Testing operations, clients and metadata support in the iRODS system

Authors:
Sławomir Kubicki
Piotr Skoczek
Artur Wesołowski

Introduction

iRODS is an open source data management software. It’s main perks consist of: data virtualization, data discovery, workflow automation, secure collaboration. According to iRODS home page (“iRODS strives to serve as the glue that can tie together many existing storage technologies”) and iRODS-chat google group (“Object Storage is currently supported”) it supports many ways to manage data storage. Therefore understating how object storage and other storage technologies work is essential to use iRODS full possibilities.

1. **Object Storage.**
   Object-based storage is a computer data storage architecture that manages data as objects. Objects are stored in a flat address space, which eliminates the complexity and scalability challenges of the hierarchical file systems. Each object typically includes the data, a variable amount of metadata, and a globally unique identifier. Some of metadata are generated by the system, others can be provided by the user or an external application.
   Objects contain additional descriptive properties which can be used for better indexing or management. As opposed to fixed metadata in file systems (filename, creation date, type, etc.), object storage provides support for custom, object-level metadata.
   Object storage provides programmatic interfaces to allow applications to manipulate data. Most API implementations are ReST-based, allowing the use of standard HTTP calls.
2. **The difference between object storage and file storage.**
   Data in file systems are stored hierarchically in subsequent folders, that create the path to the specified file. Object-storage systems have a two-level structure (bucket and object id).
   Both systems contain metadata, however in file systems they are limited (file name, creation date, modification date). In object storage system in addition to basic metadata, created by system, there is possibility to create and modify metadata defined by user.
   In file storage data typically needs to be shared locally. In object storage data can be stored across multiple regions and can be accessed from different places.
   Some object systems do not allow modification of already existing files, the result of modification is the creation of a new version of a given file. This means, among others no need to manage locks and allow better scaling for a very large number of files. This works well with binary data such as photos or movies.
3. **Open Source Data Management Software – iRODS.**
   The Integrated Rule-Oriented Data System (iRODS) is an open data management software that virtualizes resources to store them, so that users from

different zones can control the data regardless of where the data is physically stored.
The objects are organized in collections. Objects in a collection do not have to be physically stored in the same place. Storage resources are stored in iRODS zones, each has a host name and path to the source file.
You can create metadata for both objects and collections, users, resources, and other iRODS zones. The iRODS catalog for a given zone is located in a relational database hosted in PostgreSQL, MySQL, or Oracle databases.
Security.
iRODS provides Secure Collaboration through three technologies:
Tickets, Permissions, and Federation.
Tickets.
They are used to provide public access to data objects and collections. The owner of a data or collection object has the option of creating a ticket and making it available to a non-iRODS user for reading or writing access. Tickets can be revoked, and they can be set to automatically expire upon a specified date and time or a specified number of reads or writes.
Permissions.
They work analogously to the UNIX file system permissions, the owner of a data or collection object has the ability to assign read / write access to defined users and groups in iRODS. Members of the group are selected by the administrator.
Federation.
iRODS Federation extends data sharing and publication beyond a single Zone. Once the administrators of two iRODS Zones share a set of keys , the owner of a Data Object or Collection can assign read and write permissions to users from outside Zones.

Installation and configuration process

The software has been tested on three operating systems: Ubuntu 14, Ubuntu 16, CentOS 7.
The installation process was carried out in accordance with the documentation provided by the iRODS authors.
IRODS server has been configured as ‘provider’ on every system. The selected role, unlike the role of the consumer, offers zone management, and can provide Storage Resources, which is why it was chosen.

Before installing software iRODS repository needs to be added to package manager. Instruction how to do this can be found on https://packages.irods.org/.

The iRODS setup script requires database connection information about an existing database. iRODS neither creates nor manages a database instance itself, just the tables within the database. Therefore, the database instance should be created and configured before installing iRODS.
Software in accordance with the documentation supports following database management systems: Oracle, MySQL, PostgreSQL. In our project, we used PostgreSQL.
The particular type of database is encoded in /etc/irods/database_config.json.
Additionally, an iRODS database plugin is required, which for PostgreSQL can be download with command:
- apt-get install irods-server irods-database-plugin-postgres (Ubuntu)
or:
- yum install epel-release & yum install irods-server irods-database-plugin-postgres (CentOS)

Configuration files

/etc/irods/server_config.json

This file defines the behavior of the server Agent that answers individual requests coming into iRODS. It is created and populated by the installer package.

~/.irods/irods_environment.json

This is the main iRODS configuration file defining the iRODS environment. Any changes are effective immediately since iCommands reload their environment on every execution. It can also be located in /var/lib/irods/.irods.

Database

By default name of the database used by iRODS is **ICAT**. Below there is list of basic tables from ICAT database.

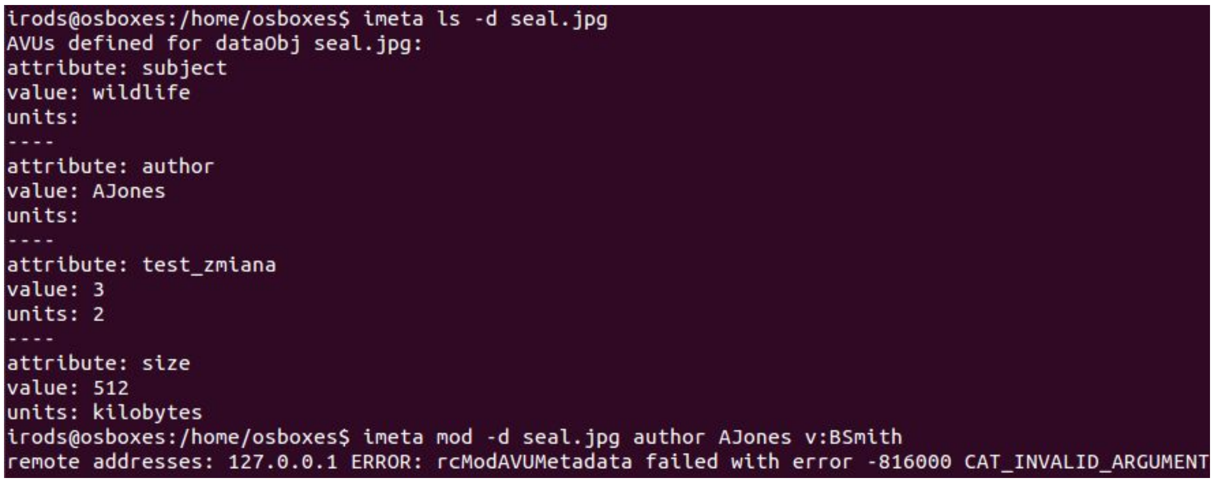
r_coll_main - contains information about collections (directories)
r_data_main - contains information about files. Data itself is stored in the filesystem
r_meta_main - contains user-defined metadata (name, value, unit, etc.; reference to file owning metadata is not stored in this table)
r_objt_metamap - connects tables **r_data_main** and **r_meta_main.** Metadata is not shared between files and this table does not contain any additional attributes so purpose of this table is unclear
r_user_main - contains list of users
r_user_passwords - contains information about users’ hashed password

4. Problems encountered

Modifying metadata using commands

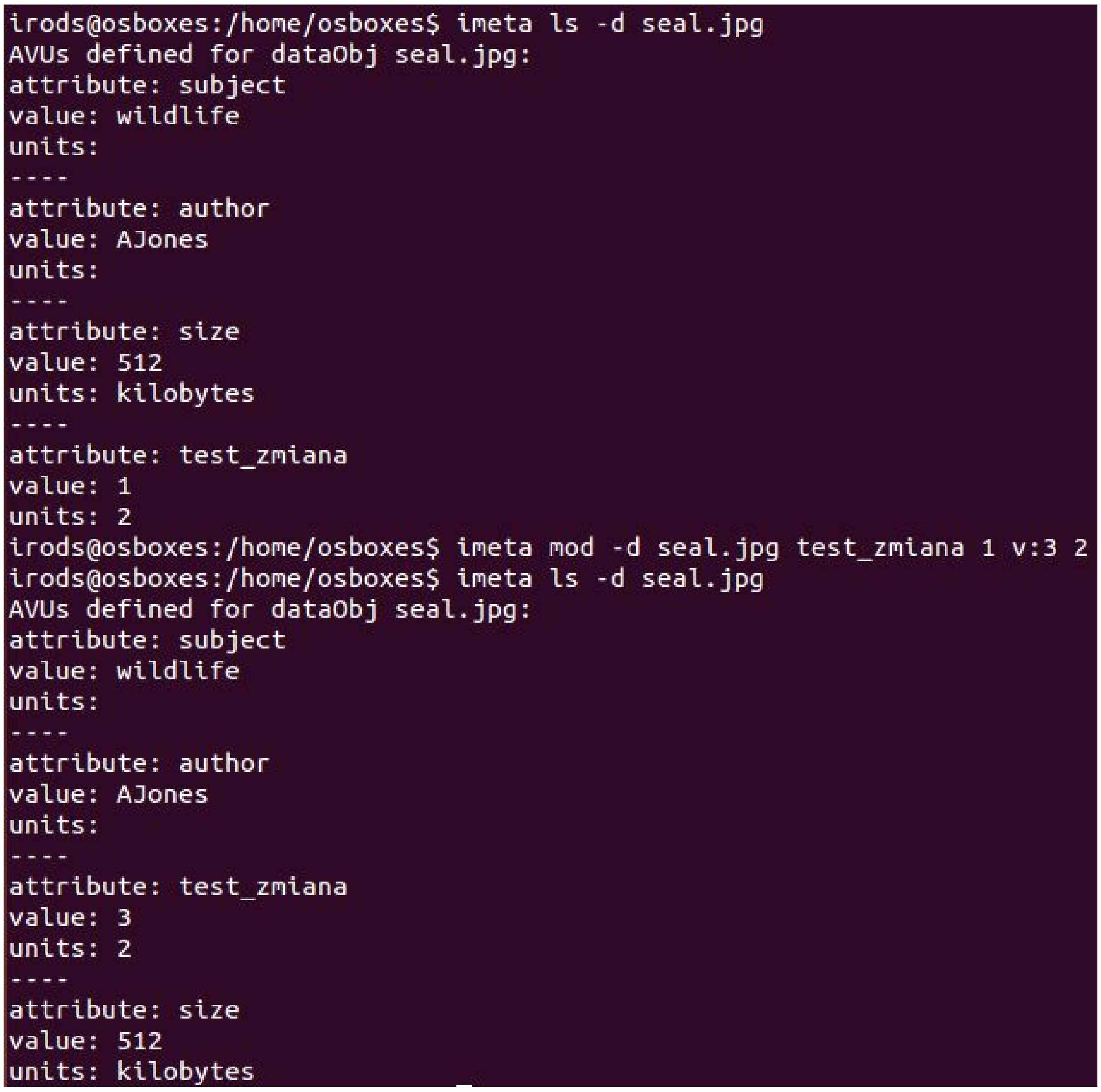
While using 'mod' command to modify metadata CAT_INVALID_ARGUMENT error occurs, if Units Attribute (AttUnits) is non existent. Even though AttUnits is optional, it is required to modify metadata, contrary to what iRODS manual states. Interestingly modifying metadata without unit works fine in cloud browser, but not in command line. Perhaps browser works around it by deleting and reinserting metadata.

Figure below shows the failed attempt of changing metadata without unit:



```
irods@osboxes:/home/osboxes$ imeta ls -d seal.jpg
AVUs defined for dataObj seal.jpg:
attribute: subject
value: wildlife
units:
----
attribute: author
value: AJones
units:
----
attribute: test_zmiana
value: 3
units: 2
----
attribute: size
value: 512
units: kilobytes
irods@osboxes:/home/osboxes$ imeta mod -d seal.jpg author AJones v:BSmith
remote addresses: 127.0.0.1 ERROR: rcModAVUMetadata failed with error -816000 CAT_INVALID_ARGUMENT
```

Below expected behaviour is shown - value of attribute ‘test_zmiana’ changes from 1 to 3:



```
irods@osboxes:/home/osboxes$ imeta ls -d seal.jpg
AVUs defined for dataObj seal.jpg:
attribute: subject
value: wildlife
units:
----
attribute: author
value: AJones
units:
----
attribute: size
value: 512
units: kilobytes
----
attribute: test_zmiana
value: 1
units: 2
irods@osboxes:/home/osboxes$ imeta mod -d seal.jpg test_zmiana 1 v:3 2
irods@osboxes:/home/osboxes$ imeta ls -d seal.jpg
AVUs defined for dataObj seal.jpg:
attribute: subject
value: wildlife
units:
----
attribute: author
value: AJones
units:
----
attribute: test_zmiana
value: 3
units: 2
----
attribute: size
value: 512
units: kilobytes
```

CentOS 7 authentication ambiguity
On CentOS 7 postgres 9.2 installation comes with “ident” as default authentication method, while iRODS expects to authenticate by password. This can fixed by changing “ident” to “md5” in pg_hba.conf file.

5. MetaInx

After meeting all the requirements (https://github.com/irods-contrib/metalnx-web/wiki/Dependencies), the installation proceeded according to the assumptions described in the link – Getting Started - https://github.com/irods-contrib/metalnx-web/wiki/Getting-Started. Installation and test on two systems Ubuntu 14.04 and Ubuntu 16.04.
Requirements.

On both systems, the required files were installed according to the instructions, with the exception of Java 1.8. For Ubuntu 14.04:

- sudo add-apt-repository ppa:webupd8team/java -y
- sudo apt-get update
- sudo apt-get install oracle-java8-installer.

PostgreSQL database was used for both systems.

Installation.

After following the next steps from the instructions, an error appears when firing the installer. On step 7/13 the installer should test the connection to the database using the given package - python-psycopg2. The connection, however, is not tested, the installer displays the message: “No DB connection test modules detected. Skipping DB connection test.”.

When negotiation with iRODS is set to use SSL:

- acPreConnect(*OUT) { *OUT="CS_NEG_DONT_CARE"; },

or

- acPreConnect(*OUT) { *OUT="CS_NEG_REQUIRE"; }

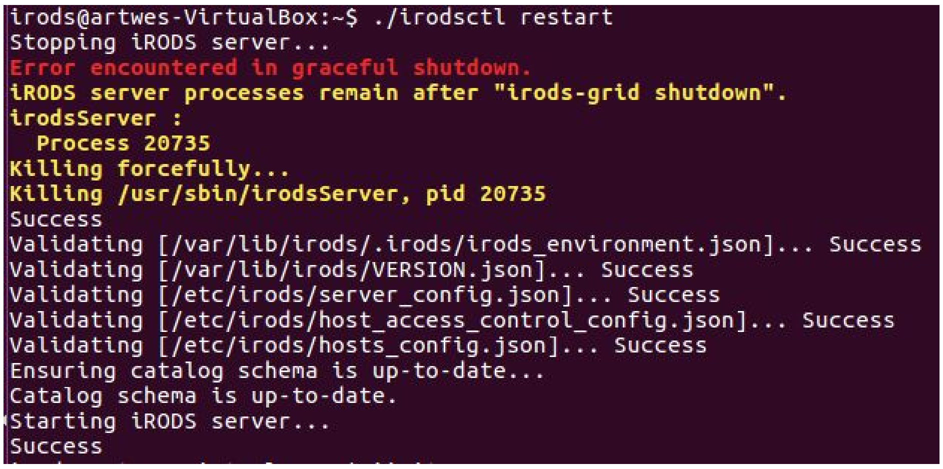
in core.re file (/etc/irods) the installation failed at step 11/13 – “Metalnx was not able to contact iRODS server. Check your parameters and try again.”.

With core.re set to not use SSL (“acPreConnect(*OUT) { *OUT="CS_NEG_DONT_CARE"; }”) installation process successful.

PAM & SSL Configuration

On PAM-&-SSL-Configuration there is post that explains how to configure Metalnx to work with an iRODS grid set up to use SSL and PAM.

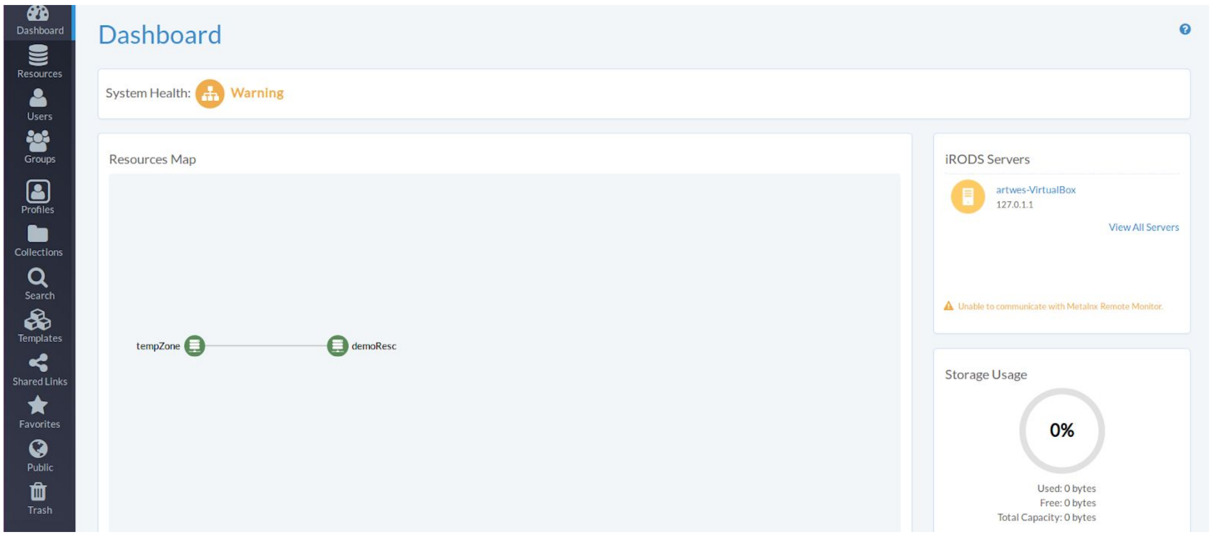
However, after a successful server reset there was a problem with logging in (iinit).



```
irods@artwes-VirtualBox:~$ ./irodsctl restart
Stopping iRODS server...
Error encountered in graceful shutdown.
iRODS server processes remain after "irods-grid shutdown".
irodsServer :
  Process 20735
Killing forcefully...
Killing /usr/sbin/irodsServer, pid 20735
Success
Validating [/var/lib/irods/.irods/irods_environment.json]... Success
Validating [/var/lib/irods/VERSION.json]... Success
Validating [/etc/irods/server_config.json]... Success
Validating [/etc/irods/host_access_control_config.json]... Success
Validating [/etc/irods/hosts_config.json]... Success
Ensuring catalog schema is up-to-date...
Catalog schema is up-to-date.
Starting iRODS server...
Success
```

- remote addresses: 127.0.1.1 ERROR: _rcConnect: connectToRhost error, server on localhost:1247 is probably down status = -1824000 CLIENT_NEGOTIATION_ERROR
- remote addresses: 127.0.1.1 ERROR: Saved password, but failed to connect to server localhost

UI



Dashboard

System Health: Warning

artwes-VirtualBox
127.0.1.1

Unable to communicate with Metalnx Remote Monitor.

0%

Used: 0 bytes
Free: 0 bytes
Total Capacity: 0 bytes

Web application provides a graphical UI that can help simplify most administration, collection management, and metadata management tasks.

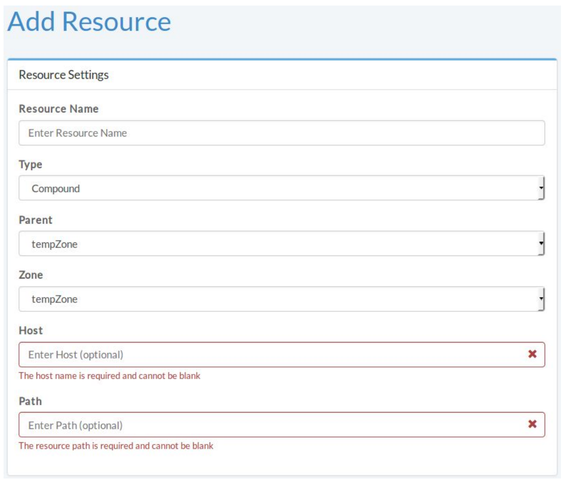
The basic elements of UI are:

Dashboard

Shows Resources Map, current iRODS Servers, Storage Usage and System Health.

Resources

Shows current resources, allows you to create new resources with a given name, type, parent, zone and optional host and path.



Add Resource

Resource Settings

Resource Name: Enter Resource Name

Type: Compound

Parent: tempZone

Zone: tempZone

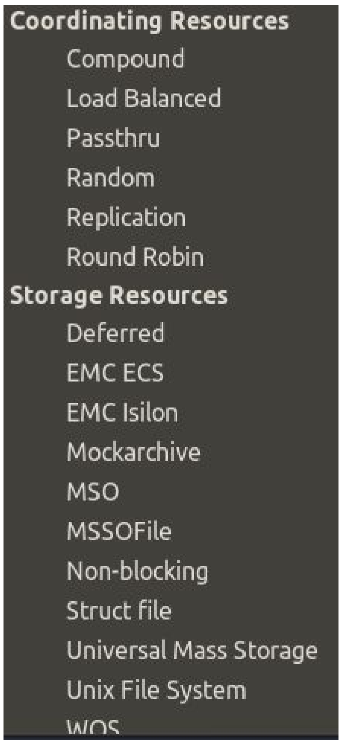
Host: Enter Host (optional)

The host name is required and cannot be blank

Path: Enter Path (optional)

The resource path is required and cannot be blank

Resources types:



Coordinating Resources
- Compound
- Load Balanced
- Passthru
- Random
- Replication
- Round Robin

Storage Resources
- Deferred
- EMC ECS
- EMC Isilon
- Mockarchive
- MSO
- MSSOFile
- Non-blocking
- Struct file
- Universal Mass Storage
- Unix File System
- WOS

Users

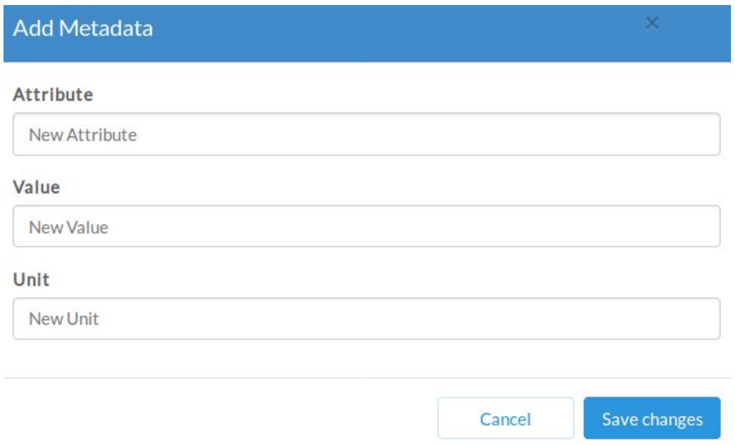
Allows adding / editing / deleting irods users.

Groups

Allows adding / editing / deleting user groups. Change of permissions for given groups, and downloading information about the group in the form of a csv document.

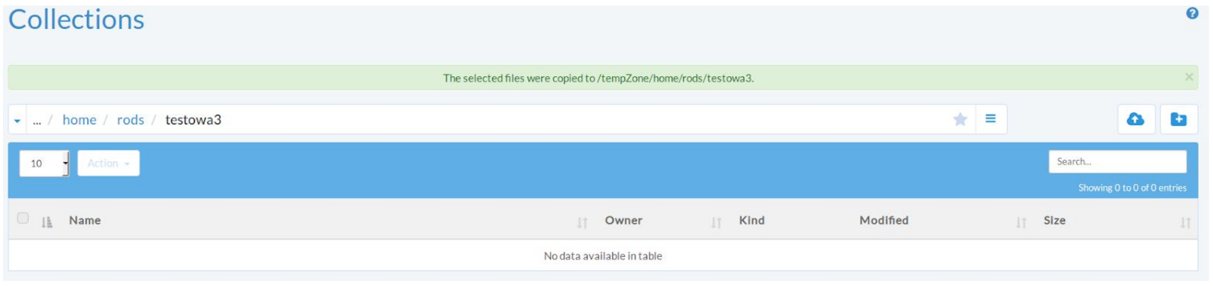
Collections

It allows you to manage data collections with the ability to freely configure metadata for individual files.



All functions, including the ability to search for files based on their metadata, worked correctly except for the Collections tab. Although the application provides for the transfer / copying of a given file to a given collection, it stays in the same place. Restarting the installer or the system does not bring any effects.

At some point, it was impossible to add new files to both old and newly created collections. Although operations on the metadata of previously attached files still work.



The selected files were copied to /tempZone/home/rods/testowa3.

Showing 0 to 0 of 0 entries

No data available in table

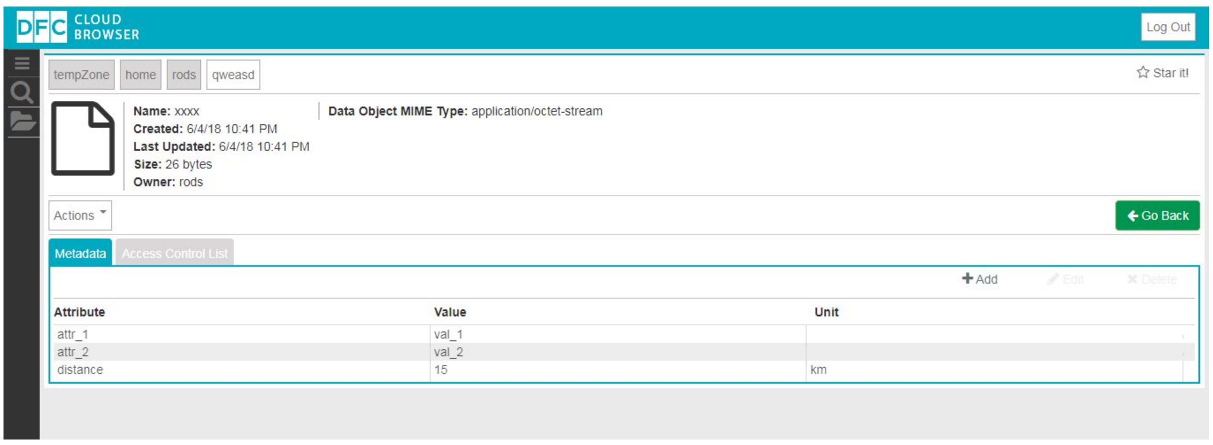
Other known issues can be find at https://github.com/irods-contrib/metalnx-web/issues/.

6. iRODS Cloud Browser

Cloud browser is web application, which allows navigation and performing actions on files stored in iRODS. Two versions were tested: 1.0.2.0-beta4 (https://github.com/DICE-UNC/irods-cloud-browser/tree/1.0.2.0-beta4) and 1.0.1-RELEASE (https://github.com/DICE-UNC/irods-cloud-browser/tree/1.0.1-RELEASE).

Release version allows browsing files managed by iRODS and simple operations on them like uploading, copying, renaming, deleting. It also allows for performing operations on metadata. It consists of two independently deployed parts: javascript frontend and JVM backend. In global.js file located in frontend application variable host needs to be configured to point to backend service (the only thing to remember is that this needs to be address from browser perspective, so in most cases this won't be 'localhost' even if both parts are deployed on the same machine). Loging form can be simplified by creating irods-cloud-backend.groovy file and setting there parameters like iRODS hostname and port, authentication method and iRODS zone.

Unlike release version, beta version is deployed as single war file on tomcat. Default SSL/TLS configuration of this version (CS_NEG_DONT_CARE) is incompatible with default iRODS configuration. Documentation claims that this option is recommended because it makes application agree upon whatever server wants to do, but it does not work. For example for non-secure connection parameter 'beconf.negotiation.policy' in irods-cloud-backend.groovy file needs to be changed to CS_NEG_REFUSE.



Name: xxxx
Created: 6/4/18 10:41 PM
Last Updated: 6/4/18 10:41 PM
Size: 26 bytes
Owner: rods
Data Object MIME Type: application/octet-stream

Attribute	Value	Unit
attr_1	val_1	
attr_2	val_2	
distance	15	km

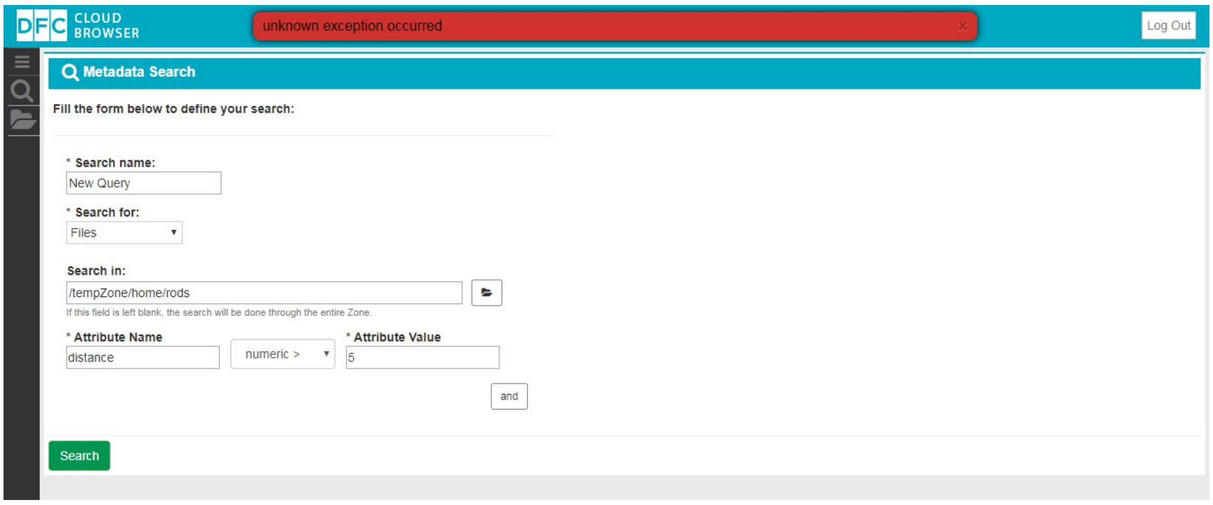
Modifying metadata in cloud browser 1.0.2.0-beta4

This version supports all previous features and additionally metadata-based file search and ACL management. When appropriate file extension is set (like .txt) application allows for in-browser editing files.

Problems encountered:

1. When two users are editing file in browser, application doesn't handle concurrent modification very well. Neither pessimistic, nor optimistic locking is used, in consequence lost update problem can occur.
2. When trying to use file search feature application is crashes (we were able to reproduce it only when deploying cloud browser and irods on the same server, so that irods hostname was set to localhost). After trying the most basic search query application displayed information about 'Unknown exception' and became irresponsive. Cloud browser logs provided no relevant

information about the exception and redeploying the war had no effect - one could not log into the application anymore.



unknown exception occurred

Metadata Search

Fill the form below to define your search:

* Search name: New Query

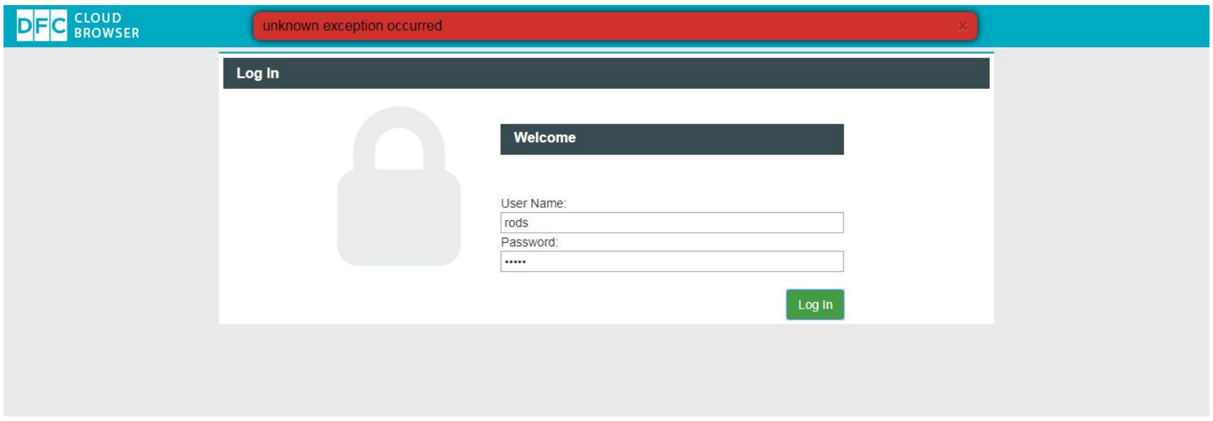
* Search for: Files

Search in: /tempZone/home/rods

If this field is left blank, the search will be done through the entire Zone.

* Attribute Name		* Attribute Value
distance	numeric >	5

Exception is thrown after hitting ‘Search’ button



unknown exception occurred

Log In

Welcome

User Name: rods

Password: •••••

After exception was thrown no user could login to application

On the irods side the following lines where logged in the rodsLog file:

```
May 21 20:10:28 pid:1494 remote addresses: ::1 ERROR: [-]
/tmp/tmp0dyJrt/server/core/src/rsApiHandler.cpp:318:int sendApiReply(rsComm_t *,
int, int, void *, bytesBuf_t *) :  status [SYS_HEADER_WRITE_LEN_ERR]  errno
[Broken pipe] -- message [wrote 0 expected 142]
        [-]     /tmp/tmp0dyJrt/lib/core/src/sockComm.cpp:1258:irods::error
sendRodsMsg(irods::network_object_ptr, const char *, bytesBuf_t *, bytesBuf_t *,
bytesBuf_t *, int, irodsProt_t) :  status [SYS_HEADER_WRITE_LEN_ERR]  errno
[Broken pipe] -- message [wrote 0 expected 142]
                [-]     /tmp/tmp0dyJrt/plugins/network/tcp/libtcp.cpp:355:irods::error
tcp_send_rods_msg(irods::plugin_context &, const char *, bytesBuf_t *, bytesBuf_t *,
bytesBuf_t *, int, irodsProt_t) :  status [SYS_HEADER_WRITE_LEN_ERR]  errno
[Broken pipe] -- message [wrote 0 expected 142]
                        [-]     /tmp/tmp0dyJrt/lib/core/src/sockComm.cpp:475:irods::error
writeMsgHeader(irods::network_object_ptr, msgHeader_t *) :  status
[SYS_HEADER_WRITE_LEN_ERR]  errno [Broken pipe] -- message [wrote 0
expected 142]
                                [-]
/tmp/tmp0dyJrt/plugins/network/tcp/libtcp.cpp:293:irods::error
tcp_write_msg_header(irods::plugin_context &, bytesBuf_t *) :  status
[SYS_HEADER_WRITE_LEN_ERR]  errno [Broken pipe] -- message [wrote 0
expected 142]

May 21 20:10:28 pid:1495 remote addresses: ::1 ERROR: [-]
/tmp/tmp0dyJrt/server/core/src/rsApiHandler.cpp:318:int sendApiReply(rsComm_t *,
int, int, void *, bytesBuf_t *) :  status [SYS_HEADER_WRITE_LEN_ERR]  errno
[Broken pipe] -- message [wrote 0 expected 142]
        [-]     /tmp/tmp0dyJrt/lib/core/src/sockComm.cpp:1258:irods::error
sendRodsMsg(irods::network_object_ptr, const char *, bytesBuf_t *, bytesBuf_t *,
bytesBuf_t *, int, irodsProt_t) :  status [SYS_HEADER_WRITE_LEN_ERR]  errno
[Broken pipe] -- message [wrote 0 expected 142]
                [-]     /tmp/tmp0dyJrt/plugins/network/tcp/libtcp.cpp:355:irods::error
tcp_send_rods_msg(irods::plugin_context &, const char *, bytesBuf_t *, bytesBuf_t *,
bytesBuf_t *, int, irodsProt_t) :  status [SYS_HEADER_WRITE_LEN_ERR]  errno
[Broken pipe] -- message [wrote 0 expected 142]
                        [-]     /tmp/tmp0dyJrt/lib/core/src/sockComm.cpp:475:irods::error
writeMsgHeader(irods::network_object_ptr, msgHeader_t *) :  status
[SYS_HEADER_WRITE_LEN_ERR]  errno [Broken pipe] -- message [wrote 0
expected 142]
                                [-]
/tmp/tmp0dyJrt/plugins/network/tcp/libtcp.cpp:293:irods::error
tcp_write_msg_header(irods::plugin_context &, bytesBuf_t *) :  status
[SYS_HEADER_WRITE_LEN_ERR]  errno [Broken pipe] -- message [wrote 0
expected 142]

May 21 20:10:28 pid:1495 remote addresses: ::1 ERROR: [-]
/tmp/tmp0dyJrt/server/core/src/rsApiHandler.cpp:540:int
readAndProcClientMsg(rsComm_t *, int) :  status
[SYS_HEADER_READ_LEN_ERR]  errno [] -- message [only read [0] of [4]]
        [-]     /tmp/tmp0dyJrt/lib/core/src/sockComm.cpp:201:irods::error
readMsgHeader(irods::network_object_ptr, msgHeader_t *, struct timeval *) :  status
[SYS_HEADER_READ_LEN_ERR]  errno [] -- message [only read [0] of [4]]
```

```
                [-]     /tmp/tmp0dyJrt/plugins/network/tcp/libtcp.cpp:194:irods::error
tcp_read_msg_header(irods::plugin_context &, void *, struct timeval *) :  status
[SYS_HEADER_READ_LEN_ERR]  errno [] -- message [only read [0] of [4]]

May 21 20:10:28 pid:1494 remote addresses: ::1 ERROR: [-]
/tmp/tmp0dyJrt/server/core/src/rsApiHandler.cpp:540:int
readAndProcClientMsg(rsComm_t *, int) :  status
[SYS_HEADER_READ_LEN_ERR]  errno [] -- message [only read [0] of [4]]
        [-]     /tmp/tmp0dyJrt/lib/core/src/sockComm.cpp:201:irods::error
readMsgHeader(irods::network_object_ptr, msgHeader_t *, struct timeval *) :  status
[SYS_HEADER_READ_LEN_ERR]  errno [] -- message [only read [0] of [4]]
                [-]     /tmp/tmp0dyJrt/plugins/network/tcp/libtcp.cpp:194:irods::error
tcp_read_msg_header(irods::plugin_context &, void *, struct timeval *) :  status
[SYS_HEADER_READ_LEN_ERR]  errno [] -- message [only read [0] of [4]]

May 21 20:10:28 pid:1495  ERROR: Agent [1495] exiting with status = -4000
May 21 20:10:28 pid:1494  ERROR: Agent [1494] exiting with status = -4000
May 21 20:10:28 pid:1074  ERROR: Agent process [1494] exited with status [96]
May 21 20:10:28 pid:1074  ERROR: Agent process [1495] exited with status [96]
```

Problem seems to lie on application side - after the incident iRODS console operations worked as before, another instance of cloud browser running on another machine was also fine.

7. REST API

Project irods-rest (https://github.com/DICE-UNC/irods-rest) allows accessing iRODS via REST. Application is provided as war file which is to deployed on servlet container (it is mostly tested on tomcat). File **irods-rest.properties** needs to be created in **/etc/irods-ext/** directory and populated with appropriate parameters (example) Basic HTTP authentication is used to authenticate the user with their iRODS username and password. By default XML response type is chosen, in order to switch to JSON query param *contentType=application/json* needs to be set.

Basic GET operations:

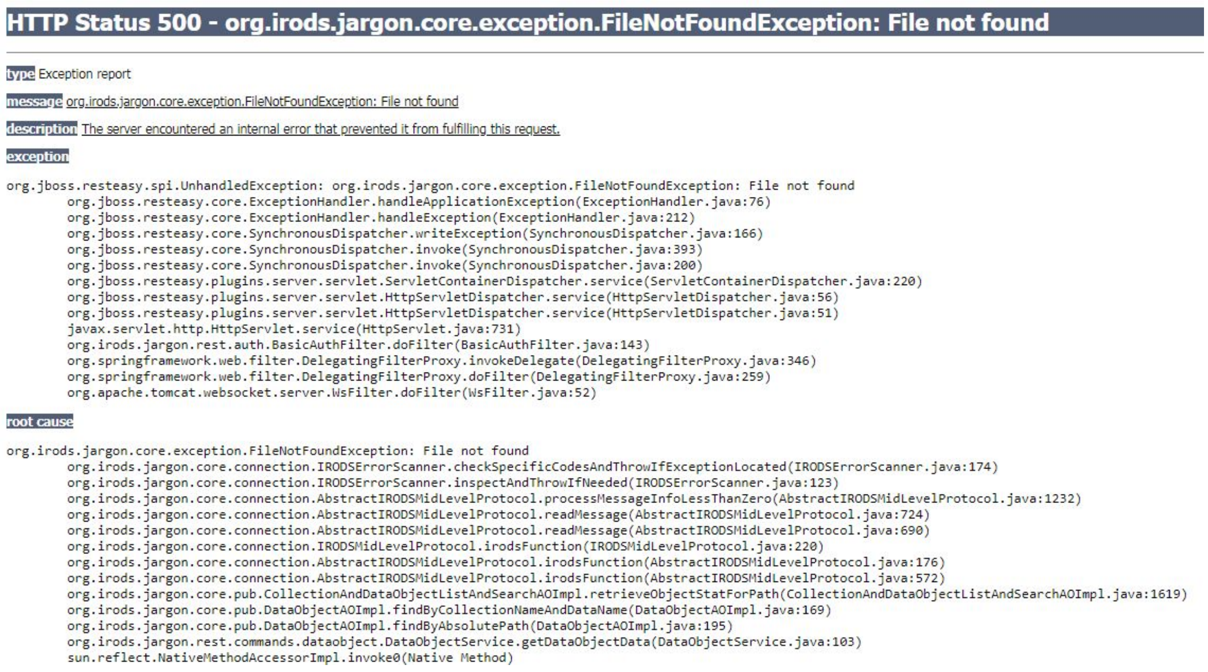
Below URLs should prefixed with protocol name, hostname and port. We also assume that we operate on “tempZone” zone.

Get information about /home/rods directory in zone “tempZone” and items contained	GET irods-rest/rest/collection/tempZone/home/rods?listing=true
Displaying information about /home/rods/qwe.txt file	GET irods-rest/rest/dataObject/tempZone/home/rods/qwe.txt
Downloading /home/rods/qwe.txt file	GET /irods-rest/rest/fileContents/tempZone/home/rods/qwe.txt
Displaying metadata of /home/rods/qwe.txt file	GET /irods-rest/rest/dataObject/tempZone/home/rods/asd.txt/metadata

Adding/modifying/deleting file/directory/metadata is done similarly using other HTTP methods (POST, PUT) based on information provided in request body. Querying objects based on attributes like object name, collection name, metadata, file size etc. is supported. More informations can be found in docs: https://github.com/DICE-UNC/irods-rest/blob/4.1.10.0-RC1/docs/iRODSRESTAPIDocumentation.pdf

Problems encountered:

Application seem to not handle exceptions at all. For example when trying to access non-existent file status 500 (server error) is returned and response contains exception stacktrace generated by tomcat in HTML format, instead of XML or JSON.



```
HTTP Status 500 - org.irods.jargon.core.exception.FileNotFoundException: File not found

type Exception report

message org.irods.jargon.core.exception.FileNotFoundException: File not found

description The server encountered an internal error that prevented it from fulfilling this request.

exception

org.jboss.resteasy.spi.UnhandledException: org.irods.jargon.core.exception.FileNotFoundException: File not found
	org.jboss.resteasy.core.ExceptionHandler.handleApplicationException(ExceptionHandler.java:76)
	org.jboss.resteasy.core.ExceptionHandler.handleException(ExceptionHandler.java:212)
	org.jboss.resteasy.core.SynchronousDispatcher.writeException(SynchronousDispatcher.java:166)
	org.jboss.resteasy.core.SynchronousDispatcher.invoke(SynchronousDispatcher.java:393)
	org.jboss.resteasy.core.SynchronousDispatcher.invoke(SynchronousDispatcher.java:200)
	org.jboss.resteasy.plugins.server.servlet.ServletContainerDispatcher.service(ServletContainerDispatcher.java:220)
	org.jboss.resteasy.plugins.server.servlet.HttpServletDispatcher.service(HttpServletDispatcher.java:56)
	org.jboss.resteasy.plugins.server.servlet.HttpServletDispatcher.service(HttpServletDispatcher.java:51)
	javax.servlet.http.HttpServlet.service(HttpServlet.java:731)
	org.irods.jargon.rest.auth.BasicAuthFilter.doFilter(BasicAuthFilter.java:143)
	org.springframework.web.filter.DelegatingFilterProxy.invokeDelegate(DelegatingFilterProxy.java:346)
	org.springframework.web.filter.DelegatingFilterProxy.doFilter(DelegatingFilterProxy.java:259)
	org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:52)

root cause

org.irods.jargon.core.exception.FileNotFoundException: File not found
	org.irods.jargon.core.connection.IRODSErrorScanner.checkSpecificCodesAndThrowIfExceptionLocated(IRODSErrorScanner.java:174)
	org.irods.jargon.core.connection.IRODSErrorScanner.inspectAndThrowIfNeeded(IRODSErrorScanner.java:123)
	org.irods.jargon.core.connection.AbstractIRODSMidLevelProtocol.processMessageInfoLessThanZero(AbstractIRODSMidLevelProtocol.java:1232)
	org.irods.jargon.core.connection.AbstractIRODSMidLevelProtocol.readMessage(AbstractIRODSMidLevelProtocol.java:724)
	org.irods.jargon.core.connection.AbstractIRODSMidLevelProtocol.readMessage(AbstractIRODSMidLevelProtocol.java:690)
	org.irods.jargon.core.connection.IRODSMidLevelProtocol.irodsFunction(IRODSMidLevelProtocol.java:220)
	org.irods.jargon.core.connection.AbstractIRODSMidLevelProtocol.irodsFunction(AbstractIRODSMidLevelProtocol.java:176)
	org.irods.jargon.core.connection.AbstractIRODSMidLevelProtocol.irodsFunction(AbstractIRODSMidLevelProtocol.java:572)
	org.irods.jargon.core.pub.CollectionAndDataObjectListAndSearchAOImpl.retrieveObjectStatForPath(CollectionAndDataObjectListAndSearchAOImpl.java:1619)
	org.irods.jargon.core.pub.DataObjectAOImpl.findByCollectionNameAndDataName(DataObjectAOImpl.java:169)
	org.irods.jargon.core.pub.DataObjectAOImpl.findByAbsolutePath(DataObjectAOImpl.java:195)
	org.irods.jargon.rest.commands.dataobject.DataObjectService.getDataObjectData(DataObjectService.java:103)
	sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
```

HTML response after trying to access to non-existent file

8. Summary assessment

Evaluation Criteria	Criteria Explanation	Score [1-5]	Result
Installation process	The product has been tested on several platforms - Ubuntu 14, Ubuntu 16, CentOS 7.		There is an instruction provided under https://docs.irods.org/4.2.2/getting_started/installation/.
- Ubuntu 14		5	The process of installation ran without any problems. Provided documentation covered it precisely and the system was up and working in no time.
- Ubuntu 16		5	As above
- CentOS 7		5	As above
Metadata management (console)	Is it possible (and simple) to add/modify/delete metadata?		
creation		5	User can add metadata using imeta add command. No errors ever occured
modification		1	Modifying metadata did not always work while using imeta mod command - CAT_INVALID_ARGUMENT error occured while trying to modify metadata without Units Attribute.
removal		5	User can remove metadata using imeta rm

			command. No errors ever occured
search		5	User can search data objects using imeta qu command. No errors ever occured
Client interfaces	Integration and functionality.		
- Cloud Browser		3	Stable, older version supports only basic operations on file and metadata. More recent, beta version include also ACL management and querying files by metadata but the second one seems to still be unstable.
Metadata management			
creation		5	works as expected
modification		5	works as expected
removal		5	works as expected
search		2	In beta version when iRODS address is localhost search feature crashes whole application, in stable version it is not supported.
- MetaLnx	File metadata support, using iRODS functionality with GUI	3	The graphical interface is user-friendly, basic irods functions work properly up to a certain point. Although the application displays a message that the file has been successfully moved / copied to another collection, the file remains in the same place. Without any information about the error, the application prevents adding new files to the collection.
Metadata management			
creation	GUI allows you to define new attributes, values and units for files in the collection. The number of metadata declared in this way depends only on the user's preferences.	5	The presented functionality works without any problems.
modification	The GUI allows you to modify previously created metadata using the 'edit' button. Both attributes, values, and unit may change.	5	The change of data proceeds without any problems. Old data is swapped, there are no errors when overwriting.
removal	The user can delete individual metadata as well as selected metadata sets.	5	The selected metadata is removed, without any errors.

search	Metalnx allows you to search for files using metadata by specifying a particular attribute value and its unit. It is possible to search for attributes with the value equal / different / having / not having the searched value. It is also possible to enter additional criteria to search for files containing more metadata.	5	There were no problems when searching for files with given attributes.
- REST API			
Metadata management			
creation		5	works as expected, uses PUT method
modification		1	Apparently this is not supported as single operation.
removal		5	works as expected, uses POST method (it relies on request body, which DELETE requests don’t have)
search		5	Basic search queries work as expected. Syntax of queries is relatively easy to pick-up using documentation.
Security	Checking different authentication methods (Standard, GSI, PAM, Kerberos).	4	Only standard authentication (password) was tested. The documentation describes also other methods of authentications, but they were not tested. Authorization is supported via ACL similar to those in UNIX. External Authorization is not possible.

Bibliography:

- https://en.wikipedia.org/wiki/Object_storage
- https://searchstorage.techtarget.com/tip/Advantages-of-using-an-object-storage-system
- https://irods.org/
- https://groups.google.com/forum/#!topic/irod-chat/zd5hErrSQ-E
- https://github.com/DICE-UNC/irods-cloud-browser
- https://github.com/DICE-UNC/irods-rest