Archipelago:
New Cloud Storage Backend of GRNET

Filippos Giannakos
philipgian@grnet.gr

Chrysostomos Nanakos
cnanakos@grnet.gr

Greek Research and Technology Network (GRNET) S.A.

16th TF-Storage Meeting
February 13, 2015
Background

- Synnefo
  - OpenStack compatible cloud platform
- Powers:
  - ~okeanos
    https://okeanos.grnet.gr
  - ~okeanos-global
    https://okeanos-global.grnet.gr
Motivation

- **Pithos (Storage Service)**
  - Swift API
  - Uploaded Files / Images
- **Plankton (Image Service)**
  - Glance API
  - Registered Images
- **Cyclades (Compute/Network/Volume Service)**
  - Nova/Neutron/Cinder API
  - Virtual Disks / Volumes
Unified View of Storage Resources

- **Files**
  User files, with Dropbox-like syncing.

- **Images**
  Templates for VM creation.

- **Volumes**
  Live disks, as seen from VMs.

- **Snapshots**
  Point in time snapshots of volumes.
Archipelago

- Storage Virtualization System
  - Powering storage in Synnefo
- Decouples storage resources from storage backends
  - Files / Images / Volumes / Snapshots
- Unified way to provision, handle, and present resources
- Decouples logic from actual physical storage
  - Software-Defined Storage
Backend Agnostic

- Advantages:
  - No vendor lock-in
  - Migration from one storage backend to another
  - Possibility of combining backends

- okeanos started with NFS and migrated to RADOS
With Archipelago we offer:
- Deduplication
- Syncable File/Image upload with partial upload
- Image Registration
- Thin clones
- Downloadable thin snapshots

with zero data movement.
Services Overview
How it is Done

- Everything is a resource on Archipelago
- The **same** resource is exposed as:
  - A file through the API of the Storage Service (Pithos)
  - An image through the API of the Image Service
  - A live disk / VM volume through the API of the Volume Service
  - A snapshot through the API of the Volume Service
- All data remain in one place
- No copying of data around
Mapfiles

- Mapfile for each resource
  - Keeps track of the mappings from resource offset to objects
  - Keeps metadata information for each resource
- Operates on mapfiles
  - not on the actual data
From Mapfiles to Data
High Level Architecture
Low level Architecture

- Same Archipelago software stack on each node
- Modular/Pluggable architecture
- Distinct processes
  - Communicate via requests on a shared memory segment
Low level Architecture
Core

- **Volume composer**
  - Composes resource I/O from individual objects
- **Mapper**
  - Keeps and updates the mappings from resource offsets to individual objects which actually hold the data
- **Flexible I/O pipeline**
  - Can be extended with other components offering common functionality
Client Endpoints

- Upstream Native QEMU virtio driver
  - VM interface
- Blktap2 driver
  - Admin/VM interface
- Pithos
  - Web interface / REST API
Backend Drivers

- Currently supported backends:
  - Files (Any FS with POSIX semantics)
  - RADOS (Using native librados interface)
  - GlusterFS (Using native libgfapi interface)
  - S3/SWIFT (Experimental)

- General: Anything that can support object semantics
Advantages over other Solutions

- Allows for resource composition from already existing objects
- Standalone resources
- Unified view of all cloud resources
- Backend agnostic
Experience from Production

- 3000 VM volumes
- ≈2.8M user uploaded files
- ≈20TB deduplicated data
- ≈80TB total data before replication
- 22M actual objects on the storage backend
Upcoming Features

- New mapfile format
  - More compact
  - Allows for larger volumes with minimal metadata overhead (1PB → 2GB)
- Garbage collection
  - Deferred reference counting
  - Automatic deletion of unused objects
- SSD Read-only caching
  - Lower latency
- ISCSI target
Thanks

- Software lives at:
  - https://www.synnefo.org
- Repository:
  - https://github.com/grnet/archipelago
- Documentation:
  - https://www.synnefo.org/docs/archipelago/latest
- Mailing list:
  - synnefo@googlegroups.com