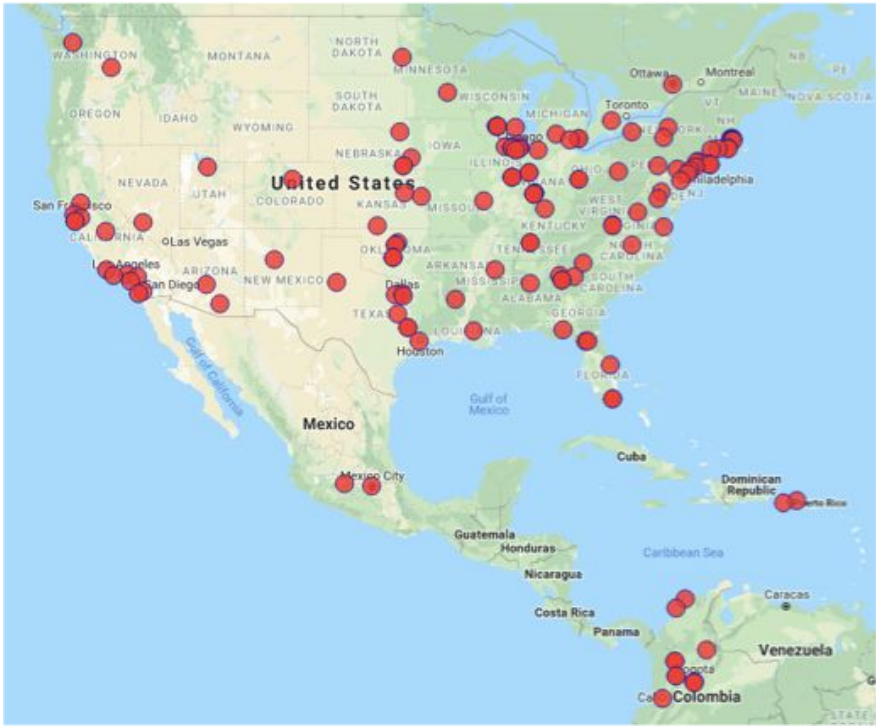
Creating a content delivery network for general science on the backbone of the Internet using XCache(s).

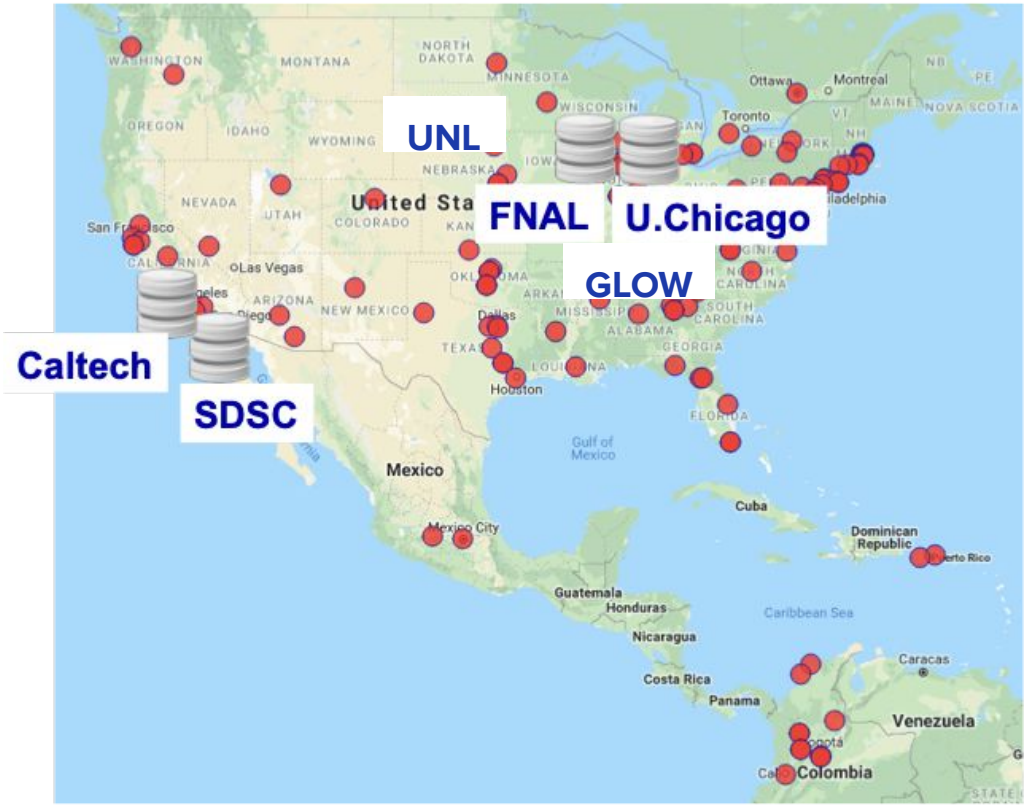
Edgar Fajardo
Presented by: Igor Sfiligoi
In Collaboration with: Brian Lin, John Hicks, Marian Zvada, Derek Weitzel, Mat Selmeci, Pascal Paschos

Introduction to Open Science Grid (OSG)

- OSG aggregates compute resources from over 100 campuses both nationally and internationally
- OSG also serves almost 40 different user communities, each with its own set of data origins
- With a handful having really large input datasets
- Networking essential to deliver data from origins to compute endpoints



OSG Data Origins



- OSG supports different scientific communities all across the science spectrum.
- These communities happen to have a “Golden copy” of their data (data origin) all around the country

FNAL: Fermilab based HEP Experiments

U.Chicago: General OSG Community

Caltech: Public LIGO Data Releases

SDSC: Simons Foundation

UNL: LIGO Data Release

Implications of this model

- Data is moved from its origin to the jobs using the network.
- If a data file is reused by several jobs the same file travels the network several times. For example:
  - LIGO - time shifter analysis
  - Biology-related communities – DNA matching
  - Any kind of parameter estimation over the same data set

Hence: Caching in the network

Benefits of Data caching in the network

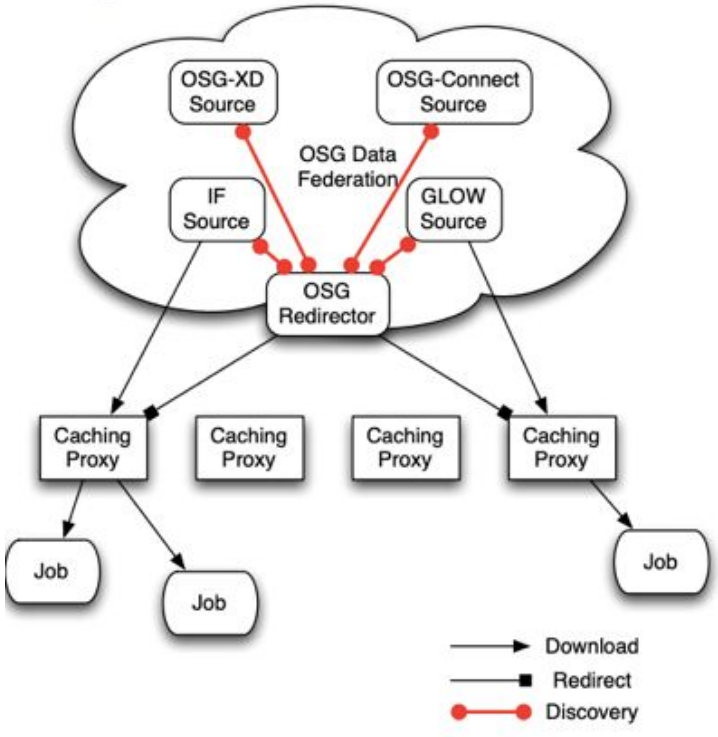
- Reduce origin to backbone data transfers:
  - Data only travels once from the origin to the cache
  - Reduces stress on data origins
  - Increases redundancy
- Increase CPU efficiency for latency sensitive applications
  - Less time wasted waiting for data
- Benefits both types of applications
  - Lower RTT greatly benefits latency sensitive applications
  - Reduced data origin server congestion allows for higher endpoint bandwidth

OSG Caching Solution: StashCache

Introduction to Stashcache

- Caching infrastructure based on SLAC XRootD server & XRootD protocol.
- Cache servers are placed at several strategic cache locations across the OSG.
- Jobs utilize GeoIP to determine the nearest cache
- Job talks to the cache using HTTP(S) via CVMFS

Image taken from Brian's slides



Powered by:

XRootD

Implications

- An organization can join the federation with their own “data origin” and their own partition of the global namespace. Like /gwosgc, /user, /pnfs/fnal/…/dune
- A cache owner can decide on caching policies for different parts of the namespace.
- This allows the owner to selectively serve only a subset of the community that uses the federation.

Caches in the backbone

- A joint project between Internet2 and OSG to place several caches on the backbone of the Internet2
- Originally three caches were deployed in the backbone: KC, Chicago and Manhattan.
- Since OSG is moving to a DevOps model all the new caches were deployed using Kubernetes for maximum flexibility of deployment (i.e one day these are caches tomorrow someone can deploy another container for bandwidth testing).
- This gave rise in 2018 to the following caches topology.

INTERNET2

Current stashcache infrastructure (US)

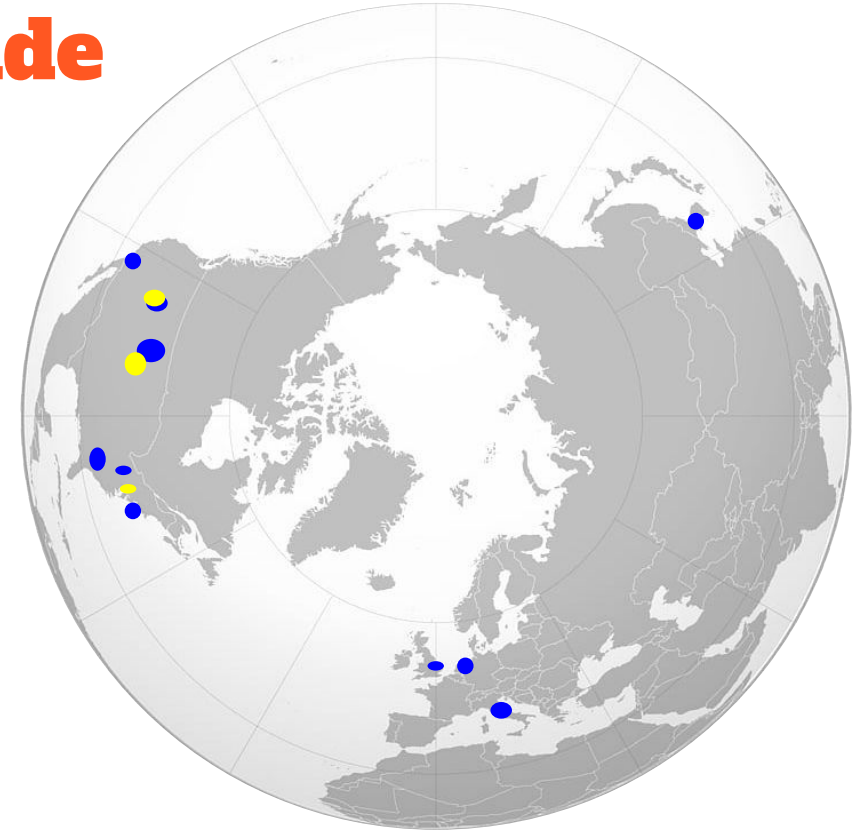
But then three become one

LIGO, Virgo, and Kagra are now in a collaboration-of-collaborations, and are working towards a data analysis common computing infrastructure called IGWN (for the International Gravitational-Wave Observatory Network)

Mr Worldwide

Cache at institution

Cache in the backbone



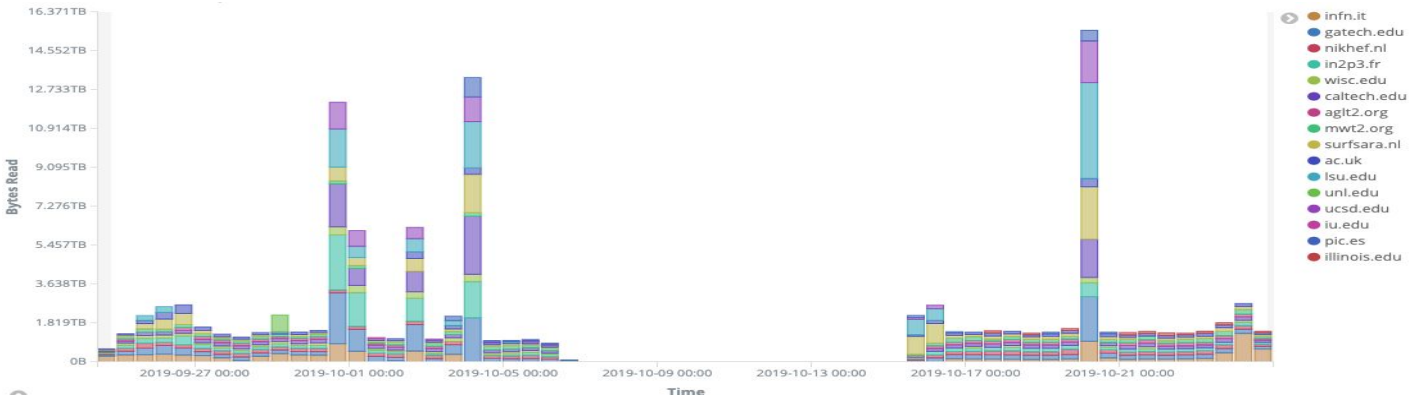
Given the worldwide location of the IGWN computing resources a need rise to expand beyond the US. This lead to a worldwide network of caches.

How does this perform?

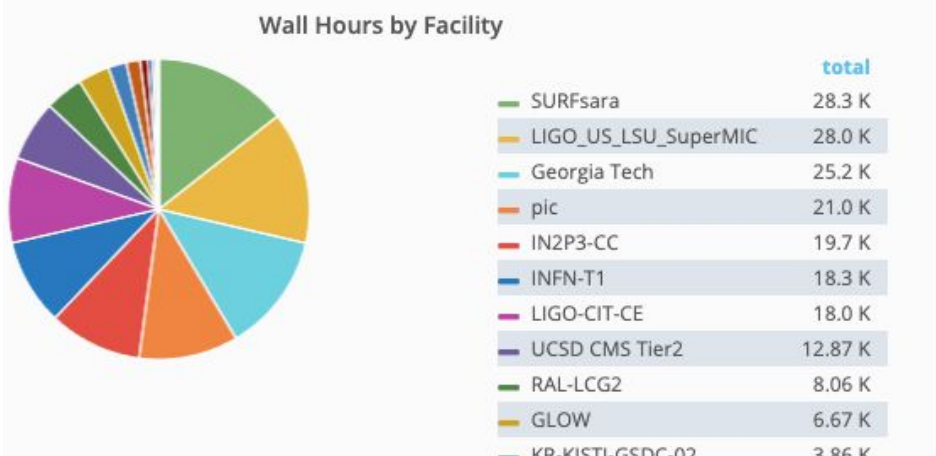
Right now the monitoring is available from three sources:

1. ElasticSearch Kibana:
    a. Aggregates of data delivered by namespace, location, and cache
    b. Can calculate working set and data reuse (then implicitly cache hit rate)
2. GRACC:
    a. CPU used at every site and by every scientific community (VO)
3. Our own testing
    a. Test same data access pattern from each site that support IGWN:
        i. Full file copy
        ii. Random read

Results: Kibana + ES + GRACC



This is an example of same 30 days of data delivery for IGWN



How did we do in the last six months?

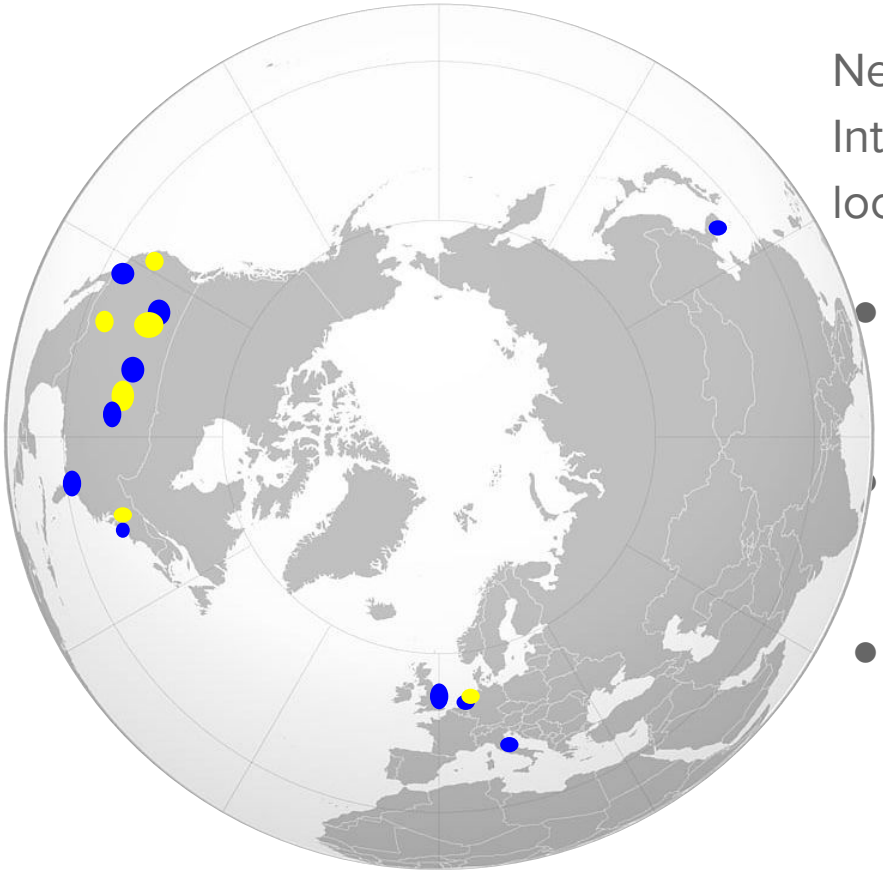
/pnfs/fnal.gov/usr/dune		14.189GB	x30k	1.843PB
/user/ligo		18.172TB	x30	595.939TB
/user/dteam	WLCG data transfer team tests	4.603TB		497.752TB
/pnfs/fnal.gov/usr/minerva		435.27GB	x700	304.667TB
/pnfs/fnal.gov/usr/des		272.478GB		127.374TB
/gwdata/O2		7.157TB	x13	96.321TB
/user/dweitzel	functionality tests	13.77GB		47.891TB
/pnfs/fnal.gov/usr/nova		86.335GB	x700	19.869TB
unknown directory	mostly CMS usage	4.831TB		8.552TB

Depending on community, files were read 10-30000 times during last six months

How the network of caches will look like

Cache at institution

Cache in the backbone



New caches in the Internet2 backbone located at:

- Sunnyvale
  - LBNL, NERSC, etc
- Houston
  - LSU (SuperMicro & QB2E
- Amsterdam
  - First one in the backbone in Europe

Conclusions

- We built a data delivery network for general science purposes that profits from in the network caching
- Kubernetes was the building block to deploy this worldwide agile infrastructure.
- There is more work to do in monitoring file hit cache rates in order to measure the exact impact of this infrastructure across different ranges of science.
-

Acknowledgments

- The authors would like to thank the funding agencies for this work, in particular the following grants:
  - PRP: NSF OAC-1541349
  - TNRP: NSF OAC-1826967
  - CESER: NSF OAC-1841530
  - OSG: NSF MPS-1148698
- Also we would like to thank Internet2 for providing the infrastructure on the backbone and their whole collaboration on this project.
- The different system administrators that have agreed to deploy hardware at the computing centers:
  - GaTech, UNL, UCSD, CNAF, U of Amsterdam, UChicago, Syracuse.
- To the PRP admins who let use use the kubernetes federation to deploy the caches.
- The Xrootd Development team for the continuous improvement of the XCache.
- OSG Software team for the XCache packaging and StashCache containers.