Using CDMI to Manage Swift, S3, and Ceph Object Repositories

David Slik
NetApp, Inc.
A Brief Overview of CDMI

- CDMI (Cloud Data Management Interface) is a RESTful API for accessing and managing cloud storage.
- The major cloud storage APIs are:
  - Amazon S3
  - CDMI
  - Microsoft Azure
  - Swift API (part of OpenStack)

- CDMI is widely implemented
  - >30 server implementations
  - CDMI gateway for S3, OpenStack support
A Brief Overview of CDMI

- 2009: SNIA Cloud Technical Working Group founded to explore API standardization
- 2011: CDMI 1.0 ratified as a US Technical Architecture
  - CDMI 1.0.1 errata released in late 2011
  - CDMI 1.0.2 errata released in mid 2012
- 2012: CDMI 1.0.2 becomes ISO/IEC 17826
- 2013: CDMI 1.1 under active development
  - 18 Extensions submitted
- 2015: CDMI 1.1.1 Submitted to ISO/IEC
  - 13 Extensions submitted, 5 incorporated
A Brief Overview of CDMI

- Why does CDMI Matter?
  - Simple and easy to implement
    - Start with HTTP and add functionality, few mandatory parts
  - Advanced functionality not found in other APIs
    - Provides a foundation for next generation cloud services, such as federation
  - Open industry standard
    - Not controlled by any one vendor, protection against patents
  - Well defined formal standard
    - Enables interoperability, testing, and cross-vendor support
  - Widespread government support and adoption
A Brief Overview of CDMI

- CDMI Standardizes:
  - CRUD operations (Create/Read/Update/Delete)
  - Data, Container, Queue and Domain objects
  - Identity and access control model
  - Metadata (including client and vendor extensibility)
  - Query and Notifications
  - Versioning
  - Serialization and Deserialization
  - Interoperability with other NAS and cloud protocols
Converged Data Management

- Objects, Files, LUNs, NoSQL databases, and many other types of persistent data entities can all be managed through the Cloud Data Management Interface (CDMI)

- This presentation focuses on how CDMI provides a common management interface that sits beside data access interfaces, such as NFS, CIFS, iSCSI, S3, & Swift.

- This is especially valuable for converged storage infrastructure that allows data objects to move between platforms and interfaces (Ceph, Sheepdog, etc)
First, Some Definitions

- What is an object?
  - At a generic level, an object is something that has a name, and represents something that can be manipulated (accessed, managed, etc).

- Examples include:
  - File
  - Directory
  - LUN
  - System
  - Snapshot
  - Etc.
First, Some Definitions

- What is an namespace?
  - Each object has a name. The organization of these names is known as the namespace.
    - Namespaces can be flat (objects in an S3 bucket)
    - Namespaces can be hierarchical (files in a file system)
    - Namespaces can be arbitrary (graph relationships)
  - Restrictions on allowable names are also important parts of namespaces:
    - Reserved characters (such as for hierarchical separators)
    - Character set encodings (Unicode)
  - The more restrictive a namespace, the less it is able to accommodate different types of objects
First, Some Definitions

- What is management?
  - Inf. Whatever doesn’t fit in the data path.
  - Management operations are applied against objects or sets of objects:
    - Snapshots
    - Migration
    - Permissions
    - Etc.
First, Some Definitions

- Putting it all together:
  - The CDMI (Cloud Data Management Interface) provides a superset **namespace** to allow **management** operations performed against cloud-resident **objects**.
  - CDMI can sit on top of almost any data repository, be it file systems, block storage systems, even NoSQL systems.
CDMI Namespaces

- CDMI supports arbitrary tree-based hierarchies
  - Superset of Swift, S3, Azure and other cloud namespaces
  - Superset of standard filesystem namespaces
  - Superset of LUN/Zone hierarchies
- CDMI references allow symlink-like cross-tree links

- This allows a single CDMI namespace to encapsulate all of these different data types into a single namespace for management purposes:
Varied Namespace Restrictions

CDMI Object Model

Root Container

Container

Data Object (or Queue)

System Capabilities

Container Capabilities

Data Object Capabilities / Requirements

Overlay for Swift Object Model

CDMI Root Container

CDMI Container (Representing Account)

CDMI Container

CDMI Data Object

Overlay for File System Model

CDMI Root Container

CDMI Container

CDMI Data Object

Overlay for S3 Object Model

CDMI Root Container

CDMI Container

Account

CDMI Data Object
Example of Subset Namespaces

- Root Container
  - Container
    - Data Object (or Queue)
  - CDMI Container
    - CDMI Container (Representing Account)
- System Capabilities
- Container Capabilities
- Data Object Capabilities / Requirements

- Exported as NFS/CIFS File Systems
- Exported as iSCSI LUNs
- Exported via Swift
- Exported via S3

* File-system-like hierarchies can be emulated on top of S3/Swift, but lack much of the operational expressiveness
CDMI Namespace Demonstration

CDMI & AJAX Client Demonstration
CDMI for LUN Management

- Typical management lifecycle
  - LUN creation
  - LUN discovery
  - LUN permissions
  - LUN SLOs
  - LUN exports
    - Taking existing files and exporting them as a LUN
- Others
  - Snapshots, serialization, etc.
CDMI LUN Demonstration

CDMI & AJAX Client Demonstration
CDMI for File Management

- Typical management lifecycle
  - Directory structures
  - File/Directory SLOs
  - Filesystem exports

- Others
  - Snapshots, serialization, etc.
CDMI File System Demonstration

CDMI & AJAX Client Demonstration
To Summarize

- CDMI provides:
  - Ability to view and manage different types of objects in a unified namespace
  - Common management API and approaches
  - A way to bundle file, object and LUNs together for mobility and management

- Extensible to arbitrary management structures
Thank you!

Questions

dslik@netapp.com