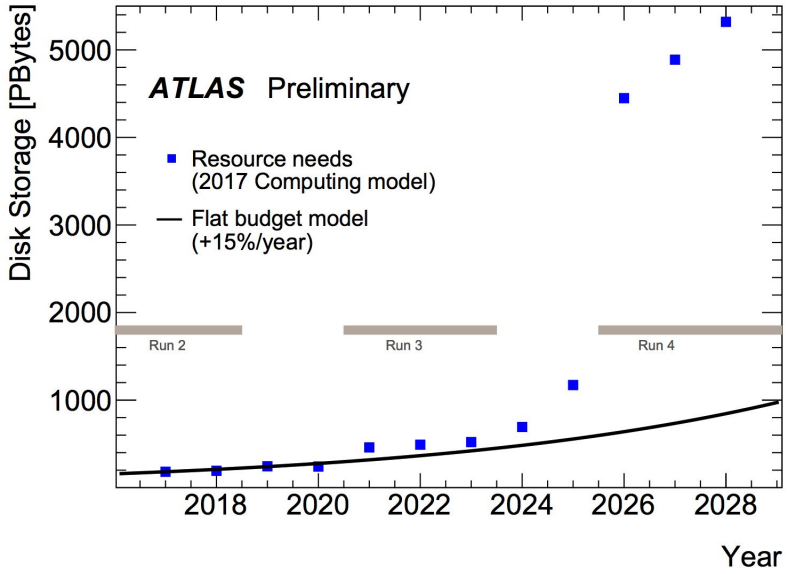
Data Lakes

Rob Gardner and Brian Bockelman
(With slides and ideas from Simone, Ilija, Benedikt, and others)

US LHC Joint Session @ University of Utah
March 19, 2018

Most significant challenge for HL-LHC

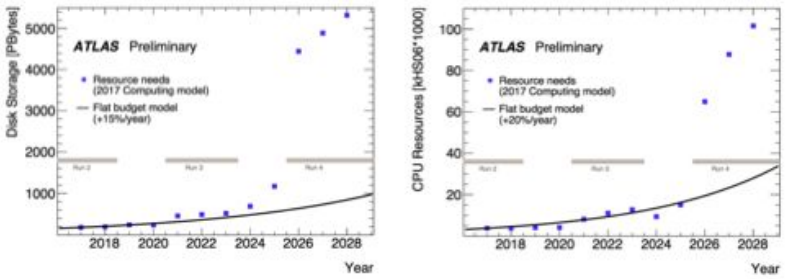
- Storage the main driver of cost in the Facility (today)
- Yet current extrapolation assuming flat budgets & a little for technology improvements fall way short in Run 4
- How can we reduce the needed disk capacity?



Storage Is (Operationally) Expensive!

As noted by Simone earlier today, disk drives operational concerns as well!

WLCG needs manage and contain the cost of HL-LHC computing



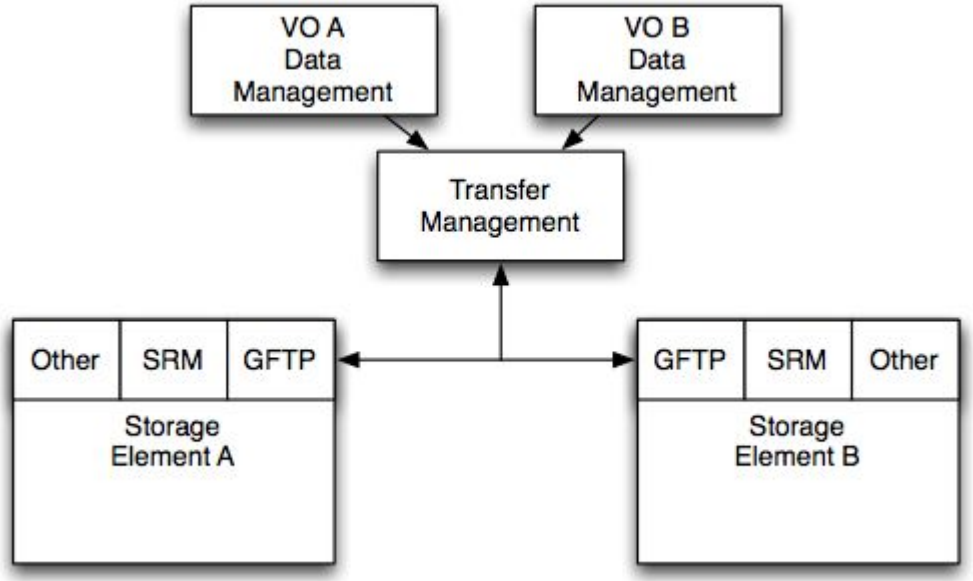
The cost comes both in terms of hardware (left) and operations

Facts:

- Storage today is the major hardware cost in most countries. Disk costs 4x more than tape per TB
- Storage is also the main operational cost at sites according to a recent (2015) survey

Storage Elements are Complex

- In this model, we have multiple services exposing a POSIX-like filesystem.
  - Each storage element acts independently.
- A higher-level transfer management layer moves files between SEs.
- VOs develop their own data management layer on top of that. Not quite so simple...
- Driving model for 15+ years!



Cost reduction factors to consider

- LHC data is mostly ‘cold’, store the majority on cheap cold storage, including tape
  - recover hardware costs
- **Reduce the number** of VO-specific data management tools
- **Reduce the number** of storage elements in the grid
  - recover operational (labor) costs
- Choose **organization & formats for the task**
  - Strategically placed datasets in regions
  - Compressed files, optimized for storage and filtering
  - Granulated & inflated for client access & processing

Data Lake – Concept

- Instead of one-SE-per-site, have a single logical SE that encompasses a significant amount of high-performance storage.
  - Sites outside a data lake have no persistent experiment storage.
  - E.g., all is cached or streamed.
- Potentially, this reduces the SE counts, allows more aggressive use of tape, and allows experimentation in the data format/organization.

Here comes the data lake

- For the purposes of discussions, we'll ignore the industry jargon and focus on large aggregates of storage in a region.
- We need to develop the appropriate cost model to design the number and configuration. Taps into an existing WLCG effort.
- Lots of management questions for a lakes-based data model.
  - Where does user data live?
  - How is tape storage incorporated and managed? In-lake or out-of-lake?
  - What would be differences in models between ATLAS and CMS?

Data Lake Implementations

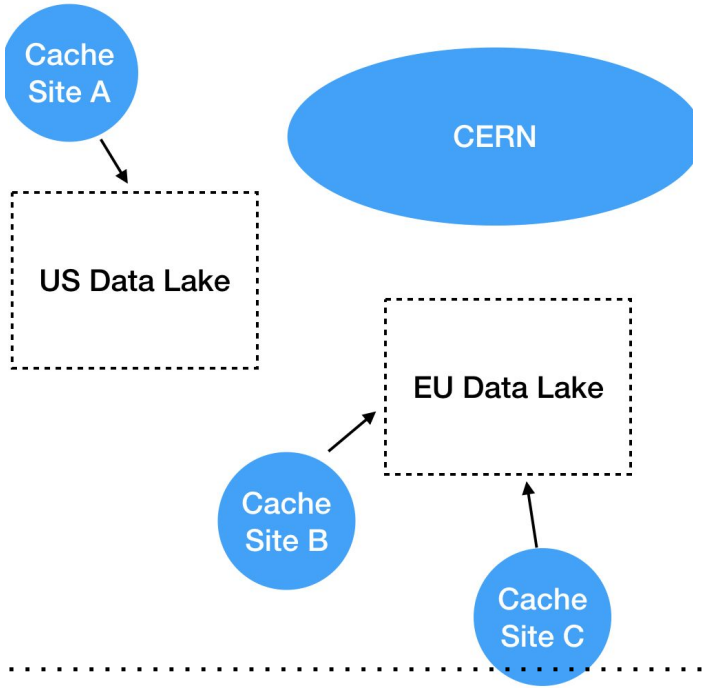
- There's a few ways to approach data lakes:
  - Take an existing high-performance SE (such as EOS or dCache) and make it work well over multiple sites.
  - Take the concept of a data federation and add functionality to make it more like a distributed SE.
  - Explore new conceptual models for reducing data replication levels.
- Do you try to save operational effort only? Or also disk space?

Evolving the LHC computing model

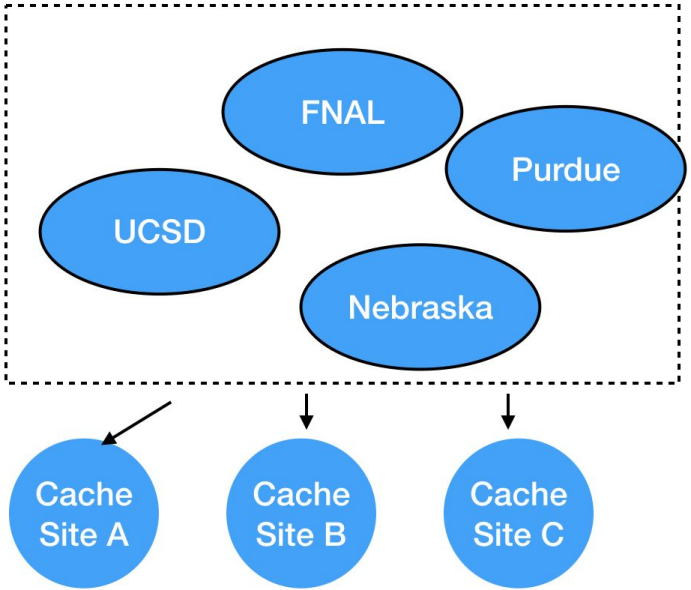
- Implies significant change to the LHC computing model
- A separation of major functions
  - A data plane (including archiving?).
  - A processing plane.
  - And a delivery & management network coupling them.
- Looks very different than any hierarchical (“tiered”) model!

Potential Data Lake Architectures

Global Architecture

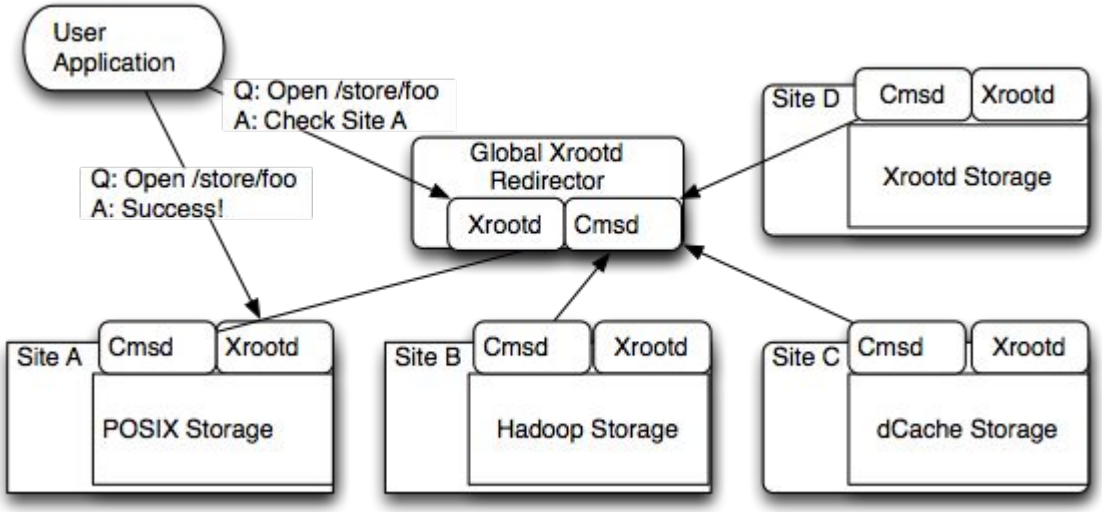


Zoom-in of a data lake.



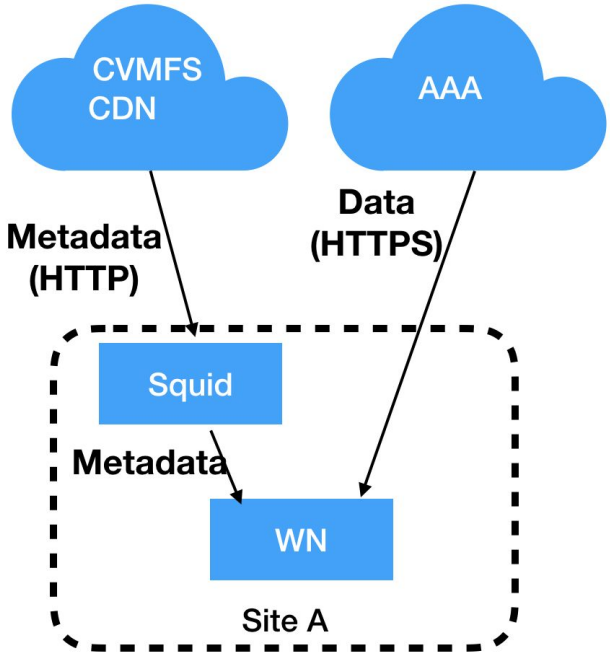
Data Lakes – Growing from Federations

- Federations provide data access.
- However, there are several things that they don't provide:
  - Namespaces. No source of authority on what should be in the federation or its contents.
  - Data movement or replication.
- These are often added by combining the federation with other technologies for data management.



Data Lake Prototype – BigCVMFS

- Starting in CVMFS 2.3, we added the ability to:
  - Have the CVMFS / FUSE client download data from files not in the existing CDN. (e.g., use AAA).
  - Utilize a separate authorization callout to retrieve credentials from the user environment. In this case, we get the GSI proxy from the user.
  - Enforce ACLs at the repository level.
- CVMFS provides an extraordinarily scalable namespace. Solves AAA problems:
  - Record of what is supposed to be accessible via AAA!
  - CVMFS client can be updated independently of CMSSW version.
- Currently publishing UCSD and Nebraska contents.



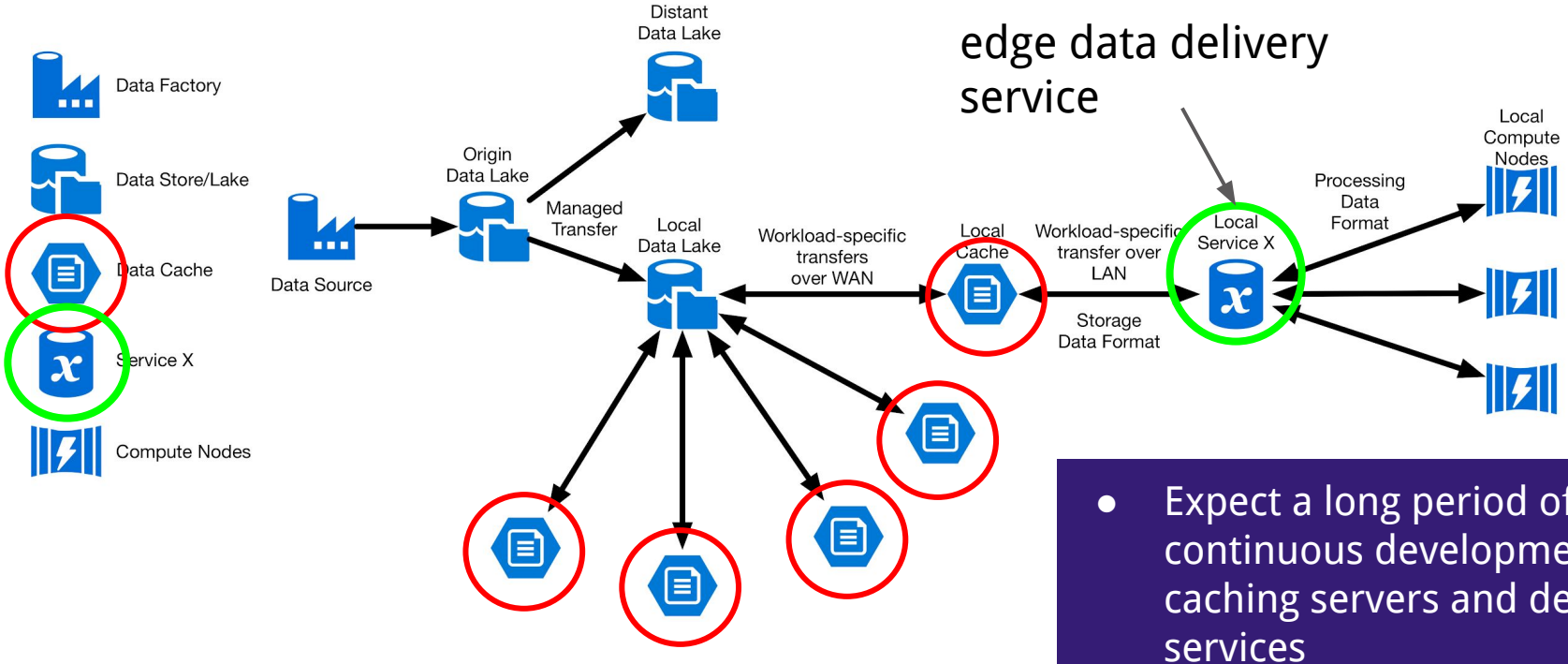
Data Lakes – New Conceptual Models

- We can also look at the data lake as a way to offer higher-level services to the LHC.
- Proposal: Rather than outfitting processing sites with fat co-located storage elements, outfit them with an **event** delivery service
  - "Service X", a new edge-service.
- Abstract away details of event format.
  - View data lake as datasets / events / branches, rather than files an byte streams
- Hide latency of access

Service X

- Translate data between the storage optimized and processing optimized formats
- Stream data to compute nodes through the LAN
- Optimizes network use by reducing number and size of needed WAN transfers
- For output, aggregate (merge) data products and reinsert into a delivery network

And delivery to a Service X?



- Expect a long period of continuous development of caching servers and delivery services
- Roll out updates centrally
- Configure & control centrally

Advantages

- Centralization of storage location and support
- Better planning for infrastructure investments (e.g. fast networks between a Data Lake and its caches or between Data Lakes)
- Separation of data storage format from data processing format, opens the way for optimizations in data compression, evolution of the data storage format, etc.
    a. E.g., popular data can be recompressed for faster reads.

Advantages, cont.

- Optimization of network usage by reducing unnecessary data transfers.
  a. Does not always need to be a “pull”, but organized placement.
- Lower startup cost and effort for adding resources (i.e. local caches and Service X instances).
- Cache deployments are significantly smaller than current Tier2 storage deployments

Prototyping lakes infrastructure

- New services will be needed in various places, e.g. in the edge networks of processing centers
- The “product” is the caching and delivery network, which is a **distributed** set of services of various types (e.g. Xcache, Service X, new-thing, …)
- We need a platform – see SLATE slides at the end (or sites jamboree talk two weeks ago)

US Computing Facilities as R&D Platform

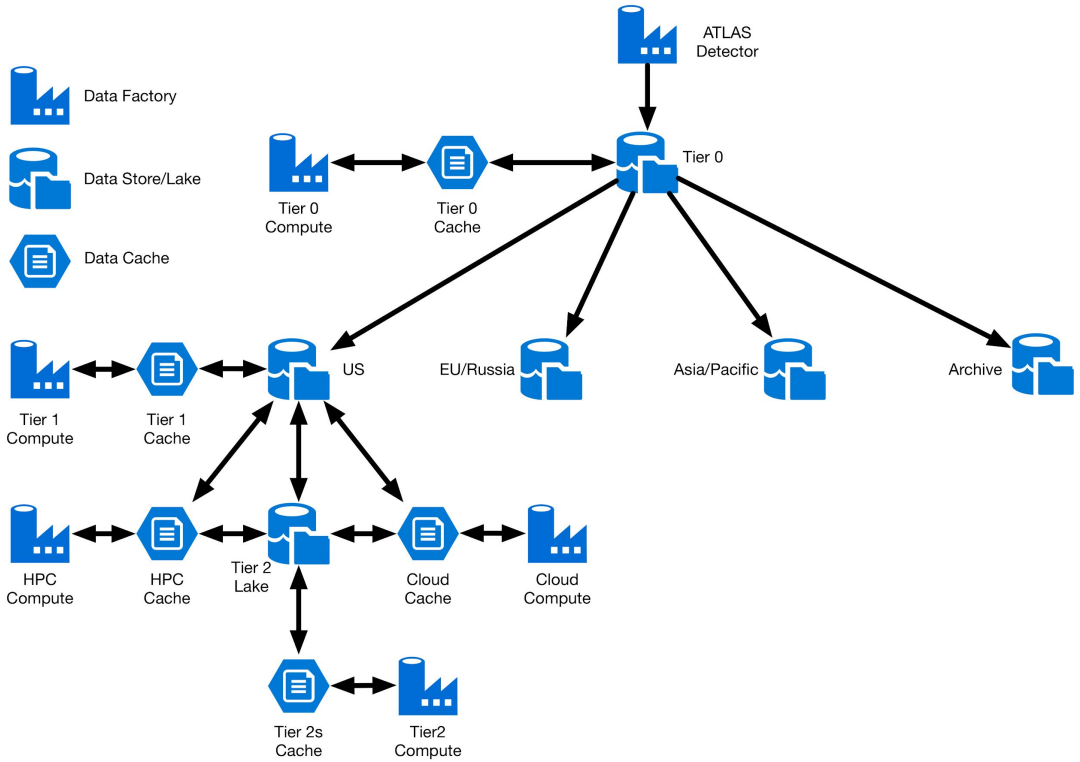
- Wider-scale usage of Xcache (and containerization). **Continued Joint Project**.
- Deployment of edge services at US Tier2s.
  - Deployment of a SLATE infrastructure for DevOps-friendly development
- Service X prototype
- Prototyping improvements to data federations.

extra slides

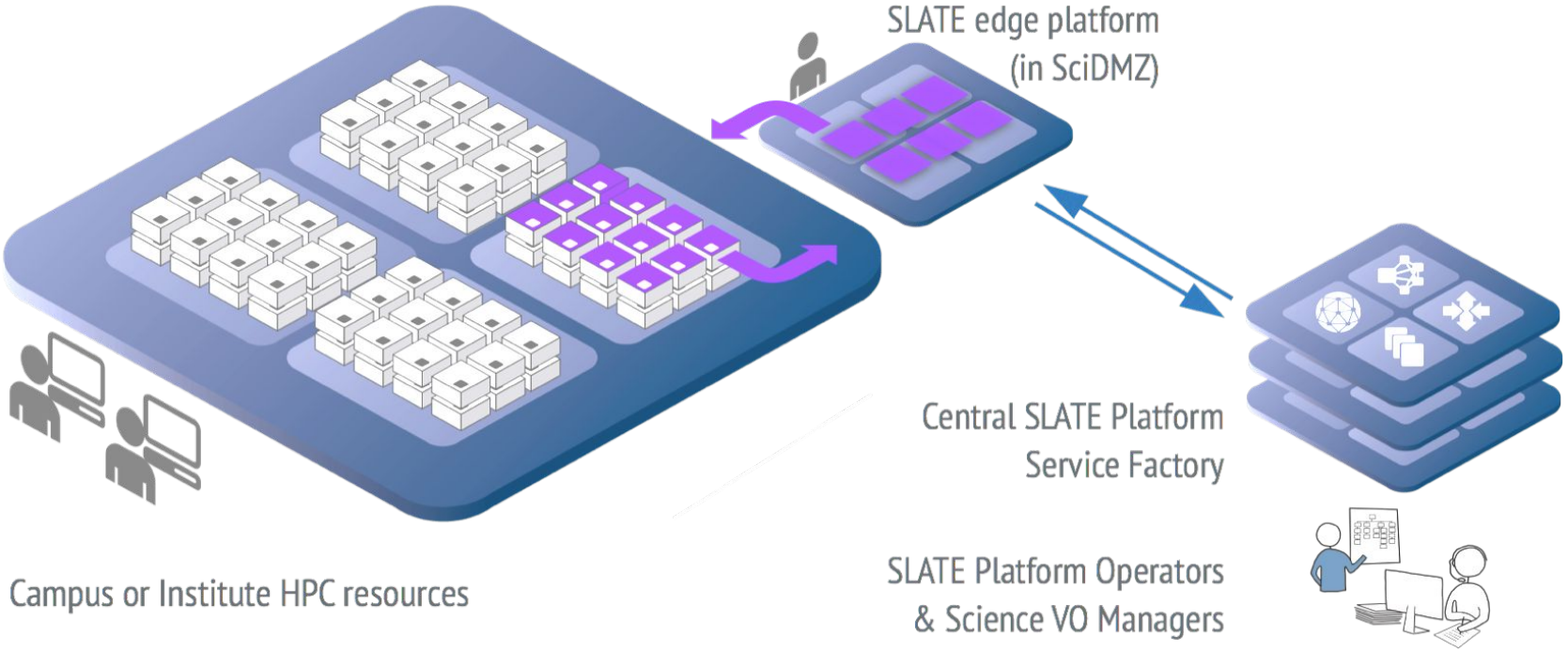
http://bit.ly/atlas-lakes

Regional lakes, ”zones”, etc

- data lakes will span geographic regions
- interoperation protocols between regions TBD



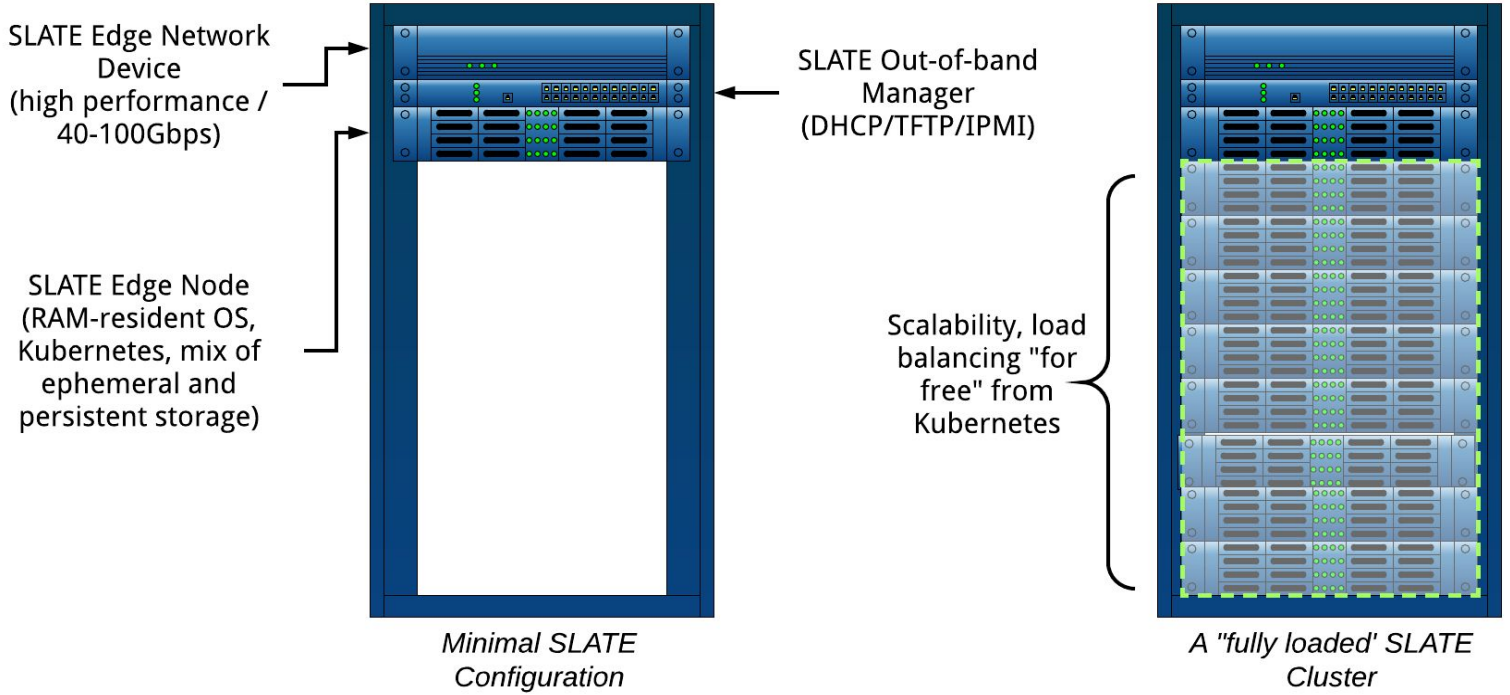
Deployment with SLATE



Services **L**ayer **A**t **T**he **E**dge

- A ubiquitous *underlayment* -- the missing shim
  - A generic cyberinfrastructure substrate optimized for hosting edge services
  - Programmable
  - Easy & natural for HPC and IT professionals
  - Tool for creating "hybrid" platforms
- DevOps friendly
  - For both platform and science gateway developers
  - quick patches, release iterations, fast track new capabilities
  - reduced operations burden for site administrators

SLATE Edge Clusters for US ATLAS



”Mock up” Examples

- `slate app install --cluster=uchicago-mwt2,umich-arc harvester:latest`
- `slate app install --cluster=alcf-edge htcondorce`
- `slate app install --cluster=mycluster arccache`
- `slate app status [appname]`
- `slate app status [appinstancename]`
- `slate app delete xcache`
- `slate app delete xcache --instance xcache-ivukotic-mwt2`
- `slate app delete xcache --cluster='uchicago-*'`
- `slate app delete --cluster=uchicago-rcc --org=ATLAS`

Containerizing XCache for **SLATE**

Ilija Vukotic

- Already several Docker containers exist.
- There is an autobuilt one in slateci/xcache.
- A simple deployment (single server) tested in three different Kubernetes clusters (CERN, MWT2, Google).
- Need a robot certificate before scale/reliability testing.
- Next steps:
  - Rucio fix for correct path construction.
  - XCache monitoring (reporting based on cache cinfo data)
  - Small scale testing
  - More complex deployment - cluster with autoscaling.