Technology Update

Kory Draughn
Chief Technologist
iRODS Consortium

June 29 - July 2, 2026
iRODS User Group Meeting 2026
Barcelona, Spain

Contributors

iRODS Release	Release Date	Commits
5.0.2	2025-10-01	37
4.3.5	2026-03-03	121

Alan King

Derek Dong

Daniel Moore

Justin James

Kory Draughn

Markus Kitsinger

Martin Jaime Flores Jr.

Ramsey Jooss

Terrell Russell

Ton Smeele

4.3.5 Release Summary

The final release of the server for the 4.3 series.

Focused effort on making the 4.3 server as stable as possible, for deployments which cannot upgrade to iRODS 5.

- Fixed several bugs
- Deprecated more functionality

Plugins will continue to receive updates, but will be limited to security fixes and trivial enhancements.

https://irods.org/2026/03/irods-4-3-5-is-released/

5.0.2 Release Highlights

- Added support for Enterprise Linux 10 and Debian 13
- Plugin versioning is independent of the server version
  - Plugin version numbers now follow three segments instead of four
  - e.g. `<plugin_name>-5.1.1-0.el10+5.0.2.x86_64.rpm`
- Improved support for Decoupled Mode in the S3 resource plugin
- Improved support for multi-byte characters when renaming collections
- Many improvements for `irsync`

https://irods.org/2025/10/irods-5-0-2-is-released/

5.1.0 Release - Where is it?

It's on the way.

There are a few more things to work out before it's ready.

We appreciate your patience.

https://github.com/irods/irods/milestone/47

Backward Compatibility

Maintaining backward compatibility is important to the Consortium.

To help in satisfying this requirement, we will do the following:

- Never modify packing instructions which are part of an official release
- Never modify data types used in public APIs
- Never modify data types sent over the network
- Never remove functionality from public APIs until a major release

These rules DO NOT apply to experimental APIs, libraries, or tools.

Main Server Process

- Disallows running with root privileges
- Logs real and effective UID/GID of user who launched the server
- Warns when launched by non-owner of the service account home directory

Towards FIPS 140-X Compliance

To support FIPS-enabled environments, the iRODS server must not use operations which rely on MD5.

- MD5 usage addressed in 5.1.0
  - Signed zone keys for server-to-server authentication
    - Introduced **`zone_key_signing_hash_scheme`** configuration property
  - Hashing rulebases and delay rules (SHA256)
  - `mockarchive` resource physical paths (SHA256)
- Other MD5 usage
  - Checksums
  - `native` authentication

New Authentication Scheme

Implemented new built-in authentication scheme called **`irods`**.

Objectives

- Strengthen security
- Time-limited, token-based authentication
- TLS required
- FIPS 140-X compliant (no MD5)

Plan

- iRODS 5 - Provided as an opt-in
- iRODS 6 - Becomes the default authentication scheme
- iRODS 7 - `native` authentication is removed

New Authentication Scheme - Usage and Password Management

Use password to get session token for authentication.

- Users set "`irods_authentication_scheme`" to "`irods`"
- **`~/.irods/.irods_secrets`** file holds returned session token

Set user passwords as normal.

- Use no-scramble option to prevent use of MD5
  - `ipasswd` **`--no-scramble`**
  - `iadmin moduser alice password apass` **`no-scramble`**

Passwords and tokens can be cleared.

- `iadmin moduser alice` **`remove_password`**
- `iadmin` **`remove_session_tokens`** `expired alice`

Script to clear legacy/native passwords packaged with 5.1.0.

New Authentication Scheme - Grid Configuration

Added new grid-wide configuration options to the `authentication` namespace.

- **`password_hashing_parameters`**
  - JSON which configures KDF
  - Key derivation `"algorithm"` (currently only supported by scrypt)
  - `"parameters"` are specific to the chosen algorithm
    - scrypt: keylen, CPU/memory cost, block size, parallelization

- **`password_storage_mode`**
  - Controls password-setting behavior
  - `"legacy"` (default/native), `"hashed"` (`irods`), `"both"`

- **`token_lifetime_in_seconds`**
  - Controls session timeouts (like TTL)

Pre-5.1.0 behavior

- Rebalance operation halts on first issue
  - Subsequent data objects are not replicated, even if they could be
  - Large amounts of manual intervention possibly required

5.1.0 behavior

- Rebalance operation continues processing after issue
  - All possible replications will be completed
- New error code to notify administrator
  - `REBALANCE_NOT_COMPLETE (-1834000)`

Resource Rebalance (cont.)

Updated documentation to include commands to locate data objects requiring intervention.

```
$ iadmin modresc thingToRebalance rebalance
remote addresses: 172.19.0.3 ERROR: rcGeneralAdmin failed with error -1834000 REBALANCE_NOT_COMPLETE

$ iquest "SELECT COLL_NAME, DATA_NAME WHERE DATA_RESC_HIER LIKE 'thingToRebalance;%' and DATA_REPL_STATUS != '1'"
COLL_NAME = /tempZone/home/alice/dataHere
DATA_NAME = foo1
----------
COLL_NAME = /tempZone/home/alice/dataHere
DATA_NAME = foo3
----------
COLL_NAME = /tempZone/home/alice/dataHere
DATA_NAME = foo5
----------

$ iquest "SELECT COUNT(DATA_ID) WHERE DATA_RESC_HIER LIKE 'thingToRebalance;%' and DATA_REPL_STATUS != '1'"
DATA_ID = 6
```

Random Scheme Vault Path Policy

Added new microservices which allow administrators to customize how physical paths are generated when using the random scheme vault path policy.

- `msi_set_random_scheme_style(style)`
  - Controls which random scheme style is applied during physical path generation

- `msi_set_random_scheme_suffix_length(length)`
  - Controls the length of the randomly-generated string appended to the physical path

Notable Server Updates

- Secure communication (TLS) can be configured via `setup_irods.py`
  - Supports generation of self-signed certificates
- Various bug fixes for Physical Quotas
- Various bug fixes and enhancements for GenQuery2
- Tracking and exposure of iRODS error codes through the dstream library
  - Makes error information available to clients such as `istream`
- iCommands detect mismatch major version numbers and warns user
- New feature test macros for detecting development library and server capabilities
  - Helps in maintaining source compatibility across versions
- More deprecations and removal of unused functionality

Building and Testing

- Plugin build hooks provide option for compiling source against released iRODS packages
  - e.g. `--irods_package_version 5.0.1-0~noble`
- Automated plugin testing via GitHub Actions
  - Covers all plugins except the Python and Audit AMQP rule engine plugins
  - Built and tested against multiple iRODS versions
  - Detects source compatibility breakage
- Removed dependency on Docker Compose Python module from iRODS Testing Environment

Policy Composition Rule Engine Plugin

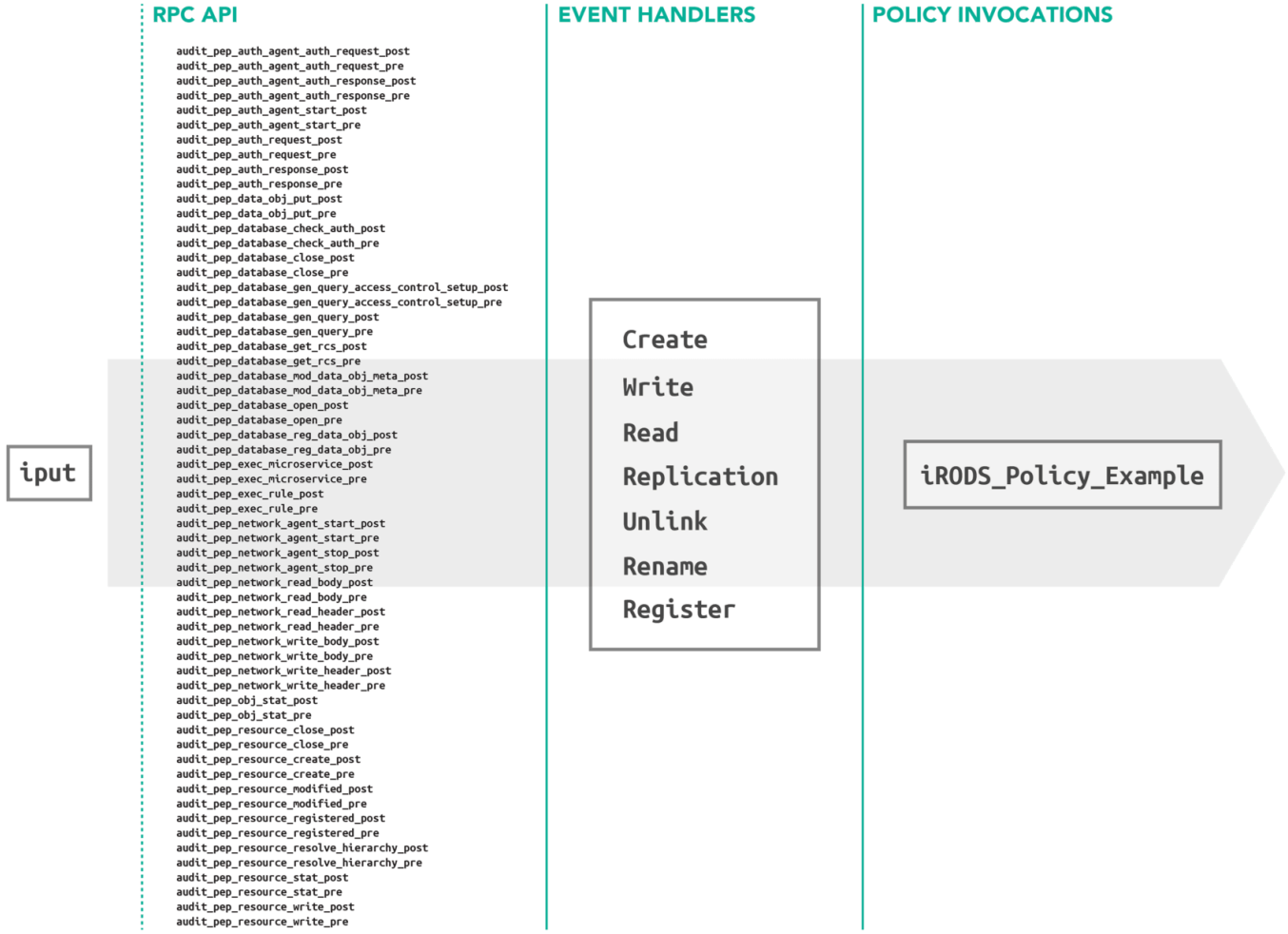
Initial release (0.1.0) is available for iRODS 5.0.2.

Designed to simplify policy enforcement by allowing administrators to think in terms of configuration and composability rather than having to write code.

Visit link to access past presentations about Policy Composition.

https://irods.org/2026/04/initial-release-of-irods-policy-composition/

Policy Composition Rule Engine Plugin - Overview



https://github.com/irods/irods_rule_engine_plugin_policy_composition/tree/0.1.0

Policy Composition Rule Engine Plugin - Example Configuration

```
{
  "instance_name": "irods_rule_engine_plugin-event_handler-data_object_modified-instance",
  "plugin_name": "irods_rule_engine_plugin-event_handler-data_object_modified",
  "plugin_specific_configuration": {
    "policies_to_invoke": [
      {
        "active_policy_clauses": [
          "post"
        ],
        "events": [
          "put",
          "create",
          "write",
          "registration"
        ],
        "policy_to_invoke": "irods_policy_data_replication",
          "configuration": {
          "destination_resource": "AnotherResc"
        }
      }
    ]
  }
}
```

https://github.com/irods/irods_rule_engine_plugin_policy_composition/tree/0.1.0

Logical Quotas Rule Engine Plugin

Six releases since UGM 2025.

- Uses rodsadmin connections for quota updates
  - Resolves issues with server redirects and permissions
  - Resolves issues with Metadata Guard interactions

https://github.com/irods/irods_rule_engine_plugin_logical_quotas/releases

Storage Tiering Capability Plugin

Three releases since UGM 2025.

- Fixed permission denied errors for checksum calculations
  - Occurred when rodsadmin user lacked ownership of the data object

https://github.com/irods/irods_capability_storage_tiering/releases

UnivMSS Resource Plugin

- Added support for escaping single quotes in paths
  - Single quotes are escaped before being passed to external executable
- Added support for semicolon-delimited properties to context string
  - `script`
    - Defines the name of an executable to run
    - Executable must exist in `<service_account_home>/msiExecCmd_bin`
  - `escape_single_quotes`
    - Controls whether single quotes are escaped
    - Defaults to 0 (disabled); Enable by setting to 1

```
$ iadmin mkresc ... 'script=file.sh;escape_single_quotes=1'
```

Audit AMQP Rule Engine Plugin

Part two of the **in-progress** overhaul is almost complete.

- Most Qpid Proton settings exposed as configuration options
- New failsafe mode to block unauditable operations
- Per-operation AMQP connections
- General modernization

https://github.com/irods/irods_rule_engine_plugin_audit_amqp/pull/185

Python iRODS Client

Two releases since UGM 2025.

- Compatible with iRODS 5
- Added support for PAM Interactive authentication scheme
- Improved support for `groupadmin` users
- Improved support for tickets
- Improved support for metadata
- Added support for aborting parallel transfers in a controlled manner
- Added support for excluding columns from GenQuery1 results
- Automated testing via GitHub Actions

https://github.com/irods/python-irodsclient/releases/tag/v3.2.0

https://github.com/irods/python-irodsclient/releases/tag/v3.3.0

Python iRODS Client - Flag Preservation for Metadata Operations

```python
import irods

sess = irods.helpers.make_session()
dobj = sess.data_objects.get('/tempZone/home/ugm_user/file.txt')

# Enable admin flag for metadata operations.
adm = dobj.metadata(admin=True)

if adm.items():
  # Enable timestamp retrieval for metadata operations.
  # This operation is cumulative, meaning the state of
  # the admin flag is preserved.
  adm_ts = adm(timestamps=True)
  avu = adm_ts.items()[0]

  # Without the bug fix, this operation would fail.
  adm_ts.set(avu.name, avu.value + ".", avu.units)
```

Python iRODS Client - `ticket_iterator`

Prints a list of tickets and their attributes, similar to **`iticket ls`**.

```
 1 import irods
 2 from irods.ticket import *
 3 from pprint import pp
 4
 5 session = irods.helpers.make_session()
 6 pp([
 7   vars(t)
 8   for t in ticket_iterator(
 9     session,
10     filter_args=[TicketQuery.Owner.name != '']
11   )
12 ])
```

Python iRODS Client - Aborting Parallel Transfers

```
1 from irods.parallel import abort_parallel_transfers
2
3 try:
4     session.data_objects.put(...)
5
6 except KeyboardInterrupt:
7     abort_parallel_transfers()
```

Globus Connector

Four releases since UGM 2025.

- Compatible with iRODS 5
- Fixed segfault on rename operation via FTP client
- Uses iRODS-provided hashers for checksum/hash calculations
- Buffer size for checksum calculations are now configurable
  - `$fileReadForChecksumCalculationBufferSizeBytes`
- Improved code quality and cleaned up CMake
  - Replaced variable-length array usage with `alloca()` function
  - Migrated from Boost.Format and `std::stringstream` to fmtlib
  - Migrated from cJSON to nlohmann-json

HTTP API

0.7.0 released on 2026-06-18.

- Fixed bugs that result in the HTTP API terminating unexpectedly
- Fixed unnecessary expansion of group permissions for data objects
- Strengthened configuration validation
- Added new configuration properties for enforcing parallel write stream limits
- Updated write operations to return true iRODS error codes
  - Implemented, but disabled until iRODS 5.1.0 development library is available

https://github.com/irods/irods_client_http_api/releases/tag/0.7.0

Four releases since UGM 2025.

- Added support for the PAM Interactive authentication scheme
- Added support for secure communication via JSSE `TrustManagers`
- Improved support for multi-byte characters
- Improved compatibility with Microsoft Windows
- Automated testing via GitHub Actions

https://github.com/irods/irods4j/releases

Refactored to use irods4j instead of Jargon.

- Initial work by intern in 2025, completed by core dev team
  - https://github.com/iterate-ch/cyberduck/pull/17341
  - https://github.com/iterate-ch/docs/pull/632
- Added support for parallel transfer over port 1247
- Added common iRODS configuration options to cyberduckprofile
- Added support for secure communication
- Added support for PAM

https://docs.cyberduck.io/protocols/irods/

iDrop

- Updated to use Jargon 4.3.7.0-RELEASE
  - Work completed by external contributor, Jakob Saternus
- Compatible with iRODS 4.3 and later
- Includes shell scripts for building application for Linux and Windows

https://github.com/irods-contrib/idrop

pam_interactive Authentication for the Python iRODS Client (PRC)

This talk introduces support for the pam_interactive authentication scheme in the Python iRODS Client (PRC). It ports the C++ plugin to enable a server-driven, conversational handshake, which is essential for complex PAM setups such as multi-factor authentication.

Updates to the iRODS Zone Management Tool and GenQuery2

The iRODS Zone Management Tool has been refactored to remove its dependence on Material UI for increased maintainability. GenQuery2 has been modified to allow for functions to be used in the GROUP-BY clause.

Refactoring Cyberduck: Migrating from Jargon to irods4j for its iRODS support

Cyberduck provides user-friendly access to remote storage systems, including iRODS. However, its reliance on the outdated Jargon library has limited performance and maintainability. This project upgrades Cyberduck's iRODS backend by replacing Jargon with the modern and streamlined irods4j library, enabling cleaner code, better performance, and improved compatibility with current iRODS 5 systems.

https://irods.org/trirods/

USF Senior Team Project - Fall 2025

Provided mentorship to a team of four students from the University of San Francisco tasked with benchmarking and optimizing client-side data transfer performance through the use of compression.

- Tested with the Python iRODS Client and irods4j
- Tested in various environments - simulated (`tc`) and real (AWS) network latency
- Zstd, LZ4, Snappy, Gzip, XZ

Findings

- Binary encoding is superior to XML encoding
- Compression is beneficial for slow networks; not so much for fast networks
- Zstd offers the best balance; LZ4 recommended for time-critical operations

This research will influence how compression is implemented in iRODS.

Future Work (after 5.1.0 release)

- Port Metadata Guard rule engine plugin logic into the server
- Improve support for archive files
- Improve database connection management
- Implement new **irods** authentication scheme for all client libraries
- Replace all use of legacy parallel transfer with multi-1247 parallel transfer
- Parallelize resource-rebalance operation
- Improve log messages in server
- Provide better/simpler tools for users
- Automate all testing for the server, plugins, and clients

Thank you!

Upcoming talks from the Development Team

- Absorbing Logical Quotas into the iRODS Server
  - Derek Dong
- AI in iRODS? A Canvassing of Community Emotion and Position
  - Terrell Russell
- iRODS S3 API: User and Bucket Mapping, Presigned URLs, and Dataverse
  - Alan King
- Verifying S3 Uploads via Direct Checksum Read from S3 Provider
  - Justin James
- iRODS Build and Packaging: 2026 Update
  - Markus Kitsinger