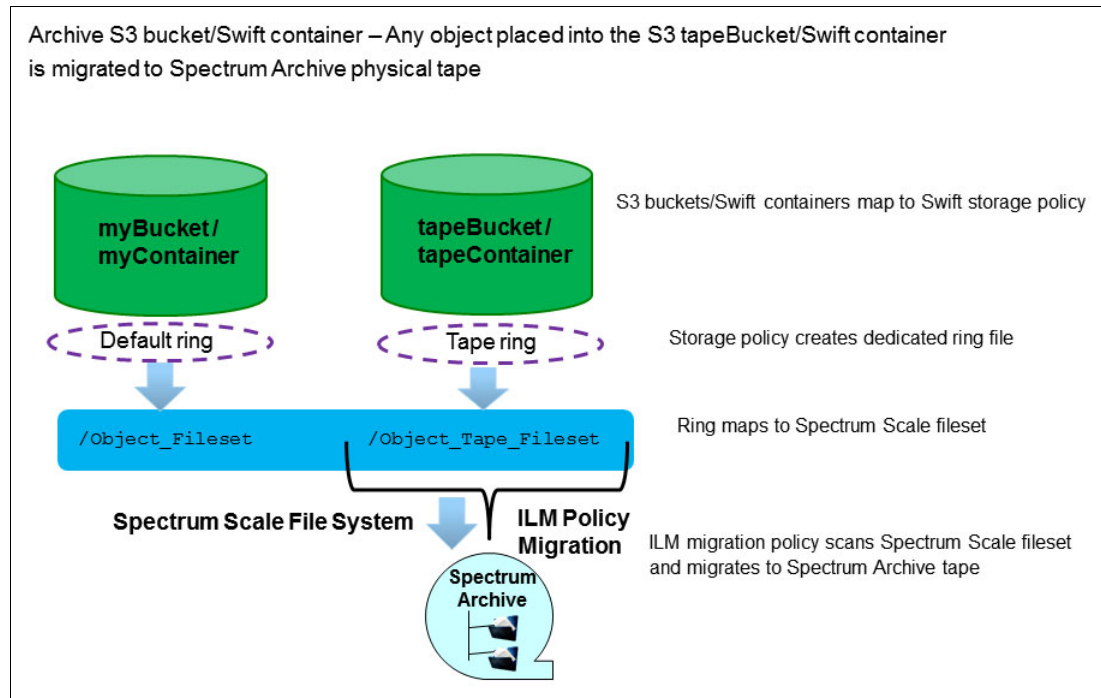


Figure 6-1 Archiving the S3 tapeBucket and Swift container to IBM Spectrum Archive tape tier

### Define object storage policy

Example 6-1 creates an object storage policy by using the **mmobj** command. This command must be run from a protocol node.

#### Example 6-1 Setting Storage Policies

```
mmobj policy create tapeTier
[I] Getting latest configuration from ccr
[I] Creating fileset /dev/fs1:obj_tapeTier
[I] Creating new unique index and building the object rings
[I] Updating the configuration
[I] Uploading the changed configuration
```

Check the policy-to-fileset mapping as indicated in Example 6-2 on page 45.

#### Example 6-2 List Object Storage Policies and Filesets

```
mmobj policy list --verbose
```

Index	Name	Default	Deprecated	Fileset	Fileset Path	Functions	Function Details
0	SwiftDefault	yes		Object_Fileset	/ibm/fs1/Object_Fileset		
44831511210	tapeTier			obj_tapeTier	/ibm/fs1/obj_tapeTier	default	regions="1"

### Create Swift container that maps to tape storage policy

Now, create a new container that maps to the tape storage policy you previously created. This example uses the **swift** command-line-tool from a protocol node.

Example 6-3 shows the syntax for this command.

#### Example 6-3 Create tape storage policy backed Swift container

```
swift post tapeContainer -H 'X-Storage-Policy: tapeTier'
```

**Note:** Swift object storage policies can be applied to Swift containers and S3 buckets only upon container or bucket creation. The Swift container or S3 bucket must not exist prior to specifying this command.

Validate that the X-Storage-Policy is set to tapeTier by using the **swift** command-line tool from a protocol node as listed in Example 6-4 (the storage policy highlighted in **bold**).

#### Example 6-4 Validate Tape Storage Policy Backed Swift Container

```
swift stat tapeContainer
  Account: AUTH_7ec0250241ba4b3ba6b00a3735400ad1
  Container: tapeContainer
  Objects: 0
  Bytes: 0
  Read ACL:
  Write ACL:
  Sync To:
  Sync Key:
  Accept-Ranges: bytes
X-Storage-Policy: tapeTier
  Connection: keep-alive
  X-Timestamp: 1448447822.16119
  X-Trans-Id: tx5155df73b08242599d4ac-005655cf45
  Content-Type: text/plain; charset=utf-8
```

### Put objects in tape bucket

Create 20 x 50 MB files and then upload them to the tape-backed Swift container (tapeContainer) by using the **swift** command line. Example 6-5 creates the first 50 MB file.

#### Example 6-5 Create 50 MB file

```
[root@Boglin.prt001st001 joe]$ dd if=/dev/urandom of=tapeObj1 bs=1M count=50
50+0 records in
50+0 records out
52428800 bytes (52 MB) copied, 4.12615 s, 12.7 MB/s
[root@Boglin.prt001st001 joe]$
```

Repeat this command 19 times, each time incrementing the number of the output file (of) as indicated in Example 6-6.

*Example 6-6 Incrementing output file tapeObjN to tapeObjN+1*

---

```
[root@Boglin.prt001st001 joe]$ dd if=/dev/urandom of=tapeObj2 bs=1M count=50
50+0 records in
50+0 records out
52428800 bytes (52 MB) copied, 4.12192 s, 12.7 MB/s
[root@Boglin.prt001st001 joe]$
```

---

Repeat the command to generate a desired amount of data. Now put the data in the Swift container named tapeContainer by issuing the command in Example 6-7.

*Example 6-7 Put object to tape bucket*

---

```
swift upload tapeContainer tapeObj1
//Repeat for each object by incrementing the object number.
swift upload tapeContainer tapeObj2
```

---

## ILM migration policy

Example 6-8 creates an IBM Spectrum Scale policy to select *only* the .data files that are over 1 MB in size and in the tape-backed container named tapeContainer, which maps to the fileset obj\_tapeTier. The other metadata files should remain on disk to allow for GETs on metadata information without causing recalls from IBM Spectrum Archive EE tapes.

**Note:** The ILM migration policy file must be created on an IBM Spectrum Archive node. The suggestion is to migrate only objects that are 1 MB or larger to tape.

*Example 6-8 Policy to select only .data files that are over 1 MB in size and in the tape tier bucket*

---

```
define(is_premigrated,(MISC_ATTRIBUTES LIKE '%M%' AND MISC_ATTRIBUTES NOT LIKE '%V%'))
define(is_migrated,(MISC_ATTRIBUTES LIKE '%V%')) define(is_resident,(NOT MISC_ATTRIBUTES Like
'%M%')) define(MB,1048576)

RULE EXTERNAL POOL 'objects'
EXEC '/opt/ibm/ltfsee/bin/ltfsee' OPTS 'Primary Copy'

RULE 'OBJECT_FILES' MIGRATE FROM POOL 'silver'
TO POOL 'objects'
FOR FILESET ('obj_tapeTier') WHERE FILE_SIZE >= (1*MB)
AND NAME LIKE '%.data'
(is_resident OR is_premigrated)
```

---

You can execute the policy by running the **mmapplypolicy** command indicated in Example 6-9.

**Note:** The **mmapply policy** command must be run on the IBM Spectrum Archive node.

*Example 6-9 Policy invocation*

---

```
mmapplypolicy <GPFS file system>/Object_Fileset/o -P <policy text file> -B 50000 -m 1
```

---

To verify the placement of the objects in the IBM Spectrum Archive pool, use the steps indicated in 8.2, “Determining the location of objects in the active archive” on page 58.

## 6.2 Sweeping migration of data to tape

This method does not require configuring Swift storage policies to map specific containers to IBM Spectrum Archive tape. Instead, all objects in all containers and buckets are scanned upon policy invocation. Figure 6-2 illustrates the sweeping migration of data to IBM Spectrum Archive tape.

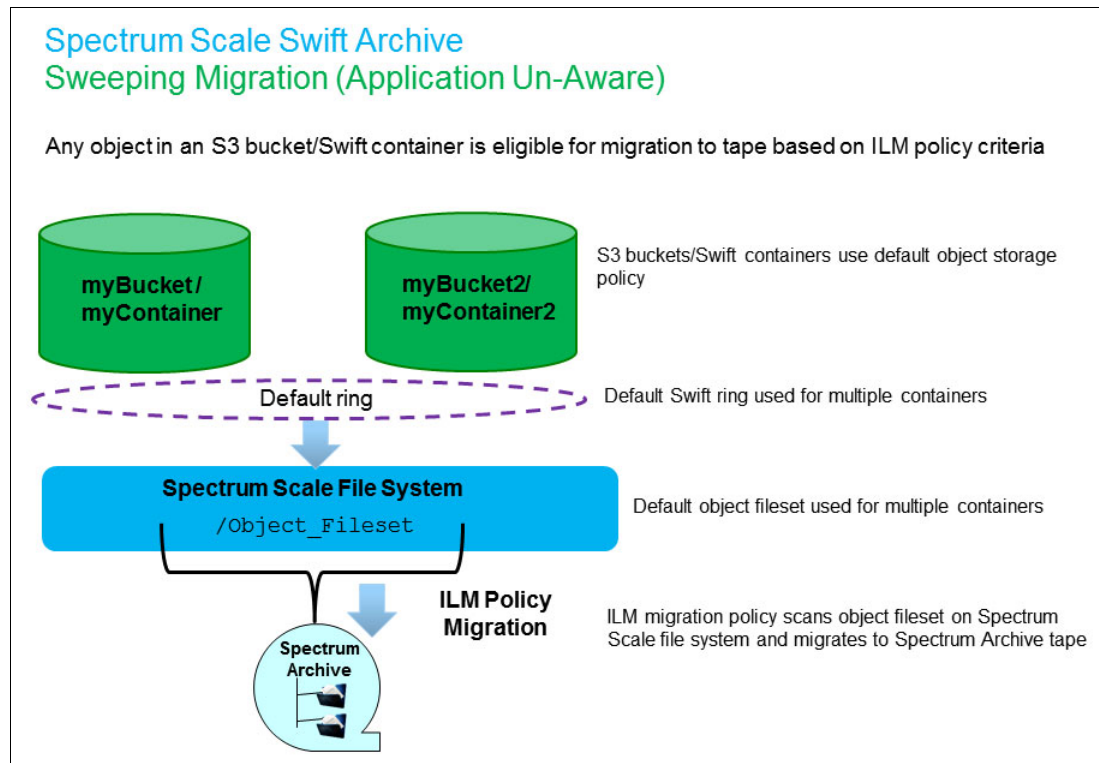


Figure 6-2 Sweeping migration of data to IBM Spectrum Archive tape

**Note:** The ILM migration policy file must be created on an IBM Spectrum Archive node. The suggestion is to migrate only objects that are 1 MB or larger to tape.

Example 6-10 shows an ILM policy

*Example 6-10 Policy to select only .data files that are over 1 MB in size across all S3 buckets and Swift containers*

```
define(is_premigrated,(MISC_ATTRIBUTES LIKE '%M%' AND MISC_ATTRIBUTES NOT LIKE '%V%'))
define(is_migrated,(MISC_ATTRIBUTES LIKE '%V%')) define(is_resident,(NOT MISC_ATTRIBUTES LIKE
'%M%')) define(MB,1048576)
```

```
RULE EXTERNAL POOL 'objects'
EXEC '/opt/ibm/ltfsee/bin/ltfsee' OPTS 'Primary Copy'
```

```
RULE 'OBJECT_FILES' MIGRATE FROM POOL 'silver'
TO POOL 'objects'
FOR FILESET ('Object_Fileset') WHERE FILE_SIZE >= (1*MB)
AND PATH_NAME LIKE '%z1device%' AND NAME LIKE '%.data'
AND (is_resident OR is_premigrated)
```

You can execute the policy by running the **mmapplypolicy** command (Example 6-11).

**Note:** The **mmapplypolicy** command must be run on the IBM Spectrum Archive node.

*Example 6-11 Policy invocation*

---

```
mmapplypolicy <GPFS file system>/Object_Fileset/o -P <policy text file> -B 50000 -m 1
```

---

To verify the placement of the objects in the IBM Spectrum Archive pool, use the steps indicated in 8.2, “Determining the location of objects in the active archive” on page 58.



## Retrieving objects from the IBM Spectrum Archive tape tier

This chapter describes two method for recalling objects from the IBM Spectrum Archive tape tier:

- ▶ The *application transparent recall* method uses the ability of IBM Spectrum Scale and IBM Spectrum Archive to automatically fetch the object from tape upon receiving a GET request in a manner that is transparent to the application. To accommodate tape recall times, Swift object and application timeouts are increased.
- ▶ The *command line pre-fetched recall* method optimizes bulk recalls from tape by using the IBM Spectrum Scale command line to pre-fetch objects from tape before issuing REST API GET requests. This method does not require increasing Swift and application timers.

These methods for retrieving objects from the IBM Spectrum Archive tape tier can be used individually or in combination. For example, you might want to use only the command-line pre-fetched recall method so that Swift and application timers do not need to be increased. Additionally, you might choose to increase Swift and application timers to provide application-transparent recall for small objects and might also use the command-line pre-fetched recall method for large objects.

## 7.1 Application-transparent object recall from tape

This section shows how to modify various time-out parameters in Swift configuration files to accommodate the recall of data from IBM Spectrum Archive tape. This method uses the ability of IBM Spectrum Scale and IBM Spectrum Archive to automatically fetch data from tape upon receiving a GET request, transparently to the application.

The amount of time spent to recall an object from tape depends on various factors such as the number of objects in the recall queue, the speed of the tape drive technology, and the size of the objects. As a result of these factors, in certain cases, application-transparent **GET** requests might encounter time-out conditions if the data cannot be recalled from tape within the allotted time. If a time-out condition occurs, the object continues to be recalled to disk even after the time-out condition occurs, therefore re-issuing the **GET** request after the object is resident in disk is possible. To avoid encountering these time-out conditions, command-line methods are provided to prefetch the objects from tape so that they are resident in disk prior to issuing the Swift and S3 based **GET** commands.

### Modify Swift timers

Example 7-1 modifies Swift timers.

*Example 7-1 Adjust Swift timers*

---

```
mmces service stop obj

mmobj config change --ccrfile proxy-server.conf --section DEFAULT --property node_timeout
--value 1200
mmobj config change --ccrfile proxy-server.conf --section app:proxy-server --property
node_timeout --value 600
mmobj config change --ccrfile proxy-server.conf --section app:proxy-server --property
conn_timeout --value 600
mmobj config change --ccrfile object-server.conf --section DEFAULT --property node_timeout
--value 1200
mmobj config change --ccrfile object-server.conf --section DEFAULT --property conn_timeout
--value 600
mmces service start obj
```

---

## 7.2 Prefetched recall of object data from tape

This section provides instructions for using the IBM Spectrum Scale command line to prefetch objects from tape prior to issuing Swift and S3 REST API GET requests, in order to avoid encountering time-out conditions, and to avoid having to change Swift and application timers. The method entails providing a large list of objects to be recalled to IBM Spectrum Archive, which allows IBM Spectrum Archive to recall the objects in an optimal order according to the placement of the objects on tape cartridges, in order to provide optimized fast recall.

**Note:** Currently this process involves determining the location of the objects on the underlying IBM Spectrum Scale system, which must be performed on a protocol node because it uses built-in Swift utilities. The location of the objects on the underlying file system to be recalled is put into a file that is copied to the *IBM Spectrum Archive node*. An **ltfsee** command is then issued from the *IBM Spectrum Archive node* to recall the objects using the file that contains the location of the objects on the underlying file system to be recalled.



The method described in this section does not use the IBM Spectrum Scale unified file and object feature. That feature can dramatically simplify the recall process as objects are placed on the underlying IBM Spectrum Scale file system in a manner that eliminates the need to perform the swift stat and swift-get-nodes object-to-file translation described in this section. In your decision of whether to use the unified file and object rather than the traditional method disclosed here, consider the current unified file and object limitations and constraints listed at the following web page:

<https://ibm.biz/BdH9sd>

## Determine Account token

The Account value provided by the swift stat output is required to map objects to files on the file system. Obtain the Account value by loading authorization credentials in the openrc file as shown in Example 7-2 followed by issuing a Swift **stat** command.

*Example 7-2 Sample openrc content*

---

```
[root@Boglin.prt002st001 ~]$ cat openrc
# Mon Jun  8 12:20:43 MST 2015
export OS_AUTH_URL="http://127.0.0.1:35357/v3"
export OS_IDENTITY_API_VERSION=3
export OS_AUTH_VERSION=3
export OS_USERNAME="admin"
export OS_PASSWORD="PasswOrd"
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_NAME=admin
export OS_PROJECT_DOMAIN_NAME=Default
```

---

Example 7-3 obtains the Account token.

*Example 7-3 Obtain Account token*

---

```
[root@Boglin.prt002st001 ~]$ source openrc

[root@Boglin.prt002st001 ~]$ swift stat
      Account: AUTH_cd3eae79e2b049e5a9a292688729a775
      Containers: 3
      Objects: 21
      Bytes: 115098151
Containers in policy "generic": 2
  Objects in policy "generic": 20
  Bytes in policy "generic": 104858151
Containers in policy "tapetier": 1
  Objects in policy "tapetier": 1
  Bytes in policy "tapetier": 10240000
      Accept-Ranges: bytes
      X-Account-Project-Domain-Id: default
      X-Timestamp: 1434206148.40249
      X-Trans-Id: tx08090d6533ee408f8e086-0055d750ad
      Content-Type: text/plain; charset=utf-8
[root@Boglin.prt002st001 ~]$
```

---

## Map Swift storage policy to Swift ring file

This example prefetches objects from tape that are in tapeContainer that uses the tapeTier custom Swift storage policy. Issue an **mmobj policy list -verbose** command as indicated in Example 7-4 and note the index number (highlighted in **bold**).

Example 7-4 List Object Storage Policies and Filesets

```
mmobj policy list --verbose
```

Index Details	Name	Default	Deprecated	Fileset	Fileset Path	Functions	Function
0	SwiftDefault	yes		Object_Fileset	/ibm/fs1/Object_Fileset		
<b>44831511210</b>	tapeTier			obj_tapeTier	/ibm/fs1/obj_tapeTier	default	regions="1"

## Map object to location on underlying file system

Issue a **swift-get-nodes** command, which requires a Swift storage ring, the auth token, the container, and the object name. The location on the file system will be the non-handoff location, highlighted in **bold** in Example 7-5.

**Note:** Swift storage rings are located in the `/etc/swift` directory and are named according to the index column listed in Example 7-4. To use the default Swift storage policy, use `/etc/swift/object.ring.gz` in the **swift-get-nodes** command.

Example 7-5 Issue swift-get-nodes to find the location of the object on the underlying file system

```
[root@boglin.prt001st001 ~]# swift-get-nodes /etc/swift/object-44831511210.ring.gz  
AUTH_7ec0250241ba4b3ba6b00a3735400ad1 tapeContainer tapeObj1
```

```
Account      AUTH_7ec0250241ba4b3ba6b00a3735400ad1  
Container    tapeContainer  
Object       tapeObj1
```

```
Partition    6092  
Hash         5f333cd31e1476579dd37715d7e7af8b
```

```
Server:Port Device      9.11.213.59:6200 s44831511210z1device60  
Server:Port Device      9.11.213.57:6200 s44831511210z1device106 [Handoff]
```

```
curl -I -XHEAD  
"http://9.11.213.59:6200/s44831511210z1device60/6092/AUTH_7ec0250241ba4b3ba6b00a37  
35400ad1/tapeContainer/tapeObj1" -H "X-Backend-Storage-Policy-Index: 44831511210"  
curl -I -XHEAD  
"http://9.11.213.57:6200/s44831511210z1device106/6092/AUTH_7ec0250241ba4b3ba6b00a3  
735400ad1/tapeContainer/tapeObj1" -H "X-Backend-Storage-Policy-Index: 44831511210"  
# [Handoff]
```

Use your own device location of servers:  
such as "export DEVICE=/srv/node"

```
ssh 9.11.213.59 "ls -lah  
${DEVICE}:/srv/node*/s44831511210z1device60/objects-44831511210/6092/f8b/5f333cd3  
1e1476579dd37715d7e7af8b" # [Handoff]
```

```
ssh 9.11.213.57 "ls -lah
${DEVICE:-/srv/node*}/s44831511210z1device106/objects-44831511210/6092/f8b/5f333cd
31e1476579dd37715d7e7af8b" # [Handoff]
```

note: `/srv/node*` is used as default value of ``devices``, the real value is set in the config file on each storage node.

---

Issue the **ls** command to resolve the full location of object on file system (Example 7-6).

*Example 7-6 Issue ls command to resolve full location of object on file system*

---

```
[root@boglin.prt001st001 ~]# ls -ltr
/ibm/fs1/obj_tapeTier/s44831512140z1device60/objects-
44831512140/6092/f8b/5f333cd31e1476579dd37715d7e7af8b
total 51200
-rw----- 1 swift swift 52428800 Dec 14 16:10 1450134636.38542.data
[root@boglin.prt001st001 ~]#
```

---

### Add object to file mapping to recall list

Now that the object-to-file mapping is known, add it to the recall list that will be passed to IBM Spectrum Archive, as shown in Example 7-7.

**Note:** Be sure that a space is used before the two dashes (" --) in the recall list file, otherwise IBM Spectrum Archive will fail the recall operation. Adding a space is accomplished by providing open quotation marks (") followed by a space, followed by two dashes.

*Example 7-7 Add object to file mapping to recall list*

---

```
[root@Boglin.prt002st001 ~]$
[root@Boglin.prt002st001 ~]$ echo " -- /ibm/fs1/obj_tapeTier/s44831512140z1device60/objects-
44831512140/6092/f8b/5f333cd31e1476579dd37715d7e7af8b/1450134636.38542.data " >>recall.list.out
```

---

Continue the steps highlighted in Example 7-4 on page 52 and Example 7-7 until you have a complete list of objects to be migrated. Example 7-8 shows a recall file list.

*Example 7-8 A recall list*

---

```
[root@Boglin.prt002st001 ~]$ cat recall.list.out
-- /ibm/fs1/obj_tapeTier/s44831512140z1device60/objects- //
44831512140/6092/f8b/5f333cd31e1476579dd37715d7e7af8b/1450134636.38542.data
-- /ibm/fs1/obj_tapeTier/s44831512140z1device15/objects- //
44831519136/6092/f8d/5a333cd31e1476579dd37715d7e7af8b/16870134787.49313.data
```

---

**Note:** In Example 7-8 the two forward slashes (//) denote a line continuation. Each file location in the recall list should be one contiguous line.

## Copy recall list to IBM Spectrum Archive Node

Example 7-9 copies the completed migration list to the IBM Spectrum Archive node.

*Example 7-9 Copy recall list to IBM Spectrum Archive Node*

---

```
[root@Boglin.prt002st001 ~]$ scp /root/recall.list.out
ltfs001st001:/root/recall.list.out
recall.list.out
100% 232    0.2KB/s   00:00
[root@Boglin.prt002st001 ~]$
```

---

## Issue recall from IBM Spectrum Archive Node

Example 7-10 recalls objects from tape by using the **ltfsee** command.

**Note:** The **ltfsee** command must be run from an IBM Spectrum Archive node in the cluster.

*Example 7-10 Recall objects from IBM Spectrum Archive Tier*

---

```
[root@Boglin.ltfs001st001 ~]$ ltfsee recall recall.list.out
GLES268I(02238): 2 file name(s) have been provided to recall.
GLES263I(02279): Recall result: 2 succeeded, 0 failed, 0 duplicate, 0 not
migrated, 0 not found
[root@Boglin.ltfs001st001 ~]$
```

---



## Runtime considerations

This chapter provides the following runtime examples:

- ▶ Putting objects in the active archive using Boto
- ▶ Getting objects from the active archive using Boto
- ▶ Determining the location of objects in the active archive

## 8.1 Verifying IBM Spectrum Scale OpenStack services with Boto

This section describes the installation of Boto and provides a sample script to use to verify the IBM Spectrum Scale and IBM Spectrum Archive solution. Boto is a Python-based interface that uses the S3 object protocol. More information about Boto is at the following web page:

[http://boto.readthedocs.org/en/latest/s3\\_tut.html](http://boto.readthedocs.org/en/latest/s3_tut.html)

### Install Boto

You can use pip to install the latest released version of Boto as follows:

1. If pip is not installed, download and install it:
  - a. Securely download pip from the following location:  
<https://bootstrap.pypa.io/get-pip.py>
  - b. Install pip by running the following command:  
**python get-pip.py**

2. You can now install Boto by using pip:

```
pip install boto
```

### Sample Boto script

You must configure the OpenStack EC2 credentials. The credentials that are used on the Amazon S3 and Elastic Compute Cloud (EC2) APIs differ from the credentials that are used by the OpenStack API. As a result, you must generate these special credentials to use them when accessing the IBM Spectrum Scale OpenStack services.

Use the instructions that are listed at the following page in the IBM Knowledge Center:

<https://ibm.biz/BdXtsm>

Example 8-1 shows how to test the connection and to print all key objects within a bucket.

*Example 8-1 Test the connection*

---

```
from boto.s3.connection import S3Connection
from boto.s3.connection import OrdinaryCallingFormat

conn = S3Connection(aws_access_key_id='<your aws access key>',
aws_secret_access_key='your aws secret access key>', is_secure=False, host='<your
CES IP host>', port=8080, calling_format=OrdinaryCallingFormat(), debug=0)

bucket = conn.get_bucket("<your bucket name>", validate=False)
for b in bucket.list():
    print(b.name)

conn.close()
```

---

## Object PUT using Boto

Example 8-2 puts an object in the active archive using Boto. Create a file before calling the upload script named 10M.bin, which is the desired file size to perform uploads.

**Note:** The prefix of the filename must match with what is in the script.

---

### *Example 8-2 Put objects using Boto*

---

```
from boto.s3.connection import S3Connection
from boto.s3.connection import OrdinaryCallingFormat
from boto.s3.key import Key
import random
import string

container = "tapeBucket"
files = ["10M"]
legals = string.letters + string.digits

def rand_string(length, char_set=legals):
    output = ""
    for _ in range(length): output += random.choice(char_set)
    return(output)

def upload_file():
    source_file = random.choice(files)
    object_name = source_file + "_" + rand_string(random.randint(5,50)) + ".bin"
    key = Key(bucket)
    key.key = object_name
    print(object_name+" "+str(key.set_contents_from_filename(source_file+".bin")))

def print_bucket_list():
    print("Start of bucket list")
    for b in bucket.list():
        print b

conn = S3Connection(aws_access_key_id='<your aws access key>',
aws_secret_access_key='your aws secret access key', is_secure=False,
host='9.11.213.56', port=8080, calling_format=OrdinaryCallingFormat(), debug=0)

bucket = conn.get_bucket(container, validate=False)
#print_bucket_list()
for i in range(0,1):
    upload_file()
#print_bucket_list()

conn.close()
```

---

## Object GET using Boto

Example 8-3 gets an object from the active archive using Boto.

*Example 8-3 Get objects using Boto*

---

```
from boto.s3.connection import S3Connection
from boto.s3.connection import OrdinaryCallingFormat
from boto.s3.key import Key
from datetime import datetime

def download_keys(p):
    for b in bucket.list(prefix=p):
        key = bucket.get_key(b.name)
        print(b.name)
        key.get_contents_to_filename(b.name)

conn = S3Connection(aws_access_key_id='<your aws access key>',
aws_secret_access_key='your aws secret access key', is_secure=False,
host='9.11.213.56', port=8080, calling_format=OrdinaryCallingFormat(), debug=0)

bucket = conn.get_bucket("tapeBucket", validate=False)
download_keys("1M_")

conn.close()
```

---

## 8.2 Determining the location of objects in the active archive

Determining the location of objects in the active archive entails the following steps:

1. Determine the object-to-file system mapping using swift-get-nodes on a protocol node.
2. Query the IBM Spectrum Scale file system on a protocol node to determine the storage pool in which the object resides.
3. If the object resides in the IBM Spectrum Archive tape-backed bronze pool, use the **1tfsee** command to determine the tape cartridges that hold the data.

**Note:** Currently the object to IBM Spectrum Scale file system mapping procedure uses built-in Swift utilities, therefore this step must be performed on a protocol node.

The method described in this section does not use the IBM Spectrum Scale unified file and object feature. That feature can dramatically simplify the recall process because objects are placed on the underlying IBM Spectrum Scale file system in a manner that eliminates the need to perform the swift stat and swift-get-nodes object-to-file translation described in this section. In your decision of whether to use unified file and object or the traditional method disclosed here, consider the current unified file and object limitations and constraints listed at the following page in the IBM Knowledge Center:

<https://ibm.biz/BdH9sd>



## Determine Account token

The Account value provided by the **swift stat** output is required in order to map objects to files on the file system. Obtain the Account value by loading authorization credentials in the openrc file shown in Example 8-4 and Example 8-5 followed by issuing a **swift stat** command, shown in Example 8-6.

**Note:** The Swift **stat** command must be performed on a protocol node.

### Example 8-4 Sample openrc content

---

```
[root@Boglin.prt002st001 ~]$ cat openrc
# Mon Jun  8 12:20:43 MST 2015
export OS_AUTH_URL="http://127.0.0.1:35357/v3"
export OS_IDENTITY_API_VERSION=3
export OS_AUTH_VERSION=3
export OS_USERNAME="admin"
export OS_PASSWORD="PasswOrd"
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_NAME=admin
export OS_PROJECT_DOMAIN_NAME=Default
```

---

### Example 8-5 Load openrc credentials into your shell

---

```
[root@Boglin.prt002st001 ~]$ source openrc
```

---

### Example 8-6 Issue Swift stat command and note account token (highlighted in bold)

---

```
[root@Boglin.prt002st001 ~]$ swift stat
      Account: AUTH_cd3eae79e2b049e5a9a292688729a775
      Containers: 3
      Objects: 21
      Bytes: 115098151
Containers in policy "generic": 2
  Objects in policy "generic": 20
  Bytes in policy "generic": 104858151
Containers in policy "tapetier": 1
  Objects in policy "tapetier": 1
  Bytes in policy "tapetier": 10240000
      Accept-Ranges: bytes
      X-Account-Project-Domain-Id: default
      X-Timestamp: 1434206148.40249
      X-Trans-Id: tx08090d6533ee408f8e086-0055d750ad
      Content-Type: text/plain; charset=utf-8
[root@Boglin.prt002st001 ~]$
```

---

## Map Swift storage policy to Swift ring file

Example 8-7 on page 60 determines the location of objects that reside in the tapeContainer, which uses the tapeTier custom Swift storage policy. Issue the command shown in Example 8-7 on page 60 and note the index number (highlighted in **bold**).

*Example 8-7 Issue Swift stat command and note account token (highlighted in bold)*

```
mmobj policy list --verbose
```

Index Details	Name	Default	Deprecated	Fileset	Fileset Path	Functions	Function
-----							
0	SwiftDefault	yes		Object_Fileset	/ibm/fs1/Object_Fileset		
<b>44831511210</b>	tapeTier			obj_tapeTier	/ibm/fs1/obj_tapeTier	default	regions="1"

### Map object to location on underlying file system

Issue a **swift-get-nodes** command, which requires a Swift storage ring, the auth token, the container, and the object name. The location on the file system will be the non-handoff location highlighted in **bold** in Example 8-8.

**Note:** Swift storage rings are located in the `/etc/swift` directory and are named according to the index column listed in Example 8-7. To use the default Swift storage policy, use `/etc/swift/object.ring.gz` in the **swift-get-nodes** command.

*Example 8-8 Map the location of the object on the underlying file system*

```
[root@Boglin.prt002st001 ~]$ swift-get-nodes /etc/swift/object-44831511210.ring.gz
AUTH_cd3eae79e2b049e5a9a292688729a775 tapeContainer tapeObj1
```

```
Account      AUTH_cd3eae79e2b049e5a9a292688729a775
Container    tapeContainer
Object       tapeObj1
```

```
Partition    609
Hash         5f333cd31e1476579dd37715d7e7af8b
```

```
Server:Port Device      9.11.213.59:6200 s44831511210z1device60
Server:Port Device      9.11.213.57:6200 s44831511210z1device106 [Handoff]
```

```
curl -I -XHEAD
"http://9.11.213.59:6200/s44831511210z1device60/6092/AUTH_7ec0250241ba4b3ba6b00a3735400ad1/tape
Container/tapeObj1" -H "X-Backend-Storage-Policy-Index: 44831511210"
  curl -I -XHEAD curl -I -XHEAD
"http://9.11.213.57:6200/s44831511210z1device106/6092/AUTH_7ec0250241ba4b3ba6b00a3735400ad1/tape
Container/tapeObj1" -H "X-Backend-Storage-Policy-Index: 44831511210" # [Handoff]
```

```
Use your own device location of servers: such as "export DEVICE=/srv/node"
ssh 9.11.213.59 "ls -lah ${DEVICE:-/srv/node*}/s44831511210z1device60/objects-
44831511210/6092/f8b/5f333cd31e1476579dd37715d7e7af8b"
ssh 9.11.213.57 "ls -lah ${DEVICE:-/srv/node*}/s44831511210z1device106/objects-
44831511210/6092/f8b/5f333cd31e1476579dd37715d7e7af8b # [Handoff]"
```

note: `~/srv/node~` is used as default value of `~devices~`, the real value is set in the config file on each storage node.

```
[root@Boglin.prt002st001 ~]$
```

In this example, /ibm/fs1/obj\_tapeTier is the base directory where the Swift devices reside for the tapeTier storage policy, therefore it must be appended to the non-handoff output in order to create the full path, as indicated in Example 8-9.

*Example 8-9 List object location*

---

```
[root@boglin.prt001st001 ~]# ls -ltr
/ibm/fs1/obj_tapeTier/s44831512140z1device60/objects-44831512140/6092/f8b/5f333cd3
1e1476579dd37715d7e7af8b
total 51200
-rw----- 1 swift swift 52428800 Dec 14 16:10 1450134636.38542.data
[root@boglin.prt001st001 ~]#
```

---

## Determine location of objects using the ltfsee command

You can determine whether an object resides in the IBM Spectrum Archive tape tier by using the **ltfsee** command, as indicated in Example 8-10. The example uses the full object path on the IBM Spectrum Scale file system that was previously determined.

### Notes:

- ▶ The **ltfsee** command must be performed on an IBM Spectrum Archive node.
- ▶ In this example the file is in the pre-migrated state meaning it exists on disk and also on tape.

*Example 8-10 List Object location in IBM Spectrum Archive*

---

```
[root@Boglin.ltf001st001 ~]$ ltfsee info files
/ibm/fs1/obj_tapeTier/s44831512140z1device60/objects-44831512140/6092/f8b/5f333cd31e1476579dd377
15d7e7af8b/1450134636.38542.data
Name:
/ibm/fs1/obj_tapeTier/s44831512140z1device60/objects-44831512140/6092/f8b/5f333cd31e1476579dd377
15d7e7af8b/1450134636.38542.data
Tape id:2FC183L5:2FC184L5 Status: premigrated [root@Boglin.ltf001st001 ~]$
```

---

## Determine the storage pool where the object resides

Example 8-11 issues an **mmisattr** command to determine the storage pool where the object currently resides. The example uses the full object path on the IBM Spectrum Scale file system that was previously determined.

*Example 8-11 Determine the storage pool where the object resides*

---

```
[root@Boglin.ltf001st001 ~]$ mmisattr -L -d
/ibm/fs1/obj_tapeTier/s44831512140z1device60/objects-
44831512140/6092/f8b/5f333cd31e1476579dd37715d7e7af8b/1450134636.38542.data
file name:
/ibm/fs1/obj_tapeTier/s44831512140z1device60/objects-44831512140/6092/f8b/5f333cd31e1476579dd377
15d7e7af8b/1450134636.38542.data

metadata replication: 1 max 2
data replication: 1 max 2
immutable: no
appendOnly: no
flags:
storage pool name: silver
fileset name: obj_tapeTier
```

```
snapshot name  
creation time: Thu Aug 20 09:22:47 2015  
Windows attributes: ARCHIVE  
Encrypted: no  
user.swift.metadata: "??}q?(U?Content-Typeq?U?application/octet-streamq?U?Content-  
Lengthq?U?10240000U?ETagq?U   eee039854de6aa9f46275df3933e89c4q?U?X-  
Timestampq?U?1440087767.20793U?nameq?UH/AUTH_cd3eae79e2b049e5a9a292688729a775/tapeContainer/tape  
Obj1"  
dmapi.IBMUID: "11377005508221052921-16782793019371167660-1243303519-804368-0"  
dmapi.IBMSGEN#: "1"  
dmapi.IBMTPS: "1 2FC183L5:2FC184L5"  
dmapi.IBMProv: "|tfs????"  
gpfs.dmapi.region: 0x000000000000000000000000000000006000000626A0000  
dmapi.IBMPMig: "???????????R4ps???????$;{?("
```



## Conclusion

This paper describes how the active archive based on IBM Spectrum Scale and IBM Spectrum Archive address the difficulties that are associated with storing large amounts of unstructured data.

IBM Spectrum Scale object storage based on OpenStack Swift combined with IBM Spectrum Archive provides an enterprise class active archive with advanced capabilities to efficiently manage large amounts of unstructured data for extended amounts of time. The IBM Spectrum Scale information lifecycle management (ILM) policy engine provides the flexibility to create custom data lifecycle management policies to efficiently place objects and object metadata in the right pool at the right time to optimize the object storage infrastructure.

Specifically, this paper describes the following advanced active archive capabilities:

- ▶ OpenStack Swift and S3 API support
- ▶ The ability to migrate data to tape at the Swift container and S3 bucket level using Swift storage policies
- ▶ Pre-fetch method for recalling archive data from tape prior to issuing Swift and S3 GET REST API calls
- ▶ Optimized Swift account and container database and object metadata placement on SSD based storage pool
- ▶ Heatmap-based tiering of object data

Figure 9-1 illustrates the IBM Spectrum Scale and IBM Spectrum Archive architecture.

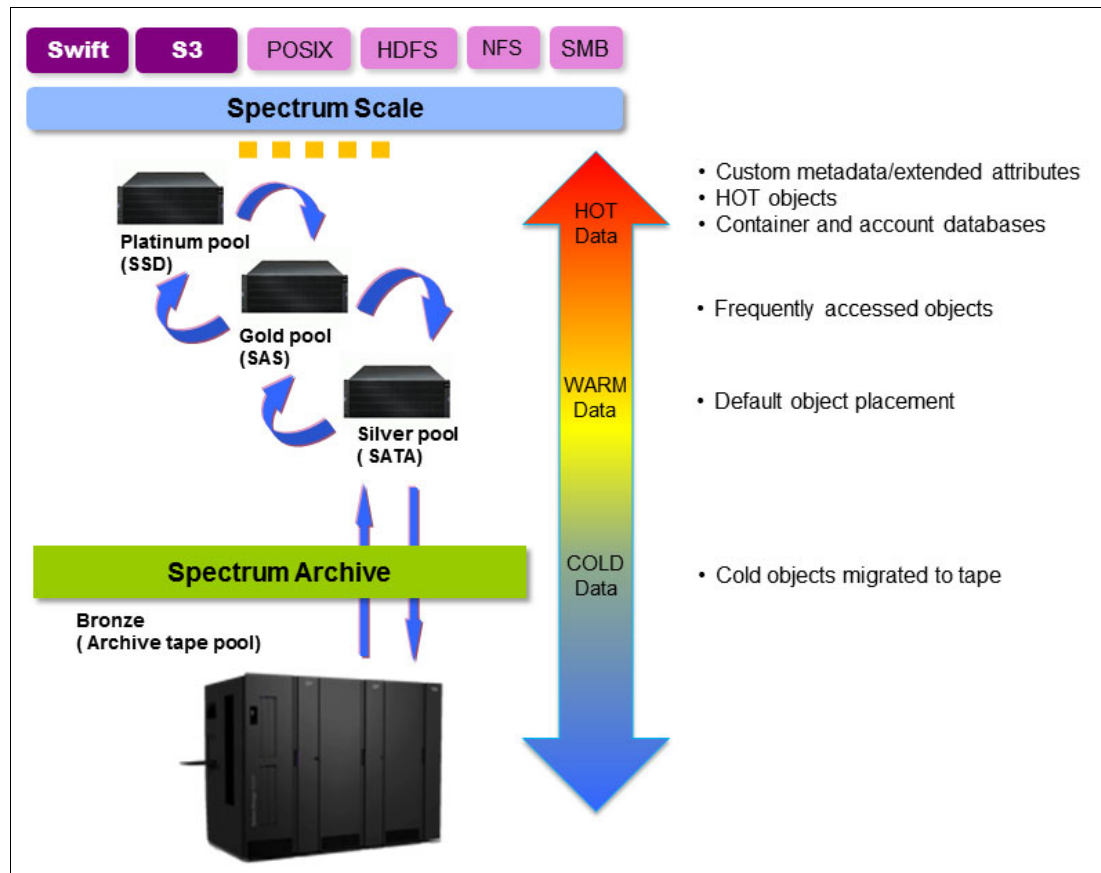


Figure 9-1 IBM Spectrum Scale and IBM Spectrum Archive architecture

# Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this paper.

## IBM Redbooks

The following IBM Redbooks publications provide additional information about the topic in this document. Note that some publications in this list might be available in softcopy only.

- ▶ *A Deployment Guide for IBM Spectrum Scale Object*, REDP-5113
- ▶ *IBM Linear Tape File System Enterprise Edition V1.1.1.2: Installation and Configuration Guide*, SG24-8143
- ▶ *IBM Private, Public, and Hybrid Cloud Storage Solutions*, REDP-4873
- ▶ *IBM Spectrum Scale (formerly GPFS)*, SG24-8254

You can search for, view, download or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following website:

[ibm.com/redbooks](http://ibm.com/redbooks)

## Other publications

These publications are also relevant as further information sources:

- ▶ *IBM Spectrum Scale V4.2: Administration and Programming Reference*, SA23-1452
- ▶ *IBM Spectrum Scale V4.2: Advanced Administration Guide*, SC23-7032
- ▶ *IBM Spectrum Scale V4.2: Concepts, Planning, and Installation Guide*, GA76-0441
- ▶ *IBM Spectrum Scale V4.2: Problem Determination Guide*, GA76-0443

## Online resources

These websites are also relevant as further information sources:

- ▶ IBM Spectrum Scale at IBM developerWorks®  
<https://ibm.biz/Bd4BQ7>
- ▶ IBM Spectrum Archive Enterprise Edition (EE) in the IBM Knowledge Center  
<https://ibm.biz/Bd4BQW>
- ▶ IBM Spectrum Scale in the IBM Knowledge Center  
<https://ibm.biz/Bd4B3D>
- ▶ IBM Spectrum Scale Wiki  
<https://ibm.biz/BdFPR2>
- ▶ IBM Elastic Storage Server  
<https://ibm.biz/Bd4B3E>

## Help from IBM

IBM Support and downloads

[ibm.com/support](https://ibm.com/support)

IBM Global Services

[ibm.com/services](https://ibm.com/services)







REDP-5237-00

ISBN 073845513X

Printed in U.S.A.

Get connected

