OSiRIS: One Year Update

Distributed Ceph and Software Defined Networking for Multi-Institutional Research

Shawn McKee
University of Michigan / ARC-TS
Michigan Institute for Computational Discovery and Engineering
October 19, 2016

- Project and participants overview
- Structural overview and site details
- Orchestration, monitoring and visualization
- Networking, NMAL, SDN
- Latency and Ceph - our experiments
- AAA infrastructure
- First science domains: ATLAS and Physical Ocean Modeling

OSiRIS Summary

OSiRIS (Open Storage Research InfraStructure) is one of 4 NSF CC*DNI DIBBs projects funded in 2015. OSiRIS is prototyping and evaluating a software-defined storage infrastructure, initially for our primary Michigan research universities, designed to support many science domains.

Our goal is to provide transparent, high-performance access to the same storage infrastructure from well-connected locations on any of our campuses.

By providing a single data infrastructure that supports computational access "in-place" we can meet many of the data-intensive and collaboration challenges faced by our research communities and enable them to easily undertake research collaborations beyond the border of their own universities.

OSiRIS Features

Scientists get customized, optimized data interfaces for their multi-institutional data needs

Network topology and perfSONAR-based monitoring components ensure the distributed system can optimize its use of the network for performance and resiliency

Ceph provides seamless rebalancing and expansion of the storage

A single, scalable infrastructure is much easier to build and maintain

- Allows universities to reduce cost via economies-of–scale while better meeting the research needs of their campus

Eliminates isolated science data silos on campus:

- Data sharing, archiving, security and life-cycle management are feasible to implement and maintain with a single distributed service.
- Configuration for each research domain can be optimized for performance and resiliency.

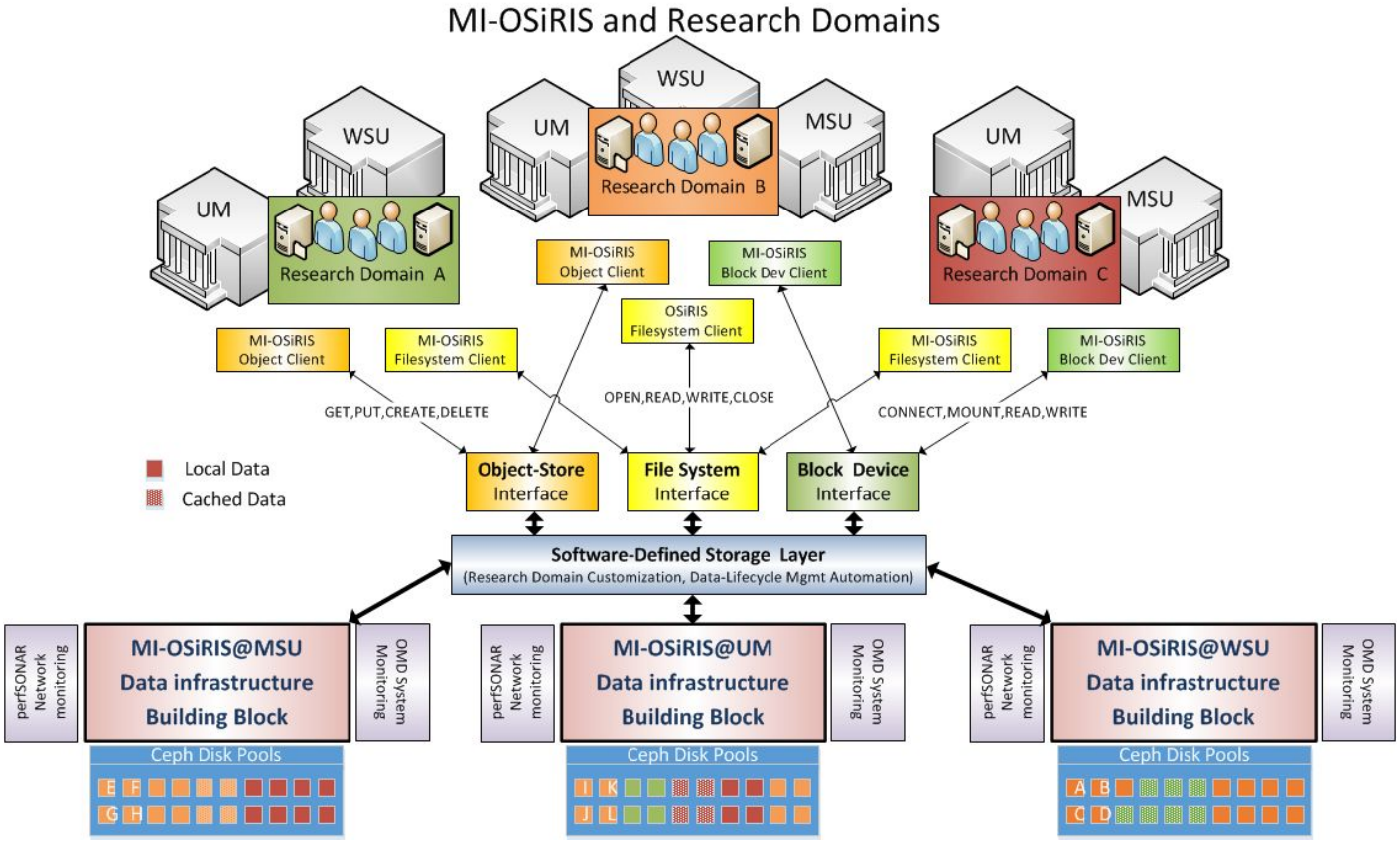
OSiRIS Team

OSiRIS is composed of scientists, computer engineers and technicians, network and storage researchers and information science professionals from **University of Michigan / ARC-TS**, **Michigan State University**, **Wayne State University**, and **Indiana University** (focusing on SDN and net-topology)

We have a wide-range of science stakeholders who have data collaboration and data analysis challenges to address within, between and beyond our campuses:

***High-energy physics, High-Resolution Ocean Modeling**, Degenerative Diseases, Biostatics and Bioinformatics, Population Studies, Genomics, Statistical Genetics and Aquatic Bio-Geochemistry*

Logical View



Ceph in OSiRIS

Ceph gives us a robust open source platform to host our multi-institutional science data

- Self-healing and self-managing
- Multiple data interfaces
- Rapid development supported by RedHat

Able to tune components to best meet specific needs

Software defined storage gives us more options for data lifecycle management automation

Sophisticated allocation mapping (CRUSH) to isolate, customize, optimize by science use case

Ceph overview:
https://umich.app.box.com/s/f8ftr82smlbuf5x8r256hay7660soafk

Ceph in OSiRIS

Ceph instances (mon, osd, rgw, mds) are all deployed with a puppet module.

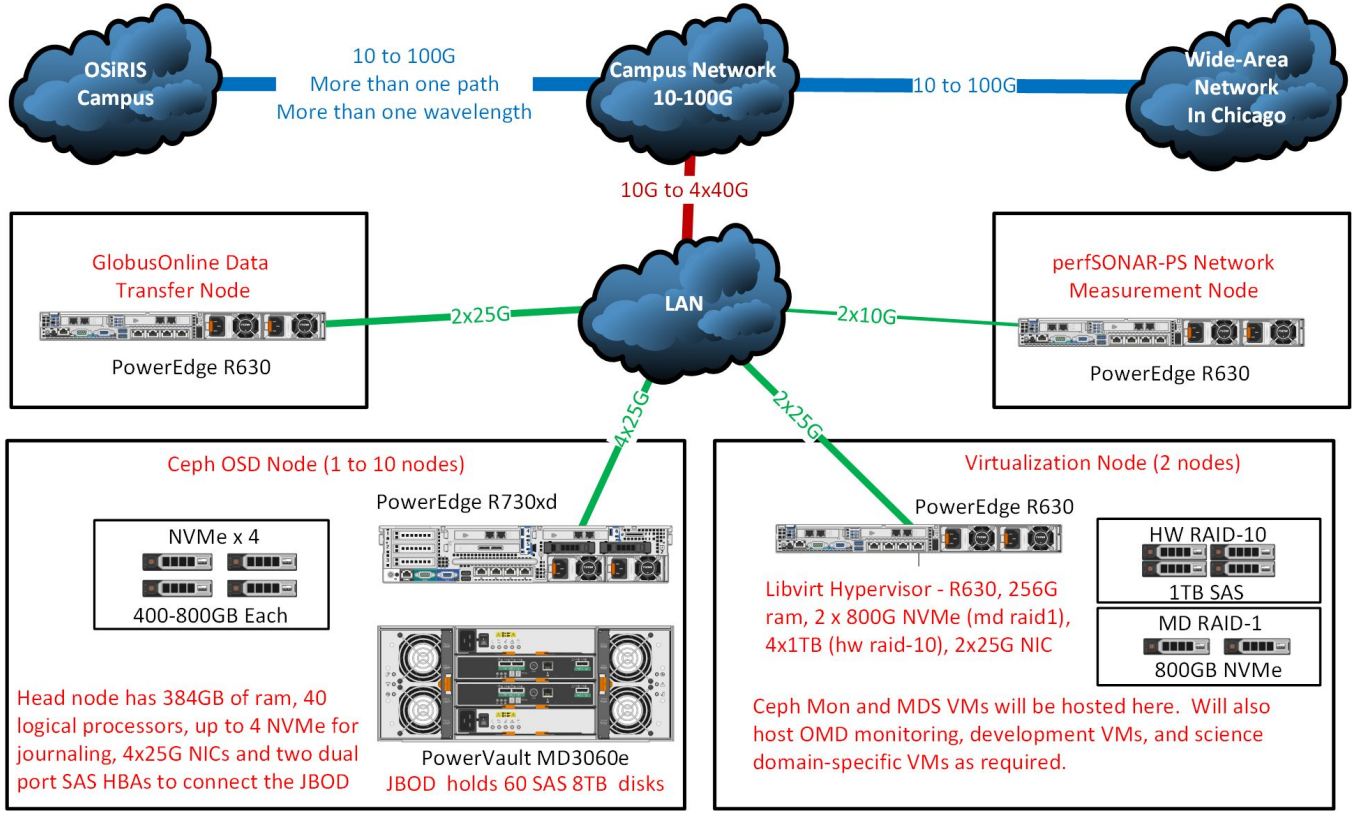
Forked from original by Openstack: https://github.com/MI-OSiRIS/puppet-ceph

Simple example of using module to deploy MDS:

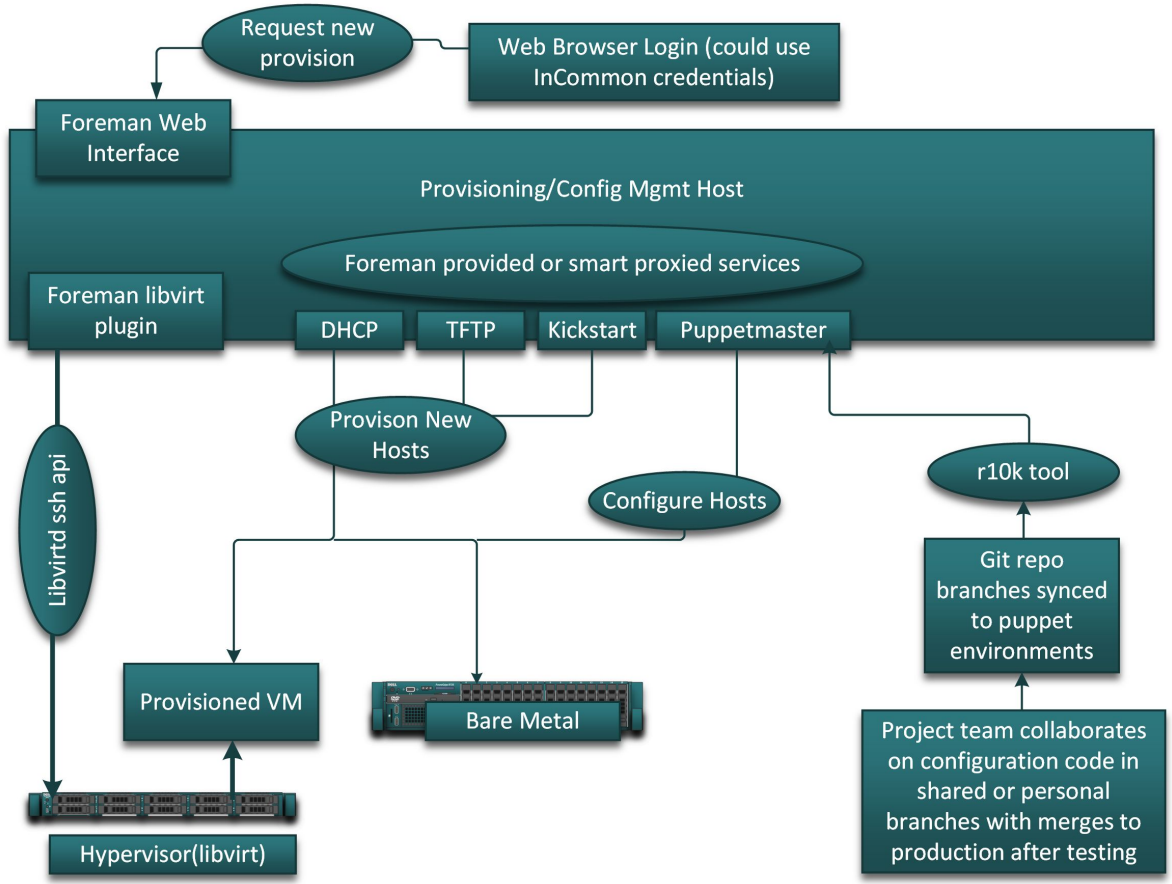
```
ceph::key { "mds.${::hostname}":
        cluster => 'ceph',
        keyring_path => "/var/lib/ceph/mds/ceph-${::hostname}/keyring",
        secret =>  hiera("eyaml-encrypted-secret"),
        user => 'ceph',
        group => 'ceph'
 }


ceph::mds { "$::hostname":
        cluster => "ceph"   # ceph and module default, but non-default names work as well
}
```

Site View

OSiRIS Data Infrastructure Building Block



Provisioning Orchestration



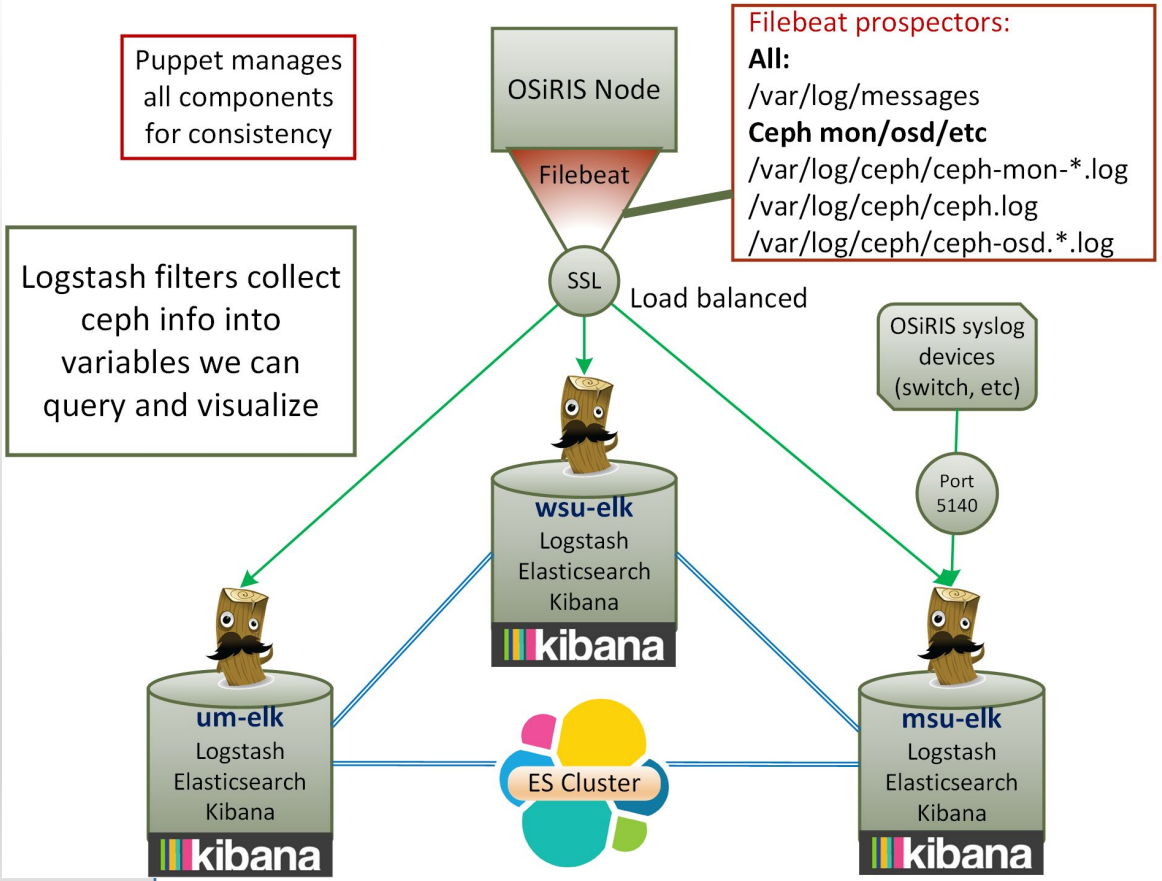
Deploying and extending our infrastructure relies heavily on orchestration with Puppet and Foreman

We can easily deploy bare-metal or VMs at any of the three sites and have services configured correctly from the first boot

Except: OSD activation requires a manual step

Open vSwitch (scripted setup)

Monitoring with ELK



A resilient logging infrastructure is important to understand problems and long-term trends

The 3 node arrangement means we are not reliant on any one or even two sites being online to continue collecting logs

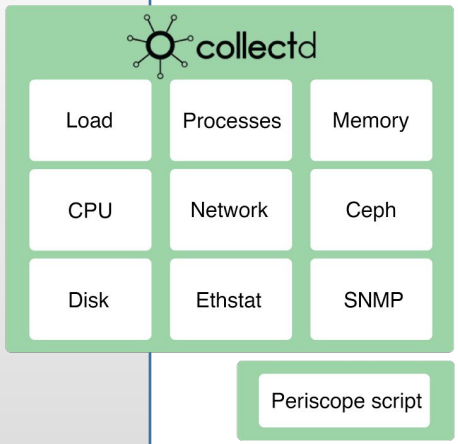
Ceph cluster logs give insights into cluster performance and health we can visualize with Kibana

Logstash: RGW logs

```
filter {
  if [type] == "rgwlog" {

 grok {
     patterns_dir => [ "/etc/logstash/patterns" ]
     match => [

       # grab the client doing rgw transfer from civetweb log

     "message", "%{TIMESTAMP_ISO8601:log_timestamp} %{DATA:ceph_hex_id} %{INT:ceph_log_prio:int} civetweb:
%{DATA}: %{IP:ceph_rgw_client_address} - - \[.*\] \"%{DATA:ceph_rgw_op} %{DATA:ceph_rgw_path} HTTP/[1.]+\"
%{INT:ceph_rgw_http_status} %{GREEDYDATA}",

     # pickup more generic logs
    "message", "%{TIMESTAMP_ISO8601:log_timestamp} %{GREEDYDATA}"

     add_field => [ "received_at", "%{@timestamp}" ]
    }
  date {
     match => [ "log_timestamp", "yyyy-MM-dd HH:mm:ss.SSSSSS", "ISO8601" ]
     remove_field => [ "log_timestamp" ]
  }
}
```

Instrumentation

System Performance Monitoring



Metrics are collected using collectd plugins. Periscope network data is collected from PS nodes using a python script.

Metrics are sent to Influxdb instances at each site where they are stored and downsampled over time. Influxdb internally supports transformations on data like derivatives, means, sums, etc.

Grafana queries Influxdb for data with various operations applied (mean, sum, etc) and graphs the resulting time-series.

We want insight into each layer of our architecture - systems, storage, network, and Ceph itself

We've been able to leverage collectd and its large collection of plugins

All of our systems feed data to instances of Influxdb and we can plot any series of data with Grafana (examples to follow)

Instrumentation

Collectd-ceph plugin interfaces with daemon admin sockets

We configure the appropriate sockets by setting puppet-collectd module params at the appropriate level in hiera - either by 'role' determined from xx-role type hostnames or at the node level (osd are different every node).

um-stor01.osris.org.yaml:

```
collectd::plugin::ceph::daemons: ['ceph-osd.0','ceph-osd.1']   (really many more)
```

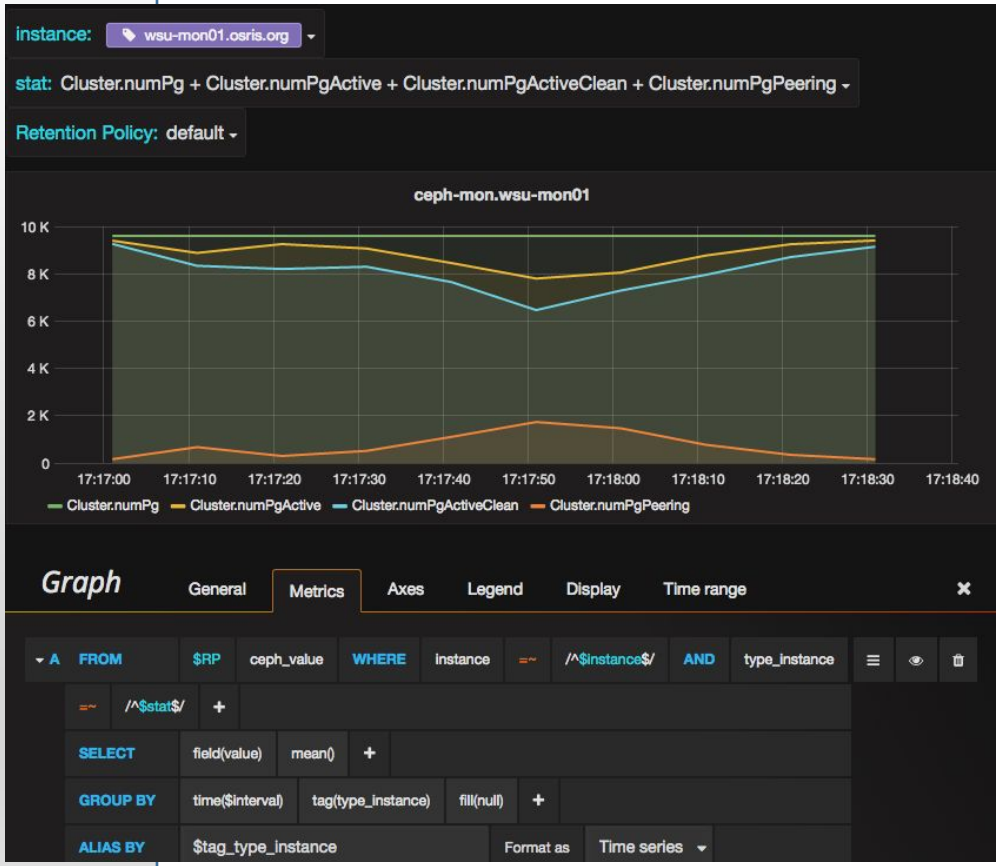
specified for all nodes matching a ceph role (puppet auto parameter lookup)

```
include collectd::plugin::ceph
```

Net result is this config on /etc/collect.d/10-ceph.conf (repeated for all daemons in array)

```
<Daemon "ceph-osd.0">
  SocketPath "/var/run/ceph/ceph-osd.0.asok"
 </Daemon>
```

Instrumentation



Templated grafana dashboards useful to explore stats and/or make permanent dashboards more flexible.

Grafana tag queries used here to select all ceph instances by host:

SHOW TAG VALUES FROM "ceph_value" WITH KEY = "host"

SHOW TAG VALUES FROM "ceph_value" WITH KEY = "instance" WHERE host = '$tag'

This is just a dev playground - panels are incorporated in more permanent dashboards (seen in other slides)

InfluxDB Retention/CQ

InfluxDB Continuous Queries constantly resample data into longer averages stored in Retention Policies that retain data longer as it is downsampled more.

Initial inputs are kept 3 months, 1 minute average for 6 months, 5 minute average for 1 year, etc to 1 day average kept forever.

We use template variables in grafana to choose the RP for visualization (no automatic switching)

```
CREATE RETENTION POLICY “six_months” ON “collectd” DURATION 26w

CREATE CONTINUOUS QUERY one_min_mean ON collectd
BEGIN
SELECT mean(value) AS value INTO collectd.six_months.:MEASUREMENT FROM
collectd."default"./.*/ GROUP BY time(1m)
END
```

Network Monitoring

Because networks underlie distributed cyberinfrastructure, monitoring their behavior is very important

The research and education networks have developed perfSONAR as a extensible infrastructure to measure and debug networks (http://www.perfsonar.net)

The CC*DNI DIBBs program recognized this and required the incorporation of perfSONAR as part of any proposal

For **OSiRIS**, we were well positioned since one of our PIs Co-leads the worldwide perfSONAR monitoring effort for the LHC community: https://twiki.cern.ch/twiki/bin/view/LCG/NetworkTransferMetrics

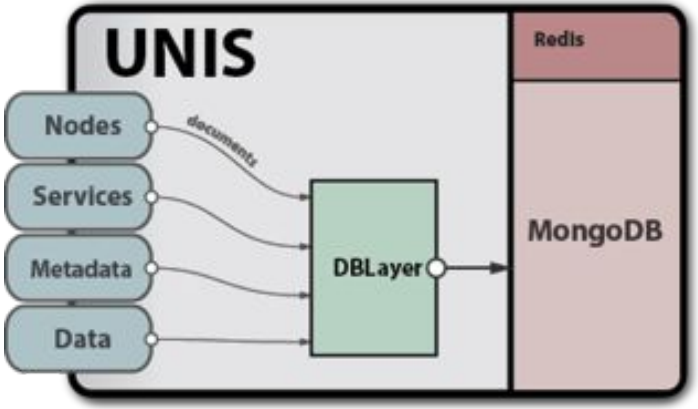
We are working to extend perfSONAR to enable the discovery of all network paths that exist between instances

SDN can then be used to optimize how those paths are used for OSiRIS

NMAL

The OSiRIS Network Management Abstraction Layer is a key part of managing our network as a dynamic resource

Captures site topology and routing information in UNIS from multiple sources: SNMP, LLDP, sflow, SDN controllers, and existing topology and looking glass services.



Package and deploy conflict-free measurement scheduler (HELM) along with measurement agents (Basic Lightweight Periscope Probe - BLiPP)

Correlate long-term performance measurements with passive metrics

Defining best-practices for SDN controller and reactive agent deployments within OSiRIS.

Latency Experiments

Part of our project is also testing the viable limits of a single Ceph instance.

We did a variety of scenarios - at left is a plot of gradually **increasing latency to a single storage block** during recovery to another node

At 320 ms there are major problems

Things don't really recover fully until we back off to 80ms...**likely 80ms is the max safe latency.**

Will test at SC16!!

ATLAS and OSiRIS

ATLAS will use the **OSiRIS** Ceph S3 gateway to read/write single events

Allows leveraging transient computing resources - lost job means only 1 lost event

Authentication is still in the early stages - doesn't yet tie in with **ATLAS** authentication.

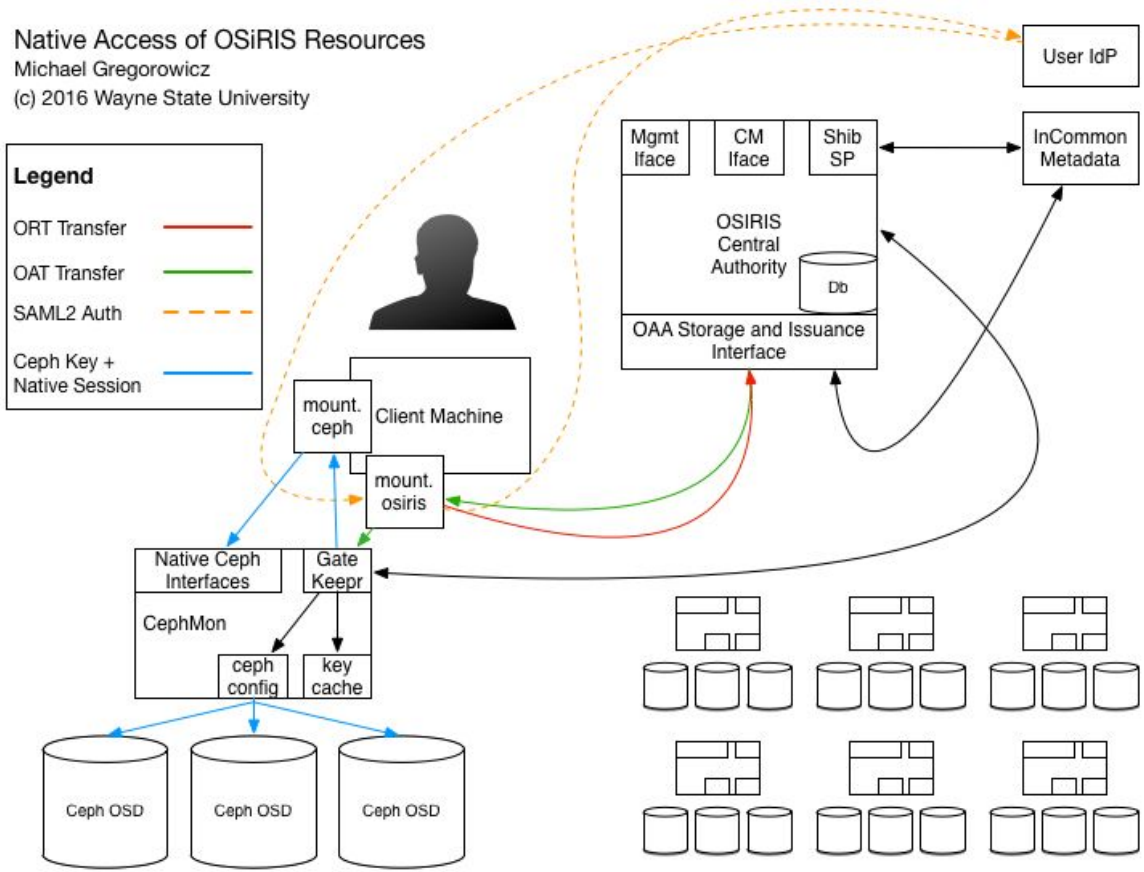
Plots of running test event code

Authentication and Authorization

High-level: **InCommon** participants use their "local" identity and computing resources. VOs self-organize and manage members and roles via OSiRIS services. OSiRIS/Ceph respect membership and roles.

- Session and affiliation data are first pulled into OSiRIS from SAML2 Assertions made by IdPs at configured or InCommon participant organizations
- Valid SAML2 sessions are combined with OSiRIS Access Assertions to create Bearer Tokens that users may use with OSiRIS' wide array of interfaces / use cases

Authentication and Authorization



Physical Ocean Modeling and OSiRIS

Still in the early stages of engagement

The **Naval Research Lab** is collaborating with researchers at **UM** to share their high-resolution ocean models with the broader community

- This data is not classified but is stored on Navy computers that are not easily accessible to many researchers

Discussions are underway to determine a suitable interface and transfer method to put this data into OSiRIS for wider use

We are exploring S3/RGW with objects mapped to a URL to provide high-level organization of the objects (e.g., the URL defines the type/location of the object data)

Our Goal: Enabling Science

The OSiRIS project goal is enable scientists to collaborate on data easily and without (re)building their own infrastructure

The science domains mentioned all want to be able to directly work with their data without having to move it to their compute clusters, transform it and move results back

Each science domain has different requirements about what is important for their storage use-cases: capacity, I/O capability, throughput and resiliency. **OSiRIS** has lots of ways to tune for these attributes.

Questions?

Questions or comments?

We gratefully acknowledge the support of the National Science Foundation . Grant information at https://nsf.gov/awardsearch/showAward?AWD_ID=1541335&HistoricalAwards=false

More information

For more information about OSiRIS see http://www.osris.org

For more information on our network and system tunings for Ceph please see http://www.osris.org/performance/tuning.html

More information on latency testing:
http://www.osris.org/performance/latency

Monitoring and logging:
http://www.osris.org/components/monitoring.html

NMAL:
http://www.osris.org/nmal

AAA Framework:
https://github.com/MI-OSiRIS/aa_services/blob/master/doc/osiris_access_assertions.md

Latency Experiments Details

Some conclusions on Ceph and latency (also see http://www.osris.org/performance/latency)

When latency to any OSD host hits 160ms RTT, throughput and ops are effectively 0. Before this point we see a steady and predictable decrease from the maximum but still usable

Max RTT latency to any given OSD is probably about 80ms. Beyond this recovery ops may not be reliable.

Ceph Monitors are not particularly sensitive to latency. At up to 800ms RTT to one monitor there were not any issues and cluster ops, including mon ops such as creating auth keys, were fine. Somewhat slower for some interactive ops.

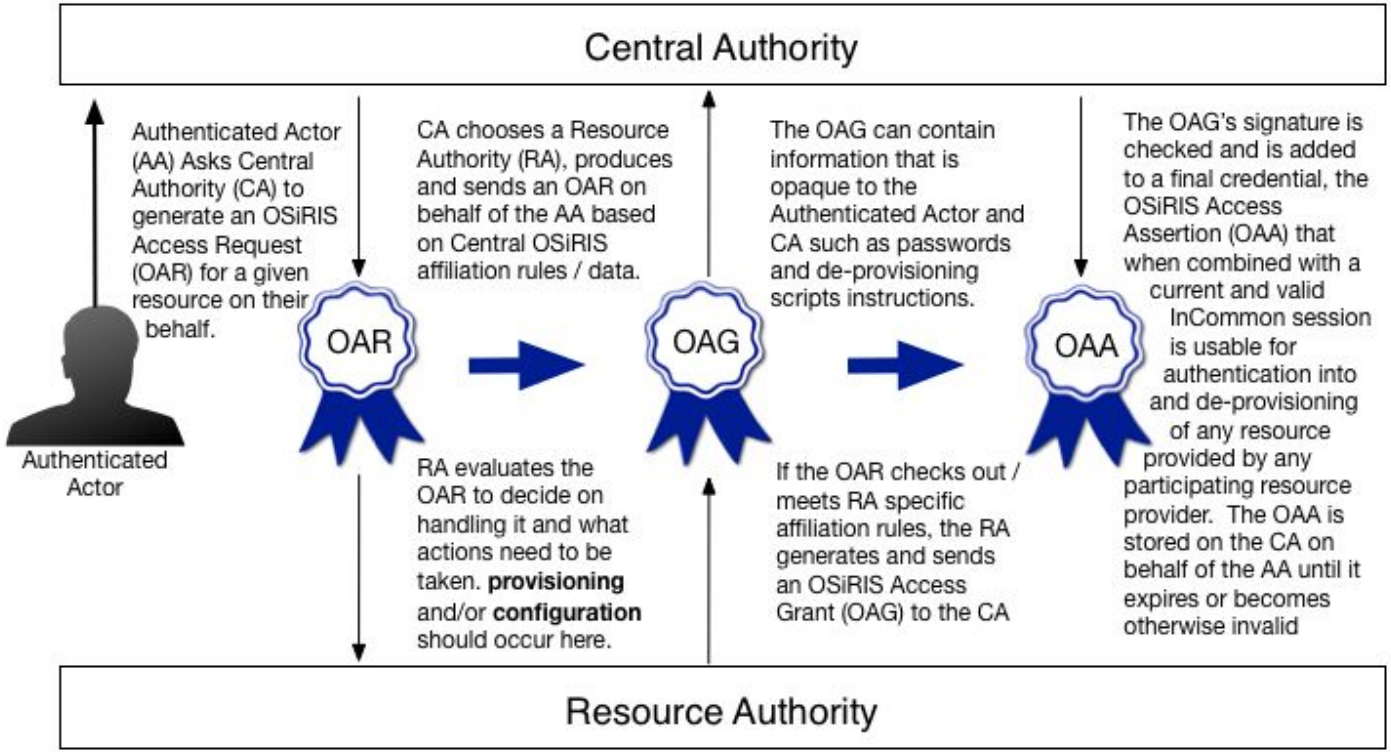
We didn't try to adjust any timeouts, etc upward to compensate, but this may be possible.

Netem Example

```
# we need an ifb device to set a delay on ingress packets
    modprobe ifb  # load 'intermediate functional block device'
    ip link set dev ifb0 up
    tc qdisc add dev eth0 ingress
    tc filter add dev eth0 parent ffff: \
            protocol ip u32 match u32 0 0 flowid 1:1 action mirred egress
    redirect dev ifb0

# now set the netem delay filter on ingress/egress
# last 3 params are delay, variation, and correlation of delay variation
    tc qdisc add dev ifb0 root netem delay 10ms 10 25%
    tc qdisc add dev eth0 root netem delay 10ms 10 25%
```

Authentication and Authorization

OSiRIS Access Assertions: Overview and Lifecycle



Versioning of OSiRIS October 2016

Scientific Linux 7.2
Ceph 10.2.3 (aka Jewel)
Puppet 3.8.7 with future parser enabled
Foreman 1.10.4
Influxdb 1.0.2
Grafana 3.0.4 (bc 3.1.1 had a stopper templating bug for us, waiting for 3.1.2 with bugfix)
Elasticsearch 2.4.1
Kibana 4.5.4
Logstash 2.3.4
Filebeat 1.3.1 (client log shipper)
Check_mk raw/client 1.2.6p16
openvswitch 2.5.0 (RPM built locally)
mlnx_en drivers 3.3 (RPMs built locally with nightly check if need to build vs new kernel versions in SL or ELrepo)

On xfer gateway hosts we use kernel-lt-4.4.22 from ELrepo so we can mount cephfs in-kernel (or whatever is latest, we don't pin to any one version). Theoretically this is so they have the fastest possible access to CephFS for Globus transfers. When Redhat mainlines CephFS support into the stock kernel we'll stop doing this.

Most of the libvirt/qemu tools from stock SL repos, these exceptions were pulled in from the CentOS Virt Special Interest Group repository (SIG). These updated packages let us take live snapshots and live merge changes back into those snaphots later. The purpose is live backups of running VMs into our own fuse mounted CephFS (maybe we get credibility points for that since nobody else is using our cluster yet).

repo:
http://mirror.centos.org/centos-7/7/virt/$basearch/kvm-common/
SIG Page:
https://wiki.centos.org/SpecialInterestGroup/Virtualization

qemu-img-ev 2.3.0-31.el7.16.1
qemu-kvm-common-ev 2.3.0-31.el7.16.1
qemu-kvm-ev 2.3.0-31.el7.16.1