LANL use case of Versity Gateway - Object Computational Storage (OCS)

- Partnership: SK hynix, AirMettle, NeuroBlade, Seagate and LANL
- LANL grid-based AMR 3D simulation output to columnar Parquet
- Leverage Apache driven Analytics tools
- Leverage object (common view of data top to bottom)
- Extend S3 semantics (similar to Iceberg)
- Push orders of magnitude of reduction to near storage

- Big Data Analytics
  - You don’t know what you are looking for, want compact model
  - You know what you are looking for but don’t know where it is

LANL simulation app writes either individual parquet row group objects into bucket
Or a multi-part object where each part is a parquet row group (applications must obey this rule)

Versity Open Src object server that does slightly enhanced S3 and can write to OCSDs - With/without Analytics Friendly Erasure

Erasure/compression

OCS api/OCS on NVME

Optional OCSA

OCSD  OCSD  OCSD  OCSD  OCSD  OCSD
LANL use case of Versity Gateway Object Access to Science Campaign Data

- Leverage massive 2 tier erasure science campaign storage data lake from posix like users and from object users.
- Data is already in immutable files
- Hundreds of gigabytes/sec
- Stable for years
- Disk cool storage and even erasure tape in the future.
What We Do:

Manage Large Unstructured Data Collections at Low Cost

- Software-Defined Storage Platform
- Mass Storage & Large Archive
Global Installed Base

Total Data

2 Exabytes Data

80 Sites
Versity Gateway

S3 -> POSIX
Enables S3 workloads to utilize POSIX file systems and tape via ScoutAM

Modular
POSIX and ScoutAM backends supported, easy to add more

High Performance
Written from scratch
Go / GoFiber
Fast processing and response times

Seamless Integration
Familiar interfaces

Scalable
Load balance across multiple instances

Open-Source
Apache2 licensed, available on GitHub

Collaborative

Versity
Los Alamos National Laboratory
Pawsey
Versity Gateway

- Scalable S3 Service
- High Performance
- Modular Backend Support
- Flexible Open Source Licensing (Apache 2.0)
Scale-Out Archive Manager: ScoutAM

A Modern Data Management Platform

- Online metadata
- Indexed attributes
- Scalable namespace
- Policy Engine
- Scheduling
- Parallel Data Movement

Sensors, Satellites, Telescopes, Spectroscopy, Video Content, DNA Sequencers, Monitoring Systems, Supercomputer Cluster

NFS, Samba, FTP, S3, Direct

Gateway

ScoutFS Filesystem

ScoutAM Application

Object, Tape, Cloud
Modular Backend

Current Backends:

- POSIX - filesystem with xattr support
- ScoutFS - [https://github.com/versity/scoutfs](https://github.com/versity/scoutfs)

Open to collaboration on new backends
ScoutFS Optimized Backend

Optimized Writes

- Multi-part upload segments are written once to the underlying storage then
- Parts of data upload are combined into a single file with a system call.
- **Eliminates one full read/write cycle**
- Speeds up the process and reduces overall upload time.

S3 Glacier Mode

- Cold storage feature for data archiving and long-term data retention.
- Data may be stored on local tape systems using Versity’s ScoutAM platform.
- Manage storage costs more effectively by accessing low cost storage.
Project Layout

- auth
- backend
- cmd/versitygw
- integration
- s3api
- s3err
- s3event
- s3log
- s3response
Project Layout

Multi-Tenant
IAM account integrations

Extensible to support new authentication systems

Currently Supported:
Local accounts
## Multi-Tenant

root account defined with cli flags/env vars
admin/user accounts stored in IAM service

<table>
<thead>
<tr>
<th></th>
<th>See All Buckets</th>
<th>Create New Buckets</th>
<th>Create New Users</th>
<th>Assign Bucket Ownership</th>
<th>See Only Owned Buckets</th>
</tr>
</thead>
<tbody>
<tr>
<td>root</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>admin</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>user</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>
Project Layout

Storage Backends

Extensible to support new storage systems

Currently Supported:
POSIX
ScoutFS
Backend

- Embed backend.BackendUnsupported
- Implement backend.Backend methods as needed to satisfy interface
- Any unimplemented will return s3err.ErrNotImplemented back to client

```go
type Backend interface {
    // bucket operations
    ListBuckets(context.Context, owner string, isRoot bool) (s3response.ListAllMyBucketsResult, error)
    ...

    // multipart operations
    ...

    // standard object operations
    ...

    // special case object operations
    RestoreObject(context.Context, *s3.RestoreObjectInput) error
    ...

    // object tags operations
    GetTags(context.Context, bucket, object string) (map[string]string, error)
    ...
}
```

package mystorage

type MyStorage struct {
    backend.BackendUnsupported
}
```
ScoutFS/Glacier Example

ScoutFS/ScoutAM - Archiving Filesystem
- POSIX filesystem
- Supports offline files where data only resides on tape/mass storage
- Automatic recall when data requested

When in Glacier Emulation Mode, scoutfs backend enables following:

- HEAD offline object returns
  storage-class: GLACIER
  x-amz-restore: (transition state)

- GET of offline file returns
  Invalid Object State

- Restore Object
  triggers backend data recall

```go
type Backend interface {
  ...

  // special case object operations
  RestoreObject(context.Context, *s3.RestoreObjectInput) error
  ...
}
```
Project Layout

CLI command support
Project Layout

Integration Tests

GitHub CI runs test suite against POSIX for PRs
Manually run test suite against any storage backend
Project Layout

Frontend API Handlers

Go Fiber framework for HTTP(s) handlers
Supporting S3 and Admin API
Project Layout

S3 errors contains all errors that clients would expect from AWS S3

Non-s3err types returned from backend treated as Internal Server Error

S3 error response types
Project Layout

- Same event structure as AWS
- Extensible event integrations, currently supported:
  - NATS
  - Kafka

S3 Event Notifications
Project Layout

Same event structure as AWS

Extensible log integrations, currently supported:
- file
- webhook

S3 Server Access Logs
AWS defines specific XML structure for some requests/responses.

For these requests, use well-specified struct field members and XML struct tags.

S3 Response formatting
Gateway enables S3 workloads for Pawsey's Large Scale Archive

- Versity's ScoutAM managing tape-based archive, Banksia
  - 150 PB tape
  - 34 tape drives
  - 2 tape libraries
  - 5 PB cache
- Incoming data is S3 via gateway
- Cluster load balanced
Pawsey Solution Architecture

ASTRONOMY DATA
- CAPTURE
- COMPUTE
- 60 PB OBJECT DATA LAKE

VERSITY SOLUTION
- ScoutAM NODES
  - 6 x INTEL R2208
- ScoutAM INTELLIGENT CACHE
  - 5.7PB USABLE

BANKSIA MASS STORAGE
- Spectra Logic TFinity
- Spectra Logic TFinity
Developed in the Open on GitHub

https://github.com/versity/versitygw

- Bugs/Features tracked in GitHub Issues
- Documentation in project wiki
- CI using GitHub Actions
Thank You

info@versity.com

@versitysoftware

https://github.com/versity/versitygw