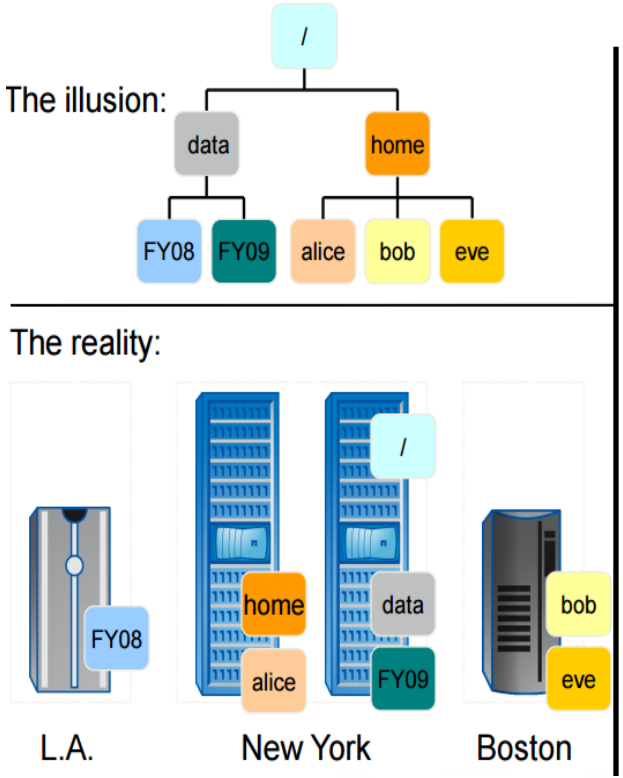
Federated Cloud File System Framework

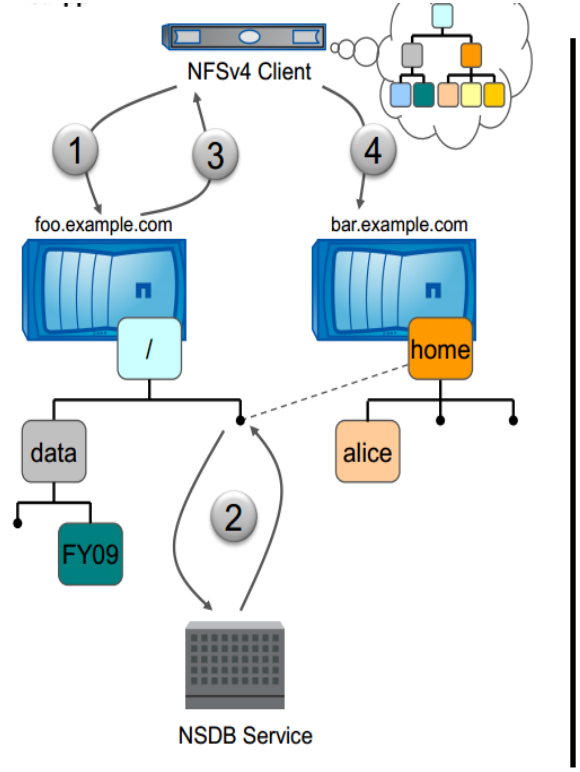
Ujjwal Lanjewar
(ujjwal.lanjewar@calsoftinc.com)

Agenda

- Background
- Federated Cloud File System (FedCFS) Overview
  - Framework
  - Filesets in Cloud
  - Global Namespace
- Cloud Storage Federation
- FedCFS Protocol Overview
- FedCFS Architecture and Components
  - Architecture Overview
  - Distributed Cloud Interface
  - Object File System
  - Global VFS
- Next Steps
- References

Federated File System (Recap)





Highlights

- Global Uniform Namespace
- Federation of many file servers
- Independent of File-Protocol
- Built on existing NFS/CIFS File Servers

Terminology

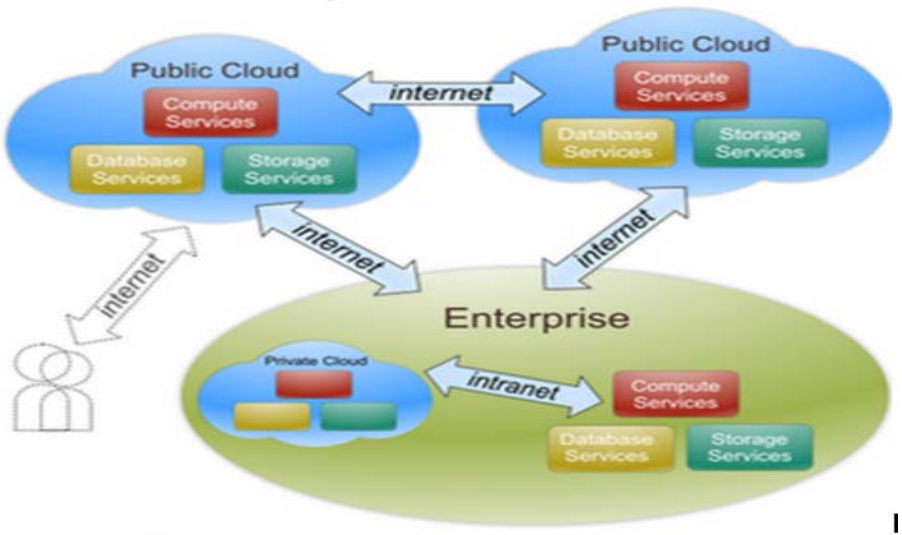
- NSDB – Namespace DB (LDAP)
- FILESET – Directory Tree (Unit of Data)
- JUNCTIONS – Mount Points
- FSL – Refers to the location of filesets

Reference: IETF '77 NFSv4 WG, 2010, James Lentini, NetApp

Federated Cloud File System (FedCFS) Framework

Motivation

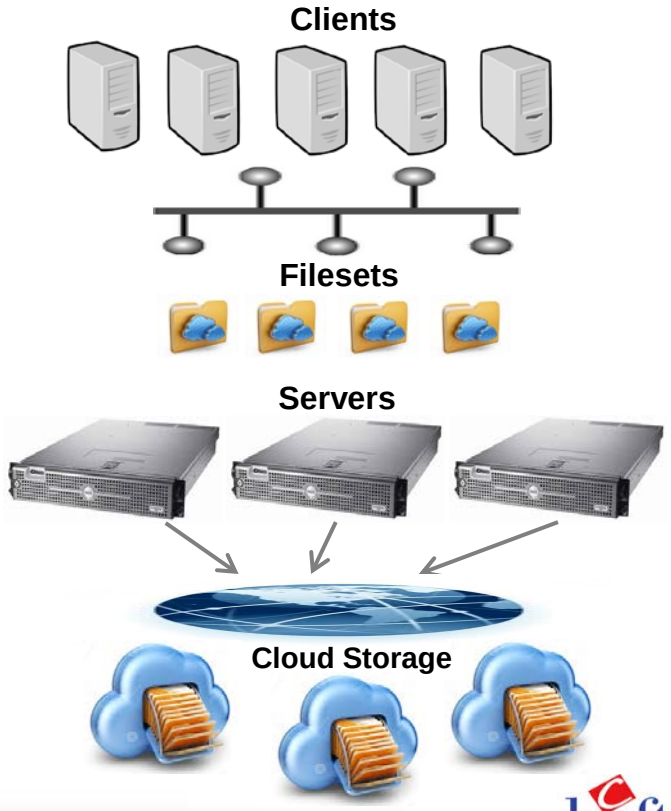
- Hybrid Cloud – Organizations using multiple clouds. Need for federation framework for cloud storages.
- Global namespace - Need to view distributed cloud data under a common global namespace
- Possible FedFS extension (e.g. integration with cloud, NSDB alternative, etc).



FedCFS Framework (cont…)

Overview

- Global Uniform namespace for data stored in the hybrid clouds
- File Services over Cloud Object Storage
- Distributed File Services
- Data organized in the form of **Filesets (Directory Tree)** in cloud
- Cloud Storage accessible over Heterogeneous Protocols
- Replication HA with dynamic namespace resolution (policy, etc)
- Namespace is rendered at the file server itself
- Caching of cloud data for performance reasons (Lease for consistency)
- Uniform Distributed Cloud Interface (**DCI**)
- Ability to add extensions to file services
- Transparent to NFS/CIFS Clients

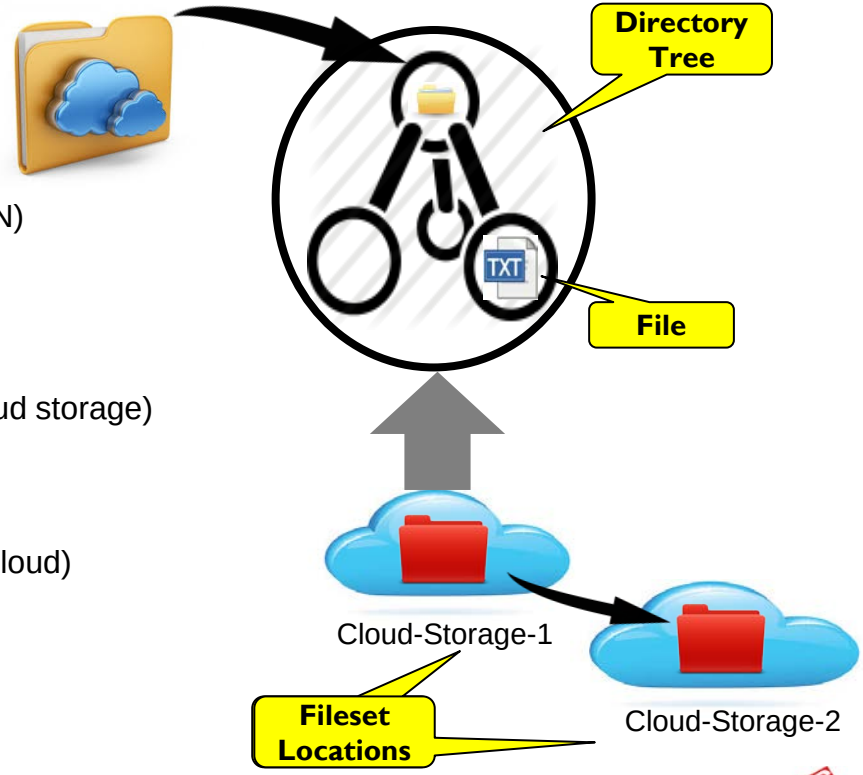


Filesets

- **Fileset**
  - Logical entity representing a Directory Tree
  - Accessible from one or more servers
  - Typically identified with UUID or a globally unique name (FSN)

- **Fileset Location (FSL)**
  - Cloud Object Storage (e.g. Containers on the cloud)
  - Primay and Secondary Locations
    (Secondary Locations are Read only, hosted on different cloud storage)
  - Can also be Directory on the Server (FedFS)

- **Fileset Migration / Replication**
  - Fileset can have one or more locations (hosted on different cloud)
  - Fileset can be migrated/replicated between cloud



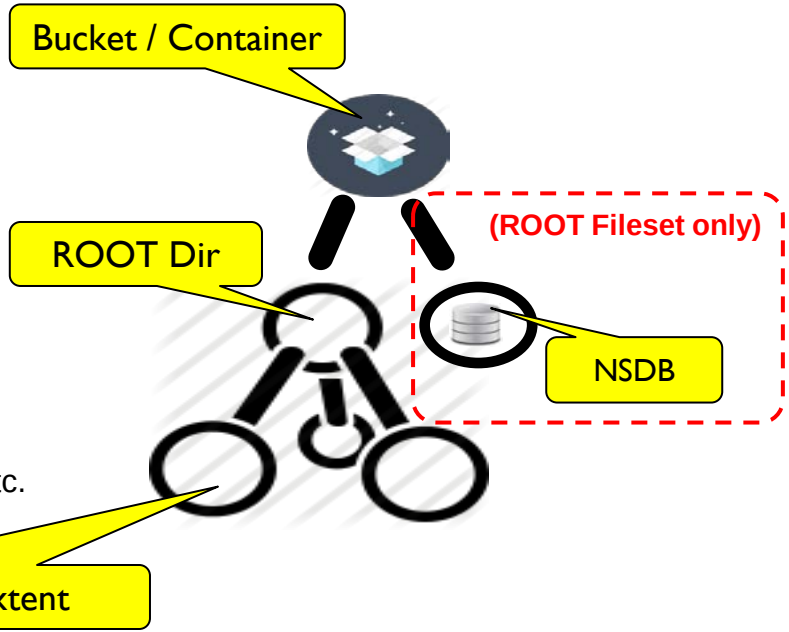
Filesets (cont...)

Fileset in Cloud…

- Fileset is represented with a container / bucket.

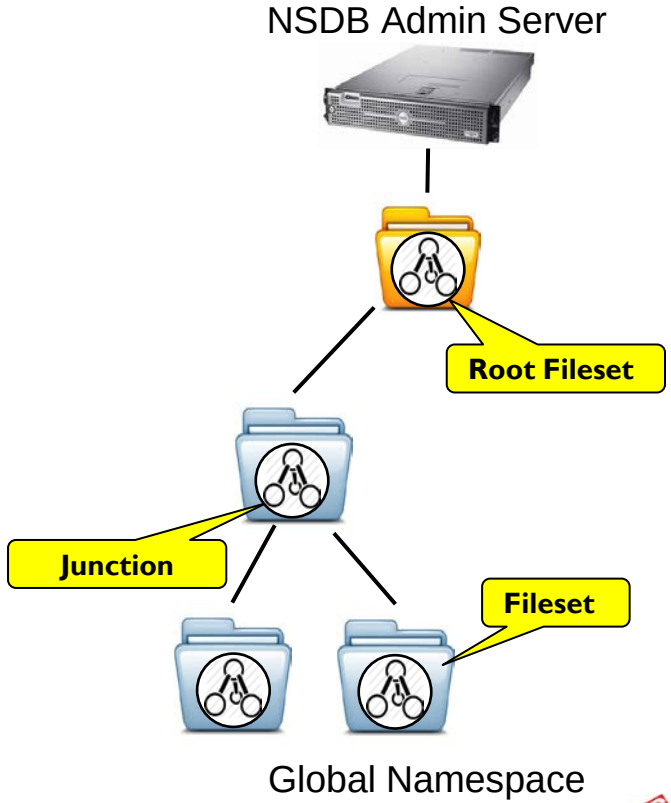
- **Cloud Objects**
  - **Directory –** Cloud storage object containing information of directories / files underneath in namespace hirarchy
  - **File –** Object Storing file data or Information about extents
  - **Extents –** Used for really large files and stores parts of file data
  - **Junctions –** Objects that store mount point target FSN (Leaf)
  - **File Group** – Object storing group of files (For small files)

- **ROOT Fileset**
  - Links parts of namespaces together to provide global namespace.
  - Read-only (Can be updated with FedCFS admin commands)
  - Contains **NSDB** to Information about filesets, domain, junctions, etc.



Global Namespace

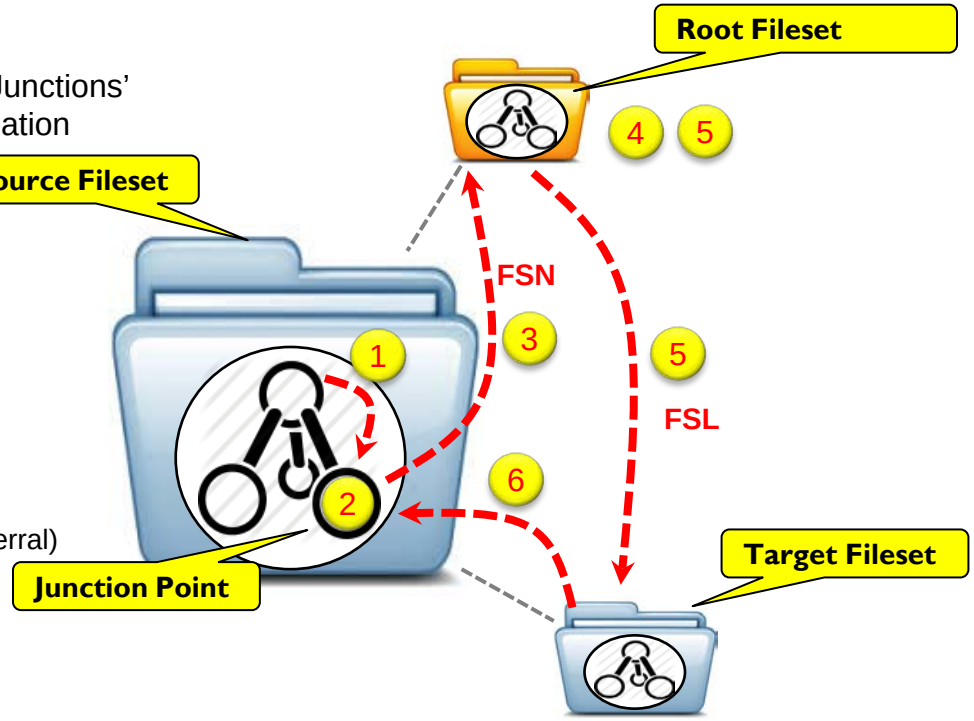
- **Namespace**
  - Single Logical Directory Tree
  - Spans multiple filesets
  - Filesets are connected together by way of Junctions
- **Namespace DB**
  - Hosted in Root Fileset in cloud. Equivalent to NSDB (FedFS).
    Can be accessed from multiple admin server. Used to map FSN to FSL
  - DBs stores information about
    - All the registered filesets (FSN as the key)
    - Directories and Subdirectories (within the ROOT fileset)
    - Junctions and Referral Information
- **Namespace Management**
  - NSDB Adminstration Server (Manages Root Fileset)
  - NSDB Admin Commands (allows namespace management)



Global Namespace (cont...)

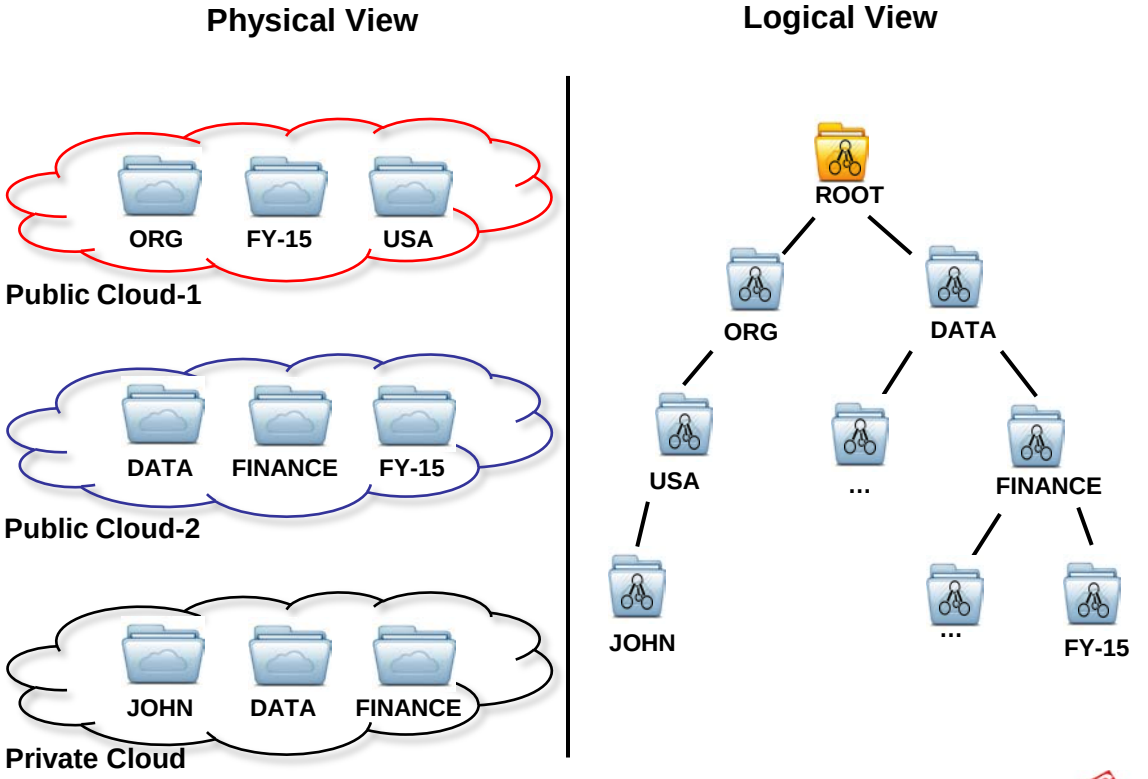
- **Junctions / Mount Points**
  - Filesets can be mounted on other filesets through ‘Junctions’
  - Junction – It contains URL with target fileset information
  - Junctions are independent of FSL
  - Junction Targets are FSN

- **Junction Resolution**
  - Juntction resolution happens through root fileset
  - Happens on the server side
  - Client agnostic
  - Protocol Independant
  - Steps: (Access)
    1. Checks for the object type
    2. If object is junction, get the Type+Target FSN (Referral)
    3. Query ROOT fileset with FSN for FSL
    4. Obtain FSL details from ROOT fileset
    5. Execute Policy Manager for the best fit FSL
    6. Return the FSL as Target for alllowing access



Cloud Storage Federation

- **Physical Structure**
  - Stored in Hybrid Cloud Environment
  - Fileset data stored as objects
  - Data Replication
    - **Filesets replicated across Clouds**
    - **Clouds can be positioned like Tiers**
- **Logical View**
  - Data is viewed as hirarchical structure
  - In the form of files and folders.



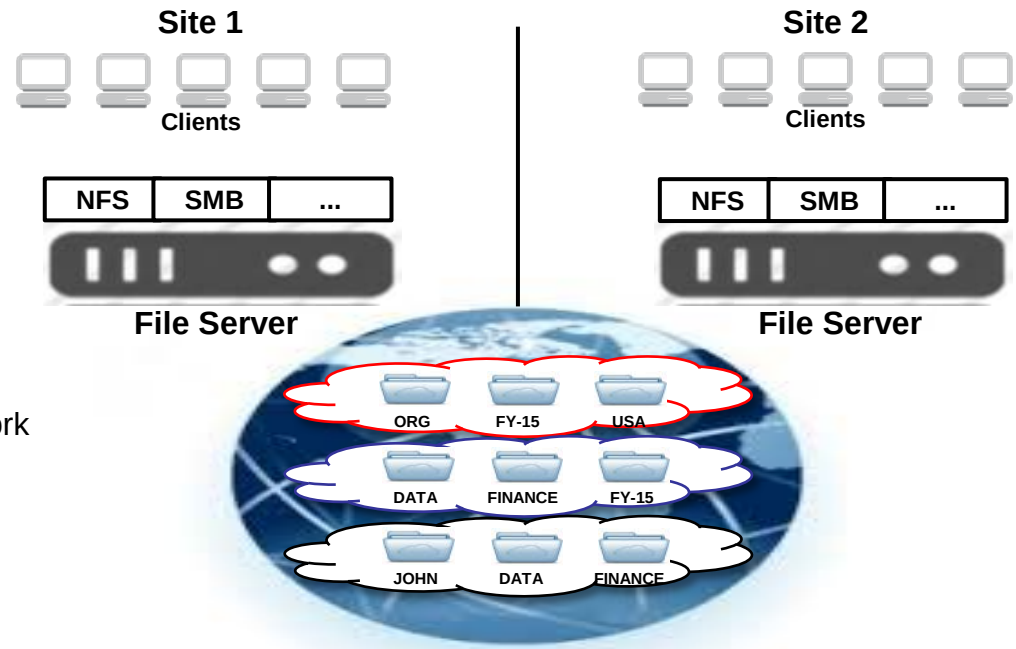
Cloud Storage Federation (cont…)

FedCFS Cell / Domain
- ❑ Logical boundary of a global namespace
- ❑ Each Cell contains a ROOT Fileset
- ❑ A Cell contains logically grouped filesets
- ❑ Fileset can technically mount a foreign fileset

Multi-Site
- ❑ Site - the location where data is accessed
- ❑ Site contains one or more file servers

Multi-Protocol
- ❑ Federation is Protocol Independent
- ❑ File Services are built on top using federation framework

Cloud Tiers
- ❑ Cloud can be viewed as Storage Tiers
- ❑ Fileset can be replicated and migrated between Tiers



FedCFS Protocols

FedCFS Administration
- ❑ Fileset Management
- ❑ Namespace Management (Junctions)

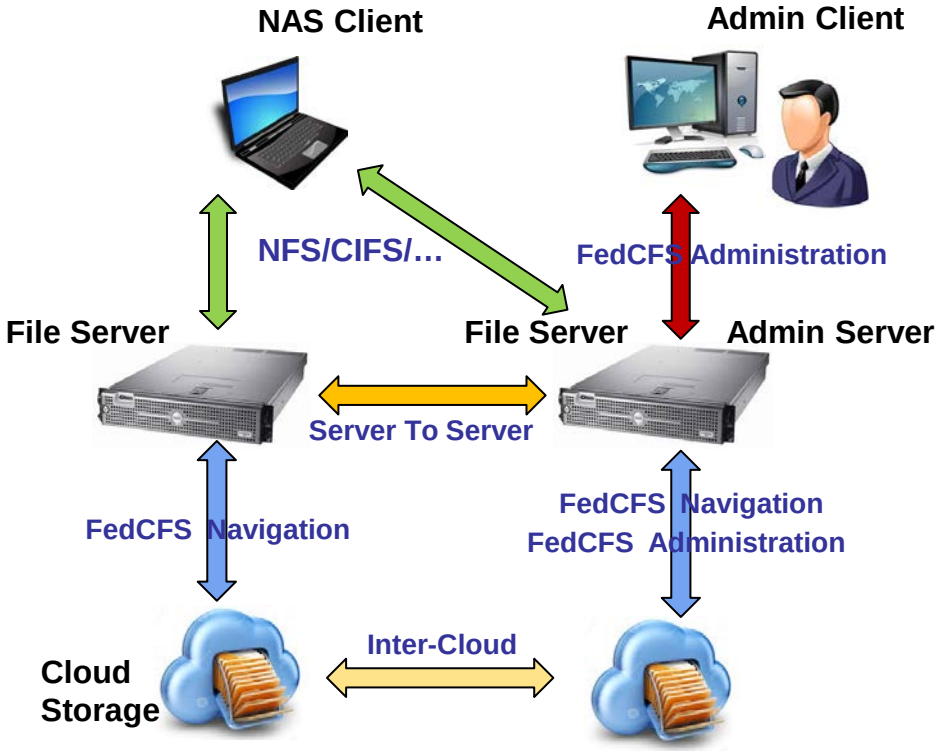
Namespace Navigation
- ❑ Cloud Interface
- ❑ Object Layout in Cloud
- ❑ Junction Resolution

File-Protocol
- ❑ File-Protocol (NFS/CIFS)
- ❑ Federation is Protocol Independent
- ❑ File Services are built on top using federation framework

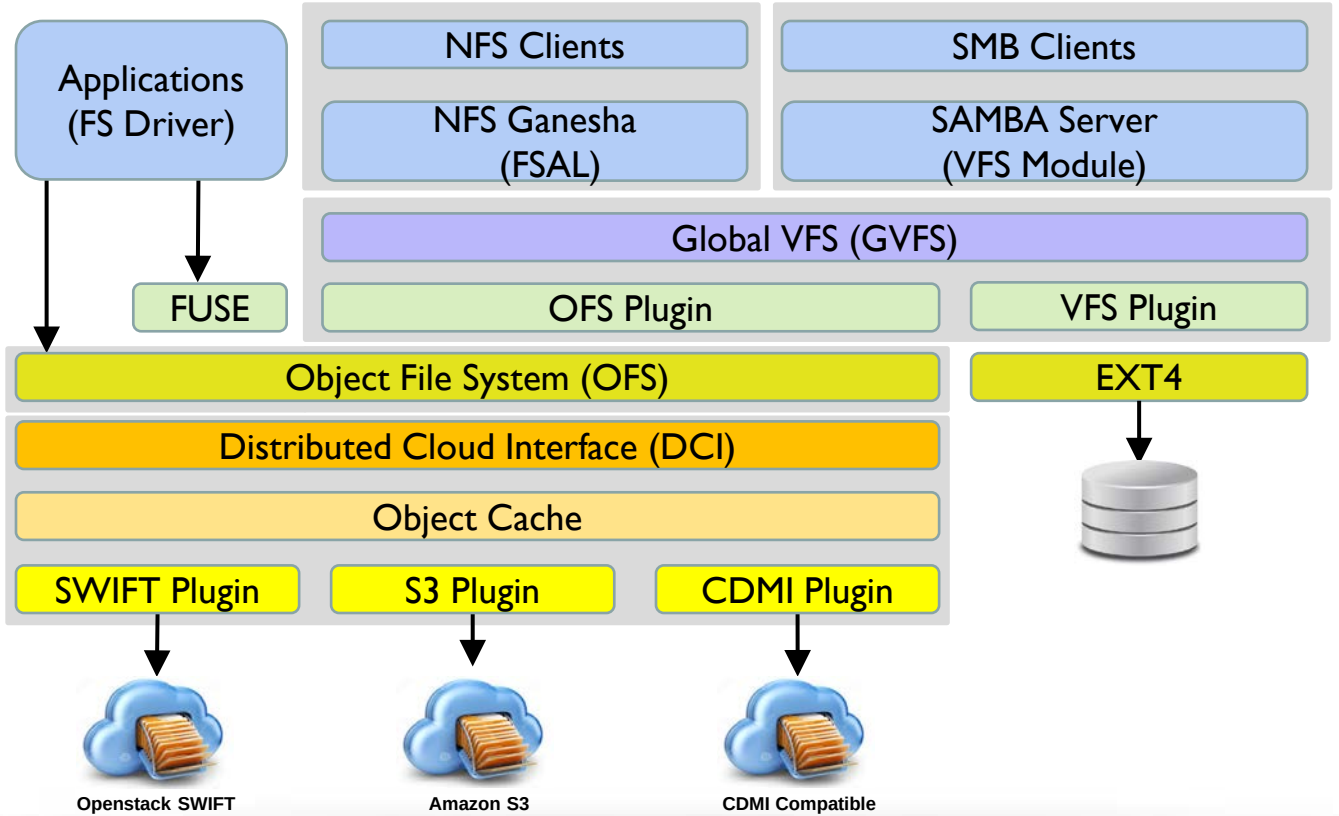
Server to Server Protocol **
- ❑ Sync / Lock
- ❑ Clustering

Cloud to Cloud Protocol **
- ❑ Inter-Cloud data replication

**** Currently outside of Federation protocol**



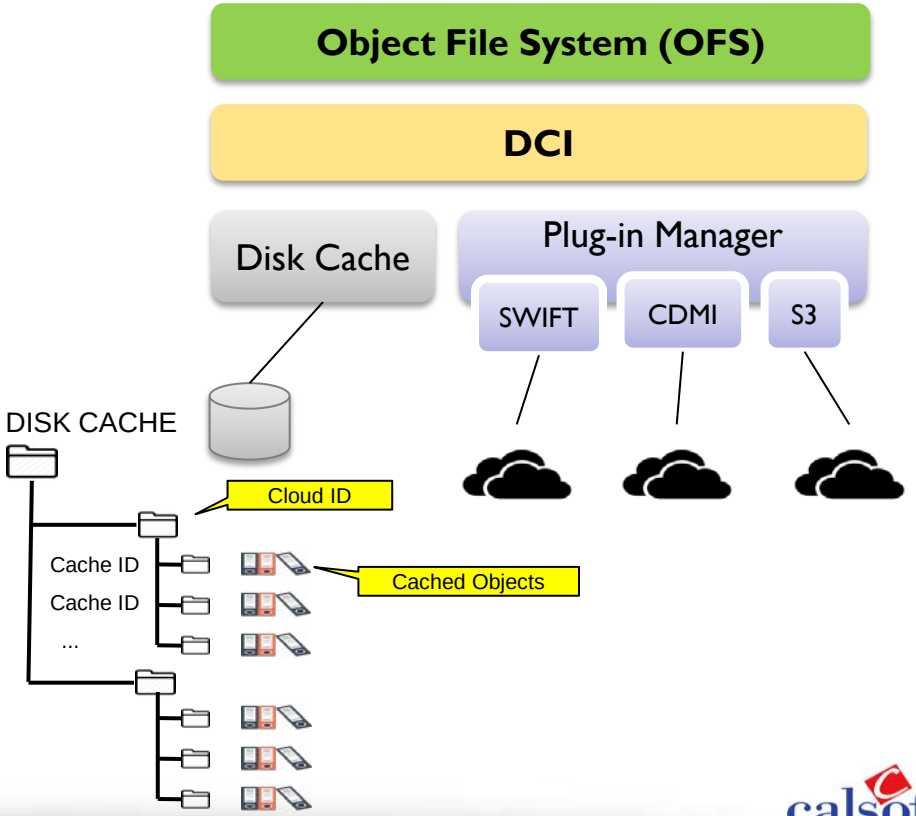
FedCFS Software Architecture



FedCFS Software Architecture (cont...)

Distributed Cloud Interface (DCI)

- Uniform Interfaces across clouds
- Plug-in based approach
- Object Cache
- Open-Close Semantics for accessing objects
- Support for Swift, S3 and CDMI
- Lease based synchronization
- Example Caching Interfaces
  - open_object
  - read_object
  - write_object
  - close_object
- Example Non-caching Interfaces
  - delete_object
  - put_object
  - get_object
  - set_object_metadata
  - get_object_metadata
  - create_container
  - delete_container
  - set_container_metadata
  - get_container_metadata



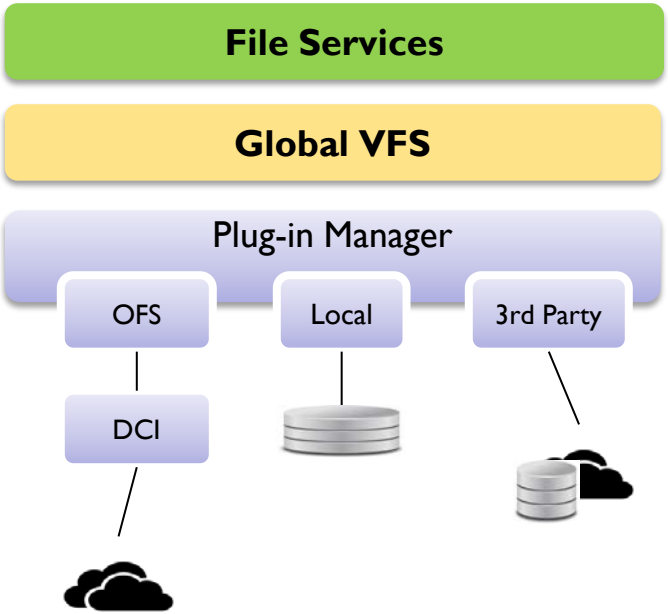
FedCFS Software Architecture (cont...)

Global VFS

- Backbone to file servers (Making it protocol independent)
- Providing access to data hosted on cloud / local disk
- Plug-in based (for extension to 3rd party storage)
- Plan to support server side replication, migration, snapshots

Object File System (OFS)

- File System on top of object storage from cloud
- File System APIs to easy integration info Applications
- Understands organization of data stored in cloud



Next Steps

- FedFS and FedCFS compatibility and co-existence
- Federation of Cloud data with Non-cloud data stored on the disk (local to file server)
- NSDB Protocol Enhancements
- Clustering (Share Management / Lock)
- File Services and Use Cases (e.g. Applications)
- Define and Standardize
  - Unified DCI APIs and Packages
  - Cloud Data Structure / Organization
  - Inter Cloud Interface (Replication or Migration)
  - Clustering of File Services (Distributed Applications)

References

- Glamor: An architecture for file system federation, IBM Journal of R&D, 2008, U. Lanjewar, M.Naik, R.Tewari, IBM
- Federated File System Overview, IETF ’77 NFSv4 WG, 2010, James Lentini, NetApp
- Federated Namespace BOF, USENIX FAST 2010, James Lentini, NetApp
- pNFS, NFSv4.1, FedFS and Future NFS Developments, 2013, Alex McDonald, NetApp
- FedFS Standards and Implementation, 2013, Chuck Lever, Oracle
- Introducing FedFS on Linux, SDC 2014, Chuck Lever, Oracle
- Cloud Data Management Interface (CDMI) v1.0, SNIA.org
- Object Storage API v1, developer.openstack.org
- Amazon Simple Storage Service, API Reference, docs.aws.amazon.com

Thank You

About Author

Ujjwal Lanjewar – Principal Architect, Calsoft Pvt. Ltd.

- Masters Degree in System Software from BITS PILANI, INDIA.
- A veteran of storage industry with 17+ years of experience in architecting and developing products for NAS, Global Namespace with FedFS, Distributed File Systems, Data Protection, Replication and Migration, etc.
- Working as Architect for designing enterprise scale products across stacks of different layers.

About Calsoft Pvt. Ltd.

- Product Development Services Company (in Storage / Virtualization) with 16 years of existence, headquerterd in Pune, India.
- Six out of top 10 storage companies work with Calsoft including NetApp, EMC, Cisco, etc.