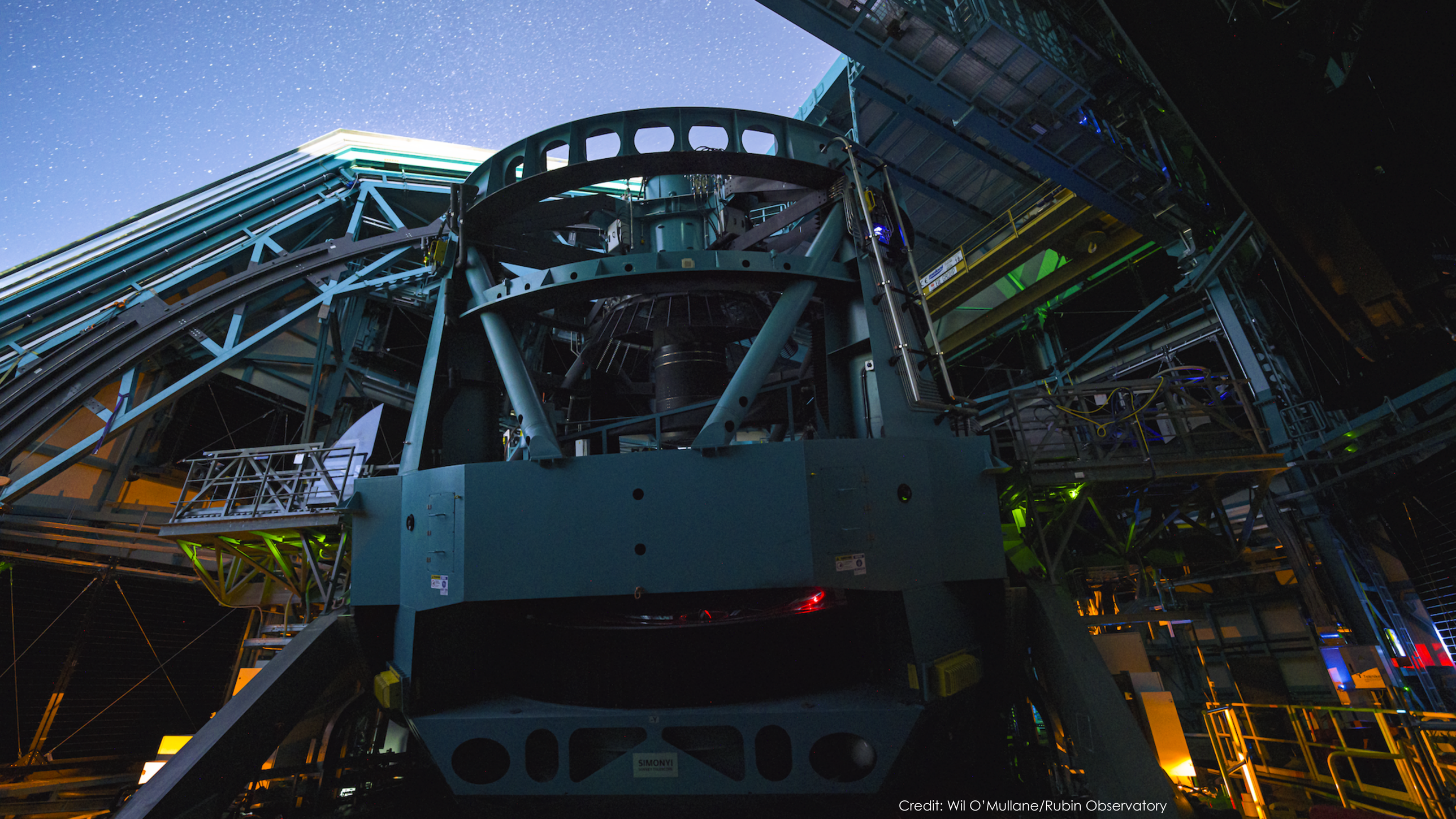
**Centre de Calcul**
de l'Institut National de Physique Nucléaire
et de Physique des Particules

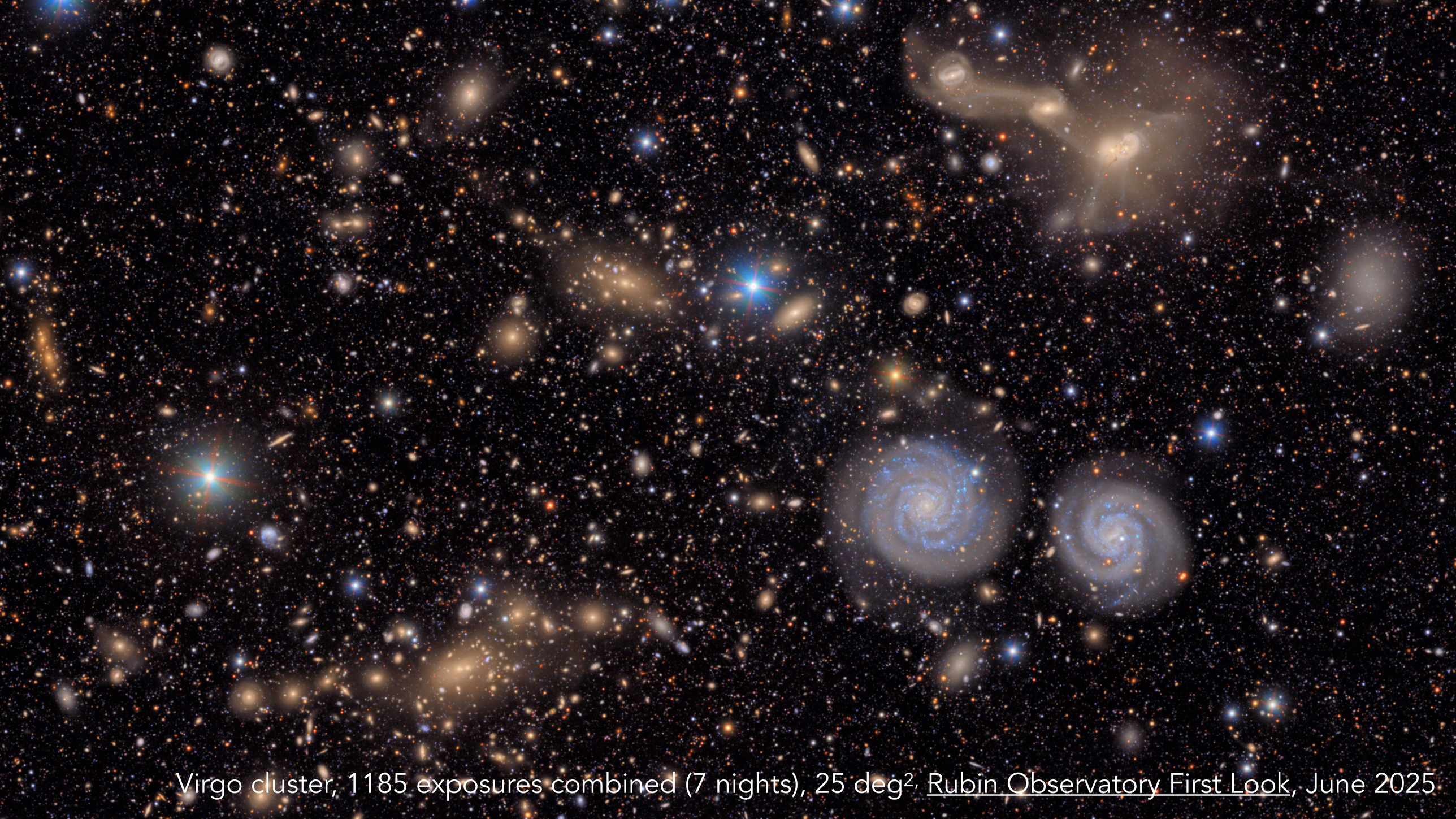
# Update on the usage of dCache for Rubin Observatory at CC-IN2P3

**adrien georget, fabio hernandez, quentin le boulc'h**

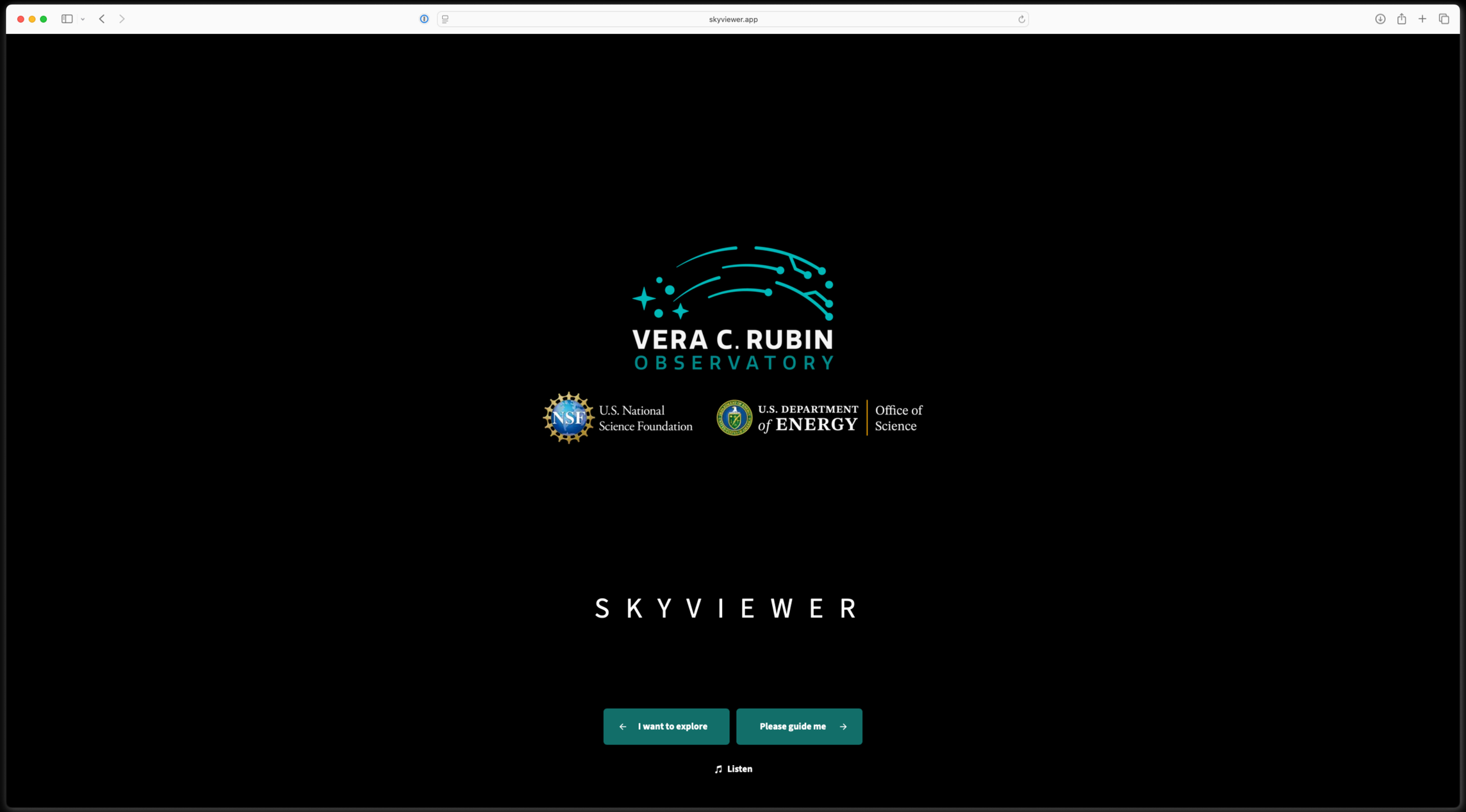
*20th International dCache Workshop, Amsterdam, May 7th, 2026*




Credit: Wil O'Mullane/Rubin Observatory



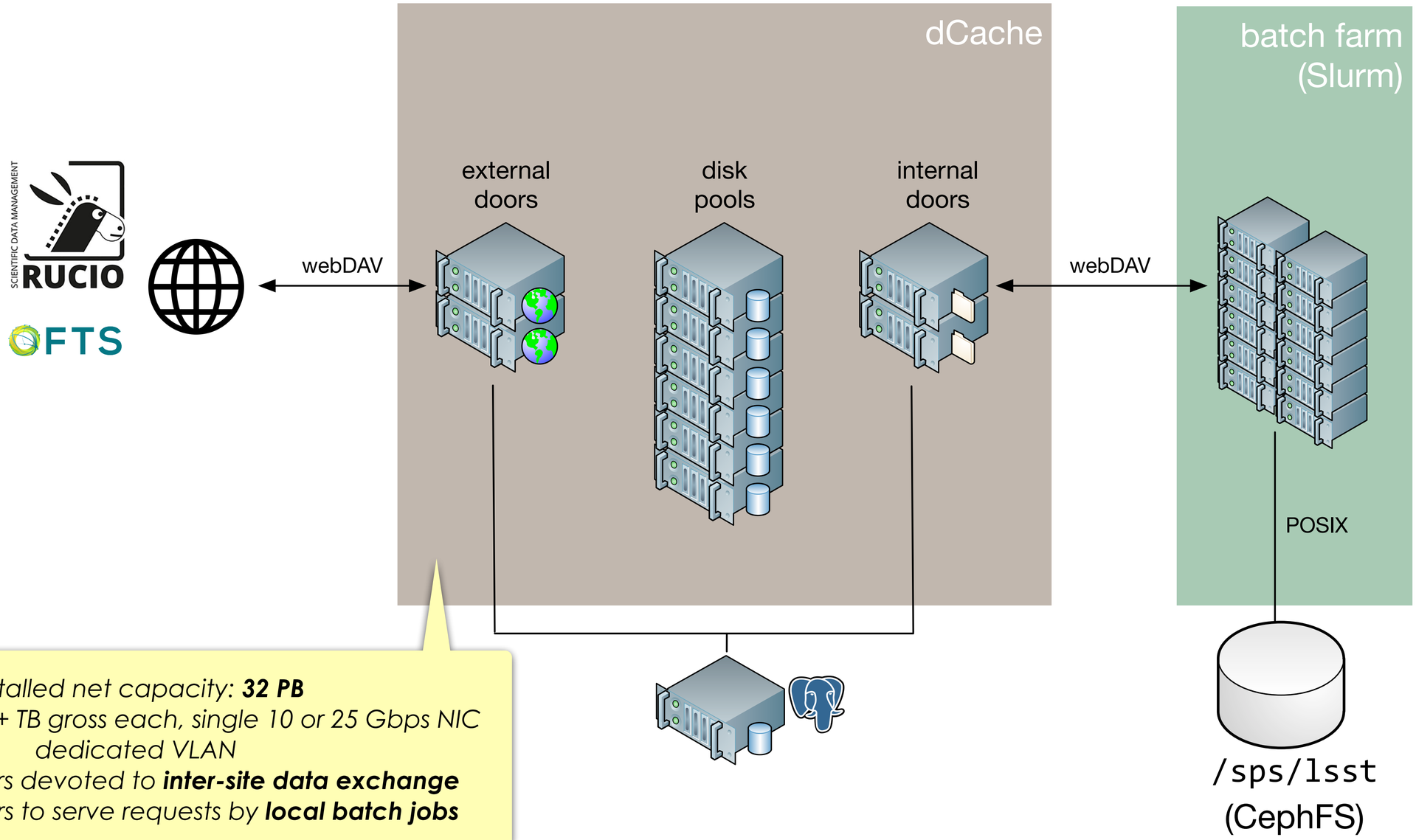
Virgo cluster, 1185 exposures combined (7 nights), 25 deg$^2$, Rubin Observatory First Look, June 2025




https://skyviewer.app

# How we use dCache

# RUBIN INSTANCE OVERVIEW

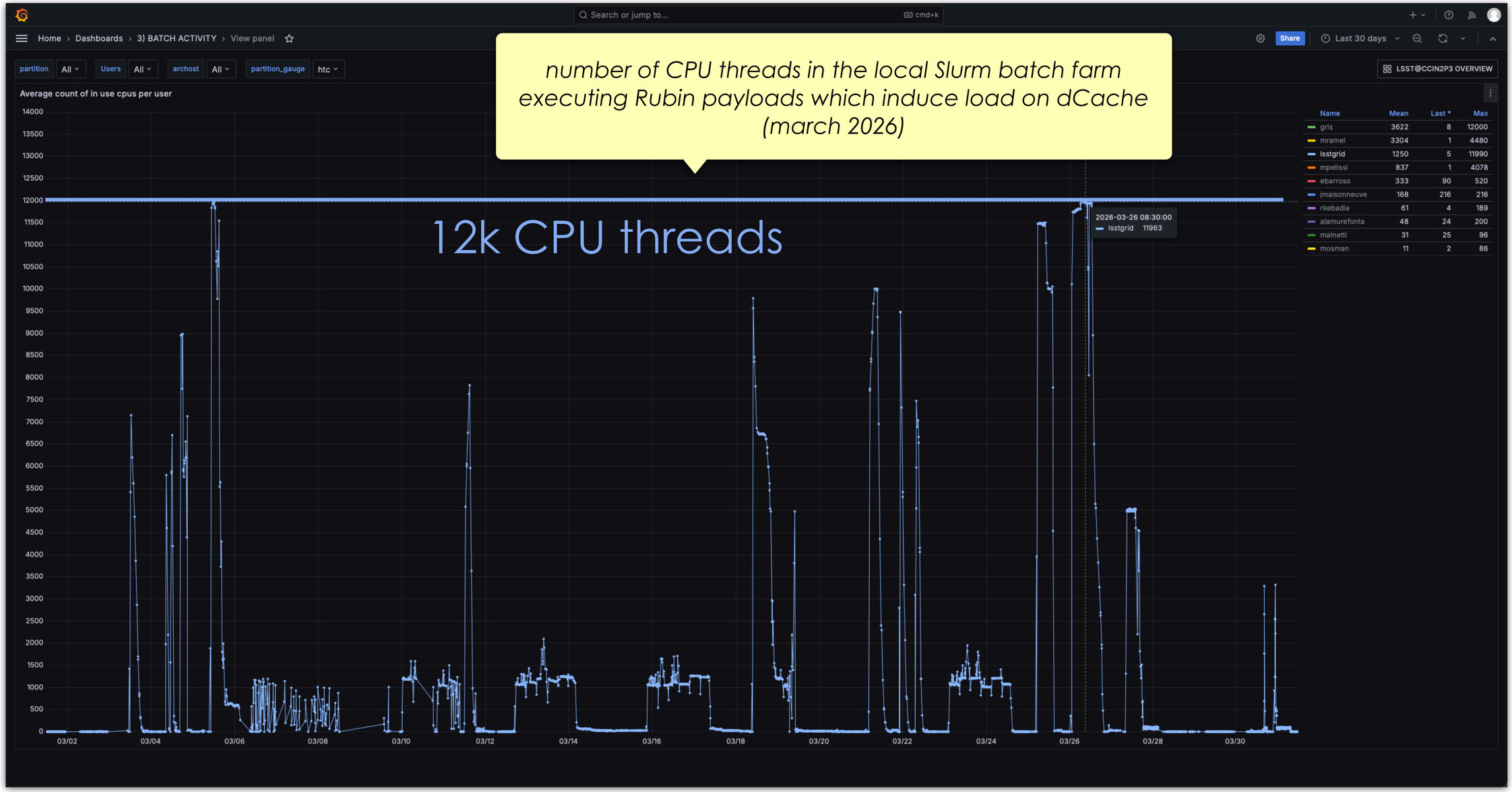



*installed net capacity:* ***32 PB***
*97 disk pools, 500+ TB gross each, single 10 or 25 Gbps NIC dedicated VLAN*
*2 webDAV doors devoted to* ***inter-site data exchange***
*2 webDAV doors to serve requests by* ***local batch jobs***

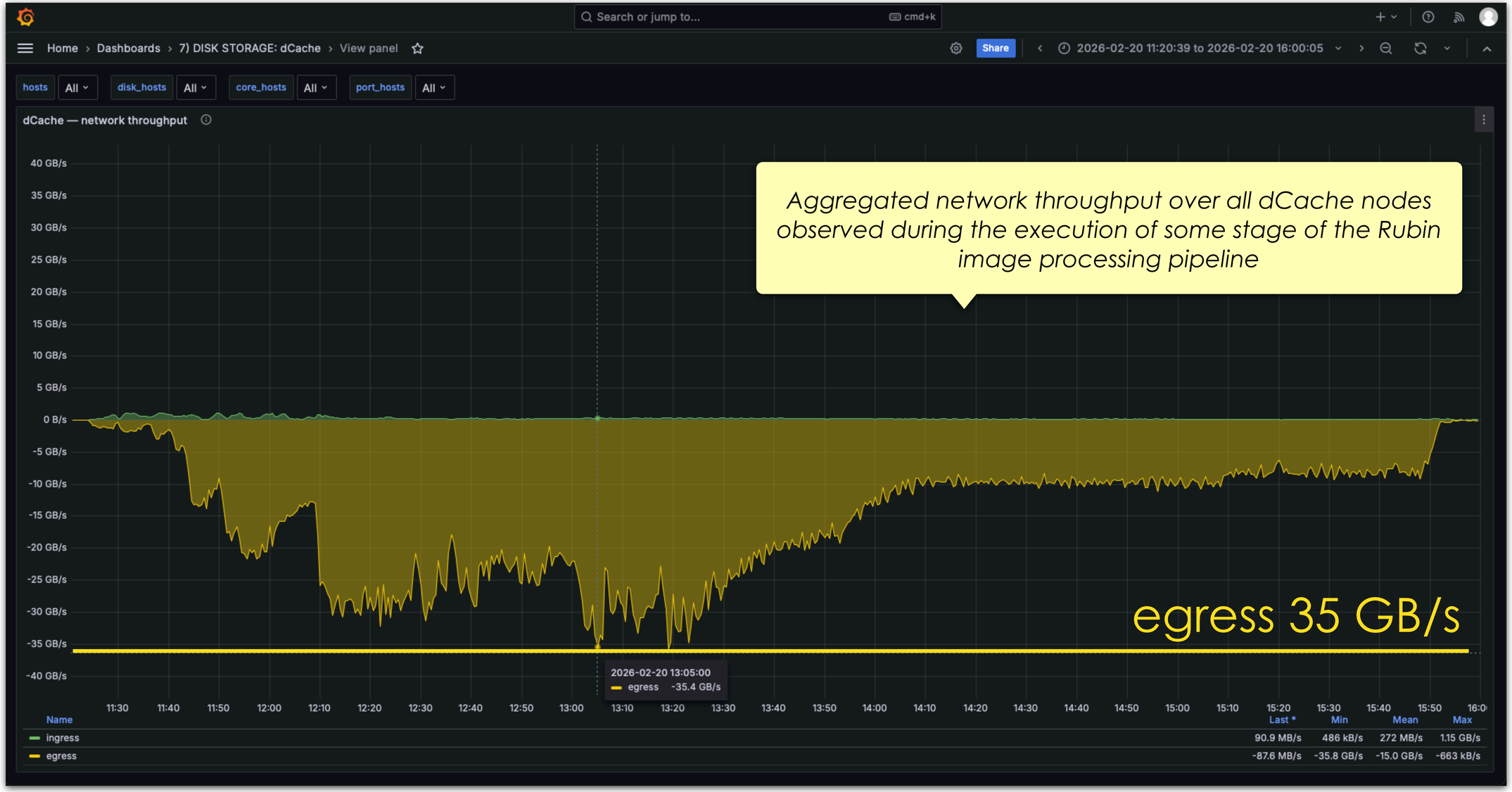
# RUBIN INSTANCE OVERVIEW (CONT.)

- Production version: mix of `dCache v10.2.14` and `v10.2.23`
- Usable capacity: 32 PB
  *planned growth of about 10 PB per year for 10 years*
- External doors use **secure HTTP** to redirect external clients to pools
  *all inter-site transfers use end-to-end confidential channels*
  *currently using X.509 proxy certificates, preparing introduction of OIDC tokens*
- Internal doors use **plain HTTP**
  *significant reduction of CPU load on the doors, pools, and clients: reduced total service time highly beneficial, in particular for metadata requests (more on this later)*
  *establishing a TCP connection over the LAN takes 0.5ms, to be compared to TLS which takes ~20ms*
  *currently using HTTP basic authentication for production jobs, humans use either bearer tokens or X.509 proxies*
  *tokens not usable with plain HTTP*
- Full flash, highly available Postgres database
  *current data volume: 650 GB*

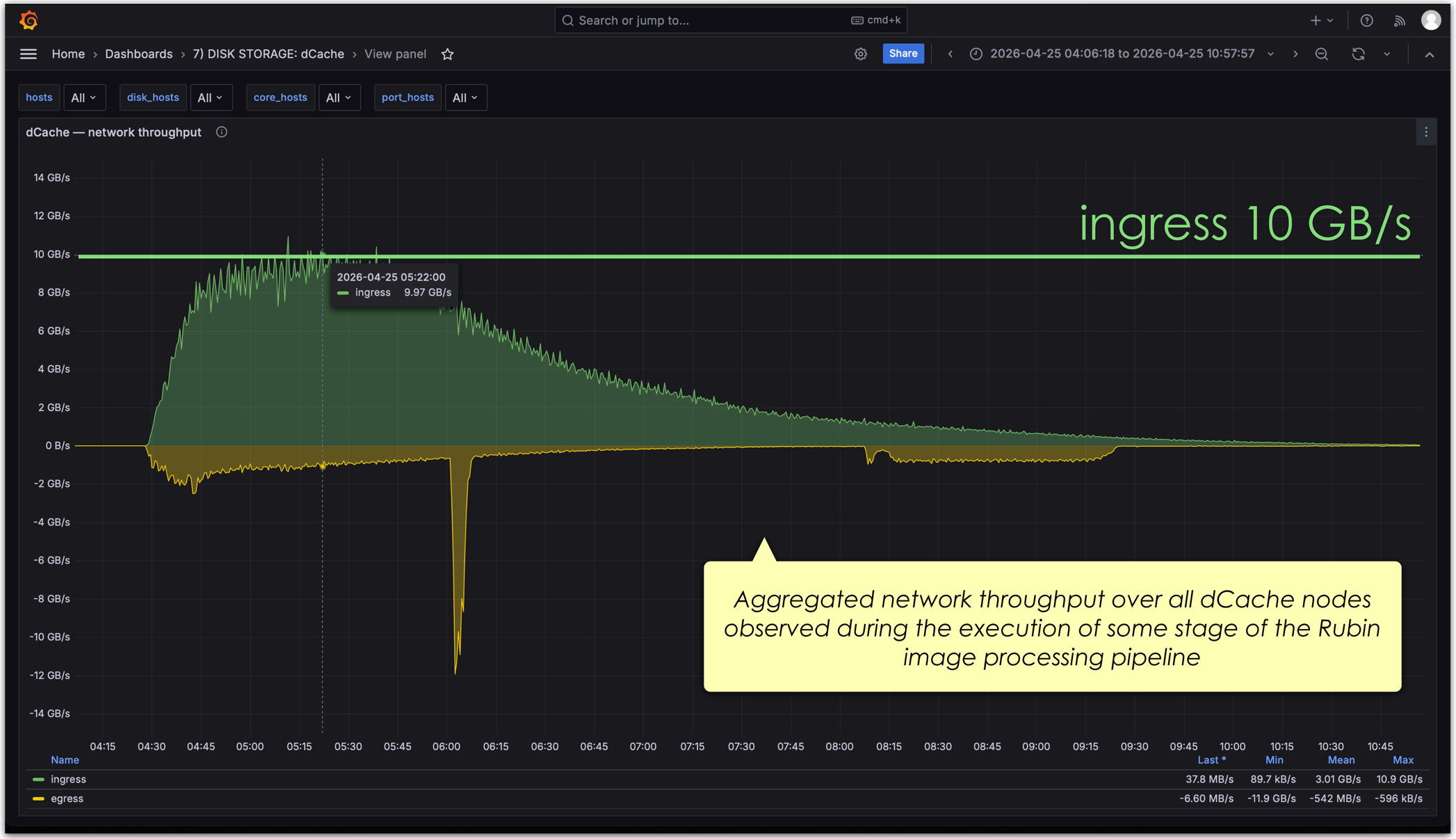
# RUBIN INSTANCE OVERVIEW (CONT.)



# RUBIN INSTANCE OVERVIEW (CONT.)



# RUBIN INSTANCE OVERVIEW (CONT.)




*Aggregated network throughput over all dCache nodes observed during the execution of some stage of the Rubin image processing pipeline*

# CLIENT

- Support for RFC 4918 WebDAV developed and maintained by CC-IN2P3

*Python, on top of urllib3, thread-safe, compatible with fsspec for reading*

*WebDAV compliance class 1 is sufficient, we don't use* `LOCK`

*both X.509- and token-based client identification*

*installable via the Python Package Index, can be used independently of the Rubin software stack*

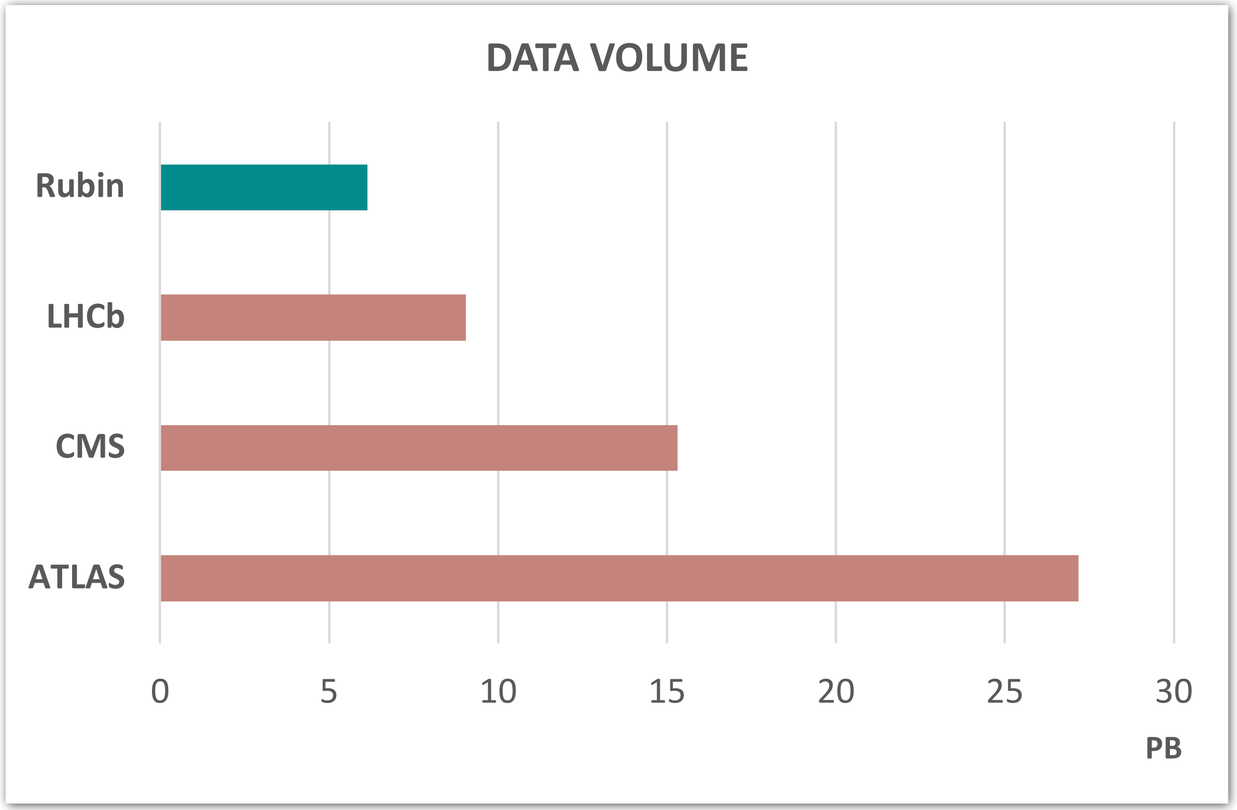
*further details https://pipelines.lsst.io/modules/lsst.resources/*

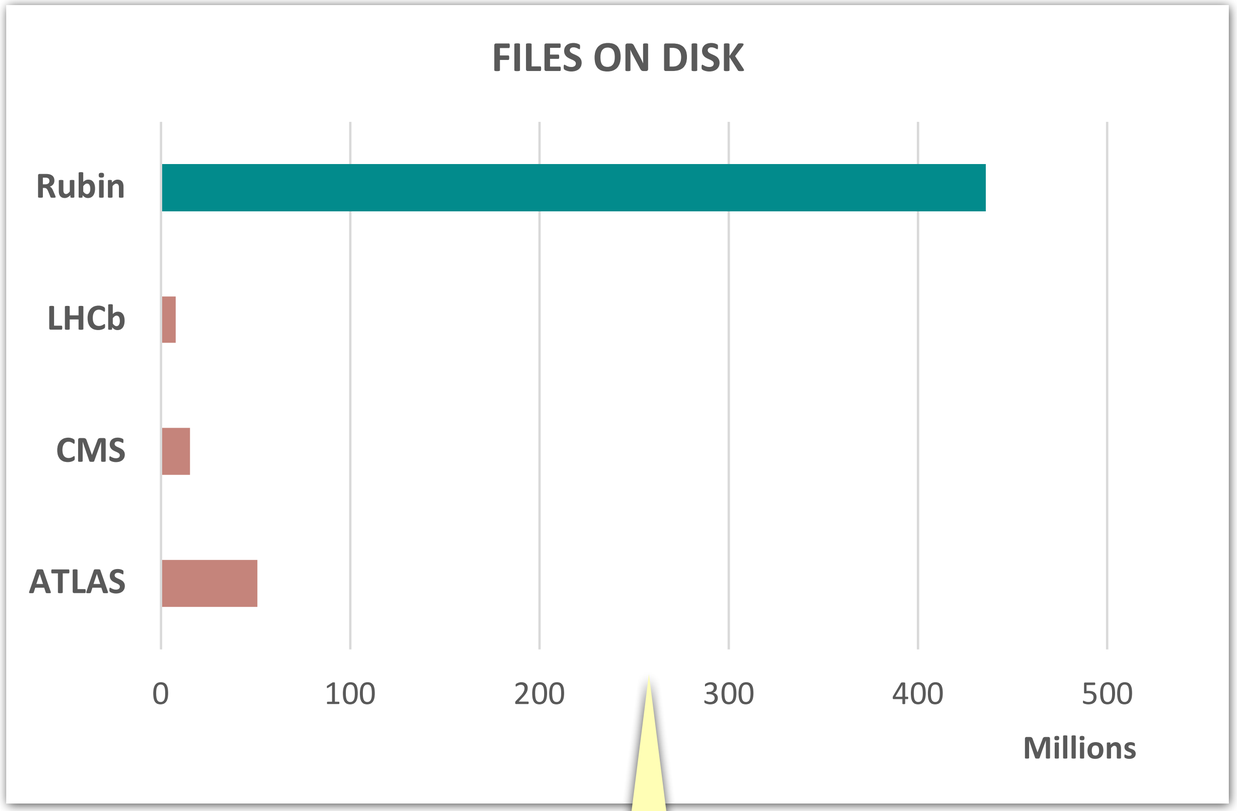
# CLIENT (CONT.)

```python
from lsst.resources import ResourcePath
file = ResourcePath("davs://dcache.example.org:1234/path/to/file.parquet")

# Read the whole file contents
data = file.read()

# Read a chunk of the file
with file as f:
    data = f.read(4096)
```

*Rubin requires to be able to make partial reading of files (e.g. zip, parquet, FITS)*

# USAGE PROFILE VS WLCG




*Usage profile of Rubin's dCache instance differs from WLCG's, in terms of both number of files and access patterns: many files of relatively small size O(100 MB)*

# UPDATES SINCE LAST WORKSHOP

# ARCHIVAL OF RAW IMAGES ON TAPE

- Daily CC-IN2P3 receives a copy of recently unembargoed raw images

  *for both custodial storage and later processing*

  *imported from the observatory's archive center located at SLAC*

- We set the QoS of precious files to "`disk+tape`" via dCache's frontend API

  *which triggers a copy to tape (HPSS)*

  *this has been working smoothly since activated 9 months ago*

  *currently performed once per day*

# Configuration of HAProxy

- Rubin software induces significant load on the doors, in particular because it issues many metadata requests

  *typical aggregated load on the doors is about 1.5k requests per second during periods of intense activity*

  *(to be compared to 200 requests per second served by the WLCG instance)*

- We were using DNS round robin for balancing load among the internal doors

  *updated every 5 minutes*

  *this configuration means that basically, at any given time, only one door was serving most of the requests*

- Introduced HAProxy to simultaneously use the capacity of all the available webdav doors

  *one HAProxy daemon colocated with each door proxies requests to any of the dCache doors using a round robin policy*

  *achieved better utilization of all the available internal webdav doors*

# CREATION OF A DEEP HIERARCHY

- Rubin uses deep file hierarchies

  *a typical file name is of the form*

  `/pnfs/in2p3.fr/lsst/a/b/c/d/e/f/g/h/i/j/k/l/m.fits`

- A previous version of Rubin software systematically verified the existence of all the ancestors before uploading a file

  *via a combination of several PROPFIND + MKCOL requests*

  *now modified to avoid that check: when a deep hierarchy is required, many MKCOL requests are replaced by exactly two requests, both handled by the door:*

  *PUT empty file with a temporary name + DELETE temporary file*

  *as a beneficial side effect, all the ancestor directories are automatically created if needed*

  *see issue #7945 for a discussion about a possible extension of MKCOL implementation to reduce this operation to a single request*

# ATOMICITY OF FILE CREATION

- To emulate atomicity of file creation, every file upload is implemented via two requests:
  1. *PUT the file content with a temporary name*
  2. *MOVE the temporary name to its final name*

  *in reality, these operations imply three requests: one PUT request to the door, one PUT request to the pool and one MOVE request served by the door*

- This helps in situations when, for some reason, the upload operation fails

  *e.g. the door created the (empty) file but uploading the file content to the pool fails*

  *Rubin software was confused by finding files with unexpected content*

# TLS CONNECTION REUSE

- Our previous configuration used secure HTTP for connections from the client to the internal door and plain HTTP for connections to the pools

  *previously we configured the client to **reuse the TLS connections** to the door*

  *the connection to the pool is systematically closed after GET and PUT to release the movers (except when doing GET with Range header)*

- We noticed that it was actually not possible for the client to reuse the TLS connection to the door after some kind of requests (e.g. `DELETE`, `GET`, `HEAD`, …)

  *see issue #8005 for details about our not-fully-understood observations*

- We modified our configuration of the doors to use TCP connections instead

  *using basic HTTP authentication*

  *the client requests the connection to the door to be closed after each request is served*

  *so far this has proven to be a good balance between the number of open connections to the doors and the load induced on the doors: we routinely observed 50k network connections open to a each door*

# TUNING THE DOORS FOR METADATA REQUESTS

- In our case, webdav doors redirect to the pools for PUT and GET requests and serve all metadata requests

  *PROPFIND, MKCOL, HEAD, MOVE, DELETE, …*

- We spent some time trying to understand the parameters to tune the doors

  *our goal was to serve as many metadata requests as possible, ideally keeping the p95 of the service time per request not higher than 50ms*

# TUNING THE DOORS FOR METADATA REQUESTS (CONT.)

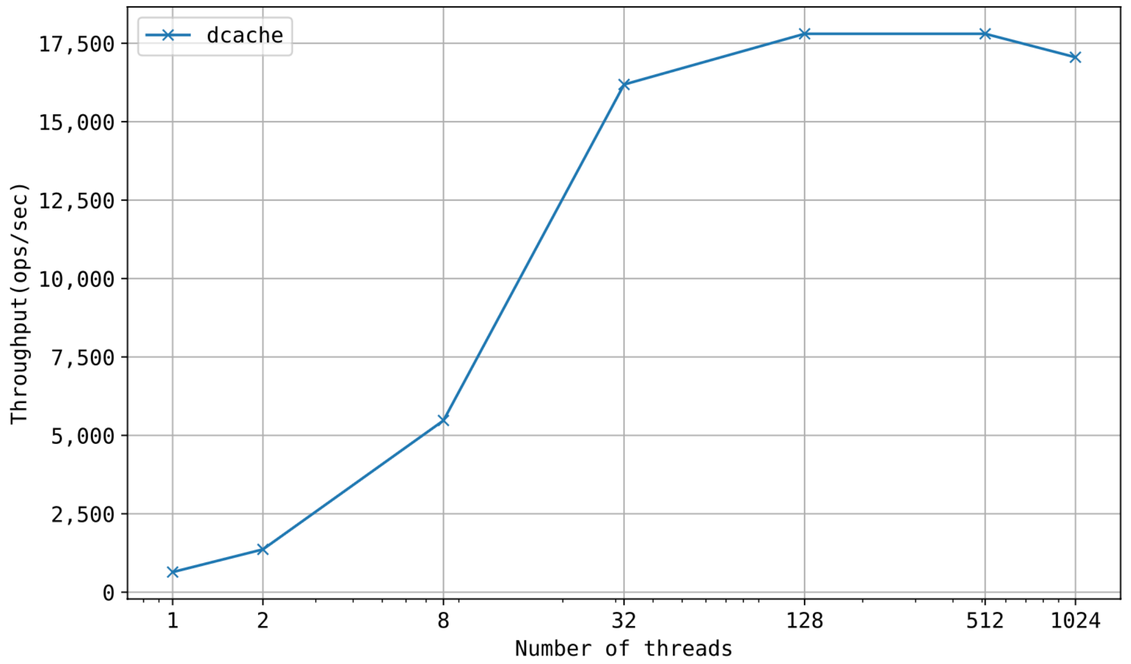



**Fig. 4** dCache aggregated file creation rate. The mdtest workload was creating zero-byte files to benchmark only the namespace latency and throughput

Computing and Software for Big Science (2025) 9:20
https://doi.org/10.1007/s41781-025-00152-5

BRIEF REPORT

## dCache: The Storage System of Choice for Data-Intensive Applications

Tigran Mkrtchyan[1] · Christopher Green[3] · Karen Hoyos[1] · Dmitry Litvintsev[3] · Paul Millar[1] · Lea Morschel[1] · Albert Rossi[3] · Marina Sahakyan[1] · Darren Starr[2,4]

Received: 15 April 2025 / Accepted: 12 November 2025


**Abstract**
The ever-increasing volumes of data produced by modern scientific facilities like EuXFEL and LHC put significant stress on data management infrastructure operated by laboratories and research centers. The challenges to be addressed span the entire data life cycle, from ingest and efficient data analysis to long-term preservation, typically involving large tape libraries. dCache, a storage system developed in collaboration between the Deutsches Elektronen-Synchrotron (DESY), Fermi National Accelerator Laboratory, and Nordic e-Infrastructure Collaboration (NeIC), is designed to manage a large number of disk servers and to facilitate transparent data migration to and from archival storage. Its multifaceted approach offers a unified method to support a variety of scientific use cases with the same storage infrastructure, including high-throughput data ingest, data sharing over wide area networks, efficient access from HPC clusters, and long-term data preservation on tertiary storage. Initially developed for high energy physics (HEP) experiments, dCache is now used by various scientific communities, including astrophysics, biomedical research, and life sciences, each having specific requirements. This paper presents architecture, deployment strategies, performance and scalability enhancements, and recent advancements in dCache addressing the needs of scientific communities. Finally, we touch on the development and release process, ensuring the software's high quality.

**Keywords** DCache · Storage · HSM · Data access

### Introduction

The dCache project started in 2000 as a collaboration between Deutsches Elektron-Synchrotron (DESY) and Fermi National Accelerator Laboratory. The mission then was to develop a common storage software for the two

Christopher Green, Karen Hoyos, Dmitry Litvintsev, Paul Millar, Lea Morschel, Albert Rossi, Marina Sahakyan and Darren Starr have contributed equally to this work.

✉ Tigran Mkrtchyan
tigran.mkrtchyan@desy.de

Christopher Green
greenc@fnal.gov

Karen Hoyos
karen.hoyos@desy.de

Dmitry Litvintsev
litvinse@fnal.gov

Paul Millar
paul.millar@desy.de

Lea Morschel
lea.morschel@desy.de

Albert Rossi
arossi@fnal.gov

Marina Sahakyan
marina.sahakyan@desy.de

Darren Starr
darren.starr@usit.uio.no

[1] Deutsches Elektronen-Synchrotron DESY, Notkestrasse 85, 22607 Hamburg, Germany

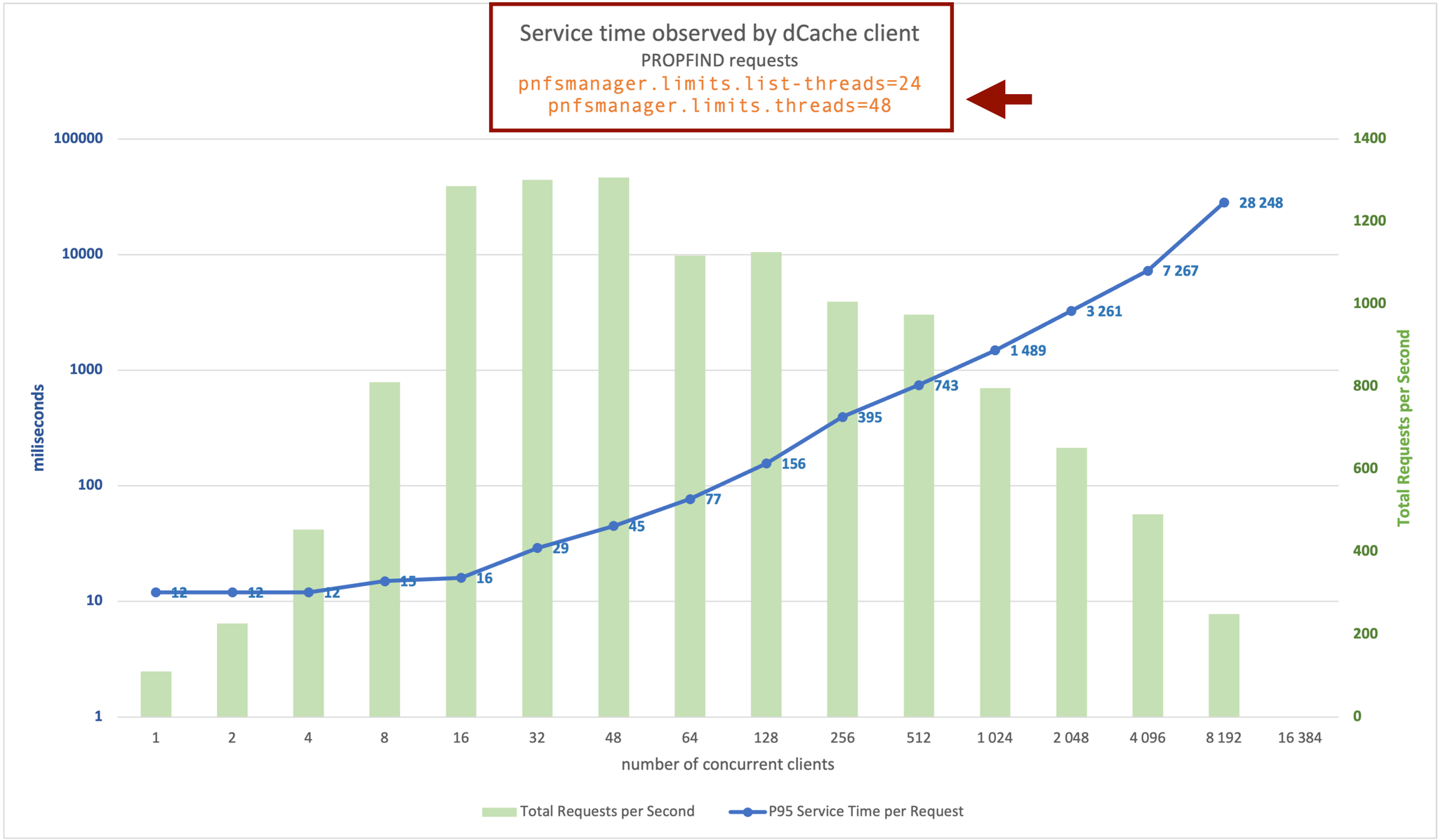
[2] The University of Oslo, Boks 1072, NO-0316 Blindern, Norway

[3] Fermi National Accelerator Laboratory (FNAL), PO Box 500, Batavia 60510-5011, IL, USA

[4] Nordic e-Infrastructure Collaboration (NeIC), c/o NordForsk, Stensberggata 25, NO-0170 Oslo, Norway

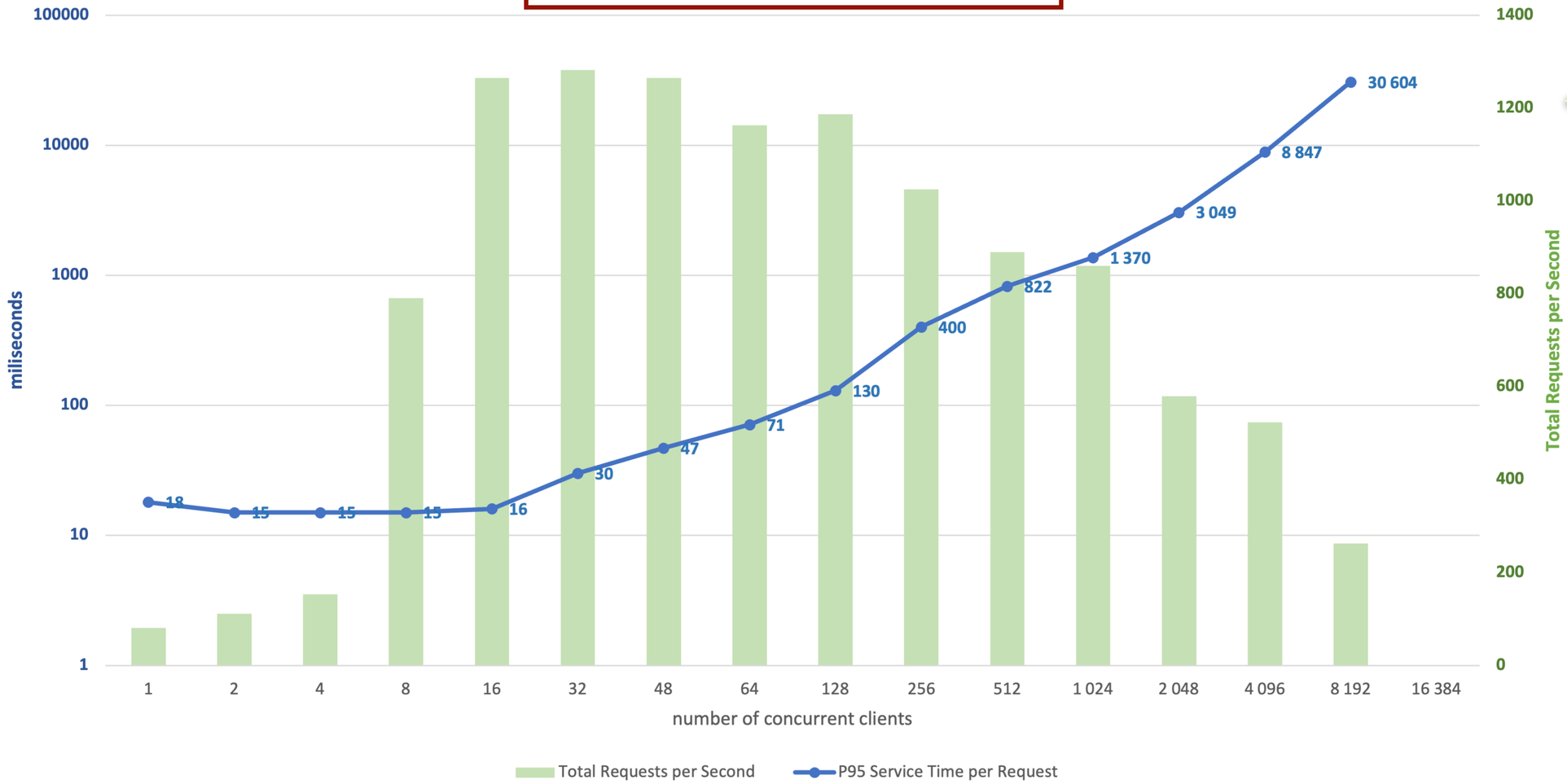
Published online: 25 November 2025

# TUNING THE DOORS FOR METADATA REQUESTS (CONT.)



# TUNING THE DOORS FOR METADATA REQUESTS (CONT.)

Service time observed by dCache client
PROPFIND requests
`webdav.limits.threads.max=2000`
`webdav.limits.acceptors=16`




*Should we instead add more webdav doors to increase our capacity to serve metadata requests?*

# Ongoing work

# Bulk deletion

- We must be able to regularly delete files in bulk
  - *at the latest  at the end of each annual reprocessing campaign but ideally even in the middle of an ongoing campaign*
  - *e.g. intermediate files generated for consumption by a downstream stage in the pipeline which already completed*
  - *we estimate the amount of files generated by each annual campaign to be stored in dCache to be of the order of **several hundred millions**, most of which we need to delete to make room for the next campaign*
- Naive attempt: webdav client recursively retrieves the content of each directory and emits one DELETE request per file to delete and per empty directory
  - *it works but takes time and feels like not that door-friendly*
  - *via a NFS mount removing several directories in parallel is faster to execute*
- We started investigating the possibilities of dCache's frontend API for bulk delete operations
  - *see issue #8062 for details on our attempts: some limit related to parameter* `bulk.limits.dir-list-semaphore` *reached and a single delete request involving too many files and directories makes the request to stall*
  - *submitting several requests with the "right number" of items to delete works, though*
- Does any other site have experience or best practice to share?

# Conclusions

# Conclusions

- dCache is a key tool in the infrastructure that CC-IN2P3 deployed for processing Rubin image data

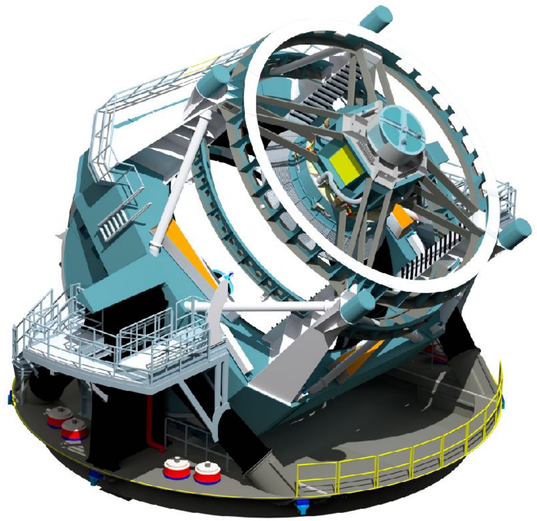
- It has been working very well for us and we intend to continue using for the years to come

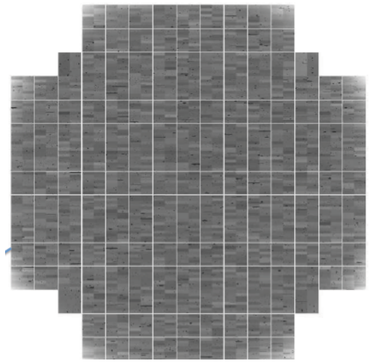
  *currently working on fine tuning our system to respond to the specific needs of Rubin and adjusting the behavior of Rubin software for it to be as friendly as possible with its storage endpoints*

- Many thanks to the dCache team and to the community of dCache users

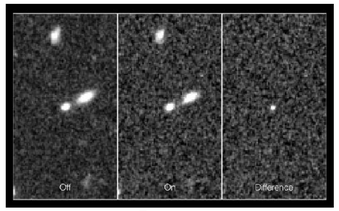
# BACKUP SLIDES

# RUBIN OBSERVATORY LEGACY SURVEY OF SPACE AND TIME

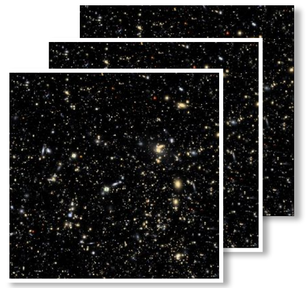




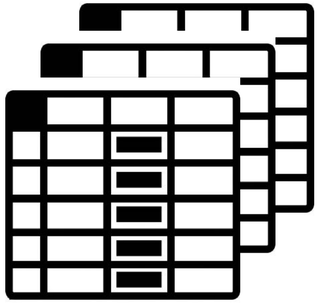
raw images




alerts



science-ready images



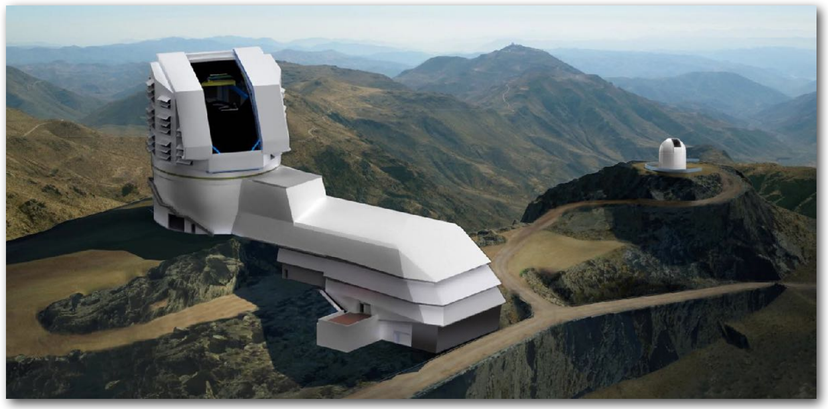
astronomical catalog

science collaborations

LSST aims to deliver a catalog of 20 billion galaxies and 17 billion stars with their associated physical properties

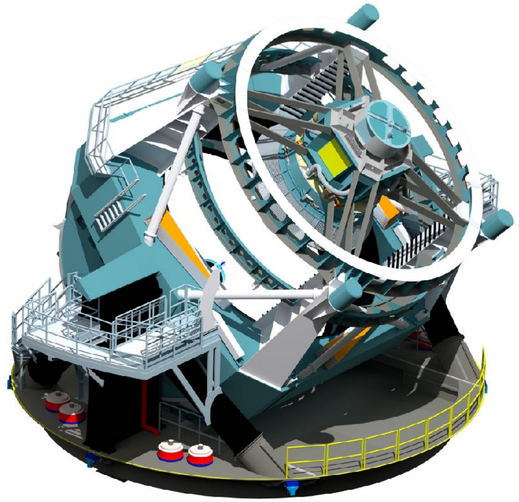
# LSST OVERVIEW

## OBSERVATORY



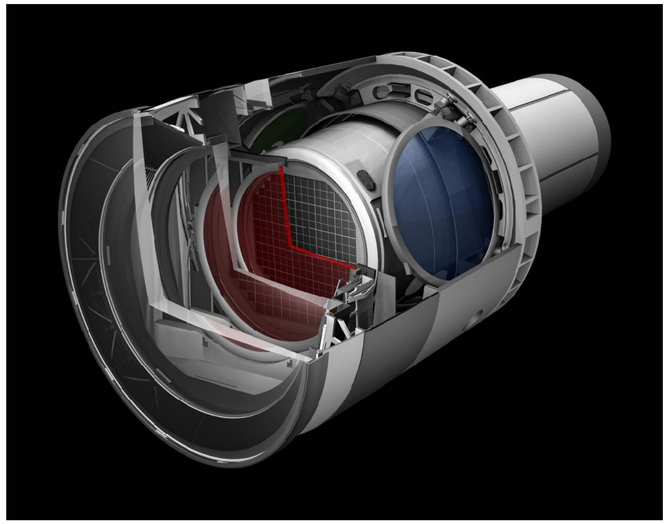
southern hemisphere | 2647m a.s.l. | stable air | clear sky | dark nights | good infrastructure

## TELESCOPE



main mirror ⌀ 8.4 m (effective 6.4 m) | large aperture: f/1.234 | wide field of view | 350 ton | compact | to be repositioned about 3M times over 10 years of operations

## CAMERA



3.2 G pixels | ⌀ 1.65 m | 3.7 m long | 3 ton | 3 lenses | 3.5° field of view | 9.6 deg² | 6 filters *ugrizy* | 320-1050 nm

Source: LSST: from Science Drivers to Reference Design and Anticipated Data Products


30°14'40.7"S 70°44'57.9"W

Source: Rubin Observatory

Source: Rubin Observatory

# Raw data

*6.4 GB per exposure (compressed)*

*2000 science + 500 calibration images per night*

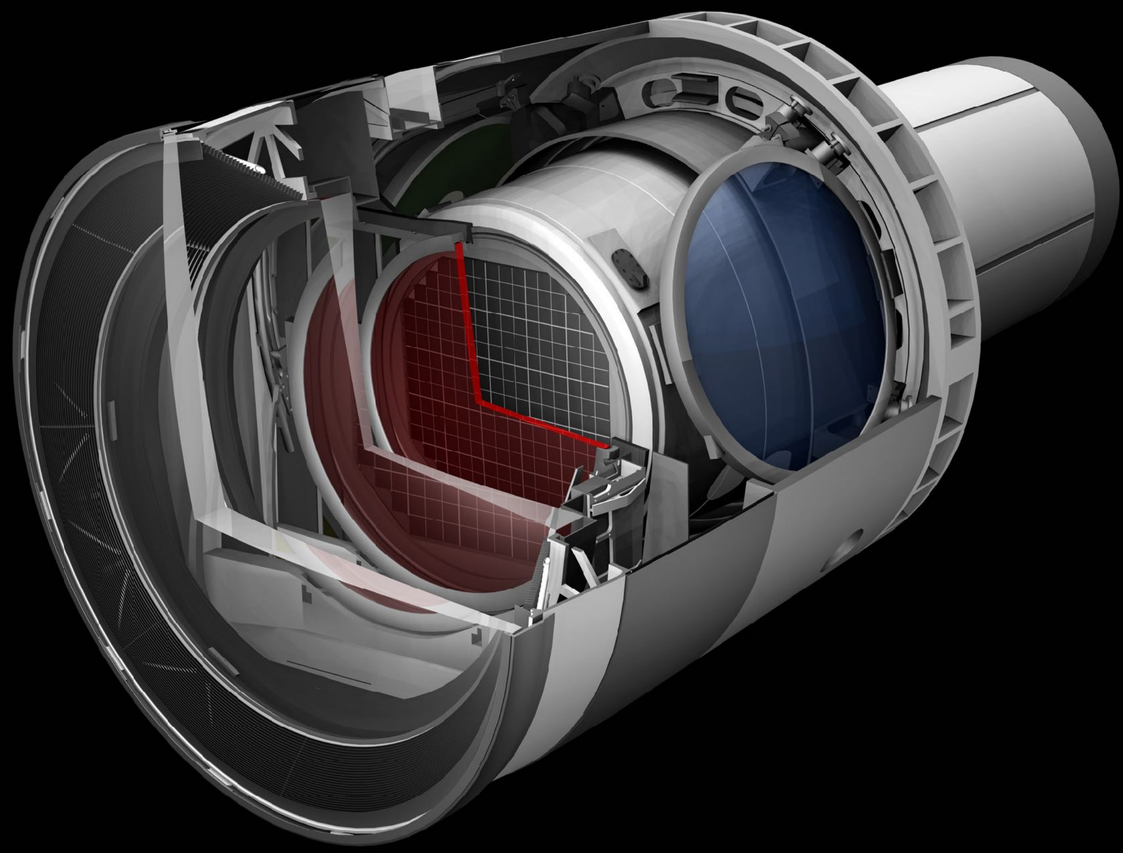
*16 TB per night*

*300 nights per year, ~5 PB per year*

# Aggregated data over 10 years of operations*, including derived data

*image collection: ~6M exposures*

*final catalog database: 15 PB*



* source: LSST Key Numbers

Credit: Rubin Obs/NSF/AURA/H. Stockebrand

## Cloud

EPO Data Center

## Dedicated Long Haul Networks

Two redundant 100 Gb/s links from Santiago to Florida (existing fiber)
Additional 100 Gb/s link (spectrum on new fiber) from Santiago-Florida (Chile and US national links not shown)

## UK Data Facility
## IRIS Network, UK

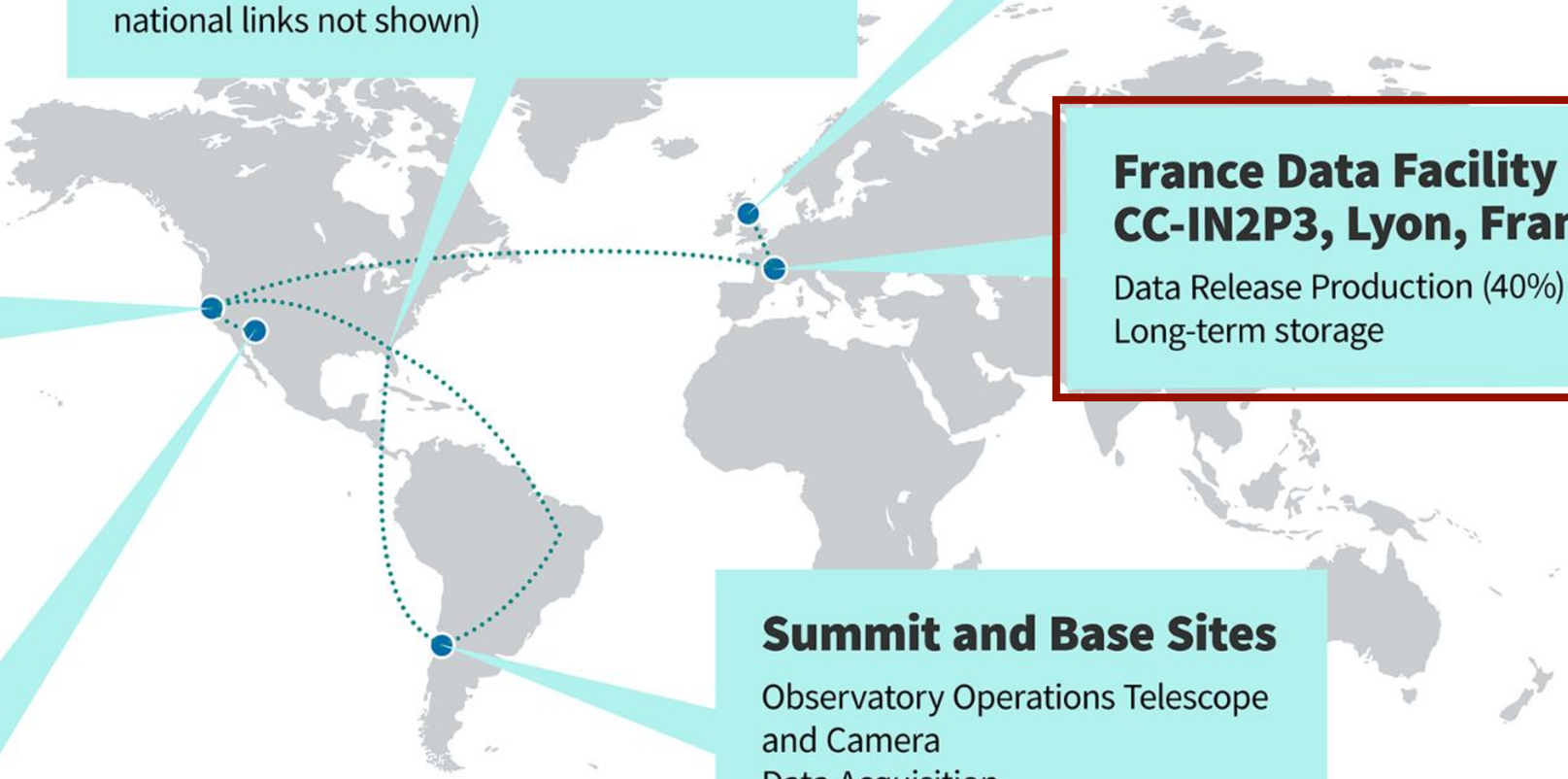
Data Release Production (25%)

## US Data Facility
## SLAC, California, USA

Archive Center
Alert Production
Data Release Production (35%)
Calibration Products Production
Long-term storage
Data Access Center
Data Access and User Services

## France Data Facility
## CC-IN2P3, Lyon, France

Data Release Production (40%)
Long-term storage



## HQ Site
## AURA, Tucson, USA

Observatory Management
Data Production
System Performance
Education and Public Outreach

## Summit and Base Sites

Observatory Operations Telescope and Camera
Data Acquisition
Long-term storage
Chilean Data Access Center

VERA C. RUBIN OBSERVATORY