# Spectrum Scale Expert Talks

Episode 7:

## Manage the lifecycle of your files using the policy engine

IBM Spectrum Scale

**Show notes:**
www.spectrumscaleug.org/experttalks

**Join our conversation:**
www.spectrumscaleug.org/join

# SSUG::Digital

## Welcome to digital events!

IBM Spectrum Scale

**Show notes:**
www.spectrumscaleug.org/experttalks

**Join our conversation:**
www.spectrumscaleug.org/join

## About the user group

- Independent, work with IBM to develop events
- Not a replacement for PMR!
- Email and Slack community
- https://www.spectrumscaleug.org/join

#SSUG

## We are …

- Simon Thompson (UK)
- Kristy Kallback-Rose (USA)
- Bob Oesterlin (USA)
- Bill Anderson (USA)
- Chris Schipalius (Australia)

IBM CHAMPION

## Check https://www.spectrumscaleug.org/experttalks for charts, show notes and upcoming talks

- Past talks:
  - 001: What is new in Spectrum Scale 5.0.5
  - 002: Best practices for building a stretched cluster
  - 003: Strategy update
  - 004: Update on performance enhancements in Spectrum Scale (file create, MMAP, direct IO, ESS 5000)
  - 005: Update on functional enhancements in Spectrum Scale (inode management, vCPU scaling, NUMA considerations)
  - 006: Persistent Storage for Kubernetes and OpenShift environments
- Today:
  - Oct 21: Manage the lifecycle of your files using the policy engine
- Next:
  - Nov 4: Multi-node scaling of AI workloads using Nvidia DGX, OpenShift and Spectrum Scale
  - Nov 16: User Meeting at SC20 (Session 1) (more details will follow)
  - Nov 18: User Meeting at SC20 (Session 2) (more details will follow)

## About the Speaker

Nils Haustein

IBM European Storage Competence Center

Responsible for solution design and implementation of file and object storage solutions with focus on data protection and archiving

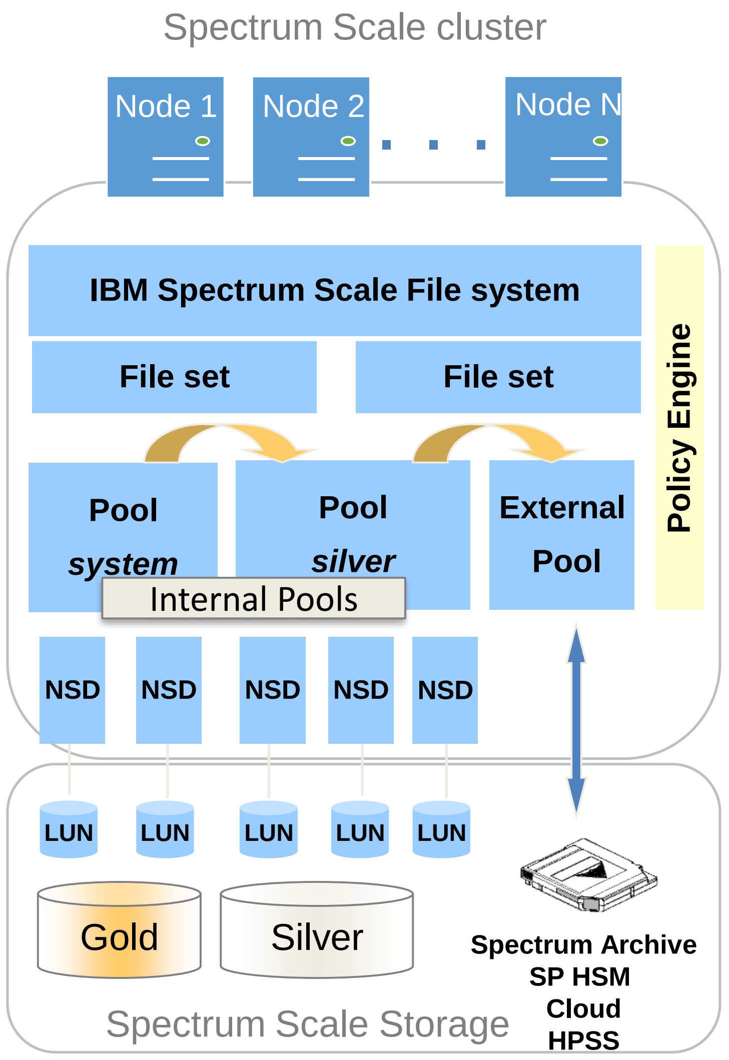
# Disclaimer

## Agenda

- **ILM policies and rules**
  - IBM Spectrum Scale policy engine
  - Placement
  - Migration
  - Lists
  - Hints and Tips

# Spectrum Scale ILM architecture

- Files are stored in **file system** and underlying **pools**
  - Storage pool is a collection of disks of the same type
  - ILM is carried out on pools
- **Placement policy** controls pool where files are placed
  - Placement policy is applied during file creation
- **Migration policies** control transparent migration of  files from one pool to another
  - Applied during life cycle
  - Migrated files can be transparently accessed
  - Files can also be migrated to tape
    - Transparent access through file system
- **Policy engine** applies and executes policies



# File system storage pools

- **Internal pool** – allows placement and migration
  - A collection of disks or arrays with similar properties that are managed together
  - Every file system has at least a “system” pool
    - File system metadata is only stored in “system” pool
    - Maximum of 8 storage pools are possible per file system

- **External pool** – allows migration to external store
  - External pool is an interface (script) to an external application
  - Scripts supported for Spectrum Archive, SP HSM, Cloud Object Storage or HPSS
  - External application is invoked by policy engine or by command

- Pools are designated by names
  - Pool name *system* is reserved and must always exist in file system

# What is the Spectrum Scale policy engine

- Policy engine is a **SQL interpreter** that executes rules
  - Rules define selection criteria and action in an SQL-like language
  - Rules are created by human experts

- Policy engine runs **multi-threaded** on multiple nodes
  - One master and multiple helper that share the work
  - Reads metadata (from inodes), matches these to selection criteria and creates weighted lists

- Policy engine is used to implement ILM
  - Placement, migration, deletion, compression, statistics ...
  - **Internal storage services** also rely on policy engine
    - such as AFM, Backup, Audit logging, Watch folder, etc.

## Agenda

- ILM policies and rules
- ➢ **IBM Spectrum Scale policy engine**
- Placement
- Migration
- Lists
- Hints and Tips

# Spectrum Scale policy engine

- Policy engine evaluates and applies policy rules by:
  - Selecting files based on criteria defined in rules
  - Initiating an action defined by the rules, such as:
    - Placement, Migration, List, Deletion, Encryption, Compression ….

- Policy rules are provided in a policy file
  - Describes the **selection criteria** in a SQL-like language
  - Describes **action** to be performed
  - Example:

```
MIGRATE FROM POOL 'System' to POOL 'data'
WHERE FILE_SIZE > 1024000
```

# Invocation of policy engine

- **Automated invocation** of active policy
  - Each file system can have **one active policy**
  - Is automatically invoked based on rule type and definitions
  - Active ILM policy for one file system can include:
    - File placement rules
    - Threshold migration rules (space triggers)

- **Manual invocation** of scheduled policies
  - Are invoked manually or via scheduler
  - Scheduled policies include migration rules
    - Based on file attribute (size, age)

- Active migration policies must be threshold based and simple
- Scheduled migration policies do not require threshold and can be more advanced

# Starting the policy engine

- Active policy for a file system is configured with command: `mmchpolicy`
  - Executes rules for **placement and threshold-based migration**

```
mmchpolicy filesystem policyfile [-I yes|test]
```

  - When threshold is reached `mmapplypolicy` is invoked via callback

- Scheduled policy for a file system is invoked with command: `mmapplypolicy`
  - Executes rules **manually for migration**

```
mmapplypolicy filesystem -P policyfile [-I yes|test|defer|prepare]
```

# mmapplypolicy command

- Executes the rules provided in a policy files
  - Invoked with command line or via callback
  - Distributes task to selected nodes (-N) running in multiple threads (-m)

- Syntax example:

```
mmapplypolicy {Device|Directory} [-P PolicyFile] [-I {yes|defer|test|prepare}]
              [-B MaxFiles] [-m ThreadLevel] [-f FileListPrefix]
              [-N {all | mount | Node[,Node...] | NodeFile | NodeClass}]
              [-g GlobalWorkDirectory] [-s LocalWorkDirectory]
              [-i InputFileList] [-L n] [-M name=value...]
              [--choice-algorithm {best | exact | fast}] [--split-margin n.n]
              [--split-filelists-by-weight]
              [-S SnapshotName] [--scope {filesystem | fileset | inodespace}]
              [--single-instance]
```

# Some key mmapplypolicy options explained

| Option | Description |
|---|---|
| -m | Number of threads dispatched for file management operation (e.g. migration) on every participating node |
| -B | Bucket size defines the number of files each file management thread obtains |
| -n | Number of directory scan threads on every node participating in the policy scan |
| -N | Node names executing the policy scans |
| -s <dir> | Directory to store local temporary files |
| -g <dir> | Directory to store global temporary files |
| --single-instance | Only one instance of mmapplypolicy can run for this file system |
| -I | test: scan the file system and do not run the actual operation (e.g. migration)<br>defer: create the file lists, but do not run the actual file operation (e.g. migration)<br>yes: run the policy and file operations (default) |
| -L | Debug level for mmapplypolicy |

## Execution phases of the policy engine

1. Selecting files
   - Read metadata for files of the specified file system device, fileset or directory  and select files by matching against the policy rules

2. Choosing and scheduling
   - Creating file lists with all or subsets of selected files
   - Order of files can be controlled with options (`--choise-algorithm, --split-filelists-by-weight`)
   - Size of file lists can be controlled with
     - Option `–B`, specifying the number of files
     - `SIZE` clause, specifying the size of the files in the list

3. Process files according to rule types (migrate, delete, external list)
   - File processing can be done on multiple nodes and multiple threads in parallel
     - Option `–N` denotes the nodes processing the files (default all)
     - Option `–m` specifies the number of threads per node
   - Each processing thread processes one file list
     - When done, the next thread is started

# Some key file attributes for phase 1

| File attribute | Description |
|---|---|
| NAME | Name of the file (excluding path name) |
| PATH_NAME | Path and file name |
| FILE_SIZE | Size of the file in Bytes. Note, after migrating a file to an external pool, the file size remains the same. |
| KB_ALLOCATED | Number of kilobytes of disk space allocated for the file data. If the file is migrated to an external pool then KB_ALLOCATED is 0 (if the stub-size is 0) |
| ACCESS_TIME | Date and time that the file was last accessed (POSIX atime) |
| FILE_HEAT | Access temperature of a file based on the frequency of file access |
| POOL_NAME | Name of the pool where the file is currently stored. Note, for files migrated to an external pool the pool name will be the name of the source pool from which the file has been migrated. |
| MISC_ATTRIBUTES | A collection of file attributes represented as string value. For example, file migration states (resident, pre-migrated and migrated). |
| XATTR | A collection of system defined file attributes. For example, the tape ID migrated files (dmapi.IBMTPS). |

## Agenda

ILM policies and rules

IBM Spectrum Scale policy engine

➢ **Placement**

Migration

Lists

Hints and Tips

# Placement policy

- Placement rules belong to active policy for a given file system
  - Activated with command `mmchpolicy`

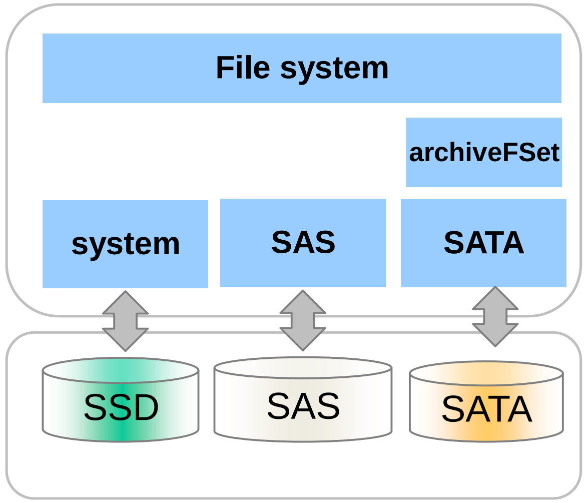
- Placement rules are evaluated and applied during file creation
  - Places file on internal storage pool only

- Placement rule syntax example

```
RULE ['RuleName'] SET POOL 'PoolName'
     [LIMIT (OccupancyPercentage)]
     [REPLICATE (DataReplication)]
     [FOR FILESET (FilesetName[,FilesetName]...)]
     [WHERE SqlExpression]
```

- If no placement rule is specified, then files are placed on
  - System pool if no other pool exists, or on next pool after system pool

## Placement policy examples



- Example
  - Place all files ending with .mp3 on pool SAS
  - Place all files in fileset archiveFSet in pool SATA
  - Place all other files in pool system

```
RULE 'archive' SET POOL 'SATA' FOR FILESET ('archiveFSet’)
RULE 'mp3' SET POOL 'SAS' WHERE LOWER(NAME) like '%.mp3’
RULE 'default' SET POOL 'system’
```

- Multiple placement rules per file system are combined in one policy file
  - Evaluation is top-down, first hit is applied
  - Last rule must be a default placement rules catching all other files
  - Caution: too many placement rules can slow down the file system

## Agenda

ILM policies and rules

IBM Spectrum Scale policy engine

Placement

➢ **Migration**

Lists

Hints and Tips

# Migration policy overview

There are two kinds of migration policies:

1. **Active migration** policy is executed when pool capacity threshold is met
   - Activated with command: `mmchpolicy`
2. **Manual or scheduled** migration policy is executed by command
   - kicks in by command: `mmapplypolicy`

- Active migration policy should be simple,
  - Scheduled policies can be more sophisticated
- Design the scheduled policy to prevent active policy to trigger

# Active migration policy (threshold based)

- Active migration policy triggers automated migration based on capacity threshold

```
RULE 'rule-name' MIGRATE FROM POOL 's-pool'
THRESHOLD(%high,%low) [WEIGHT(expression)] TO POOL 'd-pool'
[FOR FILESET ('fileset-name’)] WHERE (conditions)
```

- THRESHOLD statement defines threshold at which this rule is triggered
  - %high: Threshold triggers migration
  - %low:  continue migration until this threshold is reached

- Activated for entire file system with command: `mmchpolicy`
  - When threshold is reached migration rules are applied automatically
  - Invokes callback (must be configured)
  - Callback invokes policy engine with this policy

# Migration Callback

- To invoke the policy engine when threshold is met a callback must be configured
  - For internal and external pool migrations
- Callback is invoked upon Spectrum Scale events
  - Relevant migration events are: `lowDiskSpace,noDiskSpace`
- Callback executes a customizable script with selective parameters
  - Parameters are typically: `'%eventName %fsName' (%storagePool`)
  - Default script is: `/usr/lpp/mmfs/bin/mmstartpolicy` - invokes `mmapplypolicy`
- Example for callback setup:

```
mmaddcallback MIGRATION --command /usr/lpp/mmfs/bin/mmstartpolicy
--event lowDiskSpace,noDiskSpace --parms '%eventName %fsName'
```

# Active migration policy example

- Active Policy includes placement and threshold migration
  - Migrate files when *system* pool is 90% full, oldest files first:

```
RULE 'ilm_auto' MIGRATE FROM POOL 'system'
THRESHOLD(90,70)
WEIGHT (DAYS(CURRENT_TIMESTAMP) – DAYS(ACCESS_TIME))
TO POOL 'silver'
```

  - Place *mp3* files in *silver* pool:

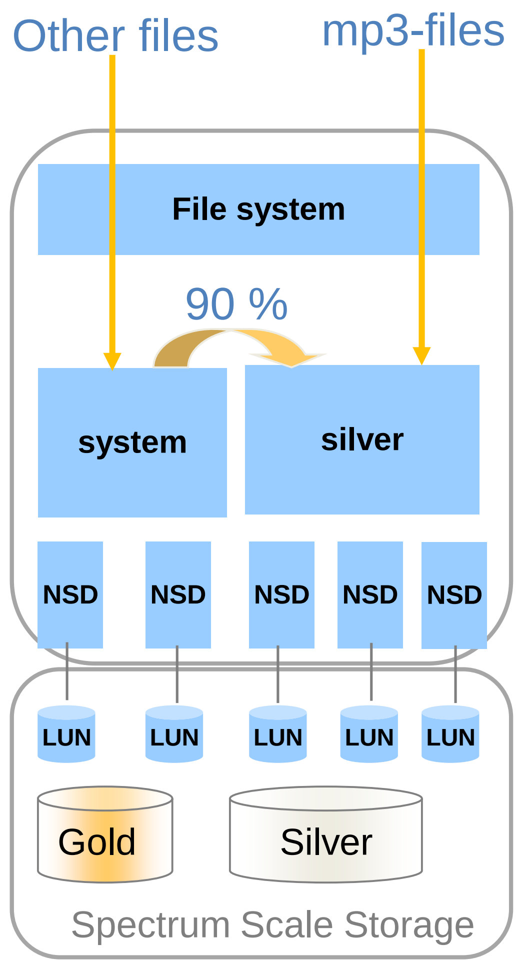
```
RULE 'mp3_rule' SET POOL 'silver'
WHERE LOWER(NAME) LIKE '%.mp3%'
```

  - Place all other files in *system* pool:

```
RULE 'default' SET POOL 'system’
```

  - Activate the policy for the file system:

```
# mmchpolicy fsname policyfile [-I test]
```



# Schedule migration policy

- Manual migration is started on-demand

```
RULE 'rule-name' MIGRATE FROM POOL 's-pool'
[WEIGHT(expression)] TO POOL 'd-pool'
[FOR FILESET ('fileset-name')] WHERE (conditions)
```

- Conditions describe files to be selected for migration
  - Can be based on file attributes (names, size, age, owner)
  - Use of threshold is optional

- Policy is executed with command: `mmapplypolicy`

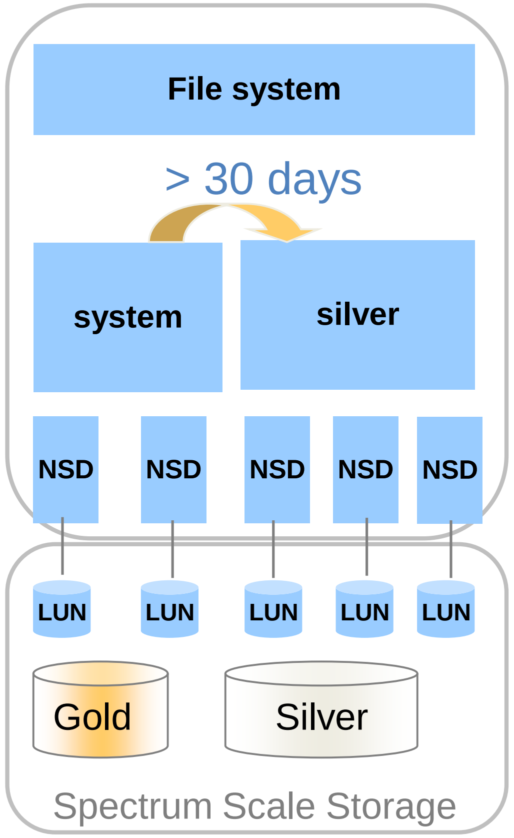
# Scheduled migration policy example

- Scheduled policy migrates data from *system* to *silver* pool
  - Migrate files not accessed for 30 days:

```
RULE 'ilm_age' MIGRATE
FROM POOL 'system' TO POOL 'silver'
WHERE (DAYS(CURRENT_TIMESTAMP) –
       DAYS(ACCESS_TIME) > 30)
```

  - Execute policy:

```
# mmapplypolicy fsname –P policyfile
[–N nodes – m threads –B size]
```



## Agenda

- ILM policies and rules
- IBM Spectrum Scale policy engine
- Placement
- Migration
- ➢ **Lists**
- Further guidance and examples

# List rules

- List rules allow identifying files based on file attributes using the policy engine
  - This prevents crawling through the file system and is much faster

- List policies can include two rules:

```
RULE EXTERNAL LIST 'listName' EXEC 'program' OPTS option
RULE 'rule-name' LIST 'listName' WHERE (conditions)
```

- Execute a list policy:

```
mmapplypolicy /filesystem -P policyfile -f ./fileprefix -I defer
```

  - Output file is stored in file `./fileprefix.list.listName`
  - Format of the output (inodenumber, inodegeneration, snapid – path and filename)

```
27648 1566354680 0  -- /mnt/user1/test/file.doc
```

(inodenumber, inodegeneration, snapid – path and filename)

# List policy example

- Create a list of non-resident files including the tape ID (Spectrum Archive)

```
/* macro defining resident state */
define( is_resident,(MISC_ATTRIBUTES NOT LIKE '%M%') )

/* External list rule showing the tape ID */
RULE EXTERNAL LIST 'tape-id' EXEC ''
RULE 'tape-id' LIST 'tape-id' SHOW(xattr('dmapi.IBMTPS'))
where NOT (is_resident)
```

- Run the policy

```
mmapplypolicy fsname -P policyfile -f /tmp/mia -I defer
```

  – Output file /tmp/mia.list.tape-id

```
27648 1566354680 0  1 SLE033L6@[lib-ID] -- /mnt/user1/test/file.doc
```

# Agenda

- ILM policies and rules
- IBM Spectrum Scale policy engine
- Placement
- Migration
- Lists
- ➢ **Hints and Tips**

## Macros

- Macros allow pre-defining certain conditions to make rule less complex to read

```
define( macro-name, (conditions & expressions) )
```

- Example: define age of last access

```
define( access_age,(DAYS(CURRENT_TIMESTAMP) - DAYS(ACCESS_TIME)) )

RULE 'MigToTape' MIGRATE FROM POOL 'system' TO POOL 'ltfs'
WHERE (KB_ALLOCATED > 0) AND (access_age > 30)
```

- More examples:

```
define(is_migrated,(MISC_ATTRIBUTES LIKE '%V%'))
define(gb_allocated, (INTEGER(KB_ALLOCATED / 1048576 )))
define(is_immutable,(MISC_ATTRIBUTES LIKE '%X%'))
define(is_replicated,(MISC_ATTRIBUTES LIKE '%2%'))
```

# Exclude rules

- Allow to exclude files and directories from migration and deletion (not for placement)

```
RULE ['RuleName'] EXCLUDE [DIRECTORIES_PLUS]
[FOR FILESET ('FilesetName'[,'FilesetName']...)]
WHERE SqlExpression
```

- Exclude rules must come first in the policy
  - Files matching the EXCLUDE rule are skipped by subsequent rules
  - Exclude rules are more efficient and faster than exclusion in WHERE clause

- Example:

```
RULE 'exclude' EXCLUDE DIRECTORIES_PLUS WHERE
(PATH_NAME LIKE '%/.SpaceMan/%' OR
 PATH_NAME LIKE  '%/.snapshots/%' OR
 NAME LIKE  '%/.mmbackupShadow%')
```

# Strings substitution in policies

- Policy engine allows to substitute strings in policy rules
  - With this you can use one policy file for different file systems

- Example: substitute the fileset name in a MIGRATE policy:
  - Rule includes the string **'FILESETNAME’** which gets substituted

```
RULE 'MigToTape' MIGRATE FROM POOL 'system' TO POOL 'ltfs'
FOR FILESET ('FILESETNAME') WHERE (access_age > 30)
```

  - Substitute the string when running the policy

```
mmapplypolicy fsname -P policyfile -M "FILESETNAME=test“ …
```

# WEIGHT clause

- WEIGHT clause can be used to sort the files provided in file lists
  - Files with largest WEIGHT value come first in file lists
  - Default weight is `KB_ALLOCATED`
- Example: Sort files by use access age with oldest files first

```
WEIGHT(DAYS(CURRENT_TIMESTAMP) - DAYS(ACCESS_TIME)
```

- Example: Sort file by their path name:

```
define(dir_weight,
  (CASE
  /*=== dir1 has high priority ===*/
  WHEN PATH_NAME like '%/dir1/%'  THEN 4
  /*=== dir2 has medium priority ===*/
  WHEN PATH_NAME like '%/dir2/%'  THEN 3
  END))

RULE 'MigToTape' MIGRATE FROM POOL 'system' WEIGHT(dir_weight) TO POOL 'ltfs'
```

# Using input files to the policy engine

- File names to be matched against rules can be provided to the policy engine
  - Eliminates the need to scan the entire file system

- Input file list includes fully qualified path or directory names:

```
/gpfs/dir1
/gpfs/dir2/file1.txt
```

- Using input files with the policy engine

```
mmapplypolicy fsname –P policy –i inputfile
```

- Example: list all files with the extended attribute `user.backup` set to yes

```
RULE EXTERNAL LIST 'backup' EXEC ''
RULE 'userBackup' LIST 'backup’
WHERE XATTR('user.backup') like UPPER('yes')
```

# Key references

Quick Start Guide

Policies samples and scripts

Knowledge Center

IBM
Spectrum
Scale

IBM® Spectrum Scale™ Information Life cycle Management Policies

An introduction to IBM Spectrum Scale ILM policies for file archiving and statistics with IBM Spectrum Archive Enterprise Edition

IBM Knowledge Center

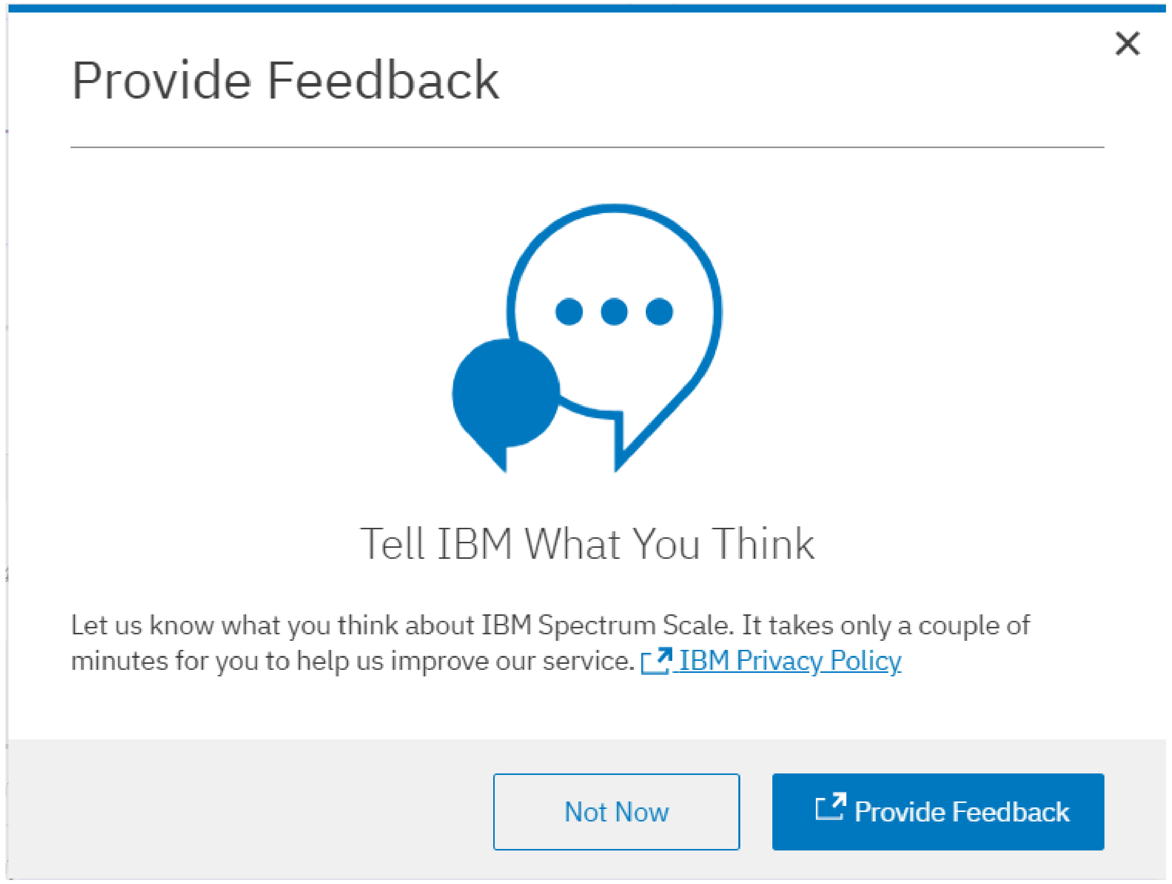
Check https://www.spectrumscaleug.org/experttalks
for charts, show notes and upcoming talks

- Past talks:
  - 001: What is new in Spectrum Scale 5.0.5
  - 002: Best practices for building a stretched cluster
  - 003: Strategy update
  - 004: Update on performance enhancements in Spectrum Scale
    (file create, MMAP, direct IO, ESS 5000)
  - 005: Update on functional enhancements in Spectrum Scale
    (inode management, vCPU scaling, NUMA considerations)
  - 006: Persistent Storage for Kubernetes and OpenShift environments
- Today:
  - Oct 21: Manage the lifecycle of your files using the policy engine
- Next:
  - Nov 4: Multi-node scaling of AI workloads using Nvidia DGX, OpenShift and Spectrum Scale
  - Nov 16: User Meeting at SC20 (Session 1) (more details will follow)
  - Nov 18: User Meeting at SC20 (Session 2) (more details will follow)

# Thank you!

Please help us to improve Spectrum Scale with your feedback

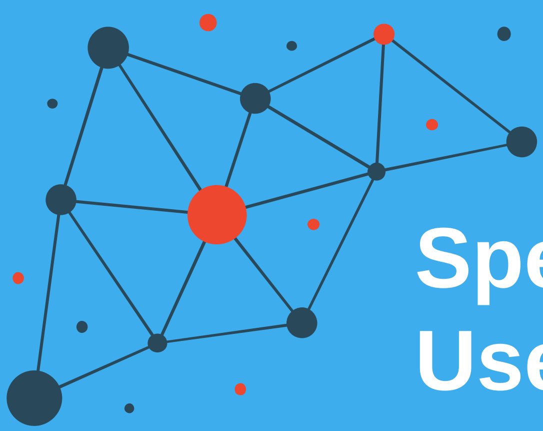
- If you get a survey in email or a popup from the GUI, please respond
- We read every single reply



Provide Feedback

Tell IBM What You Think

Let us know what you think about IBM Spectrum Scale. It takes only a couple of minutes for you to help us improve our service. IBM Privacy Policy



## Spectrum Scale User Group

The Spectrum Scale (GPFS) User Group is free to join and open to all using, interested in using or integrating IBM Spectrum Scale.

The format of the group is as a web community with events held during the year, hosted by our members or by IBM.

See our web page for upcoming events and presentations of past events. Join our conversation via mail and Slack.

**www.spectrumscaleug.org**

# Links

- IBM Spectrum Scale ILM policies – quick start guide
  https://www.ibm.com/support/pages/node/6260749

- IBM Spectrum Scale ILM policies – samples and scripts
  https://github.com/nhaustein/spectrum-scale-policy-scripts

- IBM Spectrum Scale ILM policies – Knowledge Center
  https://www.ibm.com/support/knowledgecenter/STXKQY_5.0.5/com.ibm.spectrum.scale.v5r05.doc/bl1adv_storage_mgt.htm