IT Press Tour

Terrell Russell, Ph.D
Executive Director, iRODS Consortium
Director of Data Management, RENCI

October 7, 2024
IT Press Tour
Boston, MA

renci

RESEARCH \ ENGAGEMENT \ INNOVATION

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Our Membership

iRODS

Bibliothèque et Archives nationales Québec

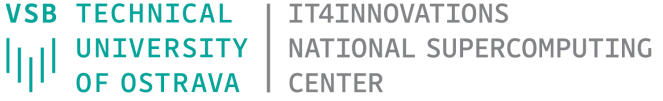
THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

Maastricht University

Universiteit Utrecht

wellcome sanger institute

renci



GRAU DATA

National Institute of Environmental Health Sciences

History

- 1995 - Storage Resource Broker (SRB)
  - 10 years of funded research - grid storage and catalog
  - San Diego Supercomputer Center, General Atomics
- 2006 - Integrated Rule-Oriented Data System (iRODS)
  - Open Source - BSD-3
  - 10 years of funded research - policy engine
- 2008 - Transitioned to UNC-Chapel Hill / RENCI
- 2013 - iRODS Consortium
  - community and membership model
  - service and support
  - installation and development

Partners and Users: Past and Present

- Supercomputing centers
- Physics
- Library / Archives
- Genomics
- Bio / Pharmaceutical
- Hydrology / Weather
- Medical
- Manufacturing
- Shipping / Logistics
- Automotive

What is iRODS

Open Source

- C++ client-server architecture
- iRODS Protocol and RPC API
- BSD-3 Licensed

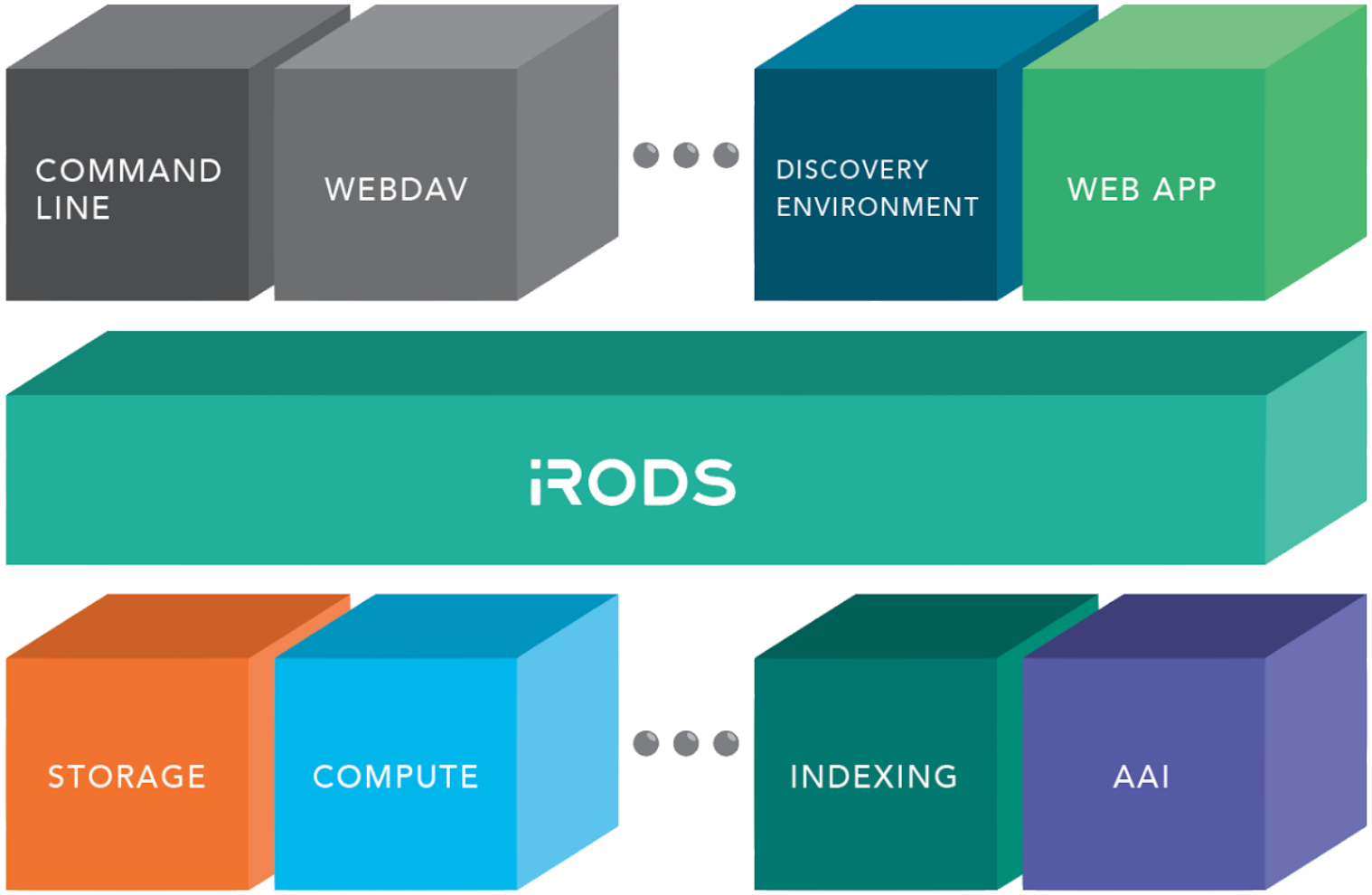
Distributed

- Runs on a laptop, a cluster, on premises or geographically distributed

Data Centric & Metadata Driven

- Insulate both your users and your data from your infrastructure over time

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

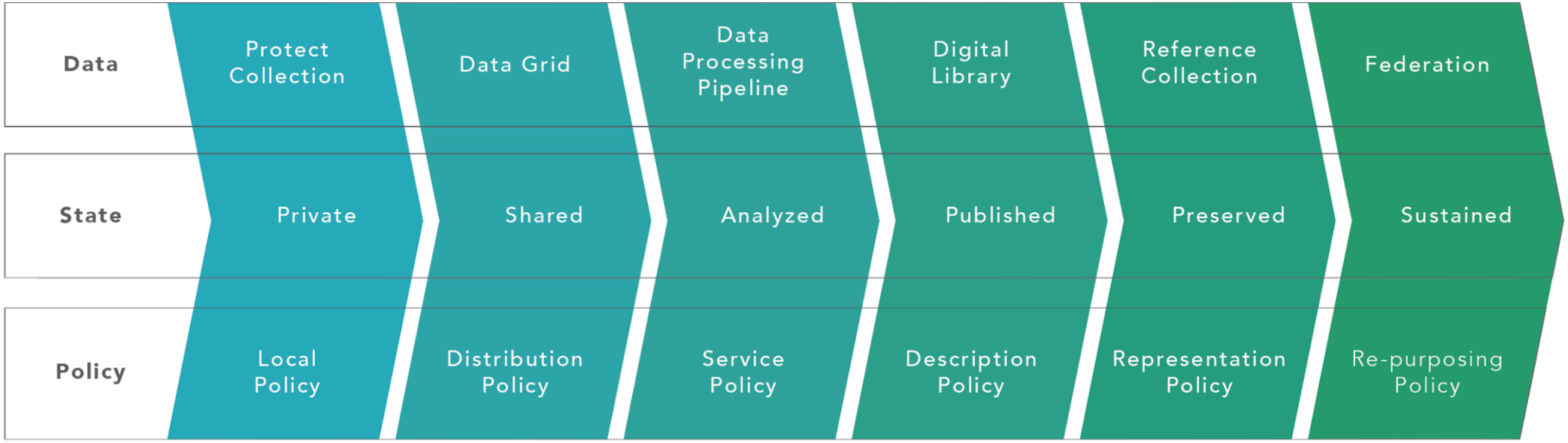
Why use iRODS?

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.

DATA LIFECYCLE



iRODS virtualizes the stages of the data lifecycle through policy evolution

As data matures and reaches a broader community, data management policy must also evolve to meet these additional requirements.

Data Virtualization

Combine various distributed storage technologies into a Unified Namespace

- Existing file systems
- Cloud storage
- On premises object storage
- Archival storage systems

iRODS provides a logical view into the complex physical representation of your data, distributed geographically, and at scale.

Attach metadata to any first class entity within the iRODS Zone

- Data Objects
- Collections
- Users
- Storage Resources
- The Namespace

iRODS supports automated and user-provided metadata which makes your data and infrastructure more discoverable, operational, and valuable.

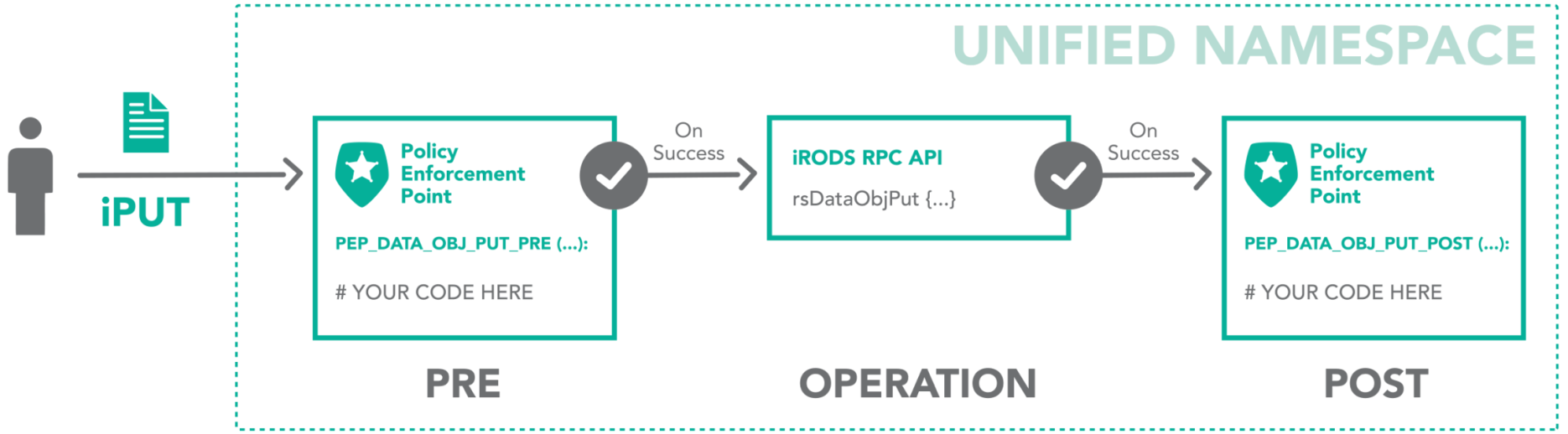
Workflow Automation

Policy Enforcement Points (PEPs) are triggered by every operation within the framework

- Authentication
- Storage Access
- Database Interaction
- Network Activity
- Extensible RPC API

The iRODS rule engine framework provides the ability to capture real world policy as computer actionable rules which may allow, deny, or add context to operations within the system.

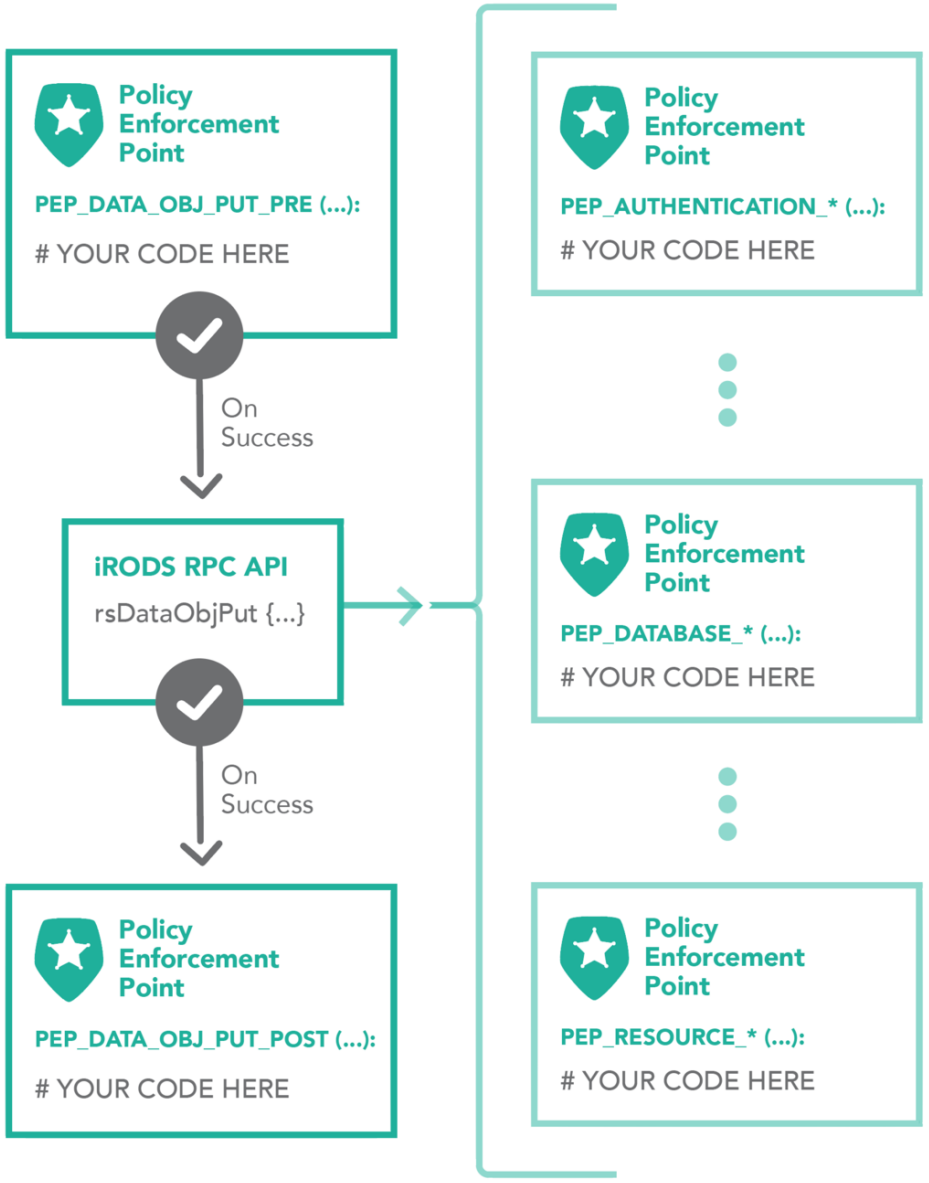
Dynamic Policy Enforcement



The iRODS rule may:

- restrict access
- log for audit and reporting
- provide additional context
- send a notification

Dynamic Policy Enforcement



A single API call expands to many plugin operations all of which may invoke policy enforcement

Plugin Interfaces:

- Authentication
- Database
- Storage
- Network
- Rule Engine
- Microservice
- RPC API

Secure Collaboration

iRODS allows for collaboration across administrative boundaries after deployment

- No need for common infrastructure
- No need for shared funding
- Affords temporary collaborations

iRODS provides the ability to federate namespaces across organizations without pre-coordinated funding or effort.

Protocol Plumbing - Presenting iRODS as other Protocols

Over the last few years, the ecosystem around the iRODS server has continued to expand.

Integration with other types of systems is a valuable way to increase accessibility without teaching existing tools about the iRODS protocol or introducing new tools to users.

With some plumbing, existing tools get the benefit of visibility into an iRODS deployment.

- WebDAV
- FUSE
- HTTP
- NFS
- SFTP
- K8s CSI
- S3

iRODS Clients

- **C/C++ (irods-dev[el])**
  API, C++ - Part of iRODS, implements the iRODS RPC API
  - **iCommands**
    CLI, C++, Part of iRODS
  - **C++ HTTP API**
    API, C++ - Presents iRODS as HTTP
    - **rirods - R Package**
      API, R - R Client Library
  - **C++ S3 API**
    API, C++ - Presents iRODS as S3
  - **C++ REST API**
    API, C++ (archived)
    - **Zone Management Tool (ZMT)**
      Web GUI, Javascript (React)
  - **GoRODS**
    API, Go - Go Client Library (archived)
    - **Minio (S3) iRODS Gateway**
      Network File System, Go - Presents iRODS as S3
  - **Perl (perl-irods-wrap)**
    API, Perl - Perl Client Library
  - **Kanki**
    Desktop GUI (Windows, Linux, MacOS), C++
  - **baton**
    CLI, C - Operates on JSON, metadata
  - **tears**
    CLI, C - Operates on streaming files
  - **fuse**
    Network File System, C++ (archived)
  - **Davrods**
    Network File System, C - Implements WebDAV
  - **Parrot Virtual File System**
    Network File System, C
  - **R Client Library**
    API, C++ (archived)

- **Python (python-irodsclient)**
  API, Python - Implements the iRODS RPC API
  - **iRODS Capability Automated Ingest**
    CLI, Python - Parallel ingest/sync from filesystems and S3
    - **BIH RODEOS Ingest**
      CLI, Python - Ingests omics data from Illumina sequencers
  - **BisQue (Bio-Image Semantic Query User Environment)**
    Platform and Web GUI, Python - Image storage, visualization, organization, and analysis
  - **BIH SODAR**
    Web GUI, Python - System for Omics Data Access and Retrieval
  - **AWS Lambda Function - S3**
    Lambda, Python - Updates an iRODS Catalog with create, rename, delete events from one or more S3 buckets
  - **irods-shell**
    CLI, Python - Simulates an ssh connection (cd, ls, mkdir, rm, ...)
  - **ii**
    CLI, Python - Command line utilities for iRODS
  - **iRODS Password Booth**
    Web GUI, Python - Simple tool for users to change their iRODS password
  - **iBridges**
    Desktop GUI (Windows, Linux, MacOS), Python
  - **Hamster**
    Desktop GUI (Windows, Linux, MacOS), Python

https://irods.org/clients

- **Java (Jargon)**
  API, Java - Implements the iRODS RPC API
  - **NFSRODS**
    Network File System, Java - Presents iRODS as NFSv4.1
  - **Java REST API**
    API, Java (archived)
    - **Qt (QRODS)**
      API, C++
      - **RODEX**
        Desktop GUI, C++
    - **R Client Library**
      API, R (archived)
  - **Cyberduck**
    Desktop GUI, Java
  - **Cloud Browser**
    Web GUI, Java
  - **MetaInx**
    Web GUI, Java
  - **iDrop**
    Desktop GUI, Java
  - **CyVerse Discovery Environment**
    Web GUI, Java
- **C# (irods-Csharp)**
  API, C# - Implements the iRODS RPC API
- **Go (go-irodsclient)**
  API, Go - Implements the iRODS RPC API
  - **iRODS gocommands**
    CLI, Go
  - **iRODS Fuse Lite**
    Network File System, Go
  - **iRODS Fuse Lite Pool**
    Network File System, Go
  - **iRODS CSI Driver**
    Kubernetes CSI Driver, Go
  - **SFTPGo with iRODS Storage Backend**
    SFTP Server, Go
- **PHP (php-irods)**
  API, PHP - Implements the iRODS RPC API (archived)

A Definition of Policy

A set of ideas or a **plan** of what to do in **particular situations** that has been agreed to officially by a group of people…

So how does iRODS do this?

The reflection of real world data management decisions in computer actionable code.

(a plan of what to do in particular situations)

Possible Policies - The What

- Data Movement
- Data Verification
- Data Retention
- Data Replication
- Data Placement
- Checksum Validation
- Metadata Extraction
- Metadata Application
- Metadata Conformance
- Replica Verification
- Vault to Catalog Verification
- Catalog to Vault Verification
- …

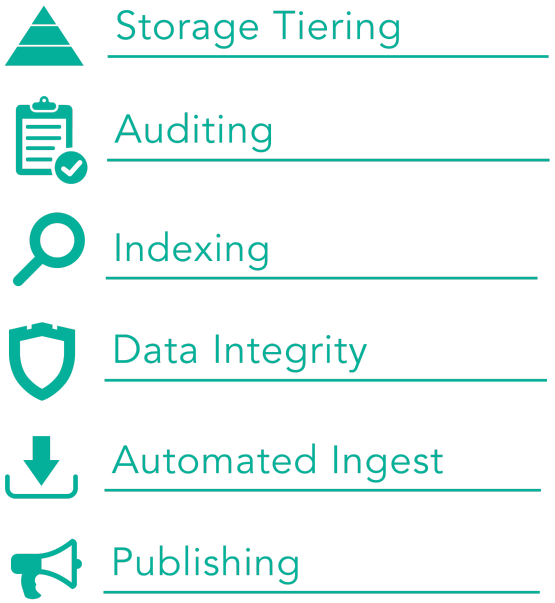
Policy Composition

Consider Storage Tiering:

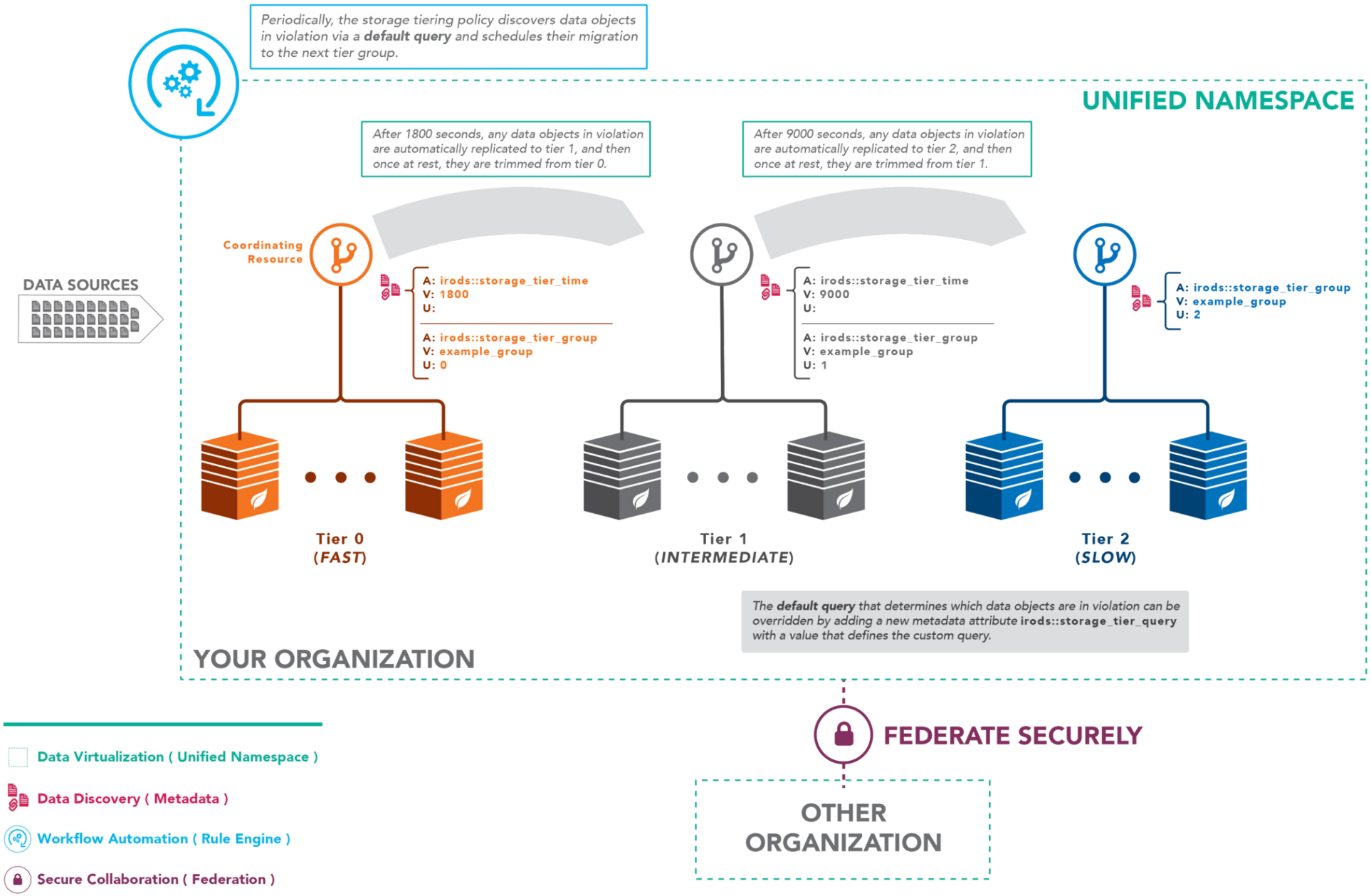
- Violating Object Identification
- Data Movement
  - Data Replication
  - Data Verification
- Data Retention

iRODS Capabilities

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



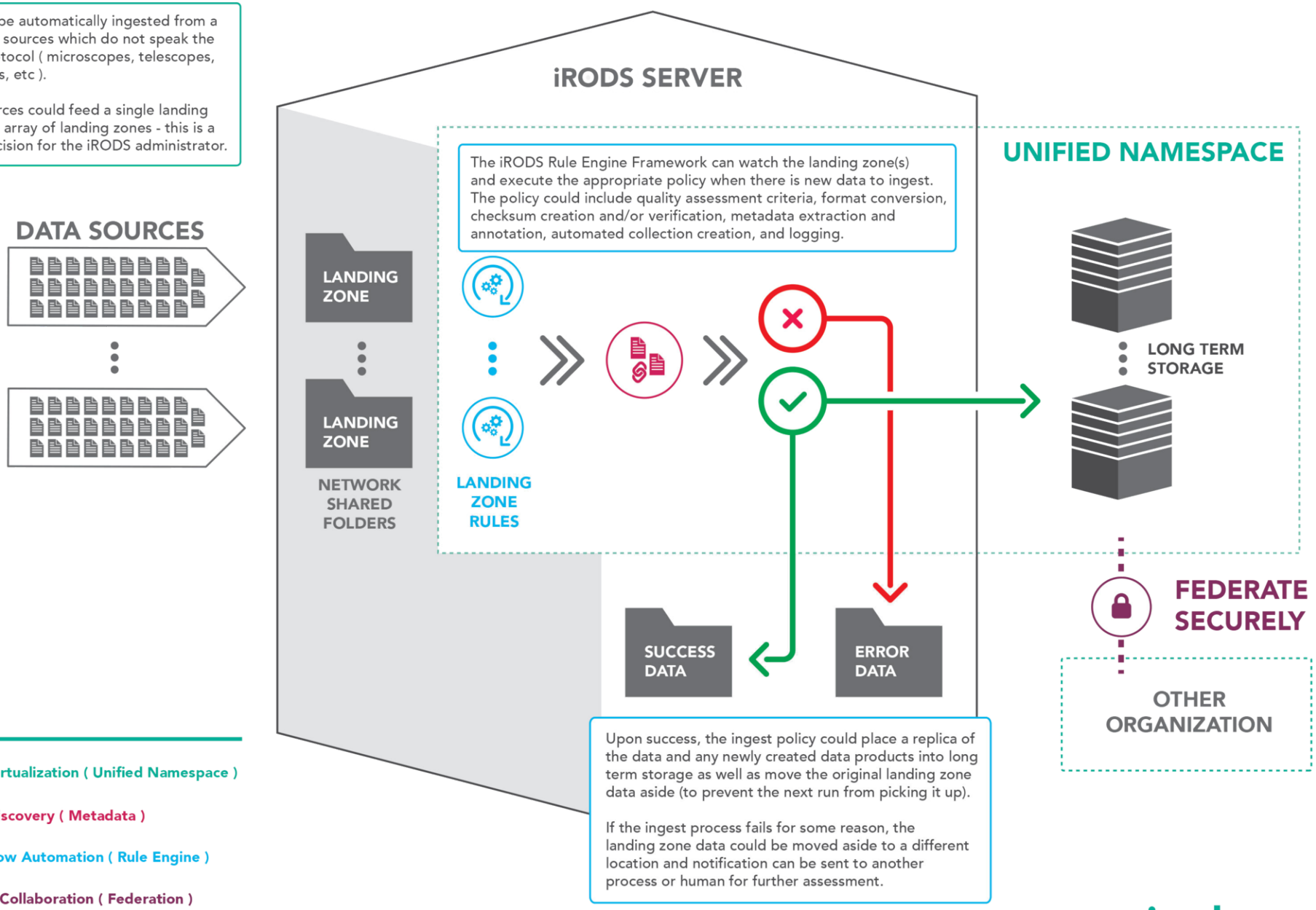
Storage Tiering



Automated Ingest - Landing Zone

Data may be automatically ingested from a number of sources which do not speak the iRODS protocol (microscopes, telescopes, sequencers, etc).

These sources could feed a single landing zone or an array of landing zones - this is a design decision for the iRODS administrator.



- Data Virtualization (Unified Namespace)
- Data Discovery (Metadata)
- Workflow Automation (Rule Engine)
- Secure Collaboration (Federation)

Automated Ingest - Filesystem Scanning

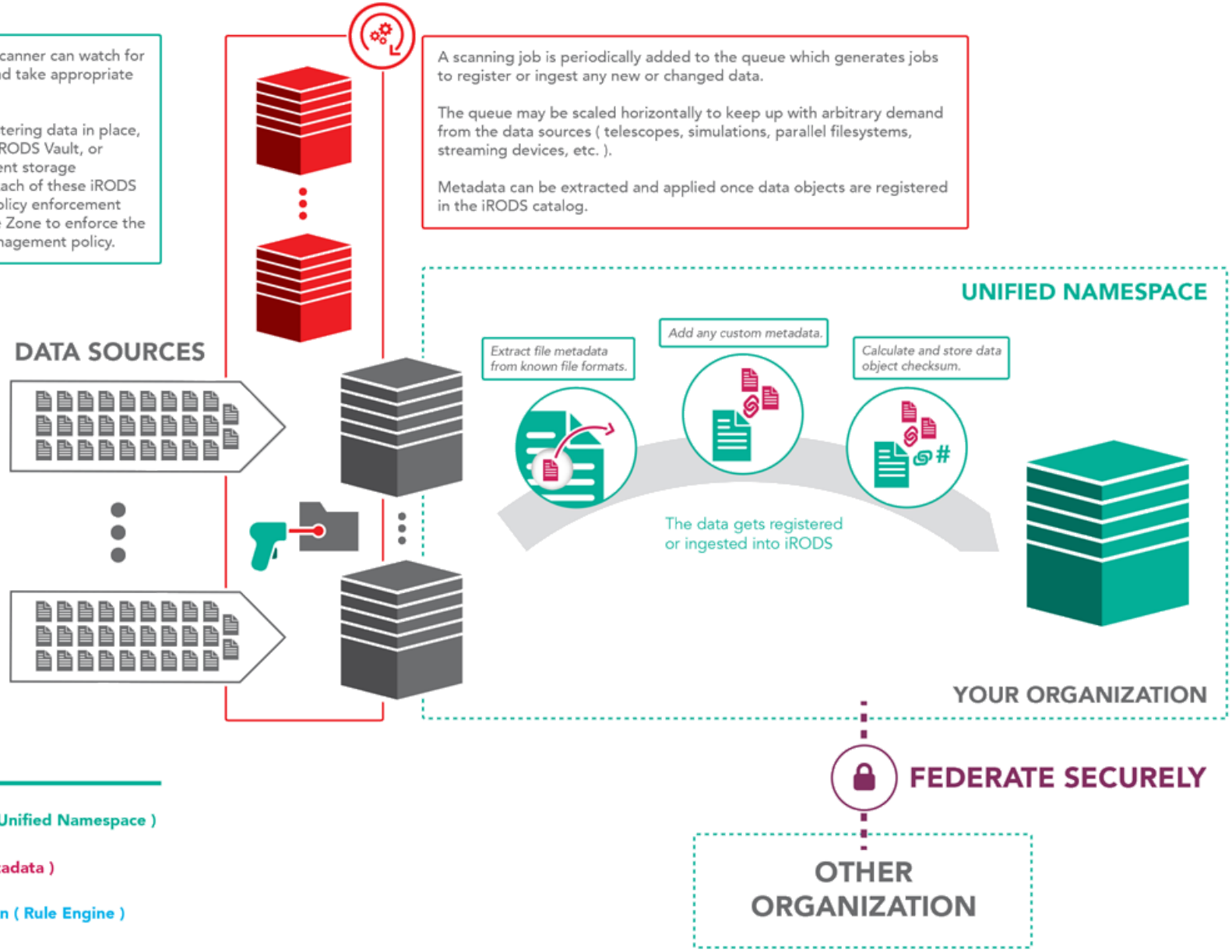
The iRODS filesystem scanner can watch for new or updated files and take appropriate action within iRODS.

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 Virtualization (Unified Namespace)
- Data Discovery (Metadata)
- Workflow Automation (Rule Engine)
- Secure Collaboration (Federation)

Indexing

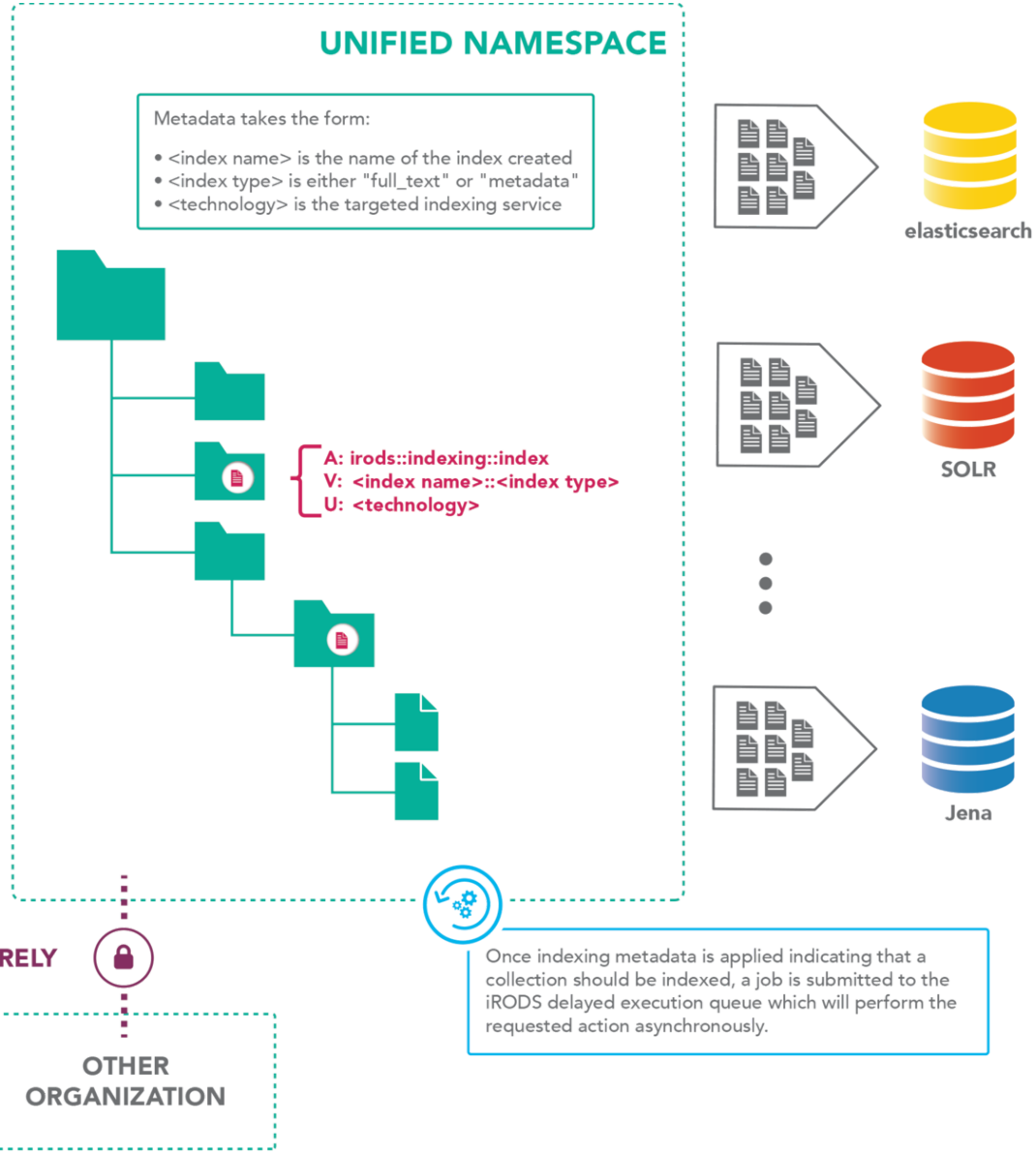
The iRODS Indexing Capability provides a policy framework around both full text and metadata indexing for the purposes of enhanced data discovery.

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>



Data Virtualization (Unified Namespace)

Data Discovery (Metadata)

Workflow Automation (Rule Engine)

Secure Collaboration (Federation)

Publishing

The iRODS Publishing Capability provides a metadata driven policy framework for the implementation of data publication to external services.

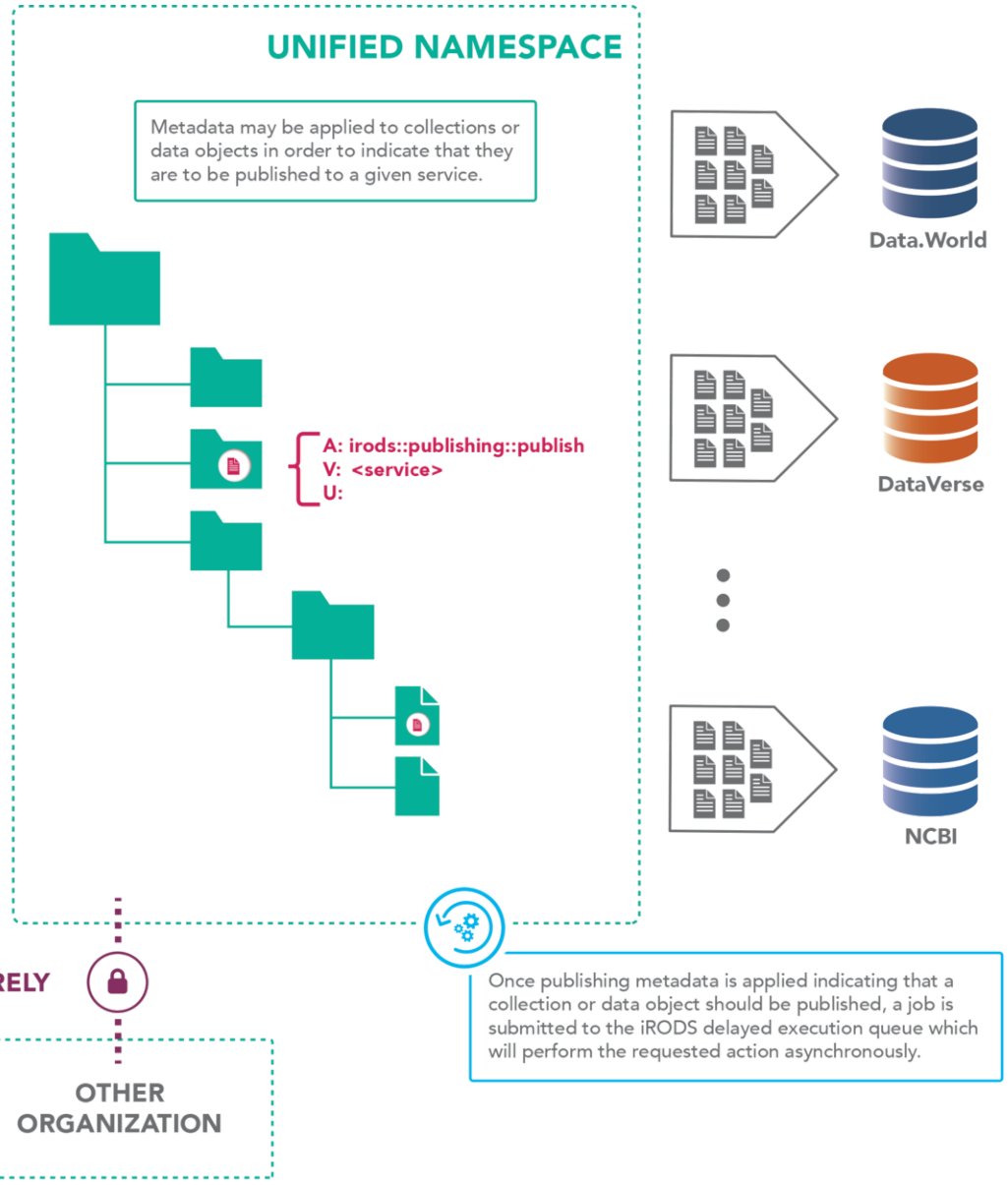
The policy framework provides:

- Protection for published data, which prevents future modification
- Invocation of secondary policy for the generation and application of persistent identifiers
- Advertisement and possible movement of published data to external catalogs

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

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

- irods_policy_publishing_object_publish_<service>
- irods_policy_publishing_object_purge_<service>
- irods_policy_publishing_collection_publish_<service>
- irods_policy_publishing_collection_purge_<service>



Data Virtualization (Unified Namespace)

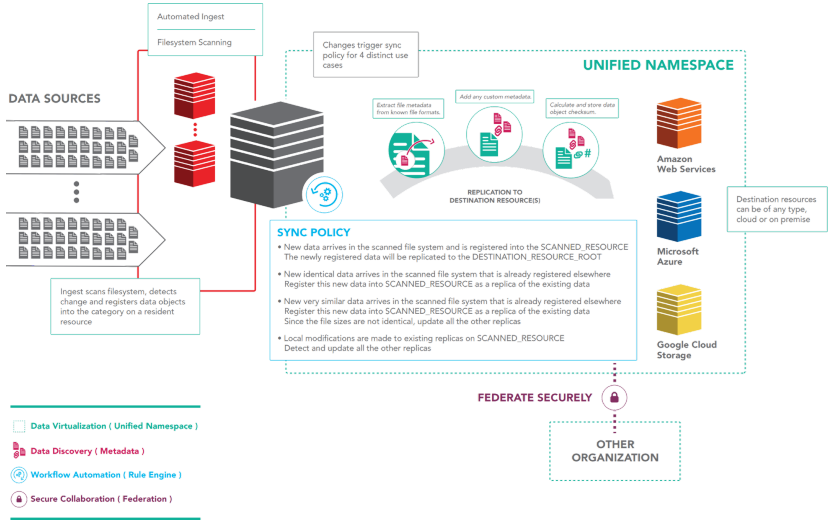
Data Discovery (Metadata)

Workflow Automation (Rule Engine)

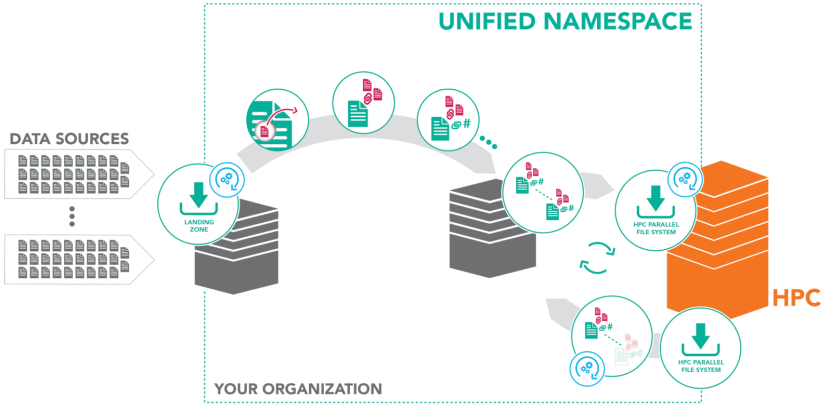
Secure Collaboration (Federation)

Deployment Patterns

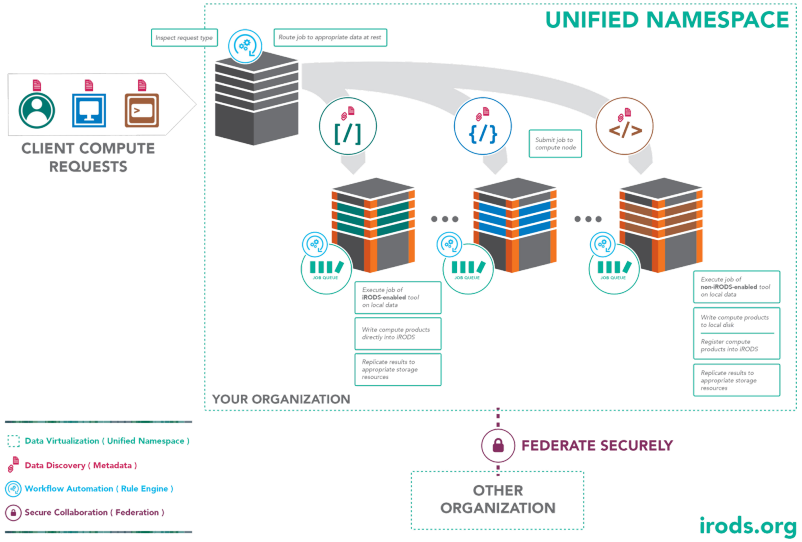
Filesystem Synchronization



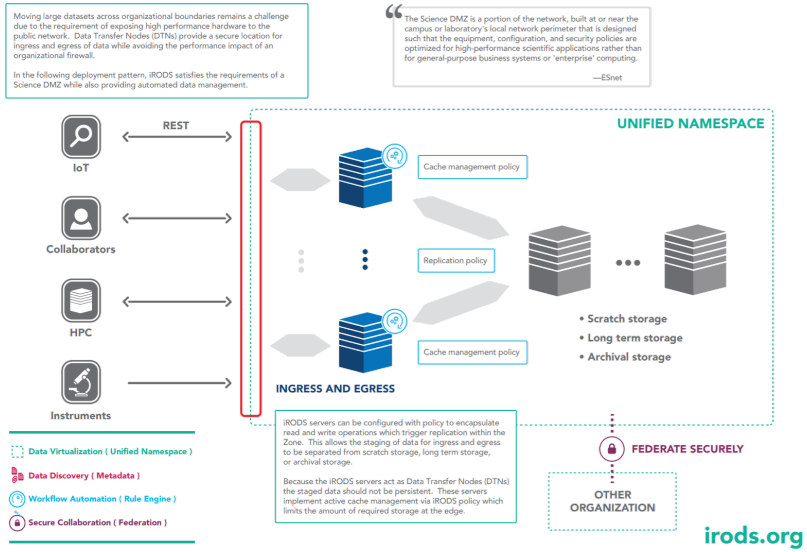
Data to Compute



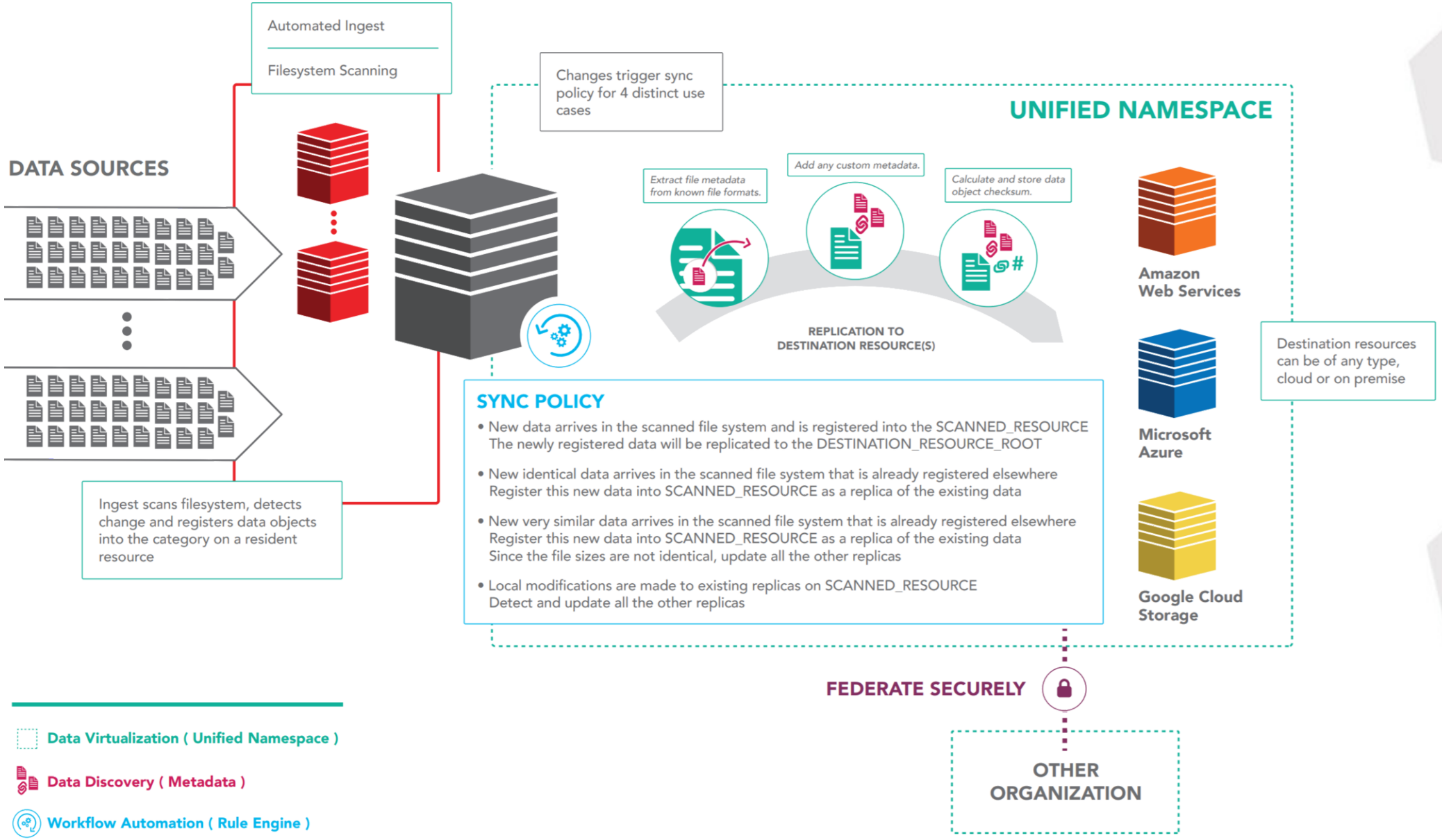
Compute to Data



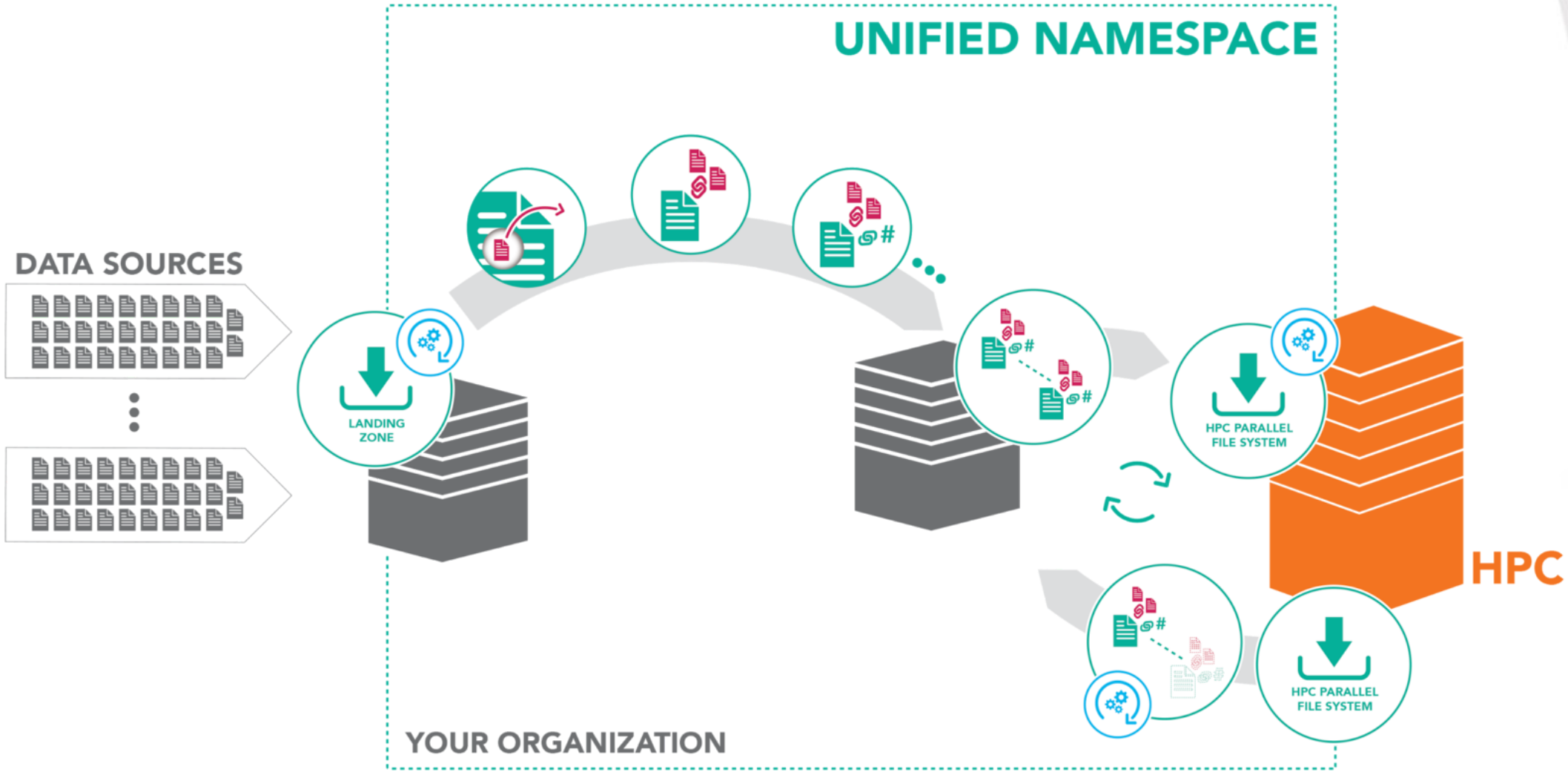
Data Transfer Nodes



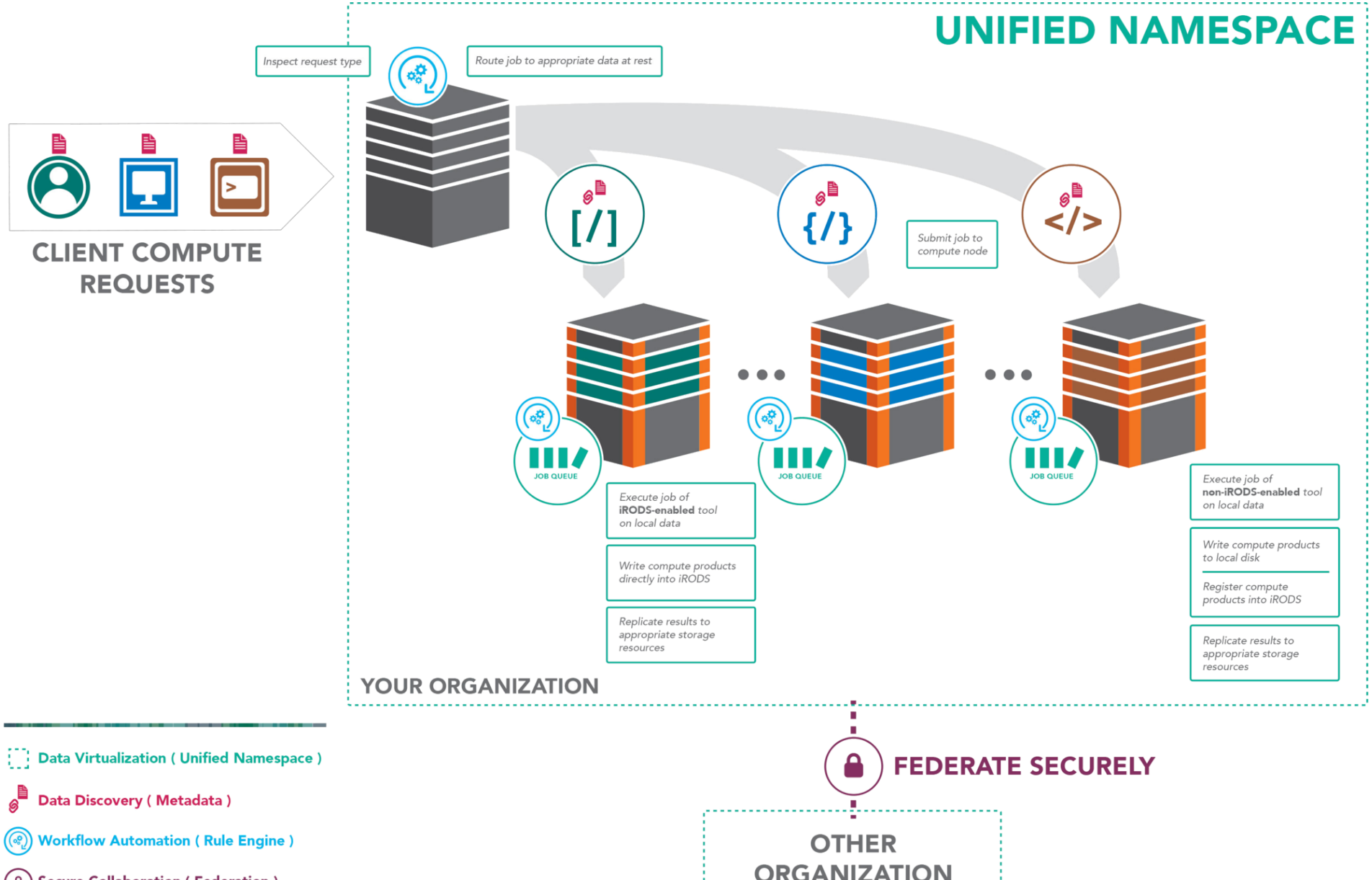
Filesystem Synchronization



Data to Compute



Compute to Data



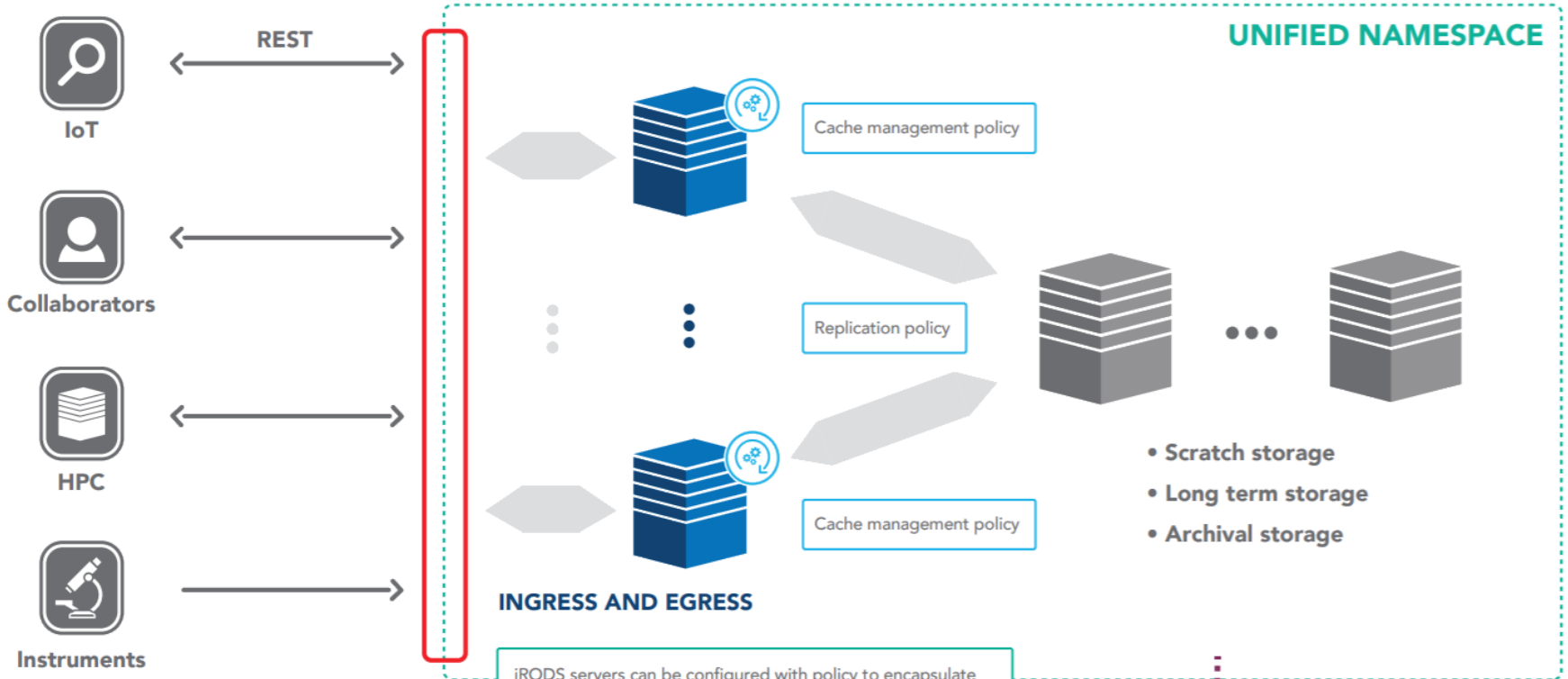
irods.org

Data Transfer Nodes

Moving large datasets across organizational boundaries remains a challenge due to the requirement of exposing high performance hardware to the public network. Data Transfer Nodes (DTNs) provide a secure location for ingress and egress of data while avoiding the performance impact of an organizational firewall.

In the following deployment pattern, iRODS satisfies the requirements of a Science DMZ while also providing automated data management.

> The Science DMZ is a portion of the network, built at or near the campus or laboratory's local network perimeter that is designed such that the equipment, configuration, and security policies are optimized for high-performance scientific applications rather than for general-purpose business systems or 'enterprise' computing.
>
> —ESnet



iRODS servers can be configured with policy to encapsulate read and write operations which trigger replication within the Zone. This allows the staging of data for ingress and egress to be separated from scratch storage, long term storage, or archival storage.

Because the iRODS servers act as Data Transfer Nodes (DTNs) the staged data should not be persistent. These servers implement active cache management via iRODS policy which limits the amount of required storage at the edge.

FEDERATE SECURELY

OTHER ORGANIZATION

- Data Virtualization (Unified Namespace)
- Data Discovery (Metadata)
- Workflow Automation (Rule Engine)
- Secure Collaboration (Federation)

irods.org

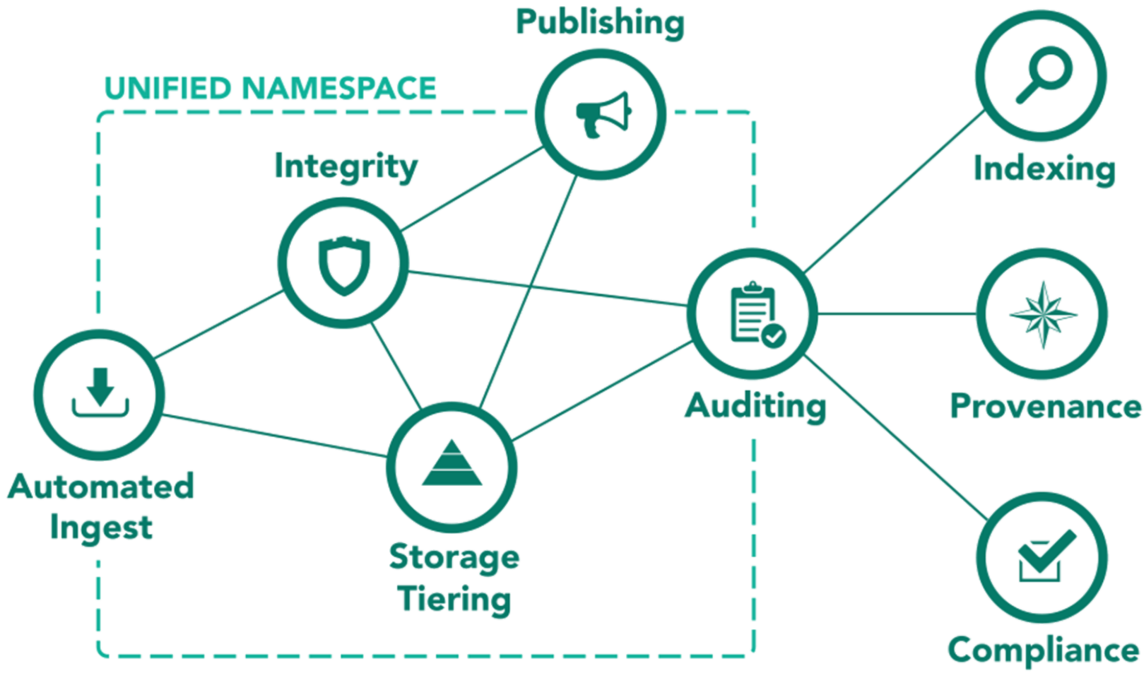
The Data Management Model

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.

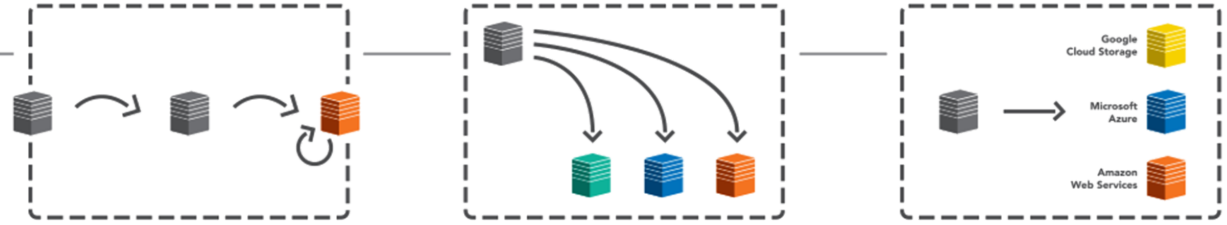
The flexibility provided by this model allows an organization to address its immediate use cases.

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

Future

- Towards Cloud-Native processes and bookkeeping
- Vertical Integrations in various domains
- Timeseries Data / Statistics
- Dashboarding
  - Visibility
  - Costs

Questions?

https://irods.org/sc24

FREE Mini-Workshop

iRODS Updates and Troubleshooting

Atlanta, Georgia

Monday, Oct. 18, 2024

9:30am-11:30am

iRODS S3 Functionality

The iRODS S3 storage resource plugin allows iRODS to use any S3-compatible storage device or service to hold iRODS Data Objects, on-premises or in the cloud.

This plugin can work as a standalone "cacheless" resource or as an archive resource under the iRODS compound resource. Either configuration provides a POSIX interface to data held on an object storage device or service.

The following S3 services and appliances (in no particular order) have been tested:

- Amazon (AWS) S3
- Fujifilm Object Archive
- MinIO S3
- Ceph S3
- Spectra Logic Vail
- Spectra Logic BlackPearl
- Google Cloud Storage (GCS)
- Wasabi S3
- Oracle OCI
- Quantum ActiveScale
- Garage S3

https://github.com/irods/irods_resource_plugin_s3