Stanford Research Computing
srcc.stanford.edu

LAD24 & LUSTRE ADMIN DEV WORKSHOP
Paris 23.24.25/9/2024

MinIO, Lustre, and Phobos: A fresh take on scalable object storage with tape for research data

Stéphane Thiell – September 2024

Research Storage Lead

Stanford Research Computing, University IT

There is no problem in computer science that can't be solved using another level of indirection. – David Wheeler

…except for the problem of too many levels of indirection 😁

Elm: Driven by research

Campus-wide need for on-prem cold storage to preserve research data

Campus-wide need for affordable cold storage

Many research fields produce vast and valuable datasets that, while not in active use, need to be stored for future research or compliance. However, the comparatively high cost of commercial cloud storage at scale can be prohibitive for researchers. Affordable, long-term storage solutions are essential to ensure that these critical datasets are preserved. These fields include:

- Neuroscience
- CryoEM (Cryogenic Electron Microscopy)
- AI and Machine Learning
- Bioinformatics
- Earth Sciences and Climate Research
- High-Energy Physics

Human Neural Circuitry program

In-patient research program using **optogenetics** to advance understanding of neuropsychiatric disorders

Continuous data acquisition at up to 5.5 GB/s over a week = 3 PB of data per participant

Raw data must be retained for future analysis

Karl Deisseroth is the D.H. Chen Professor of Bioengineering and of Psychiatry and Behavioral Sciences at Stanford University, and Investigator of the Howard Hughes Medical Institute.

Data goes from the patient's brain, transmitted by electrodes placed inside the skull or, as modeled here, by high density EEG detectors on the skull's surface, to a server across campus and back to the researchers in half a millisecond.

https://med.stanford.edu/news/all-news/2023/12/human-neural-circuitry.html
https://stanmed.stanford.edu/real-time-brain-data

Requirements and features

Drivers behind Elm and the core functionalities it offers

Elm meets **modern research needs**

Elm supports large-scale research storage needs with:

- **Scalability on multiple fronts**
  - **Frontend**: MinIO, as a distributed service, offers flexible scaling
  - **Disk tier**: Expand capacity easily by adding more space with Lustre
  - **Tape storage**: Scale the tape system with an expandable tape library
- **High-speed ingestion**
  - MinIO with Lustre enable rapid data intake for large datasets
- **Highly parallel archiving**
  - Phobos and LTFS archive data to tape efficiently, maximizing throughput
- **Cost-effective, long-term storage**
  - Lustre/HSM provides a cost-efficient disk layer for MinIO
  - Tape retrieval is slower, but its infrequent use is a cost-effective choice
  - Minimize vendor lock-in

Core capabilities for seamless integration

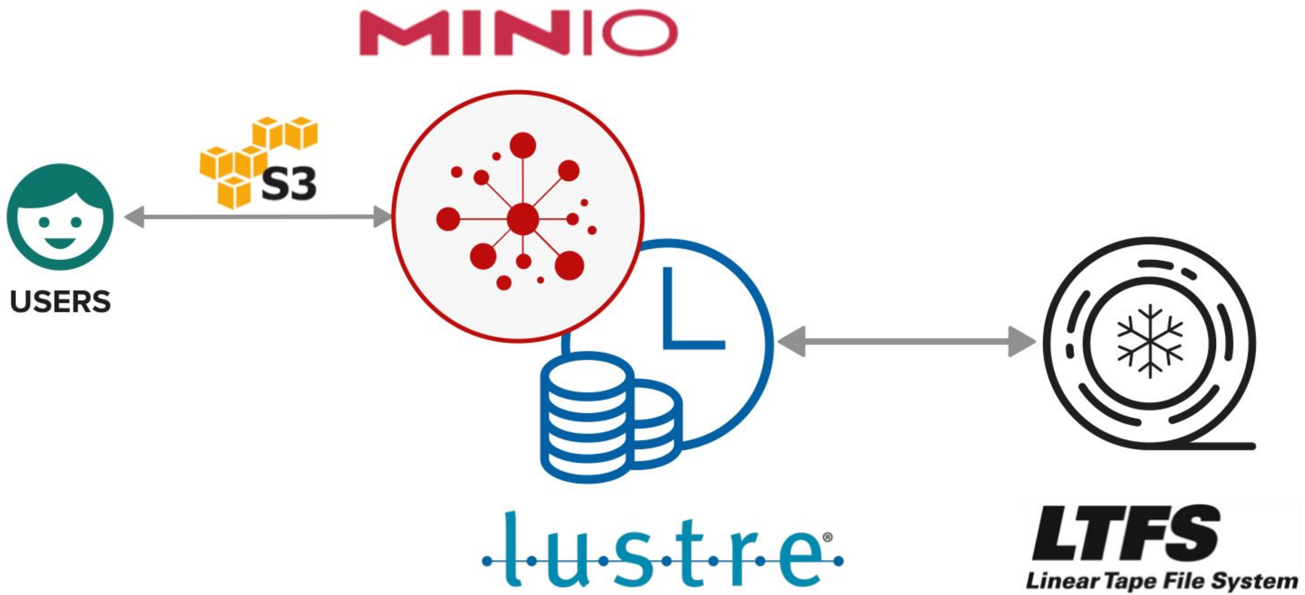
Elm provides a scalable storage solution that fit seamlessly into research environments:

- **S3 compatibility**
  - Seamlessly integrates with familiar tools, simplifying adoption
- **Data protection**
  - MinIO adds erasure coding, checksums, and encryption via the S3 protocol, ensuring data integrity and security
- **Organized storage**
  - Phobos tags are used for MinIO projects and data risk classification, ensuring clear data differentiation and management

Elm architecture

Key concepts and features

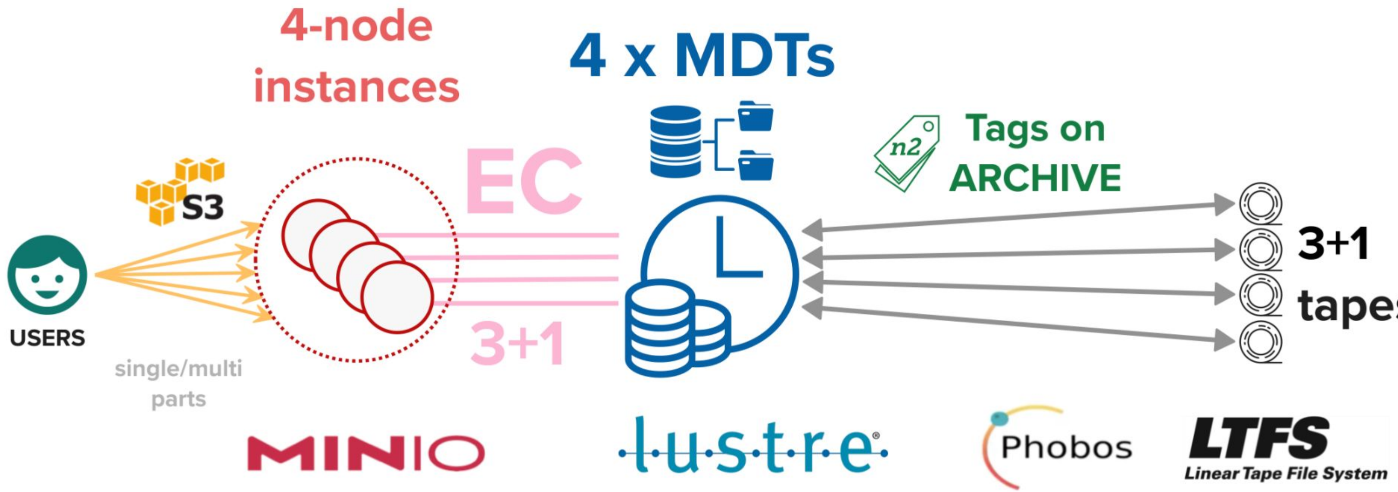
Elm big picture



Elm erasure code (EC)

Provide improved availability and resiliency

The number of MDTs (4) matches the size of MinIO's erasure coding set (3+1)



Elm backend storage capacity

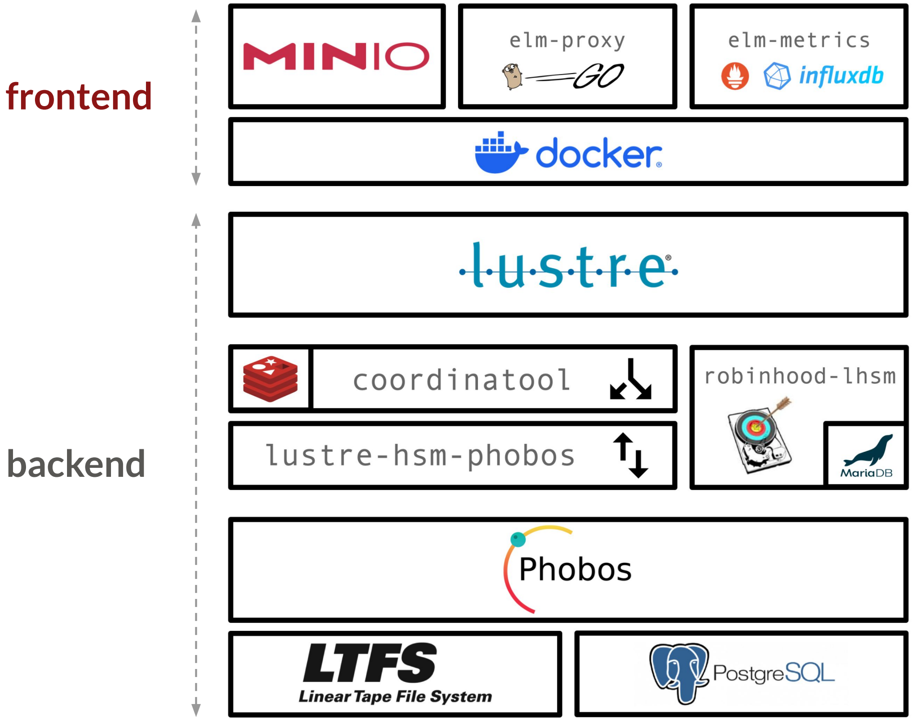
lustre

LTFS
Linear Tape File System

7.5 PB
+4 x 12TB MDTs

10-175 PB
10 PB now with LTO-9
w/o compression

Elm open source software stack



MinIO

Focus on Elm's frontend

Elm frontend: MinIO architecture

MinIO's distributed architecture on Docker Swarm

Multiple instances of MinIO servers are compartmentalized by project

- parallel data intake
- enhanced security

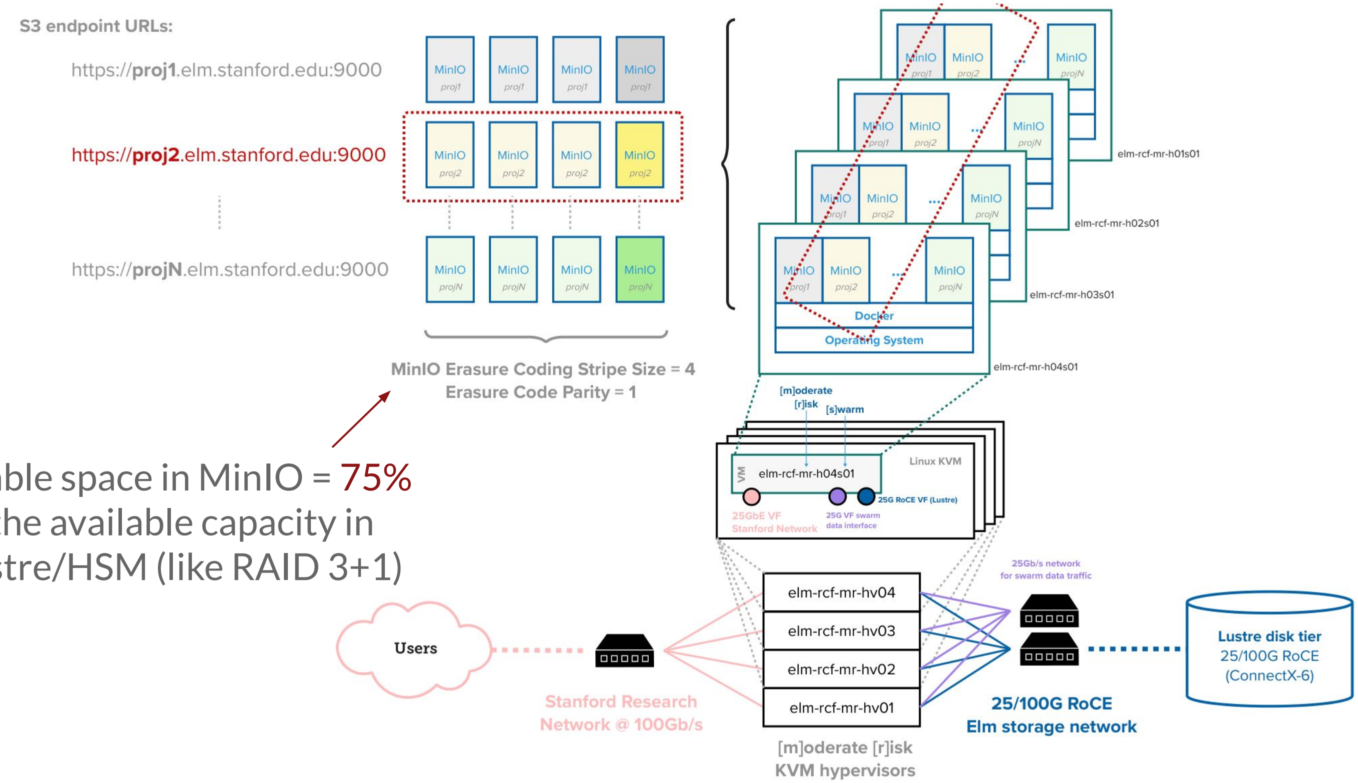
KVM virtualization as an added layer

- Isolate VMs from management network
- Several docker swarm instances are running in VMs with different data risk classifications and Stanford VLAN access
- Potentially could allow us to migrate VMs in the future

Elm frontend: MinIO S3 endpoint

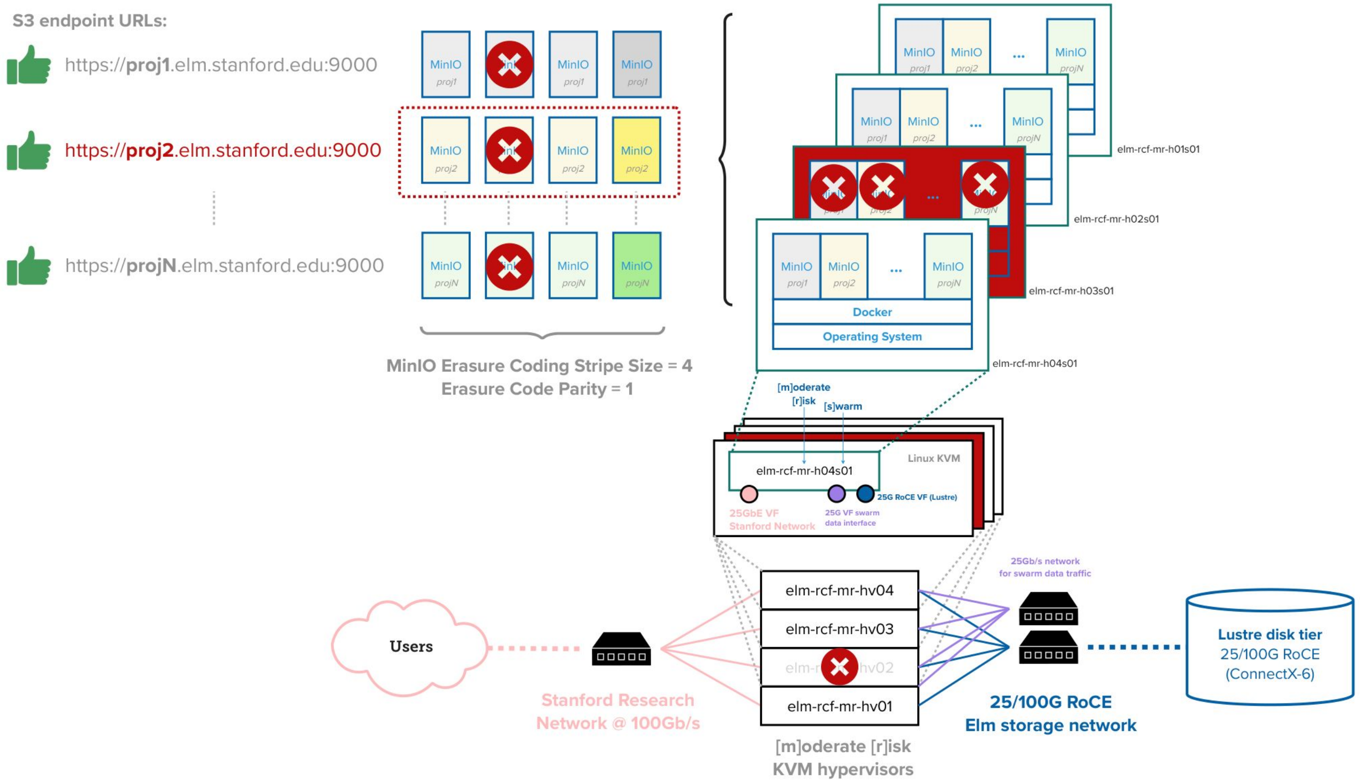
- Primary interface for data transfers
- Supports multipart uploads
- Integrates seamlessly with S3-compatible tools
- S3 checksum/metadata features for data integrity

Elm: MinIO project layout with MinIO EC



usable space in MinIO = 75% of the available capacity in Lustre/HSM (like RAID 3+1)

Elm: MinIO EC resilience showcase



Don’t try this at home

The configuration used for Elm is a custom implementation designed for specific requirements. It may NOT comply with various MinIO’s recommended practices, like:

- XFS as backend filesystem
- At least 4 drives per MinIO Server
- Slow/tape storage backend is not supported
    - Tiering is supported between immediate-access S3 tiers
- EC:1 (RSS) is usually too low for standard MinIO deployments

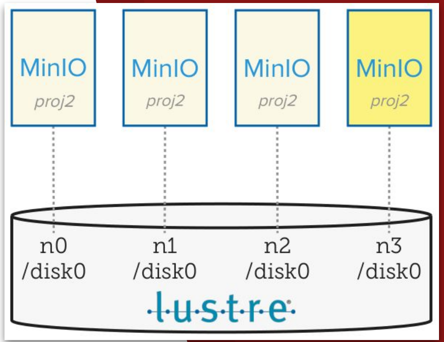
MinIO/Lustre

Focus on the interaction between MinIO and Lustre

Elm MinIO/Lustre: structured path design

Descriptive path pattern used on Elm's Lustre filesytem:

/elm/stanford/**<class>**/projects/**<project>**/minio/**<node#>**/disk0/



access via separate docker swarm per data risk classification

unique MinIO server instance per project

/elm/stanford/**mr**/projects/**hnc**/minio/**n2**/disk0/

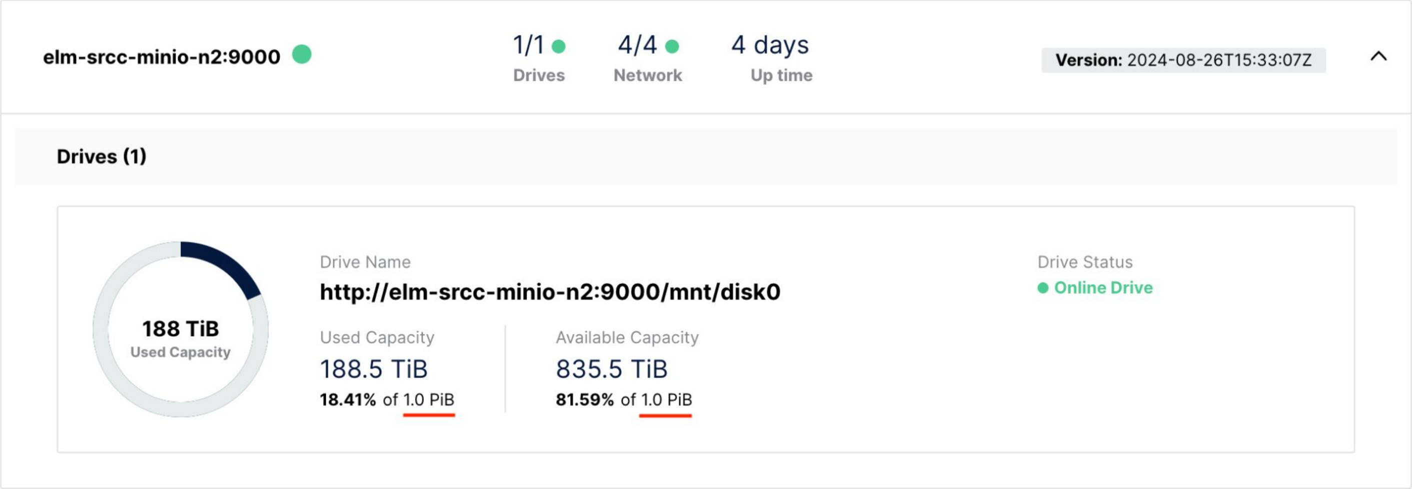
Lustre nodemap's fileset mounted exclusively on VMs with the matching risk level

directory bound to a single MinIO node and used as target disk (1 / node)

created on specific MDT
for now, *nX* always on elm-MDT#*X*

Elm MinIO/Lustre: project ID quotas

We use Lustre project disk quotas to limit each MinIO disk (primarily cosmetic), relying on projid-specific statfs(). The LU-16771 issue with **statfs_max_age** hasn't affected us here so far, unlike with SMB.

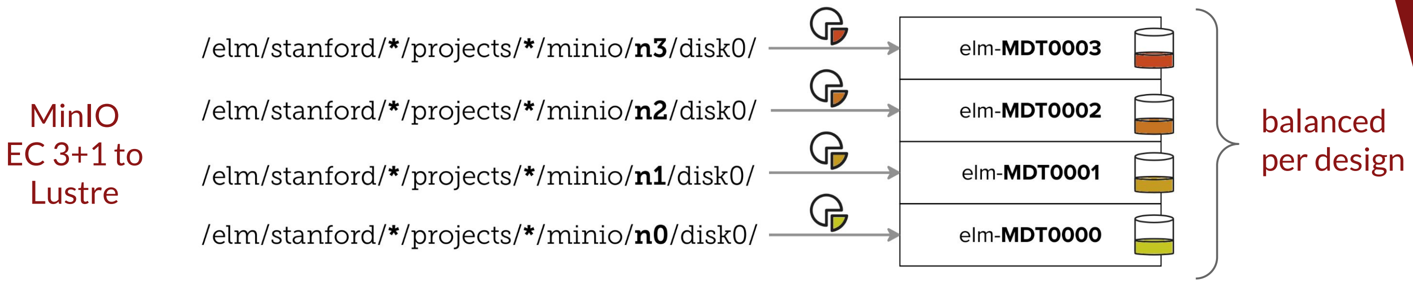


Elm MinIO EC/Lustre MDT relation

Files from each MinIO shard are distributed across four Lustre MDTs using DNEv1, ensuring balanced and uniform growth across all MDTs.

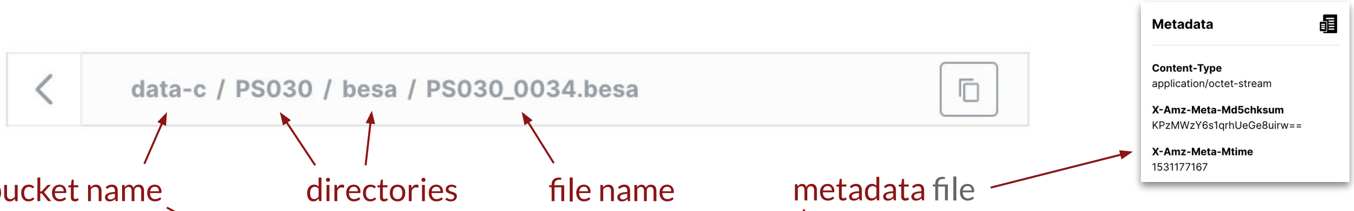
- `lmv_stripe_count=1, lmv_max_inherit=-1`

When inode capacity is reached, four new MDTs can be added, and striping for new files in existing projects will be adjusted accordingly.



Elm MinIO/Lustre: MinIO file layout

Despite its internal nature, MinIO's file layout is documented and deterministic. The layout is replicated across the shards, ensuring consistency.



file version (UUIDv4)

```
.../n{0-3}/disk0/data-c/PS030/besa/PS030_0034.besa/xl.meta
.../n{0-3}/disk0/data-c/PS030/besa/PS030_0034.besa/fa7040ce-9869-4b2a-a761-b1444706fe40
.../n{0-3}/disk0/data-c/PS030/besa/PS030_0034.besa/fa7040ce-9869-4b2a-a761-b1444706fe40/part.1
.../n{0-3}/disk0/data-c/PS030/besa/PS030_0034.besa/fa7040ce-9869-4b2a-a761-b1444706fe40/part.2
.../n{0-3}/disk0/data-c/PS030/besa/PS030_0034.besa/fa7040ce-9869-4b2a-a761-b1444706fe40/part.3
```

data parts

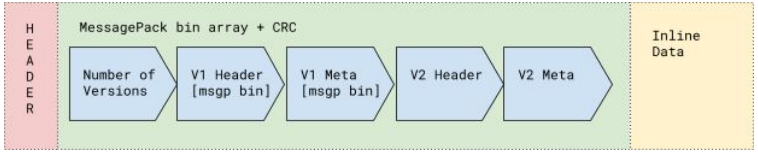
Elm MinIO/Lustre: small files and `xl.meta`

MinIO uses `xl.meta` to handle metadata and store small files (< 128kB) as inline data when possible. These `xl.meta` files are erasure coded, ensuring redundancy and protection for both data and metadata.

bucket name → bucket-1 | directory → foobar | file name → small image.jpg | metadata and data → xl.meta

(no part files in that case)

`…/n{0-3}/disk0/bucket-1/foobar/small image.jpg/xl.meta`



Overall structure of xl.meta

See also: https://blog.min.io/minio-versioning-metadata-deep-dive/

Elm **MinIO/Lustre**: MDTs and `inline_data`

If Lustre were to support ext4's `inline_data` feature, small directories could be stored directly within the inode. This would likely allow directories with up to ~50 `parts.x` files to avoid consuming additional blocks, improving storage efficiency.

LU-5603: Enable inline_data feature for Lustre

- This feature *could* improve performance by eliminating block redirection
- Smaller MDT volumes could store the same number of small directories, potentially lowering costs
- Despite these benefits, the **4-billion inode limit per MDT** (for both files and directories) would still apply

Lustre on Elm

Focus on the centerpiece

Elm with Lustre 2.16 (pre-release)

OS is Rocky Linux 9 (EOL 2032) managed by xCAT

Lustre 2.16 pre-release (2.15.62 + patches)

- Lustre 2.15 LTS does not support EL9 for servers
- Lustre/ldiskfs for MDTs (Dell ME5 ADAPT) and OSTs (Linux RAID-6)
- LU-17711: ldiskfs corruption on el9 (dx_probe: Corrupt directory)
  - Triggered with MinIO (rename after each upload)
  - Rolled back el9 ldiskfs code closer to el8 for now
    - `This revert linux upstream commit 6c0912739699 ("ext4: wipe ext4_dir_entry2 upon file deletion")`
  - Thanks to Dominique Martinet and Whamcloud!
- LU-18223: (...) ASSERTION(hsd->hsd_request_count < hsd->hsd_request_len)
- LU-18238: Ghost hsm/agents on MDTs

Elm with Lustre/RoCE

Our first Lustre/RoCE (o2ib) deployment

- Cost-effective (vs. IB) and flexible RDMA solution for a Lustre-only network
- Deployed with NVIDIA Spectrum SN3420 and SN2010 switches
- RoCE network spread across 2x DC rooms via 4 x 100Gb/s port channel
- 25Gb/s RoCE VFs on KVM virtual machines
- First, we tried with Broadcom 57414 NICs without success…
  - We were told RoCE Virtual Functions (VF) were supported…
  - Buried in a doc: `RDMA SR-IOV is supported on BCM575xx devices only`
- Working seamlessly with NVIDIA ConnectX-6 25G and 100G (OSS only)

SN3420

Lustre/Phobos

Open source Lustre/HSM solution

What is Phobos?

Parallel Heterogeneous OBject Store

Development led by CEA, source code available on GitHub:

- https://github.com/phobos-storage

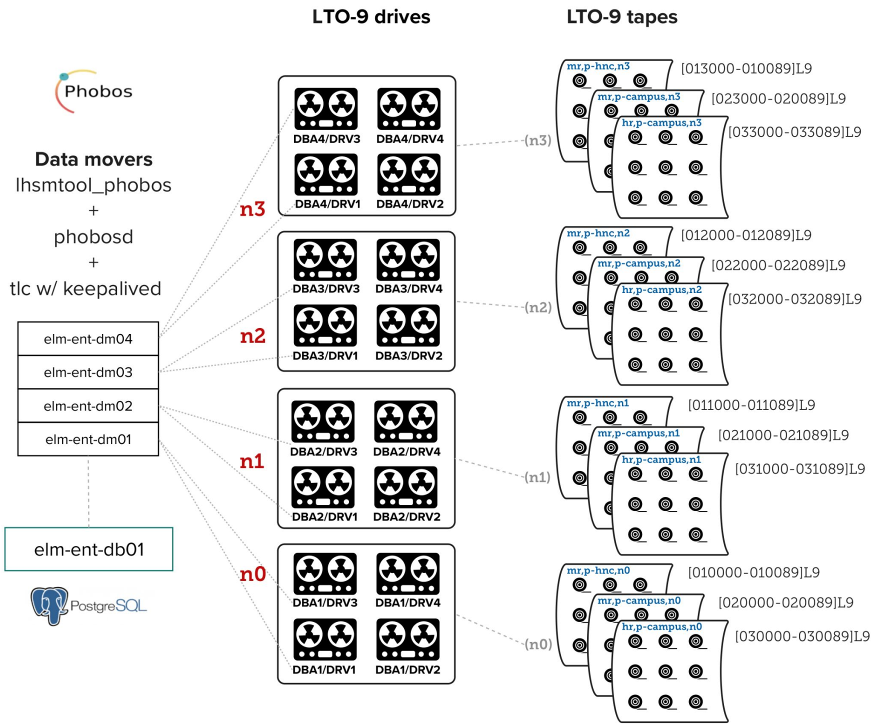
Uses LTFS as tape filesystem (open format ISO/IEC 20919:216)

Lustre/Phobos HSM copytool available (`lhsmtool_phobos`)

Lustre HSM coordinator in user-space (`coordinatool`) **required** when using multiple Phobos data movers

Phobos manages tape drives and tapes and stored objects, with **tags** support for tapes. A PostgreSQL database is required.

Phobos on Elm: tapes with tags



Lustre/Phobos archive policy

Robinhood v3 with Lustre/HSM support runs the archive policy

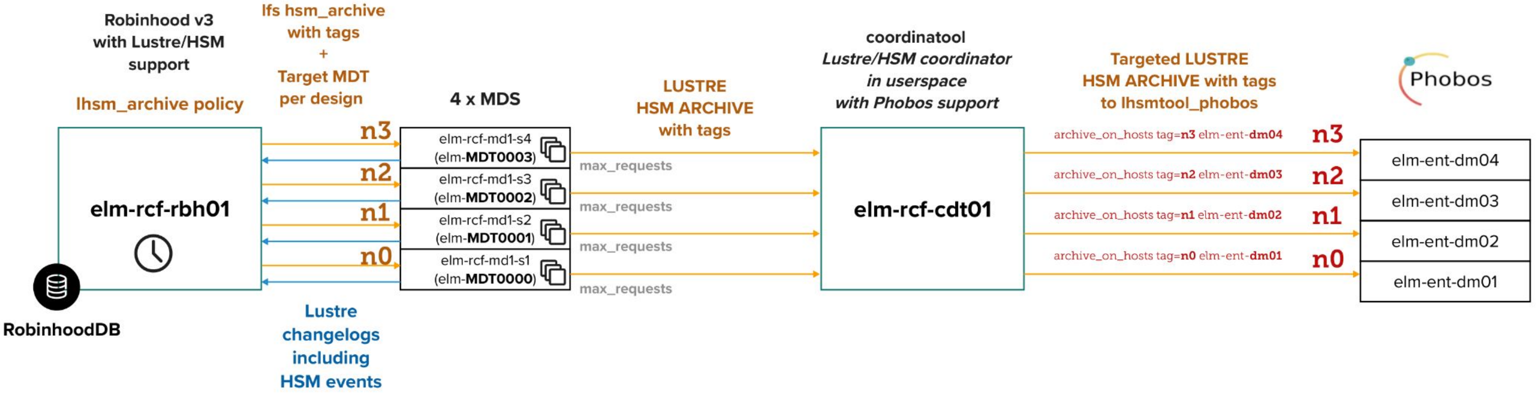
```
lhsm_archive_rules {
    ignore_fileclass = system;
    rule archive_minio {
        target_fileclass = mr_srcc_minio_n0;
        target_fileclass = mr_srcc_minio_n1;
        target_fileclass = mr_srcc_minio_n2;
        target_fileclass = mr_srcc_minio_n3;
        # Archive to Phobos with tags
        action = cmd("lfs hsm_archive --archive {archive_id} --data 'tag={risk},tag={project},tag={minio_n}' {fullpath}");
        condition { tree != "/elm/*/*/*/*/minio/*/*/.minio.sys" and
                    size > 0 and
                    last_mod >= 1d }
    }
}
```

Lustre/HSM with Phobos

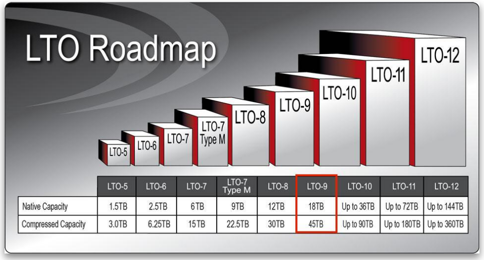
Coordinatool is a key component of a distributed Lustre/Phobos HSM.

Improvements made by Dominique Martinet (`archive_on_hosts`).

More need to be done for archiving by tag to avoid tape movements.

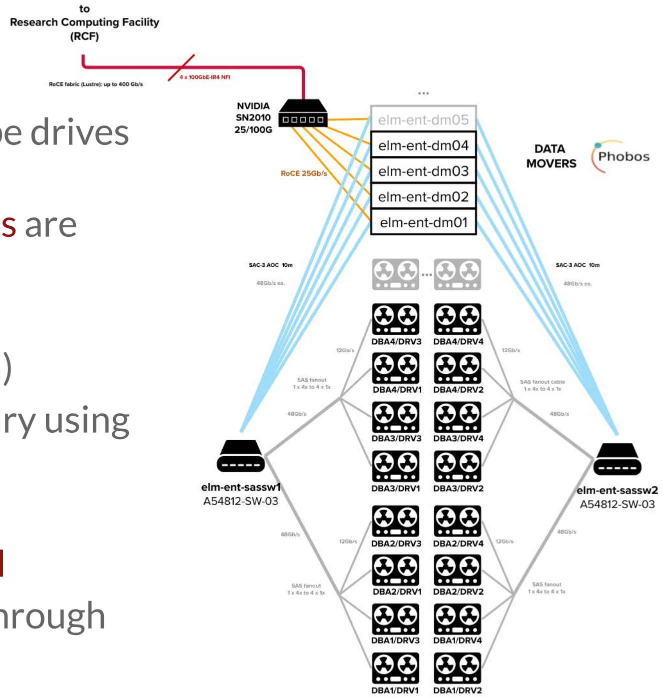


Elm **SAS tape drives**



	LTO-5	LTO-6	LTO-7	LTO-7 Type M	LTO-8	LTO-9	LTO-10	LTO-11	LTO-12
Native Capacity	1.5TB	2.5TB	6TB	9TB	12TB	18TB	Up to 36TB	Up to 72TB	Up to 144TB
Compressed Capacity	3.0TB	6.25TB	15TB	22.5TB	30TB	45TB	Up to 90TB	Up to 180TB	Up to 360TB

- 16 x IBM LTO-9 SAS 12Gb/s tape drives
- 1-to-4 fanout SAS 12Gb/s cables are used to connect the drive
- SAS switches (12 x 48Gb/s each) installed on top of the tape library using SpectraLogic's 3U bracket
- Library access (changer) via ADI (Automation/Drive Interface) through configurable SAS drive(s)

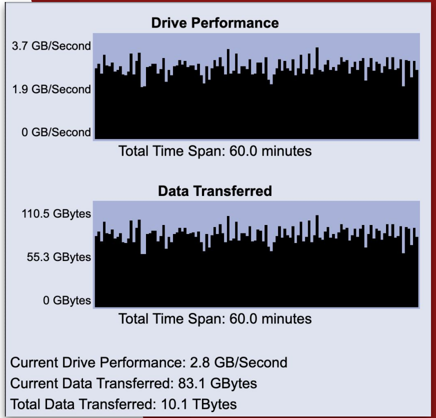


Lustre/Phobos LAD’24 “hero” run

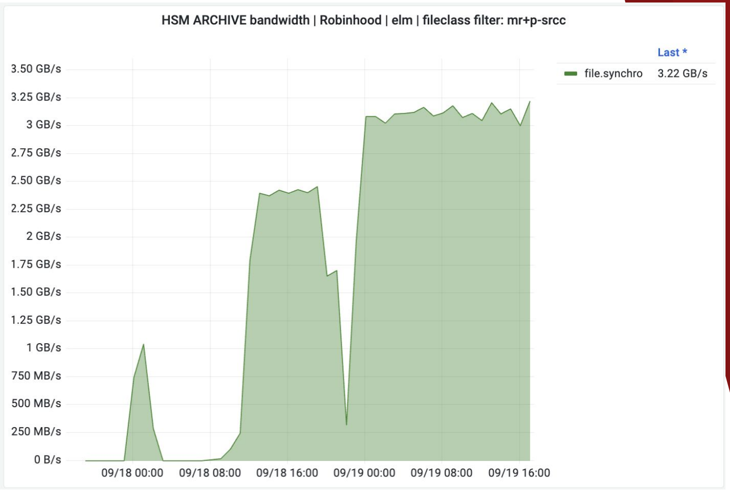
- Lustre/HSM with DNE archive run using large files, 4 data movers (one per MinIO shard), 4 LTO-9 SAS drives per mover
- Allow all drives to write:

```
# clush -w @dm phobos sched fair_share --type LTO9 --max 0,4,0
```

- LTFS sync every 2 mins or 1000 files or 16 GiB (set in `phobos.conf`)
- `hsm/max_requests=`750/MDT
- Results:
  - 3.22 GB/s aggregate max
  - 201.25 MB/s per drive
  - **10+ TB/hour archived**
  - **1 PB in 4 days archived**



not LumOS!



Lustre/Phobos LAD’24 “hero” run (cont’d)

```
[root@elm-rcf-hn01 ~]# clush -w@dm -x elm-ent-dm05 -b phobos drive status
---------------
elm-ent-dm01
---------------
|   address | currently_dedicated_to   | device    | media    | mount_path        | name                              | ongoing_io   | serial      |
|-----------|--------------------------|-----------|----------|-------------------|-----------------------------------|--------------|-------------|
|         0 | W                        | /dev/sg2  | 010050L9 | /mnt/phobos-sg2   | /dev/tape/by-id/scsi-10110057FB | True         | 10110057FB |
|         1 | W                        | /dev/sg12 | 010052L9 | /mnt/phobos-sg12  | /dev/tape/by-id/scsi-10120057FB | True         | 10120057FB |
|         2 | W                        | /dev/sg1  | 010023L9 | /mnt/phobos-sg1   | /dev/tape/by-id/scsi-10130057FB | True         | 10130057FB |
|         3 | W                        | /dev/sg11 | 010041L9 | /mnt/phobos-sg11  | /dev/tape/by-id/scsi-10140057FB | True         | 10140057FB |
---------------
elm-ent-dm02
---------------
|   address | currently_dedicated_to   | device    | media    | mount_path        | name                              | ongoing_io   | serial      |
|-----------|--------------------------|-----------|----------|-------------------|-----------------------------------|--------------|-------------|
|         4 | W                        | /dev/sg4  | 010045L9 | /mnt/phobos-sg4   | /dev/tape/by-id/scsi-10210057FB | True         | 10210057FB |
|         5 | W                        | /dev/sg13 | 010013L9 | /mnt/phobos-sg13  | /dev/tape/by-id/scsi-10220057FB | True         | 10220057FB |
|         6 | W                        | /dev/sg5  | 010019L9 | /mnt/phobos-sg5   | /dev/tape/by-id/scsi-10230057FB | True         | 10230057FB |
|         7 | W                        | /dev/sg15 | 010012L9 | /mnt/phobos-sg15  | /dev/tape/by-id/scsi-10240057FB | False        | 10240057FB |
---------------
elm-ent-dm03
---------------
|   address | currently_dedicated_to   | device    | media    | mount_path        | name                              | ongoing_io   | serial      |
|-----------|--------------------------|-----------|----------|-------------------|-----------------------------------|--------------|-------------|
|         8 | W                        | /dev/sg7  | 010021L9 | /mnt/phobos-sg7   | /dev/tape/by-id/scsi-10310057FB | True         | 10310057FB |
|         9 | W                        | /dev/sg17 | 010068L9 | /mnt/phobos-sg17  | /dev/tape/by-id/scsi-10320057FB | True         | 10320057FB |
|        10 | W                        | /dev/sg6  | 010044L9 | /mnt/phobos-sg6   | /dev/tape/by-id/scsi-10330057FB | True         | 10330057FB |
|        11 | W                        | /dev/sg16 | 010069L9 | /mnt/phobos-sg16  | /dev/tape/by-id/scsi-10340057FB | True         | 10340057FB |
---------------
elm-ent-dm04
---------------
|   address | currently_dedicated_to   | device    | media    | mount_path        | name                              | ongoing_io   | serial      |
|-----------|--------------------------|-----------|----------|-------------------|-----------------------------------|--------------|-------------|
|        12 | W                        | /dev/sg7  | 010042L9 | /mnt/phobos-sg7   | /dev/tape/by-id/scsi-10410057FB | True         | 10410057FB |
|        13 | W                        | /dev/sg17 | 010037L9 | /mnt/phobos-sg17  | /dev/tape/by-id/scsi-10420057FB | True         | 10420057FB |
|        14 | W                        | /dev/sg8  | 010038L9 | /mnt/phobos-sg8   | /dev/tape/by-id/scsi-10430057FB | True         | 10430057FB |
|        15 | W                        | /dev/sg18 | 010065L9 | /mnt/phobos-sg18  | /dev/tape/by-id/scsi-10440057FB | True         | 10440057FB |
```

Elm: Next steps?

- Deep dive into HSM restore for reliability and convenience
  - file restore testing at scale
  - large dataset restore scheduling
- Phobos
  - Reduce tape movements when archiving
    - Fair share distribution per tag(s) (GH issue #10)
    - Locate mounted tapes with matching tags?
  - Explore PgBouncer for PostgreSQL connection pooling
    - Mitigate DB server load spikes during mass HSM archive
- Address occasional SCSI timeouts with LTFS
- Robinhood v3 fileclass future scalability challenges: try v4?

Links

- MinIO
  - https://github.com/minio/minio
- Phobos GitHub org
  - https://github.com/phobos-storage
- Elm's coordinatool
  - https://github.com/stanford-rc/coordinatool/tree/elm
- Robinhood v3 (projid, stripe_index, creation_time, EL9, no tests)
  - https://github.com/stanford-rc/robinhood/commits/prod/
- s3up: S3 uploader tool with full-body checksum support
  - https://github.com/stanford-rc/s3up
- rpm-ltfs: EL9 spec file for LTFS v2.4
  - https://github.com/piste2750/rpm-ltfs/

Elm in pictures

KVM hypervisors
25Gb/s RoCE SR-IOV

4 x Lustre OSS
100Gb/s RoCE

Intra-building fiber optics
4x100Gb/s

SAS switches in
top-of-library bracket

SpectraLogic TFinity tape
library on ISO-Base

Stanford network
100Gb/s IP backbone

RoCE Lustre network
NVIDIA ConnectX-6

7.5 PB disk tier (OST)
8 x Western Digital
Data60 22TB SED

Phobos data movers
25G RoCE/SAS AOC

16 x Tape drives
LTO-9 SAS 12Gb/s

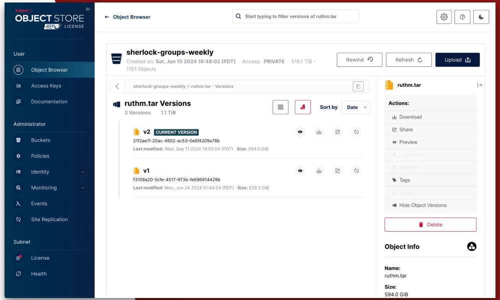
LTO-9 tapes

Stanford
University

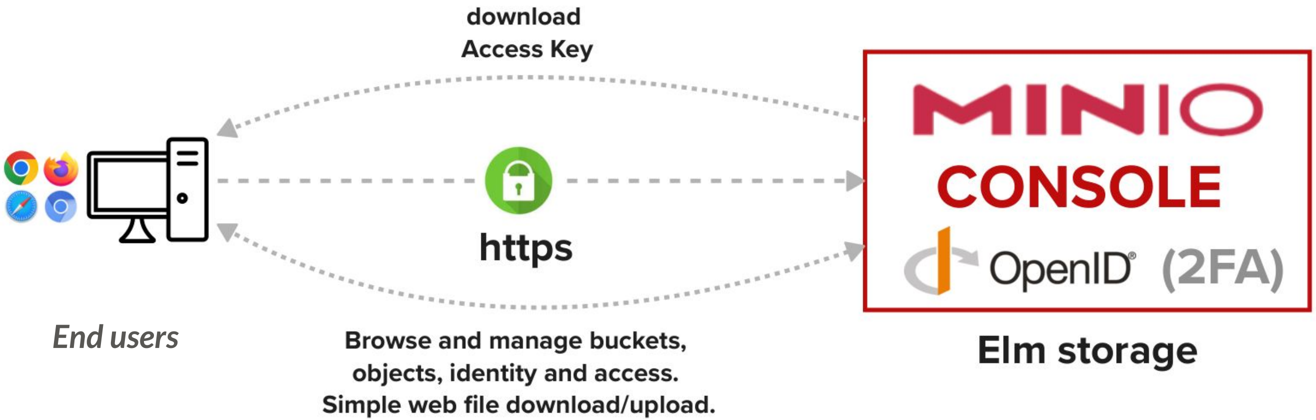
Backup slides

Too much to say

Elm frontend: MinIO console



The console simplifies interaction with the object store, offering a visual representation of data, managing access permissions, and making it easier to oversee large-scale archival tasks.



MinIO bugs discovered on Elm [FIXED]

object attributes Checksum attribute is empty for SHA1 / SHA256 multi-part uploads #20225

- https://github.com/minio/minio/issues/20225

parseObjectAttributes needs to account for the possibility of repeating headers #20267

- https://github.com/minio/minio/issues/20267

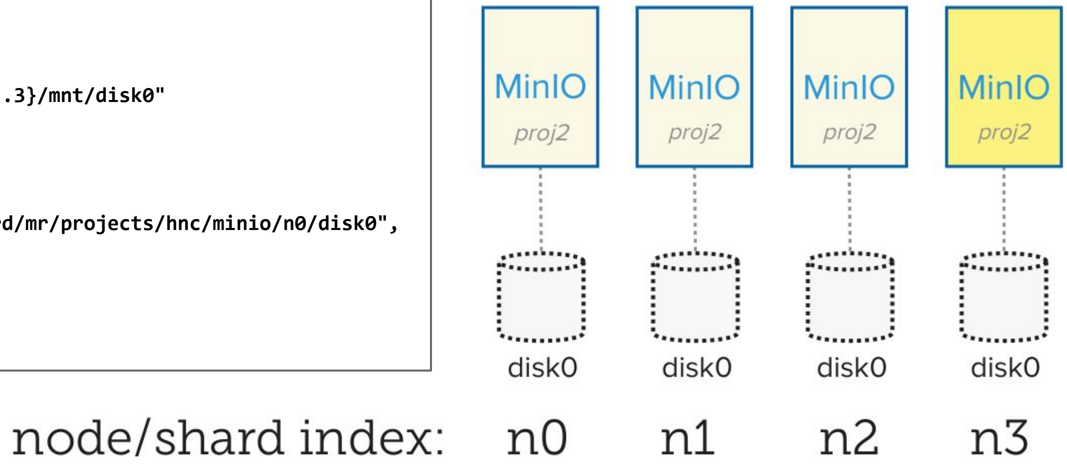
Elm MinIO disk layout

Elm currently uses a single disk per node bound to a directory in Lustre

elm-hnc-minio-n0:

```
"ContainerSpec": {
    "Args": [
        "server",
        "--console-address",
        ":9001",
        "http://elm-hnc-minio-n{0...3}/mnt/disk0"
    ],
    "Mounts": [
        {
            "Type": "bind",
            "Source": "/elm/stanford/mr/projects/hnc/minio/n0/disk0",
            "Target": "/mnt/disk0"
        }
    ...

}
```



Elm MinIO/Lustre: project ID assignment

Lustre project IDs are systematically assigned to `disk0` directories following a clear convention:

```
ProjectID: 12014624
```

- 1 — prefix for MinIO projects
- 2 — node/shard ID (and also == MDT#)
- 0 — `disk0` (always... for now!)
- 14624 — unique ID per project, from a GID in Stanford LDAP

```
# lfs project -d /elm/stanford/mr/projects/hnc/minio/n?/disk0
10014624 P /elm/stanford/mr/projects/hnc/minio/n0/disk0
11014624 P /elm/stanford/mr/projects/hnc/minio/n1/disk0
12014624 P /elm/stanford/mr/projects/hnc/minio/n2/disk0
13014624 P /elm/stanford/mr/projects/hnc/minio/n3/disk0
```

Elm MinIO/Lustre: inode growth forecasting

Understanding how inode consumption scales is essential for planning long-term capacity and ensuring efficient use of resources, ultimately saving costs.

And indeed, MinIO creates many directories in Lustre, significantly contributing to inode consumption…

1 directory per small file

```
…/n{0-3}/disk0/bucket-1/foobar/small image.jpg/xl.meta
```

at least 2 directories per larger file (more if versioned)

… per MDT!

```
…/n{0-3}/disk0/data-c/PS030/besa/PS030_0034.besa/xl.meta
…/n{0-3}/disk0/data-c/PS030/besa/PS030_0034.besa/fa7040ce-9869-4b2a-a761-b1444706fe40/part.1
…/n{0-3}/disk0/data-c/PS030/besa/PS030_0034.besa/fa7040ce-9869-4b2a-a761-b1444706fe40/part.2
…/n{0-3}/disk0/data-c/PS030/besa/PS030_0034.besa/fa7040ce-9869-4b2a-a761-b1444706fe40/part.3
```

Elm MinIO/Lustre: inodes and block usage

Inode and block usage

- Each directory uses 1 inode and 1 block per MDT, with the ability to store up to ~112 `parts.x` files per directory before requiring additional blocks
- By design, MinIO requires at least twice as many inodes as blocks to store its files

Inode configuration for Elm

- With 12TB MDTs, we chose the default MDT format options (`-i 2560`), which maximizes the inode count to the ldiskfs upper limit of 4 billion. This configuration provides approximately 1.85 billion free blocks, with an average of 2.3 inodes per block.

```
elm-MDT0000:
    Inode count:   4294376736  (max ldiskfs)
    Inode size:    1024
    Block count:   2929686528  (x 4096 = 12TB)
    Free blocks:   1854742915
    Block size:    4096
```

Lustre on Elm: hardware

Switches: NVIDIA SN3420 + SN2010 25/100GbE RoCE enabled

Metadata: **25Gb/s RoCE** and **4 large MDTs** with block level snapshots

- 1 x MGS Dell R6515 25GbE ConnectX-6 RoCE
- 2 x MDS Dell R6525 128GB 25GbE ConnectX-6 RoCE
- 1 x Dell ME5025 w/ 24 x 3.8TB SED with snapshots

IO cells (2): **100Gb/s RoCE** and **10,560TB SED raw total**

- 2 x 2 x OSS R6525 100GbE ConnectX-6 RoCE
- 2 x 4 x WD Data60 JBOD 22TB TCG

Lustre on Elm: hardware (cont'd)

HSM services (Lustre clients)

- 1 x Coordinatool server R6515 64GB 25GbE RoCE
- 1 x Robinhood server R6525 256GB 25GbE RoCE
  - 7 TB usable SSD for MariaDB
- 1 x Phobos DB server R6525 256GB 25GbE RoCE
  - 7 TB usable SSD for PostgreSQL
- 5 x Data movers (Phobos) Dell R6525 128GB 25GbE RoCE
  - Connected to the tape library and tapes drives via SAS

Enterprise DC

Frontend (Lustre clients)

- 8 x KVM Hypervisors R6515 256GB 25GbE/25GbE RoCE SR-IOV

Stanford
University