iRODS®

Research Community Requirements Drive Expanded Scale Data Management Features

Terrell Russell, Ph.D.
@terrellrussell
Executive Director, iRODS Consortium

January 24-27, 2022
CS3 2022
Virtual

iRODS

CONSORTIUM

renci

RESEARCH ENGAGEMENT INNOVATION

THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

The iRODS Consortium

Our Mission

- Continuous Improvement
- Grow the Community
- Standardization
- Show value to our Membership

Our Membership

BAYER

ddn

Bibliothèque et Archives nationales Québec

Maastricht University

SOFTIRON

Western Digital

Consortium Member

Universiteit Utrecht

Consortium Member

UCL

wellcome sanger institute

renci

Consortium Member

BIH Berlin Institute of Health
Charité & MDC

Research Computing
UNIVERSITY OF COLORADO **BOULDER**

INES

AGRICULTURE VICTORIA

TACC
TEXAS ADVANCED COMPUTING CENTER

Consortium Member

university of groningen

Consortium Member

SURF

KU LEUVEN

emagine it

OpenIO

NetApp

NIH
National Institute of Environmental Health Sciences

Research Community Requirements Drive Features

- **History**
- Philosophy
- Process
- Results

History

- 1995 - SRB started (grid storage)
- 2004 - iRODS started (added rule engine / policy layer)
- 2013 - Consortium founded by RENCI, DICE, and DDN
- 2014 - Consortium accepted the code base
- 43 releases of iRODS to date

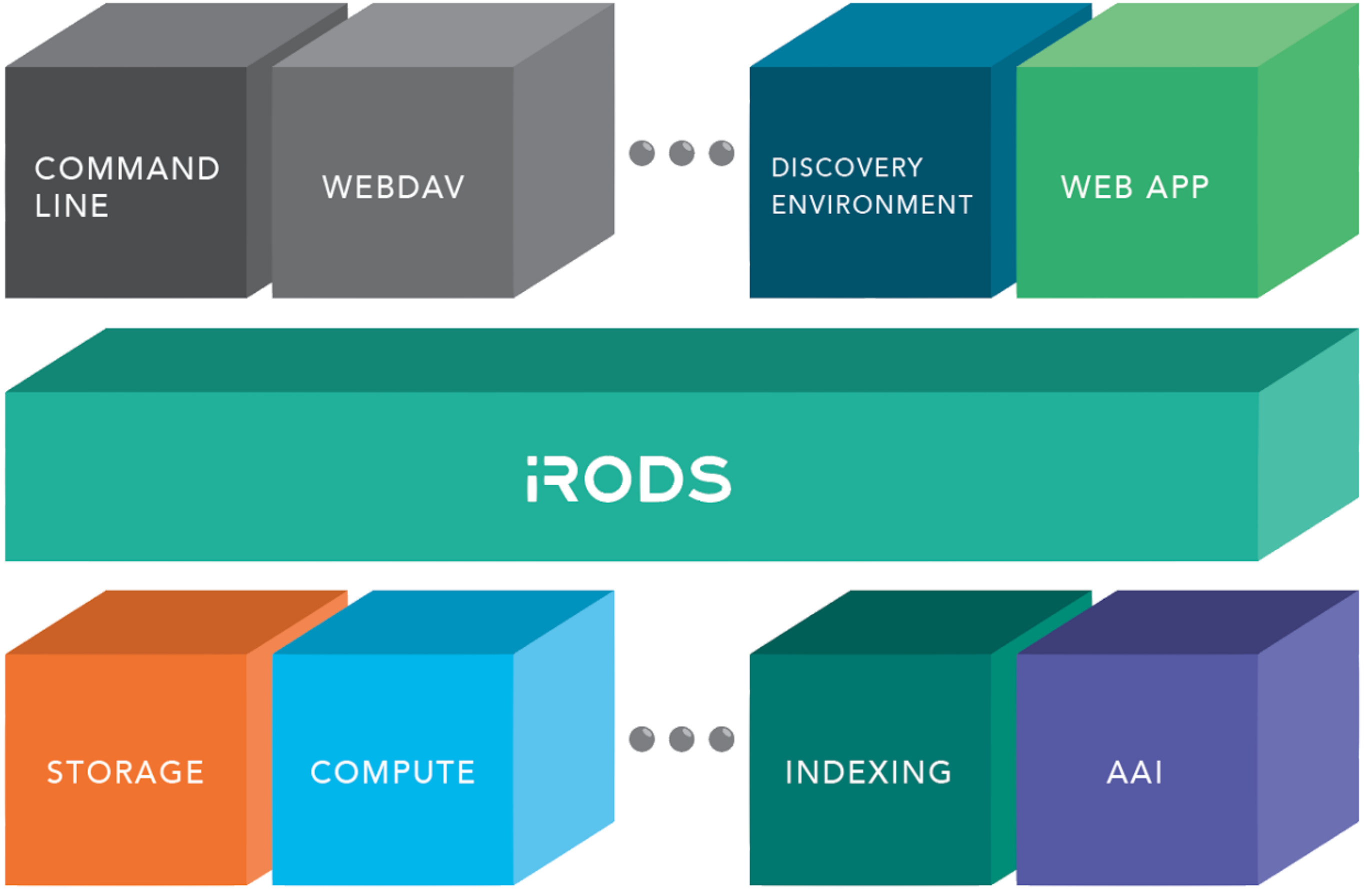
Why use iRODS?

People need a solution for:

- Managing large amounts of data across various storage technologies
- Controlling access to data
- Searching their data quickly and efficiently
- Automation

The larger the organization, the more they need software like iRODS.

iRODS as the Integration Layer



iRODS Clients

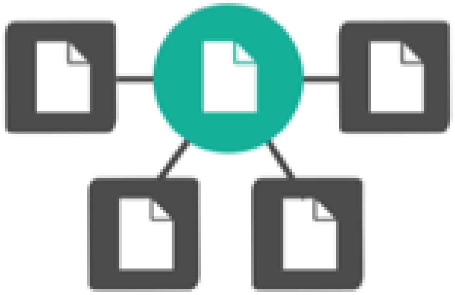
iRODS provides a layer of abstraction which integrates with your pre-existing infrastructure.

This flexibility allows your infrastructure to continue to change over time.

Existing Infrastructure

iRODS Core Competencies

DATA VIRTUALIZATION



DATA DISCOVERY

WORKFLOW AUTOMATION

SECURE COLLABORATION

iRODS Capabilities

- Packaged and supported solutions
- Require configuration not code
- Derived from the majority of use cases observed in the user community

- Storage Tiering
- Auditing
- Provenance
- Data Integrity
- Automated Ingest
- Indexing
- Compliance
- Publishing

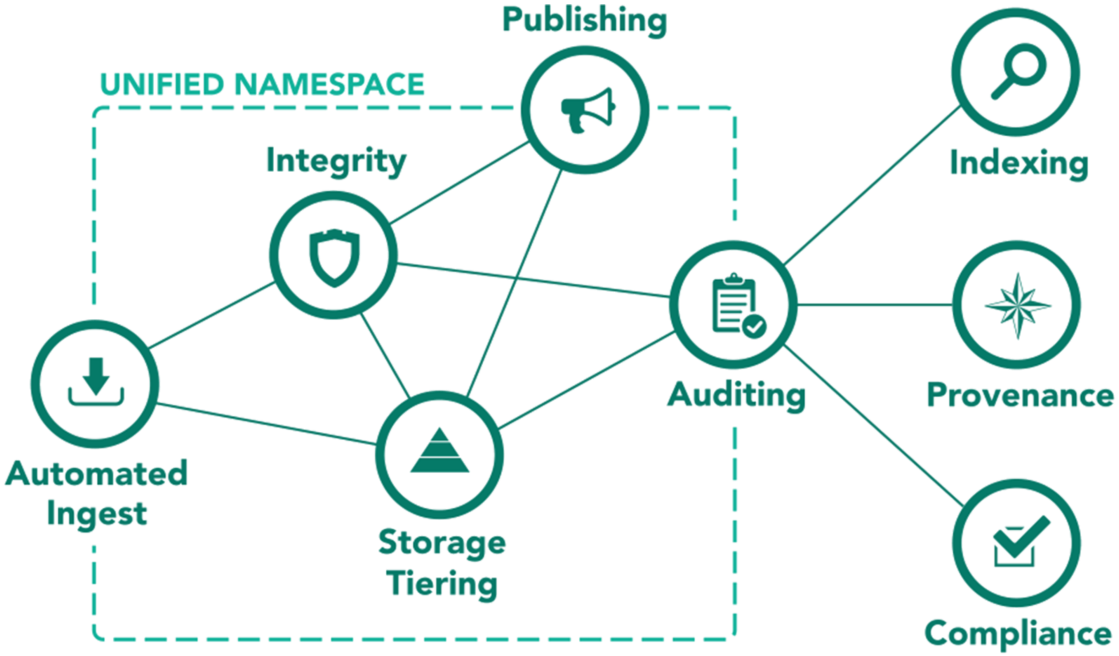
The Data Management Model

iRODS provides eight packaged capabilities, each of which can be selectively deployed and configured.

These capabilities represent the most common use cases as identified by community participation and reporting.

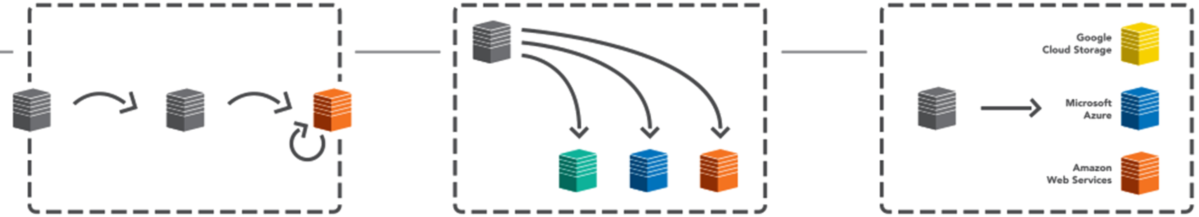
The flexibility provided by this model allows an organization to address its immediate use cases.

Additional capabilities may be deployed as any new requirements arise.



A pattern represents a combination of iRODS capabilities and data management policy consistent across multiple organizations.

Three common patterns of iRODS deployment have been observed within the community:



Data to Compute

Compute to Data

Synchronization

Research Community Requirements Drive Features

- History
- **Philosophy**
- Process
- Results

Philosophical Drivers

- 100-year view

- Plugin Architecture
  - core is generic - protocol, api, bookkeeping
  - plugins are specific
  - policy composition

- Modern core libraries
  - standardized interfaces
  - refactored iRODS internals
    - ease of (re)use
    - fewer bugs

- Configuration, Not Code

Research Community Requirements Drive Features

- History
- Philosophy
- **Process**
  - Google Group
  - GitHub
  - Working Groups
- Results

Working Groups

Technology Working Group

- Goal: To keep everyone up to date, provide a forum for roadmap discussion and collaboration opportunities

Metadata Templates Working Group

- Goal: To define a standardized process for the application and management of metadata templates by the iRODS Server
    - NIEHS / Data Commons
    - Utrecht / Yoda
    - Maastricht / DataHub+
    - Arizona / CyVerse

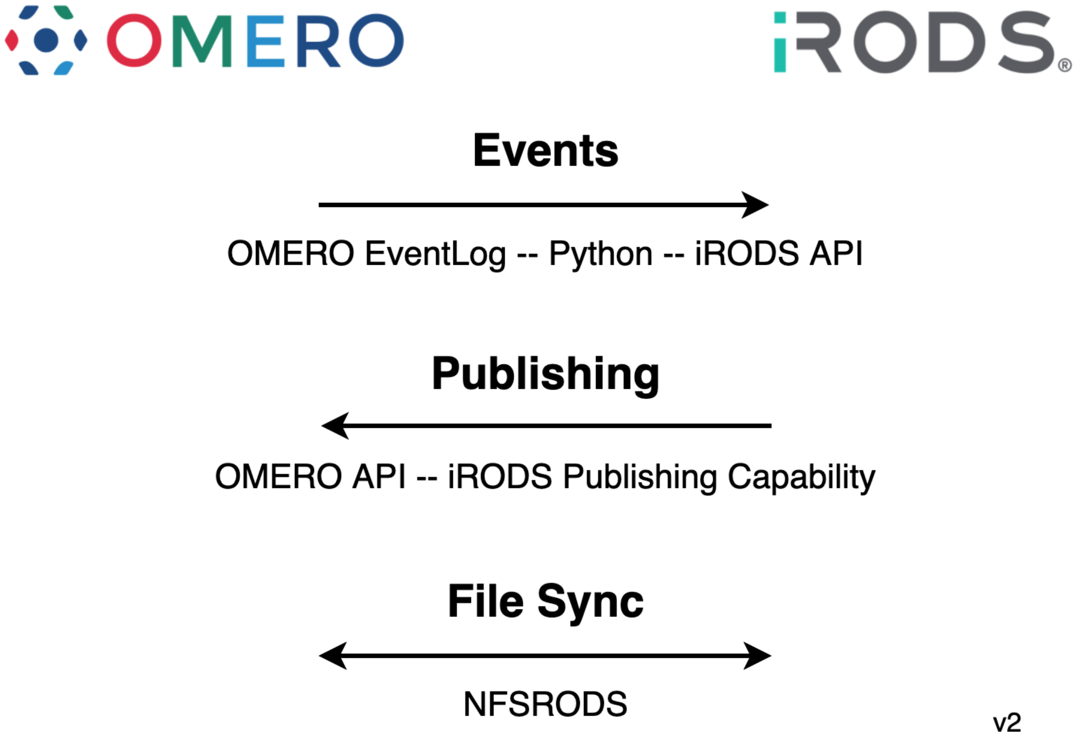
Authentication Working Group

- Goal: To provide a more flexible authentication mechanism to the iRODS Server
    - SURF
    - NIEHS
    - Sanger
    - CyVerse
    - Utrecht

Working Groups

Imaging Working Group

- Goal: To provide a standardized suite of imaging policies and practices for integration with existing tools and pipelines
  - Open Microscopy Environment (and OMERO)
  - Neuroscience Microscopy Core at UNC School of Medicine
  - New York University
  - Santa Clara University
  - UC San Diego
  - UC Santa Cruz
  - UMass
  - Harvard
  - Maastricht University
  - Wellcome Sanger Institute
  - CyVerse
  - NIEHS
  - Netherlands Cancer Institute (NKI)
  - Francis Crick Institute
  - Fritz Lipmann Institute
  - Osnabrück University
  - RIKEN



Events

OMERO EventLog -- Python -- iRODS API

Publishing

OMERO API -- iRODS Publishing Capability

File Sync

NFSRODS

v2

Research Community Requirements Drive Features

- History
- Philosophy
- Process
- **Results**

In The Last Year

Plugins

- Python Rule Engine Plugin
- Storage Tiering Rule Engine Plugin
- Auditing (AMQP) Rule Engine Plugin
- S3 Resource Plugin
- Kerberos Authentication Plugin
- Curl Microservice Plugin
- Indexing Rule Engine Plugin
- Logical Quotas Rule Engine Plugin
- Metadata Guard Rule Engine Plugin
- **Policy Composition Framework**
- **Policy Composition Event Handlers**

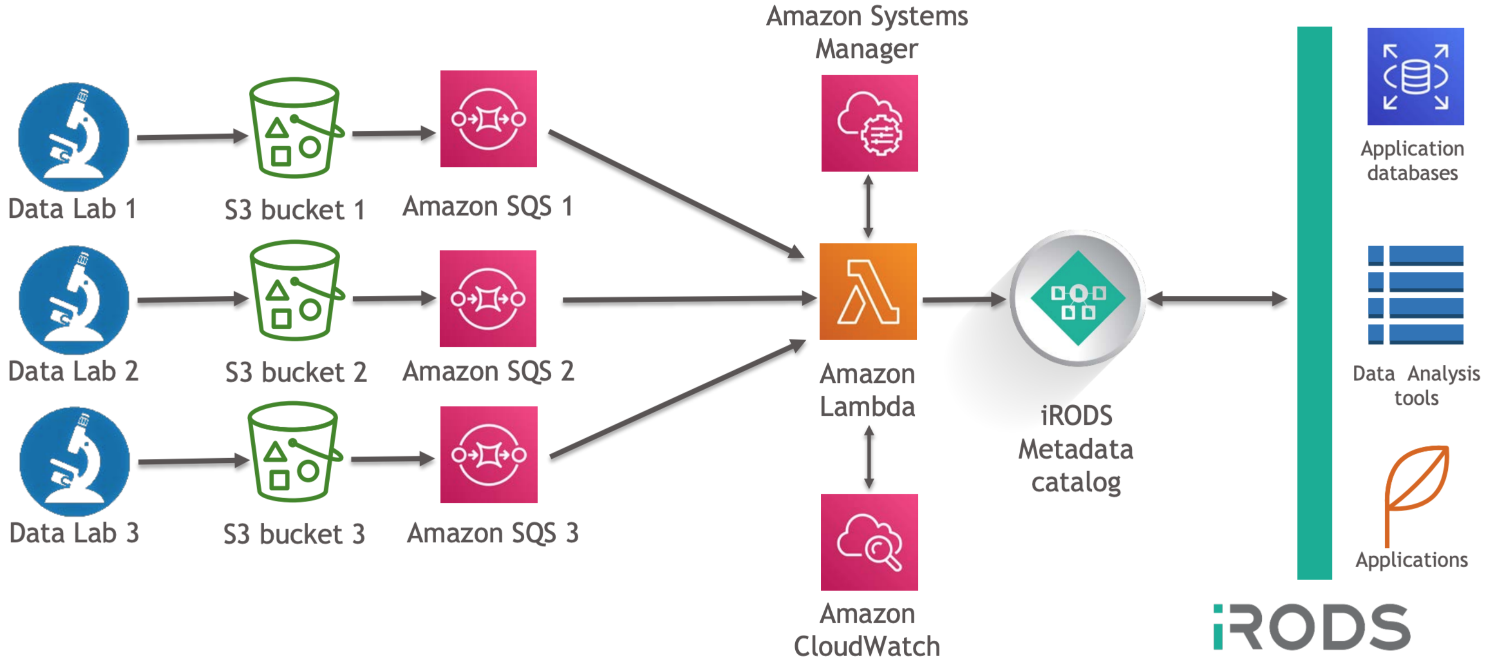
Clients

- **Python iRODS Client**
- **Metalnx**
- **NFSRODS**
- Automated Ingest Framework
- AWS Lambda for S3
- **C++ REST API**
- **Zone Management Tool (ZMT)**
- **iRODS Globus Connector**
- **iCommands**

AWS Lambda for S3

- iRODS Client
- Developed in collaboration with BMS

Bristol Myers Squibb™

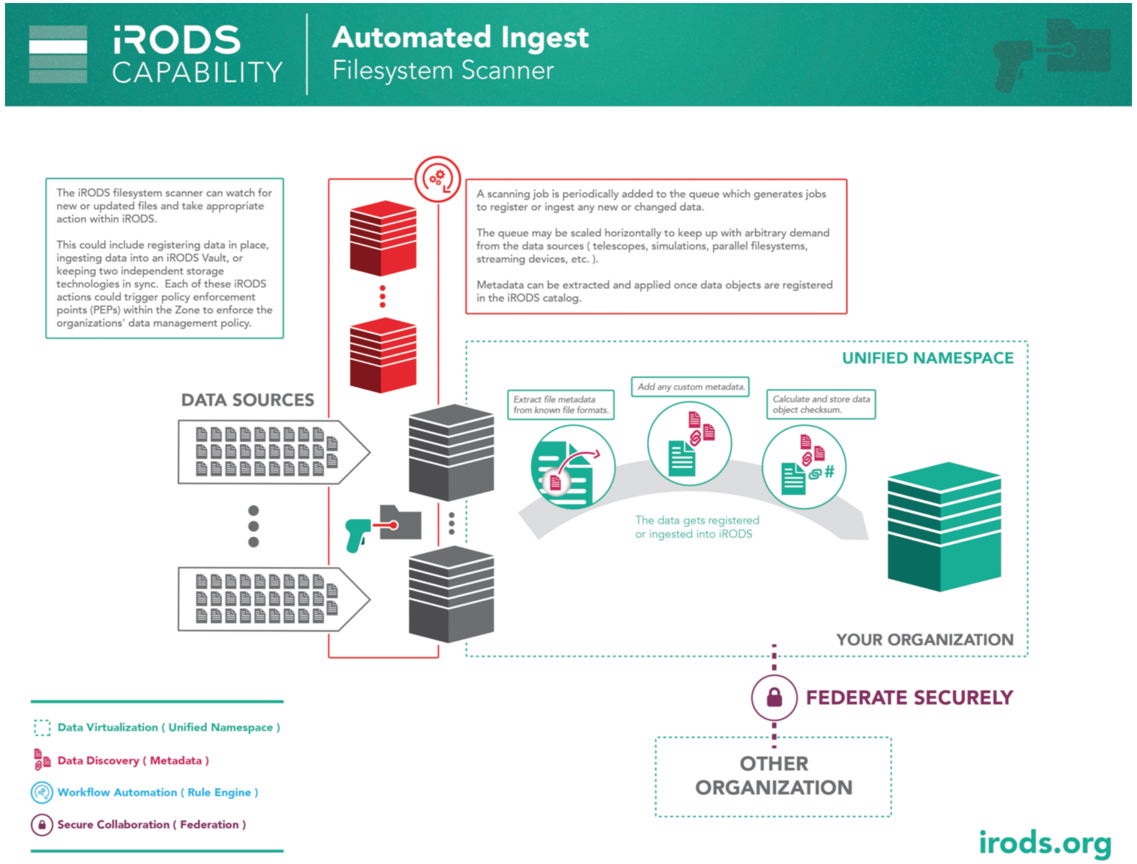


Open Source:

https://github.com/irods/irods_client_aws_lambda_s3

Automated Ingest Capability

- iRODS Capability
- Developed in collaboration with Roche



The iRODS filesystem scanner can watch for new or updated files and take appropriate action within iRODS.

This could include registering data in place, ingesting data into an iRODS Vault, or keeping two independent storage technologies in sync. Each of these iRODS actions could trigger policy enforcement points (PEPs) within the Zone to enforce the organizations' data management policy.

A scanning job is periodically added to the queue which generates jobs to register or ingest any new or changed data.

The queue may be scaled horizontally to keep up with arbitrary demand from the data sources (telescopes, simulations, parallel filesystems, streaming devices, etc.).

Metadata can be extracted and applied once data objects are registered in the iRODS catalog.

DATA SOURCES
UNIFIED NAMESPACE
Extract file metadata from known file formats.
Add any custom metadata.
Calculate and store data object checksum.
The data gets registered or ingested into iRODS
YOUR ORGANIZATION
FEDERATE SECURELY
OTHER ORGANIZATION
Data Virtualization (Unified Namespace)
Data Discovery (Metadata)
Workflow Automation (Rule Engine)
Secure Collaboration (Federation)
irods.org

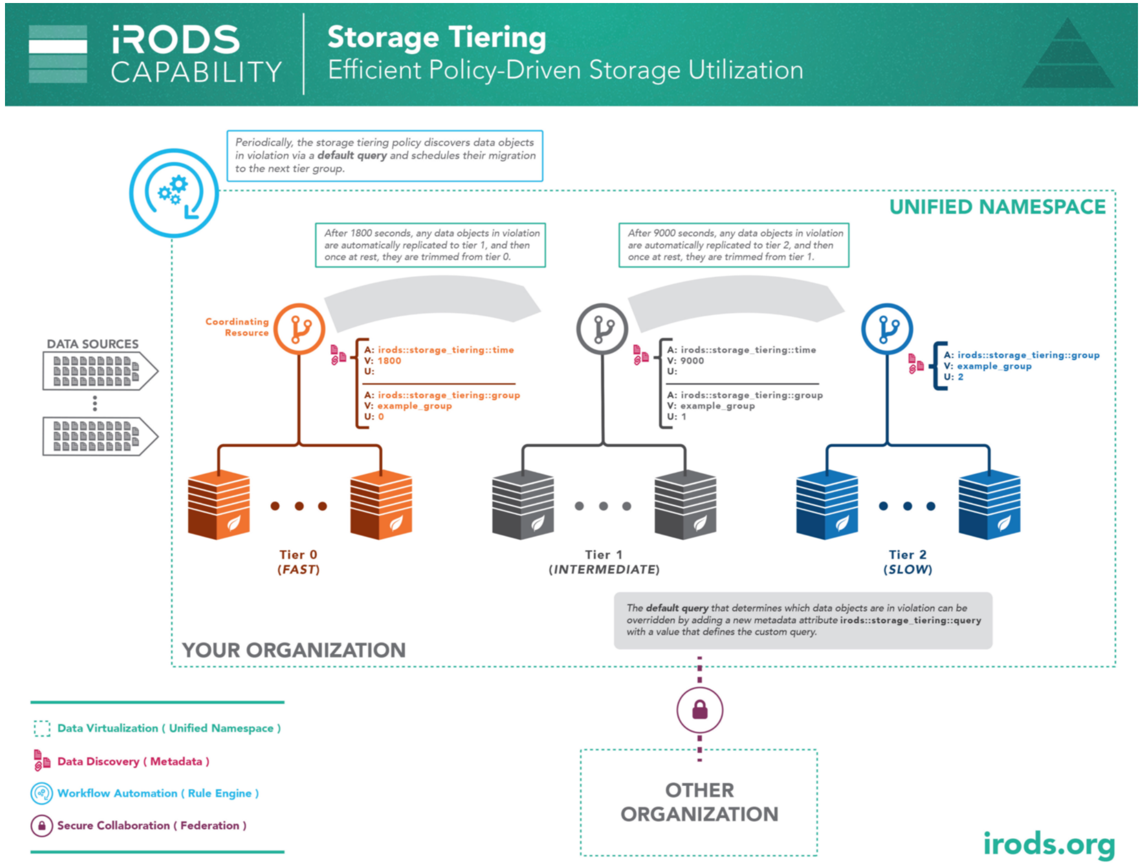
Open Source:

https://github.com/irods/irods_capability_automated_ingest

Storage Tiering Capability

- iRODS Capability
- Developed in collaboration with Roche

Roche



iRODS CAPABILITY

Storage Tiering

Efficient Policy-Driven Storage Utilization

Periodically, the storage tiering policy discovers data objects in violation via a ***default query*** *and schedules their migration to the next tier group.*

After 1800 seconds, any data objects in violation are automatically replicated to tier 1, and then once at rest, they are trimmed from tier 0.

After 9000 seconds, any data objects in violation are automatically replicated to tier 2, and then once at rest, they are trimmed from tier 1.

The ***default query*** *that determines which data objects are in violation can be overridden by adding a new metadata attribute* **irods::storage_tiering::query** *with a value that defines the custom query.*

UNIFIED NAMESPACE
DATA SOURCES
Coordinating Resource
A: irods::storage_tiering::time
V: 1800
U:
A: irods::storage_tiering::group
V: example_group
U: 0
A: irods::storage_tiering::time
V: 9000
U:
A: irods::storage_tiering::group
V: example_group
U: 1
A: irods::storage_tiering::group
V: example_group
U: 2
Tier 0
(FAST)
Tier 1
(INTERMEDIATE)
Tier 2
(SLOW)
YOUR ORGANIZATION
OTHER ORGANIZATION
Data Virtualization (Unified Namespace)
Data Discovery (Metadata)
Workflow Automation (Rule Engine)
Secure Collaboration (Federation)
irods.org

Open Source:

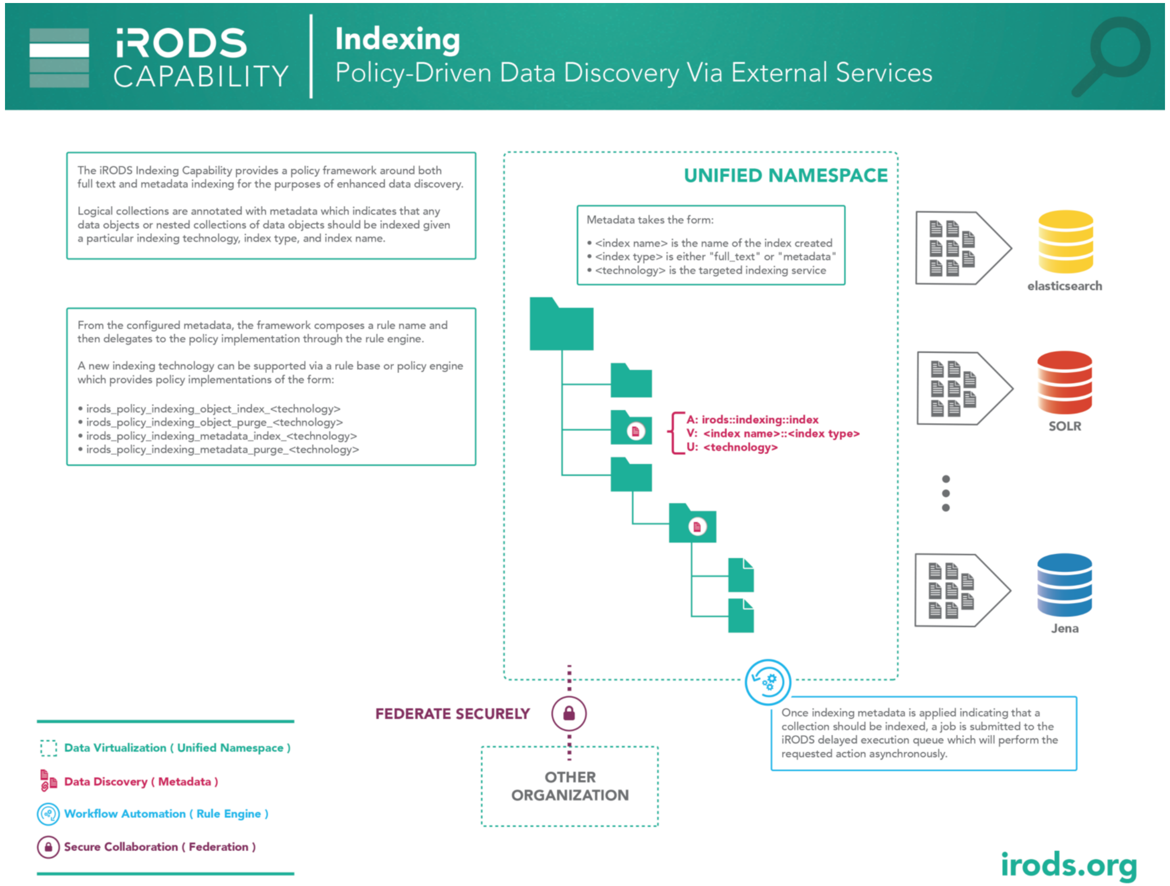
https://github.com/irods/irods_capability_storage_tiering

Indexing Capability

- iRODS Capability
- Developed in collaboration with NIEHS and BMS

National Institute of Environmental Health Sciences

Bristol Myers Squibb™



iRODS CAPABILITY

Indexing

Policy-Driven Data Discovery Via External Services

The iRODS Indexing Capability provides a policy framework around both full text and metadata indexing for the purposes of enhanced data discovery.

Logical collections are annotated with metadata which indicates that any data objects or nested collections of data objects should be indexed given a particular indexing technology, index type, and index name.

From the configured metadata, the framework composes a rule name and then delegates to the policy implementation through the rule engine.

A new indexing technology can be supported via a rule base or policy engine which provides policy implementations of the form:

- irods_policy_indexing_object_index_<technology>
- irods_policy_indexing_object_purge_<technology>
- irods_policy_indexing_metadata_index_<technology>
- irods_policy_indexing_metadata_purge_<technology>

UNIFIED NAMESPACE

Metadata takes the form:

- <index name> is the name of the index created
- <index type> is either "full_text" or "metadata"
- <technology> is the targeted indexing service

A: irods::indexing::index
V: <index name>::<index type>
U: <technology>

Once indexing metadata is applied indicating that a collection should be indexed, a job is submitted to the iRODS delayed execution queue which will perform the requested action asynchronously.

elasticsearch
SOLR
Jena
FEDERATE SECURELY
OTHER ORGANIZATION
Data Virtualization (Unified Namespace)
Data Discovery (Metadata)
Workflow Automation (Rule Engine)
Secure Collaboration (Federation)
irods.org

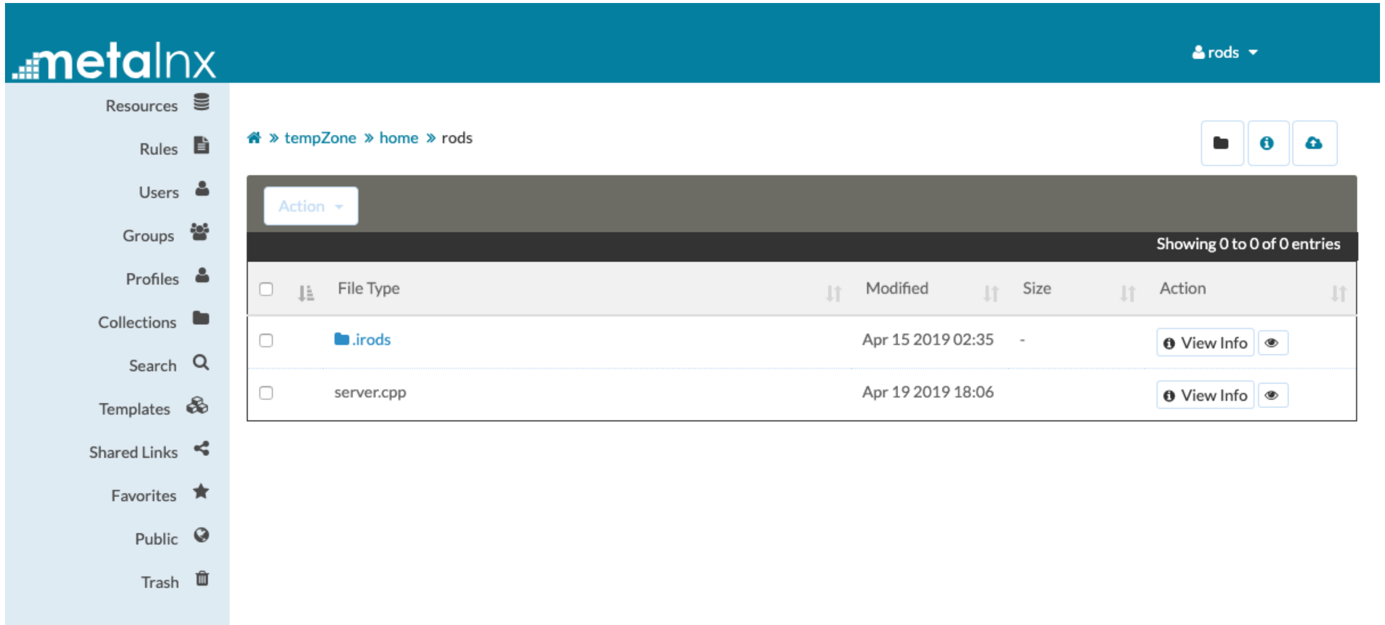
Open Source:

https://github.com/irods/irods_capability_indexing

Metalnx

- Web GUI
- Developed and Contributed by EMC

EMC^2®



tempZone » home » rods

Showing 0 to 0 of 0 entries

	File Type	Modified	Size	Action
□	.irods	Apr 15 2019 02:35	-	View Info
□	server.cpp	Apr 19 2019 18:06		View Info

Open Source:

https://github.com/irods-contrib/metalnx-web

go-irodsclient

- Golang iRODS Client Library
- Developed by CyVerse

CYVERSE®

Open Source:

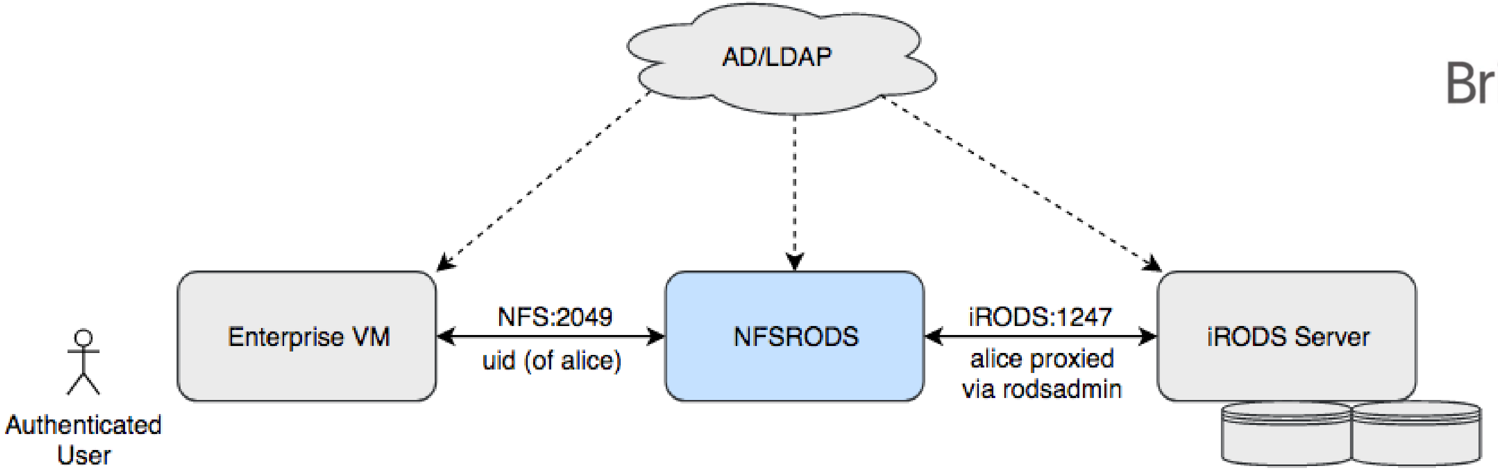
https://github.com/cyverse/go-irodsclient

NFSRODS

- iRODS Client, NFSv4.1 Server
- Developed in collaboration with CU Boulder and BMS

Research Computing
UNIVERSITY OF COLORADO **BOULDER**

Bristol Myers Squibb™



Open Source:

https://github.com/irods/irods_client_nfsrods

S3 Resource Plugin

- iRODS storage resource
- Developed Glacier-compatibility in collaboration with Fujifilm

FUJIFILM

Value from Innovation

Open Source:

https://github.com/irods/irods_resource_plugin_s3

iRODS Globus Connector

- iRODS Client, Globus Connector
- Developed in collaboration with Globus

Open Source:

https://github.com/irods/irods_client_globus_connector

Questions?

Proper data management requires policy enforcement.

These policies will change over time.

Open source is the best practice for a 100-year view.

Thank you.

irods.org

Terrell Russell

@terrellrussell

iRODS Consortium