Architecture of a Next-Generation Parallel File System

Agenda

- Introduction
- Whats in the code now
- futures

An introduction

What is OrangeFS?

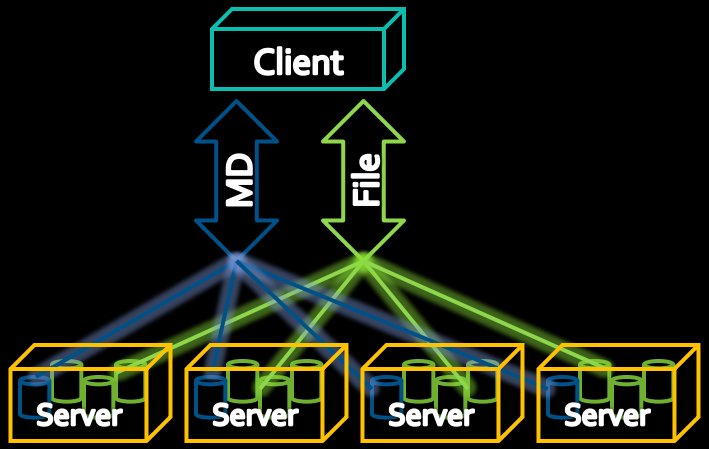
- OrangeFS is a next generation Parallel File System
  - Based on PVFS
  - Distributes file data across multiple file servers leveraging any block level file system.
  - Distributed Meta Data across 1 to all storage servers
  - Supports simultaneous access by multiple clients, including Windows using the PVFS protocol Directly
  - Works w/ standard kernel releases and does not require custom kernel patches
  - Easy to install and maintain
  - Happy member of the Linux Foundation

Why Parallel File System?

HPC – Data Intensive

- Large datasets
- Checkpointing
- Visualization
- Video
- BigData

Parallel (PVFS) Protocol



Unstructured Data Silos

- Unify Dispersed File Systems
- Simplify Storage Leveling

Interfaces to Match Problems

- Multidimensional arrays
- typed data
- portable formats

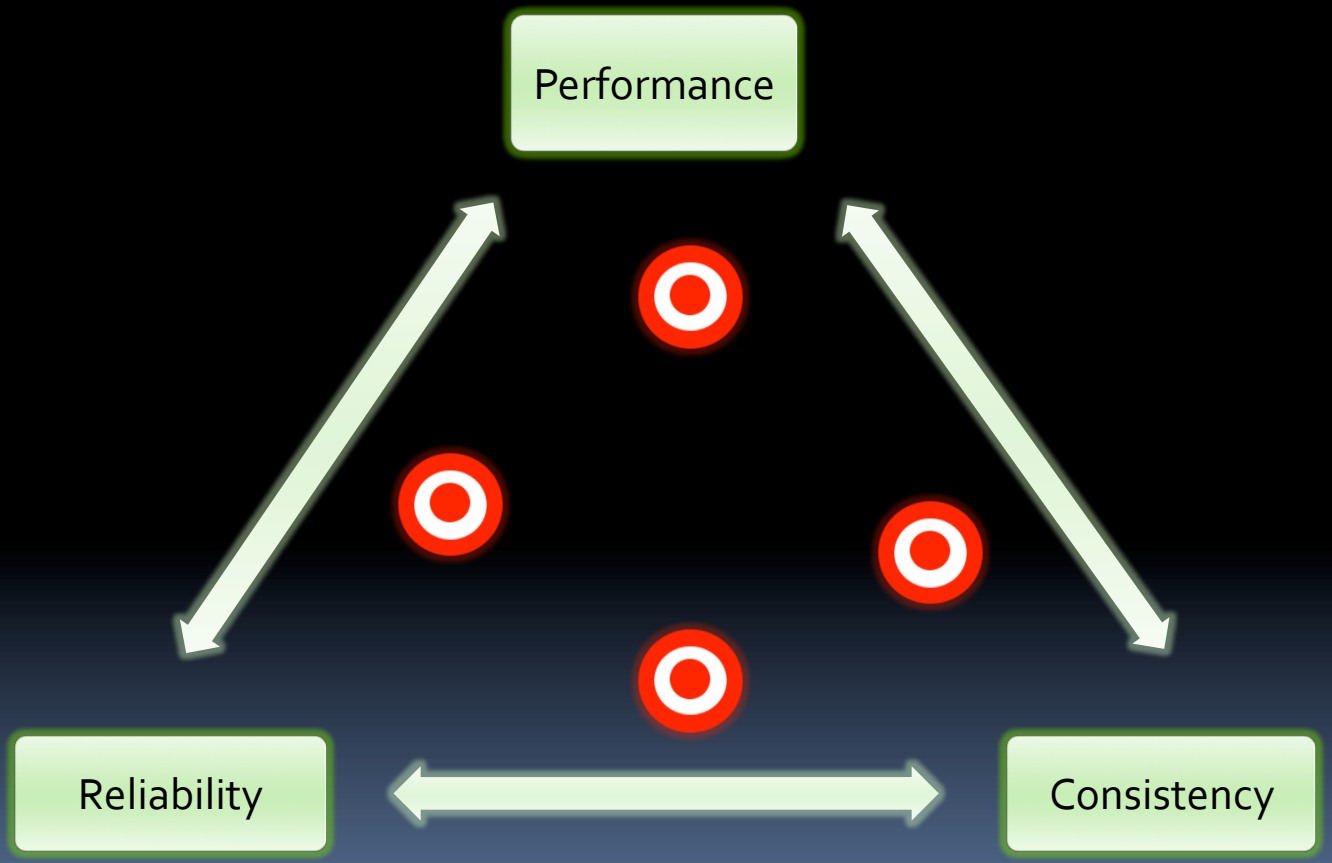
Original Design Goals

- Scalable
  - Configurable file striping
  - Non-contiguous I/O patterns
  - Eliminates bottlenecks in I/O path
  - Does not require locks for metadata ops
  - Does not need locks for non-conflicting applications
- Usability
  - Very easy to install, small VFS kernel driver
  - Modular design for disk, network, etc
  - Easy to extend -> Hundreds of Research Projects have used it, including dissertations, thesis, etc…

OrangeFS Philosophy

- Focus on a Broader Set of Applications
- Customer & Community Focused
  - (~300 Member Strong Community & Growing)
- Completely Open Source
  - (no closed source add-ons)
- Commercially Viable
- Enable Research

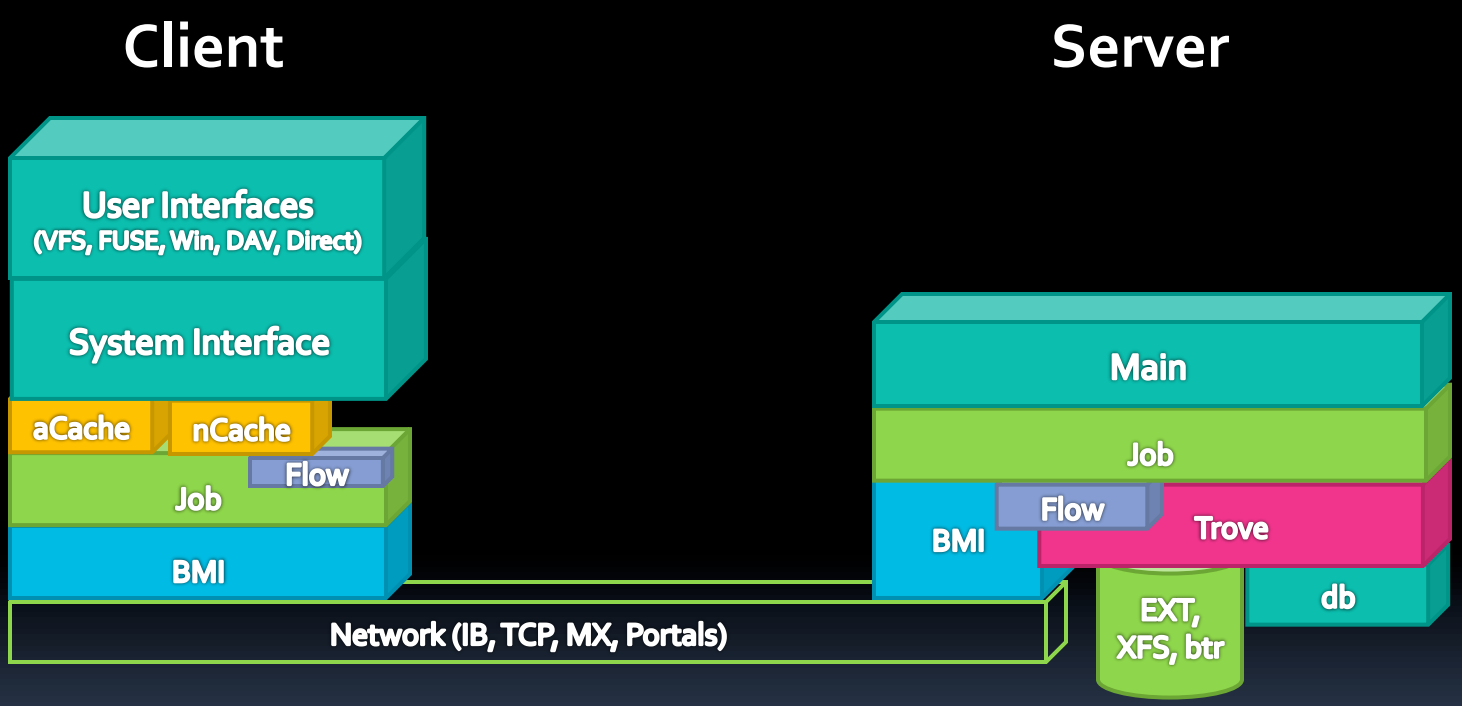
Configurability



System Architecture

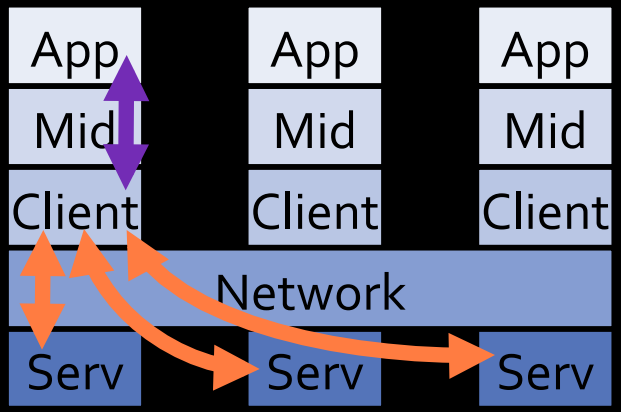
- OrangeFS servers manage objects
  - Objects map to a specific server
  - Objects store data or metadata
  - Request protocol specifies operations on one or more objects
- OrangeFS object implementation
  - Distributed DB for indexing key/value data
  - Local block file system for data stream of bytes

Current Architecture



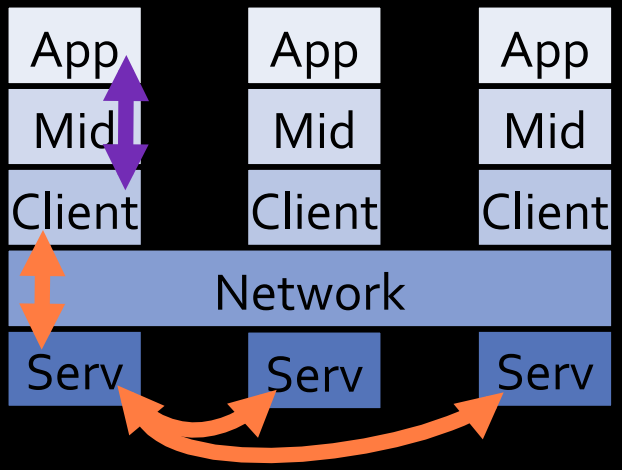
In the Code now

Server to Server Communications (2.8.5)



Traditional Metadata Operation

Create request causes client to communicate with all servers O(p)

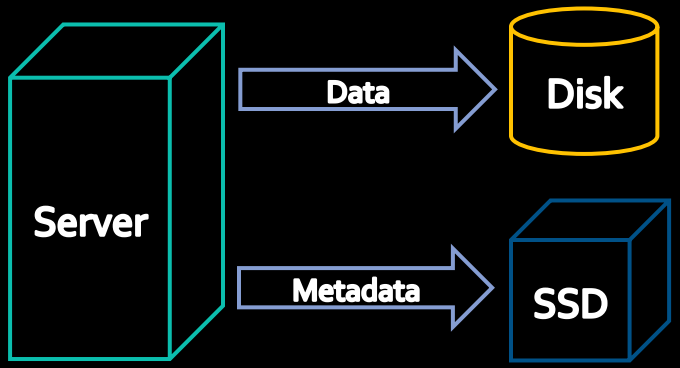


Scalable Metadata Operation

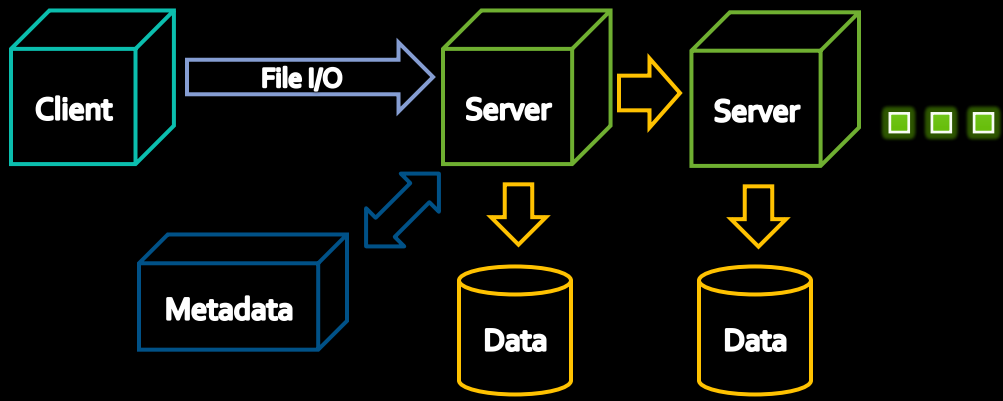
Create request communicates with a single server which in turn communicates with other servers using a tree-based protocol O(log p)

Recent Additions (2.8.5)

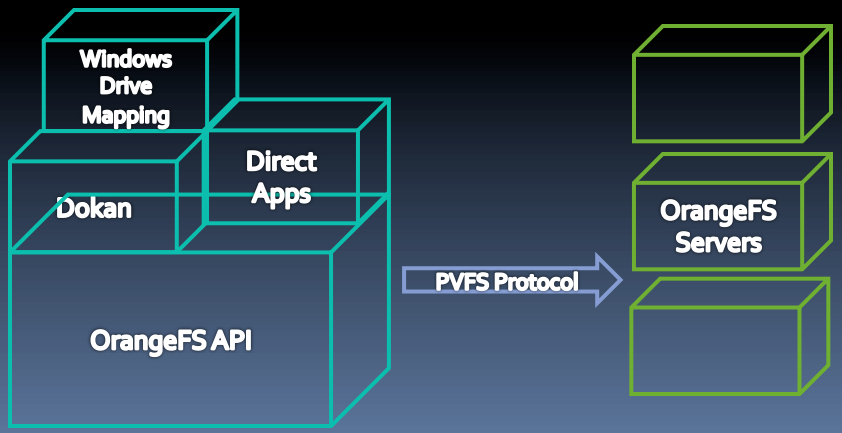
SSD Metadata Storage



Replicate on Immutable (file based)

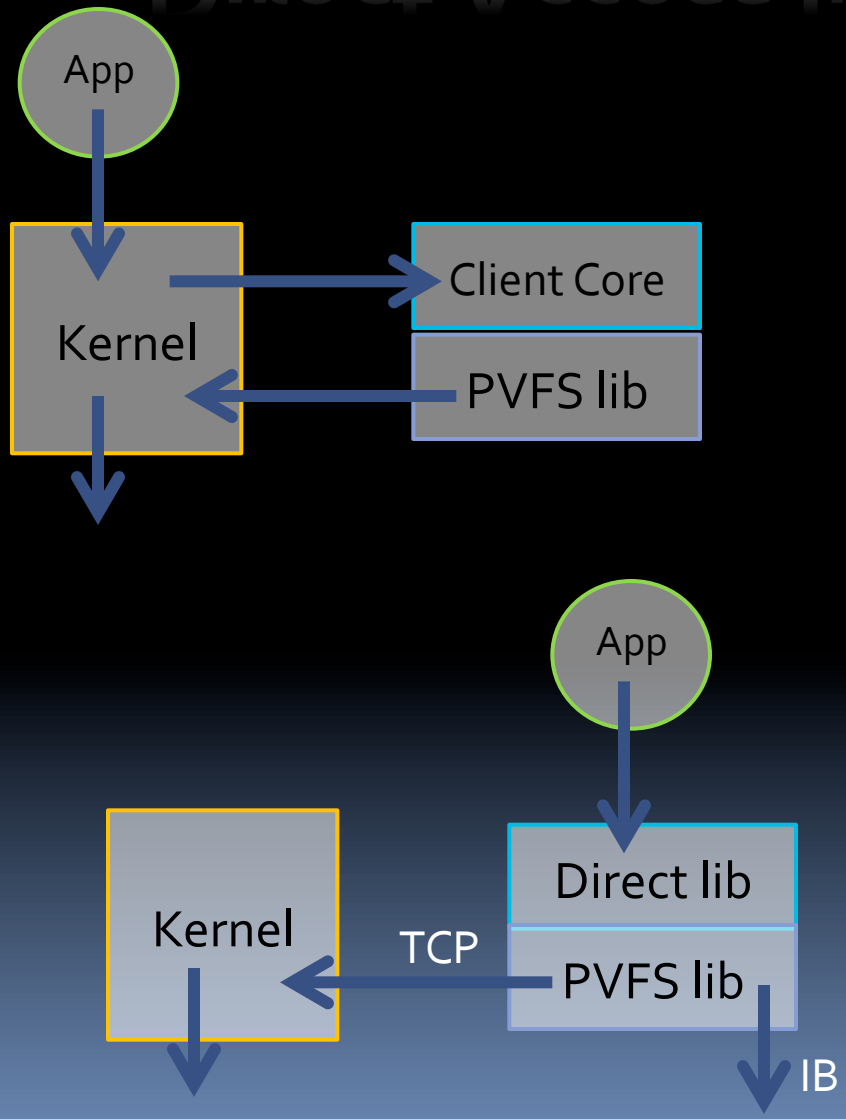


Windows Client



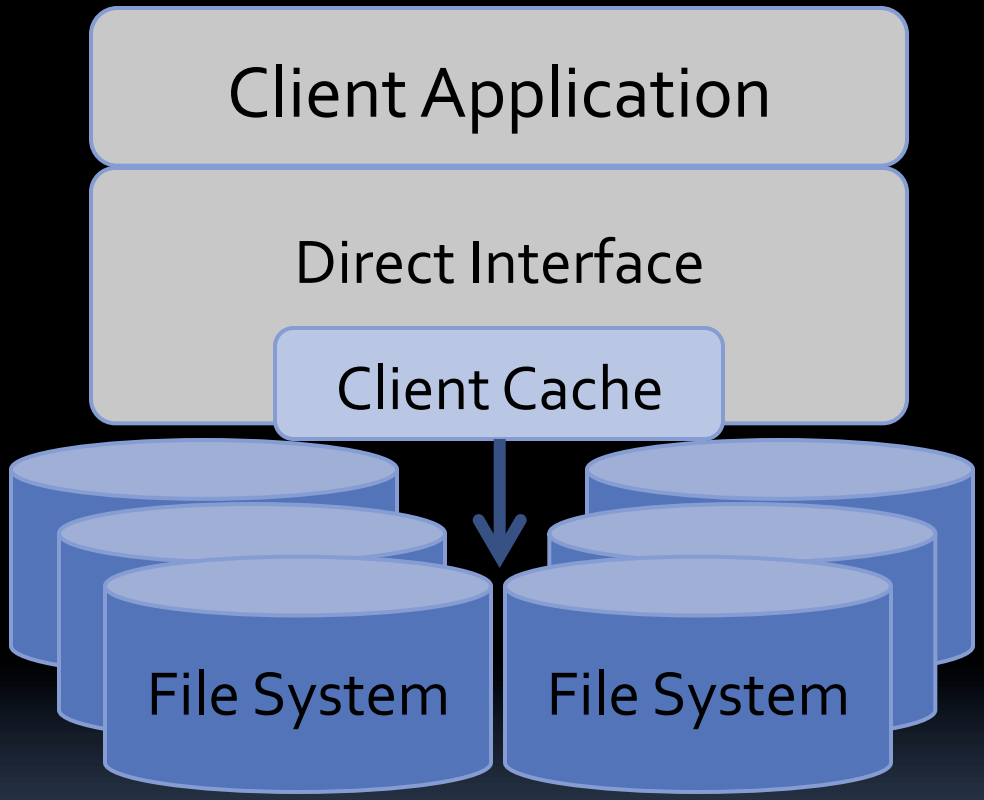
Supports Windows 32/64 bit Server 2008, R2, Vista, 7

Direct Access Interface (2.8.6)



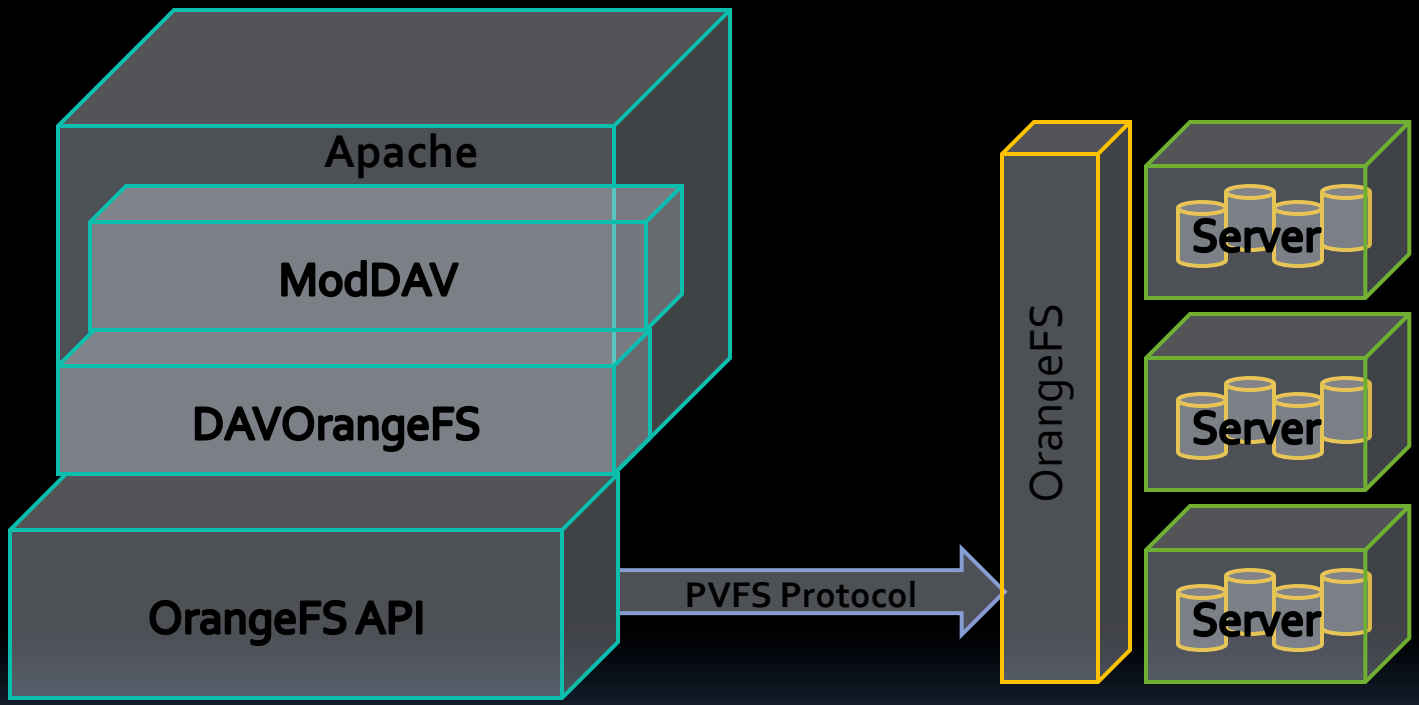
- Implements:
  - POSIX system calls
  - Stdio library calls
- Parallel extensions
  - Noncontiguous I/O
  - Non-blocking I/O
- MPI-IO library
- Found more boundary conditions fixed in upcoming 2.8.7

Direct Interface Client Caching (2.8.6)



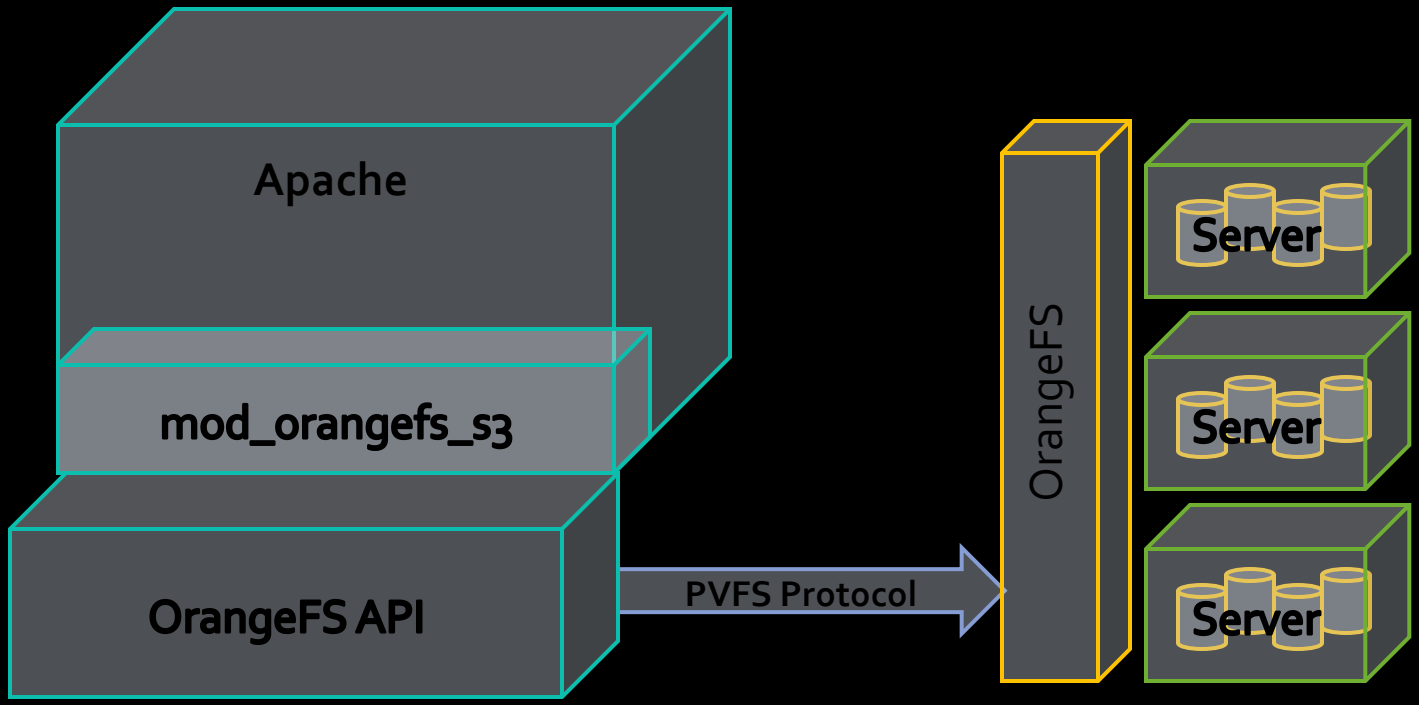
- Direct Interface enables Multi-Process Coherent Client Caching for a single client

WebDAV (2.8.6 webpack)



- Supports DAV protocol and tested with the Litmus DAV test suite
- Supports DAV cooperative locking in metadata

S3 (2.8.6 webpack)



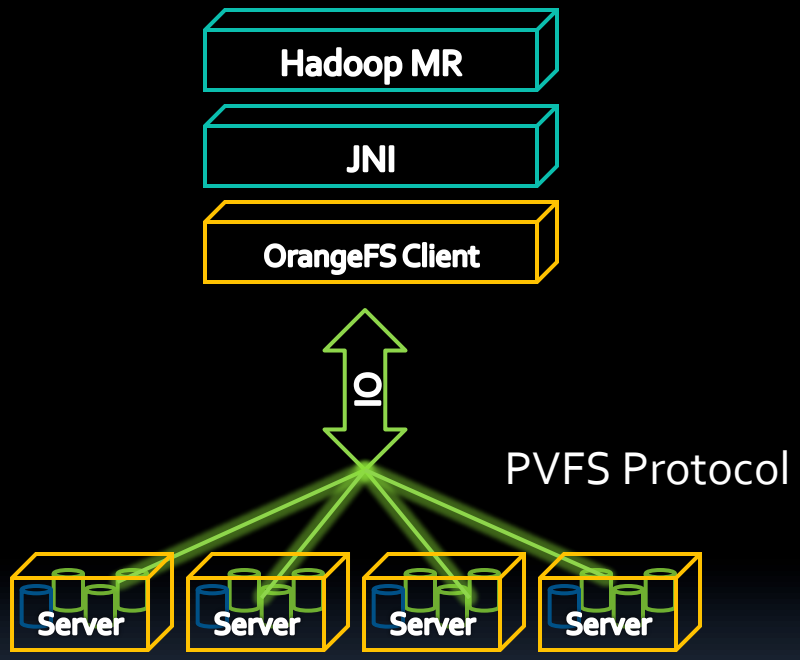
- Tested using s3cmd client
- Files accessible via other access methods
- Containers are Directories
- Accounting Pieces not implimented

Summary - Recently Added to OrangeFS

- In 2.8.3
  - Server-to-Server Communication
  - SSD Metadata Storage
  - Replicate on Immutable
- 2.8.4, 2.8.5 (fixes, support for newer kernels)
- Windows Client
- 2.8.6 – Performance, Fixes, IB updates
  - Direct Access Libraries (initial release)
    - preload library for applications, Including Optional Client Cache
  - Webpack
    - WebDAV (with file locking), S3

Coming Soon…

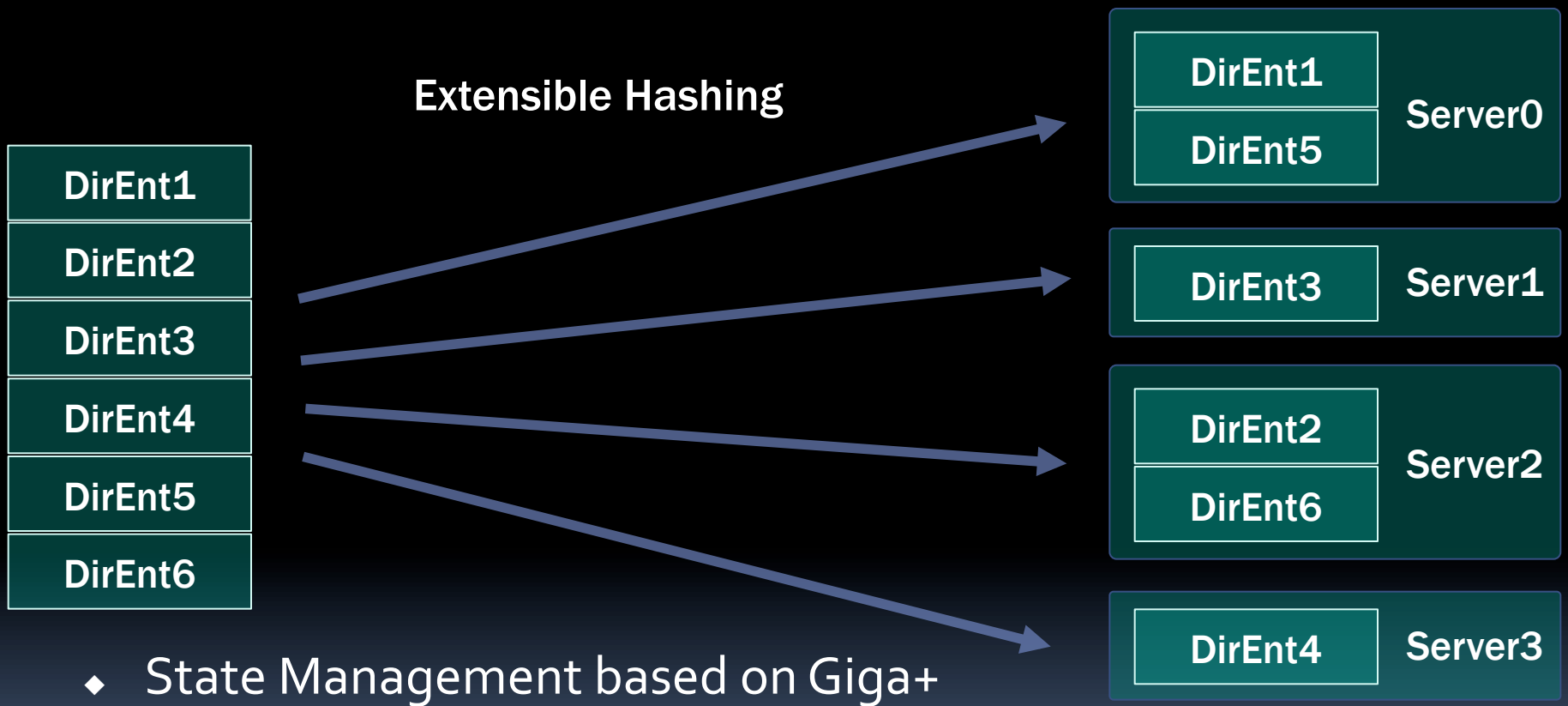
Hadoop JNI Interface (2.8.x)



- OrangeFS Java Native Interface
- Extension of Hadoop File System Class –>JNI
- Replicate on Immutable
- Buffering
- Distribution
- Based on work by

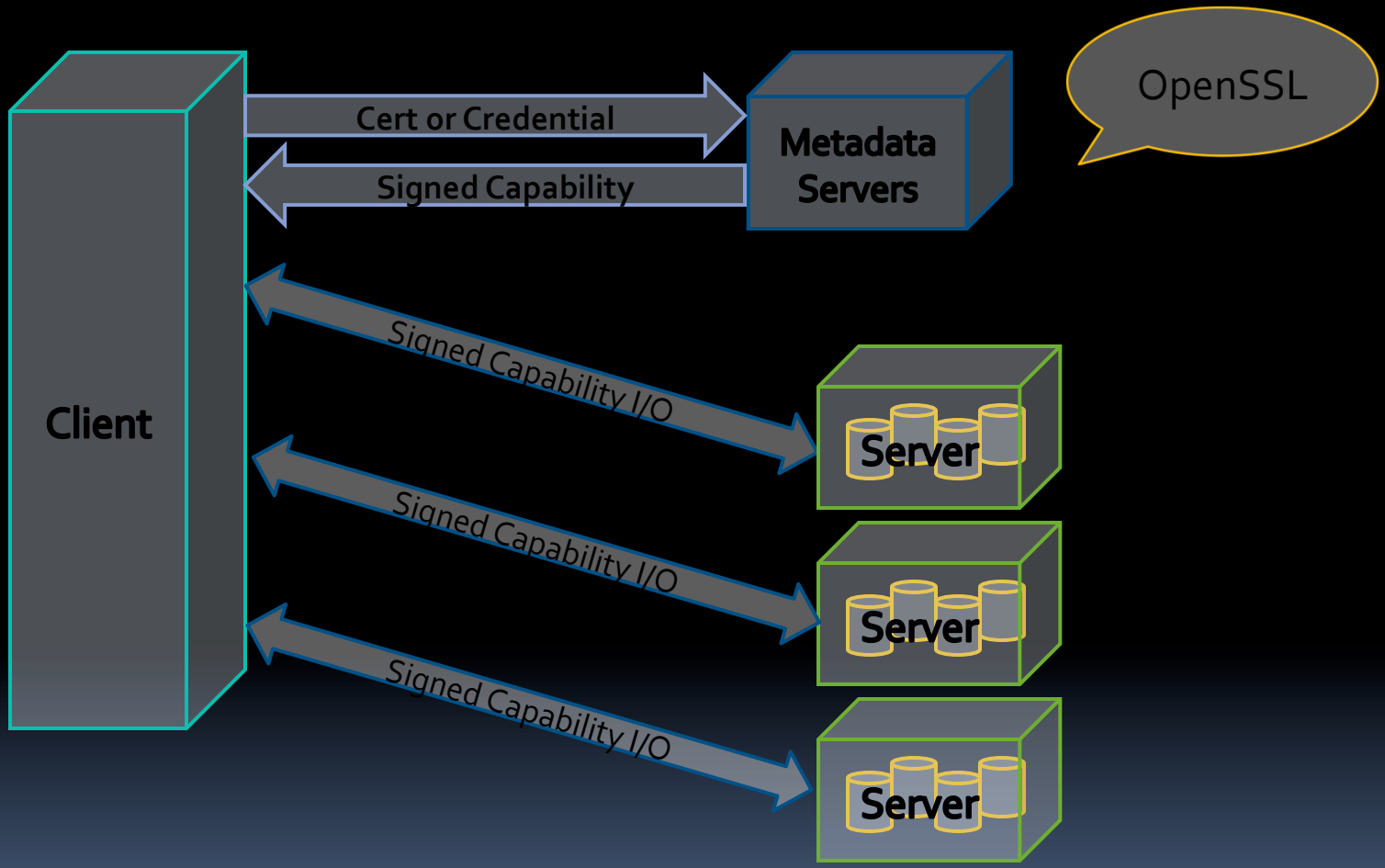
  Wittawat Tantisiriroj,
  Swapnil Patil,
  Garth Gibson, CMU

Distributed Directory Metadata (2.9.0)



- State Management based on Giga+
  - Swapnil Patil and Garth Gibson CMU
- Improves access times for directories with a very large number of entries
- Potential Extension to First Stripe of Data (10x+)

Capability Based Security (2.9.0)



- Untrusted Client support at the pvfs protocol level
- Working on how to simplify certificate integration
  - Make it so people will want to use

File Data Replication

- Forked Flow
  - small processing overhead
  - Bandwidth
- Configurable
  - Number of replicas (O .. N)
  - Update modes
    - Immutable (since 2.8.3)
    - Real Time (2.9.x)

Configurable Locking (2.8.x)

- File Locking already in WebDAV
- Bringing Locking concept into Client & KM
  - Fcntl / flock
  - Locks maintained across server in MD
- Initially support client and NFS locks through KM
- Eventually integrate with WebDAV locks
  (if semantics work out)

OrangeFS on the Cloud

- Build a Scale out Shared File System on top of Cloud Resources
- Cloud Formation Templates
  - AWS, EC2+EBS
  - OpenStack – Heat
  - Others as requested
- Spin up n number of storage servers, each with 8 or so EBS volumes and create a single large file system in Minutes.

On The Fence (2.9.x or 3.x)

Background Parallel Processing Infrastructure

(2.9.x)

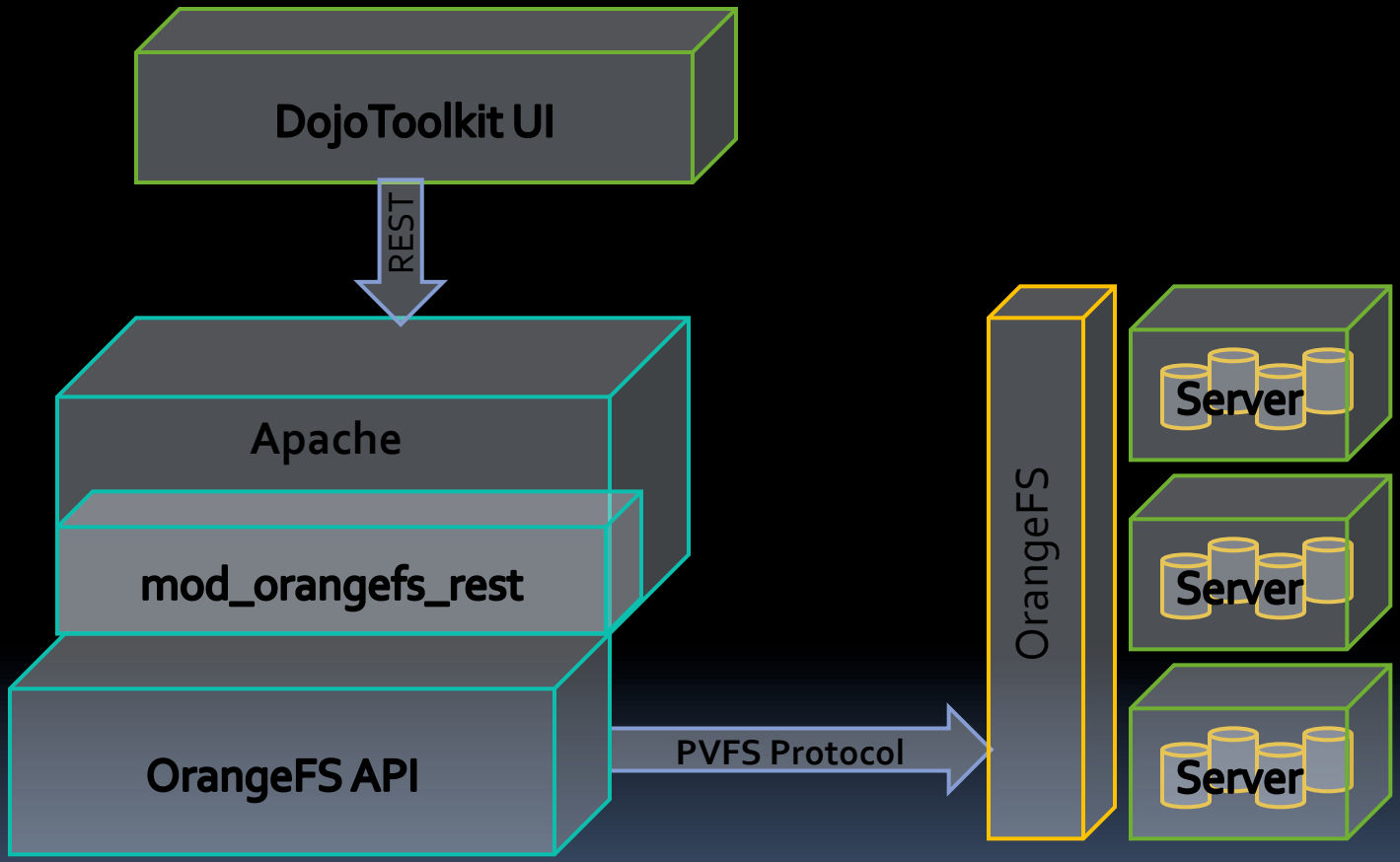
- Modular infrastructure to easily build background parallel processes for the file system

  Used for:

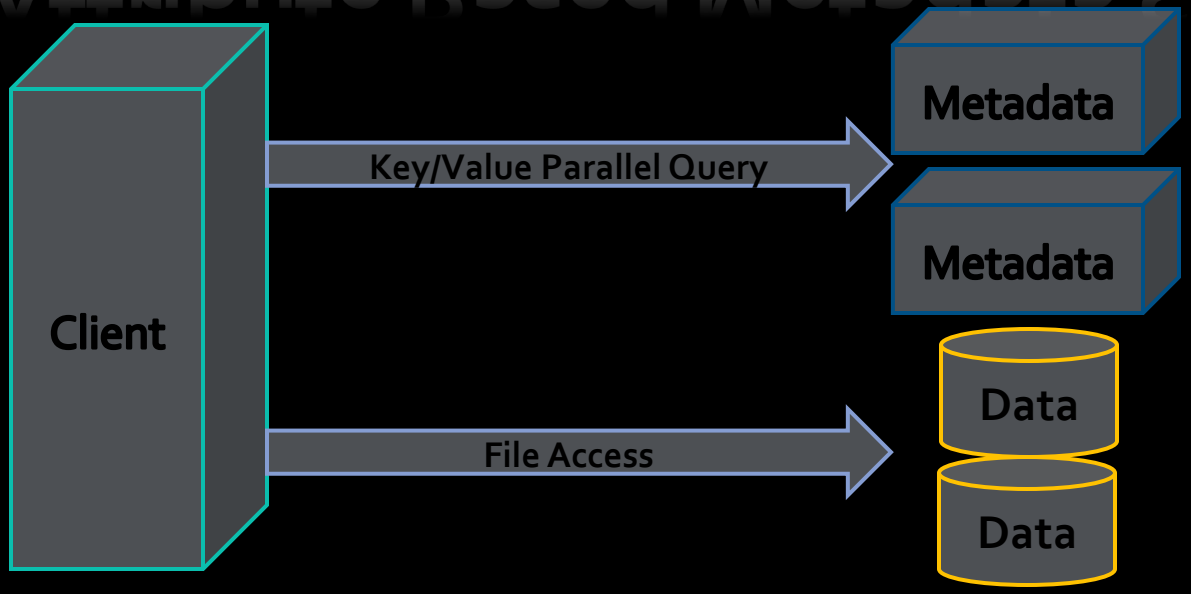
  - Gathering Stats for Monitoring
  - Usage Calculation (can be leveraged for Directory Space Restrictions, chargebacks)
  - Background safe FSCK processing (can mark bad items in MD)
  - Background Checksum comparisons
  - Parallel Cross File System rsync
  - Etc…

Admin REST interface / Admin UI

(2.x.x?)



Attribute Based Metadata Search



- Client tags files with Keys/Values
- Keys/Values indexed on Metadata Servers
- Clients query for files based on Keys/Values
- Returns file handles with options for filename and path

OrangeFS 3

OrangeFS 3

- Goals
  - Scalability
    - to tens of thousands of storage nodes
  - Security
    - support systems with untrusted root as clients
  - Fault Accepting Architecture
  - Diverse Access Methods
  - Monitoring and Management
  - Storage Tiers
  - Wide Area Capabilities

Replication / Redundancy

- Redundant Metadata
  - seamless recovery after a failure
  - Redundant objects from root directory down
  - Configurable
  - Distributed Master Objects
- Fully Redundant Data
  - Extension of 2.9.x work
  - Configurable
    - Number of replicas (O .. N)
    - Update modes (continuous, on close, on immutable, none)
- Emphasis on continuous operation
  - Fault Accepting Architecture

Policy Based Location

- User defined attributes for servers and clients
  - Stored in SID cache
- Policy is used for data location, replication location and multi-tenant support
- Completely Flexible
  - Rack
  - Row
  - App
  - Region

Server Location / SID Mgt

- In an Exascale environment
  - thousands of I/O servers
  - Servers cannot know about all other servers.
- Servers will discover a subset of their neighbors
  - At Startup or Cached
  - Servers will learn about unknown servers on an as needed basis and cache them.
    - Similar to DNS query mechanisms (root servers, authoritative domain servers).
- SID Cache, in memory db to store server attributes

Handles -> UUIDs

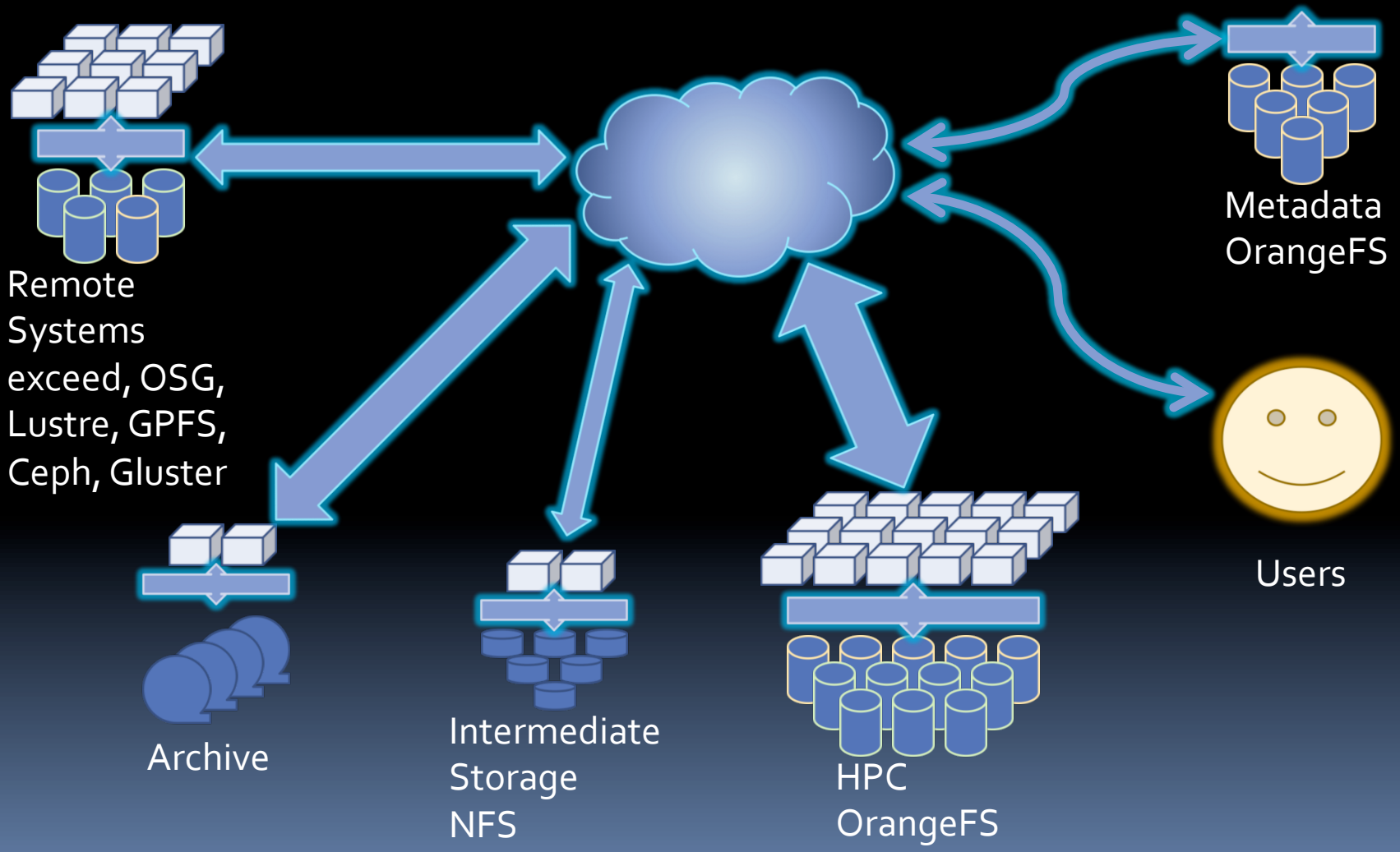
- An OID (object identifier)
  - 128-bit UUID that is unique to the data-space
- An SID (server identifier)
  - 128-bit UUID that is unique to each server.
- No more than one copy of a given data-space can exist on any server
- The (OID, SID) tuple is unique within the file system.
- (OID, SID1), (OID, SID2), (OID, SID3) are copies of the object identifier on different servers.

Data Migration / Mgt

- Built on Redundancy & DBG processes
- Migrate objects between servers
  - De-populate a server going out of service
  - Populate a newly activated server (HW lifecycle)
  - Moving computation to data
- Hierarchical storage
  - Use existing metadata services
- Possible - Directory Hierarchy Cloning
  - Copy on Write (Dev, QA, Prod environments with high % data overlap)

Hierarchical Data Management

OrangeFS 3.x



Beyond OrangeFS 3

Extend Capability based security

- Enable certificate level access (in process)
- Federated access capable
- Can be integrated with rules based access control
  - Department x in company y can share with Department q in company z
    - rules and roles establish the relationship
    - Each company manages their own control of who is in the company and in department

SDN - OpenFlow

- Working with OpenFlow researchers at Clemson
- OF separates the control plane from delivery, gives ability to control network with SW
- Looking at bandwidth optimization leveraging OF and OrangeFS

ParalleX

ParalleX is a new parallel execution model

- Key components are:
  - Asynchronous Global Address Space (AGAS)
  - Threads
  - Parcels (message driven instead of message passing)
  - Locality
  - Percolation
  - Synchronization primitives
  - High Performance ParalleX (HPX) library implementation written in C++

PXFS

- Parallel I/O for ParalleX based on PVFS
  - Common themes with OrangeFS 3
- Primary objective: unification of ParalleX and storage name spaces.
  - Integration of AGAS and storage metadata subsystems
  - Persistent object model
- Extends ParalleX with a number of IO concepts
  - Replication
  - Metadata
- Extending IO with ParalleX concepts
  - Moving work to data
  - Local synchronization
- Effort with LSU, Clemson, and Indiana U.
  - Walt Ligon, Thomas Sterling

Community

Learning More

- www.orangefs.org web site
  - Releases
  - Documentation
  - Wiki
- Dell OrangeFS Reference Architecture Whitepaper
- pvfs2-users@beowulf-underground.org
  - Support for users
- pvfs2-developers@beowulf-underground.org
  - Support for developers

Support & Development Services

- www.orangefs.com & www.omnibond.com
  - Professional Support & Development team
  - Buy into the project


OrangeFS
project