libnova
the most advanced digital preservation platform

LABDRIVE
Research Data Management

Platform training

LIBNOVA EU Paseo de la Castellana 153 - 28046 Madrid, Spain - Tel: +34 91 449 08 94
LIBNOVA USA 14 NE First Ave (2nd floor) - Miami, Florida 33132, USA - Tel: +1 844-894-6532
contact@libnova.com

Agenda

LABDRIVE introduction	(60 mins)	9:30-10:20
Architecture and overview	35 mins	9:30-10:05
How research content is to be organized	25 mins	10:05-10:30
Break	*15 mins*	*10:30-10:45*
Configuration	(70 mins)	10:45-12:10
Users and Permissions	15 mins	10:45-11:00
Archival organization	10 mins	11:00-11:10
Container – concept and usage	15 mins	11:10-11:25
Introduction to metadata – concept and usage (container, item & tags)	15 mins	11:25-11:40
Metadata configuration	15 mins	11:40-11:55
Break	*15 mins*	*11:55-12:10*
Operations	(110 mins)	12:10-13:20
Create a data container	10 mins	12:10-12:20
Upload content	10 mins	12:20-12:30
Download content	10 mins	12:30-12:40
Searching	20 mins	12:40-13:00
Lunch Break	30 mins	13:00-13:30
LABDRIVE functions & workflows	20 mins	13:30-13:50
Storage mode transitions	10 mins	13:50-14:00
Reports	10 mins	14:00-14:10
Advanced operations – Jupyter Notebooks & reproducibility	20 mins	14:10-14:30
Q&A & Conclusions	(15 mins)	14:30-14:45

Main concepts and platform benefits

Adapting it to your needs

Day to day operation

libnova

About LIBNOVA

About LIBNOVA

LIBNOVA is market leader in digital preservation and digital content archiving.

Organizations use our solutions **to safeguard and provide access** to their valuable digital assets for the long term, either by using our cloud platform or by deploying our software on their own premises.

Present in 17 countries. Some of the largest and **most demanding** organizations worldwide are using our platforms.

Founded in 2009. **Modern** architecture. **Massively adopted** during the last 5 years.

Self-sustained company, no risk capital, no debt.

We work with them with a **long-term, partnership approach**.

LABDRIVE Introduction: Overview

Introduction

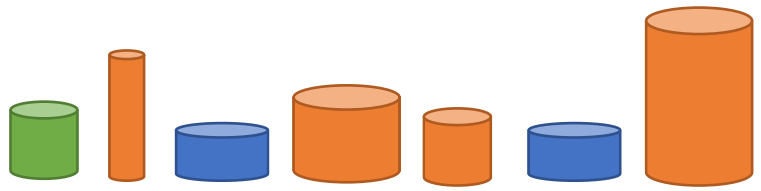
LABDRIVE is a **Research Data Management** and Preservation platform.

Its design principles are:

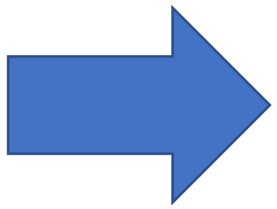
- Unified: It enables organizations to **combine** their research content **in a single, unified, rationalized platform**.
- Comprehensive: It allows organizations to **capture the research** data they produce, helping them to **properly manage, preserve and allow access to it**, during the whole data lifecycle (and not only at the end of the cycle).
- Open: ISO16363, ISO27001, ISO27017, ISO27018 certified. 100% of the information cab be easily exported. No exit barriers. User-extensible.

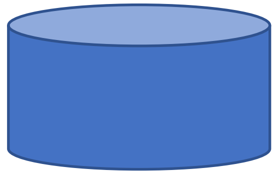
Unified

LABDRIVE allows organizations to **organize and unify** their content:



Transition from **a siloed approach** in which each series of datasets, experiments, departments or units are using multiple, disaggregated systems to keep content



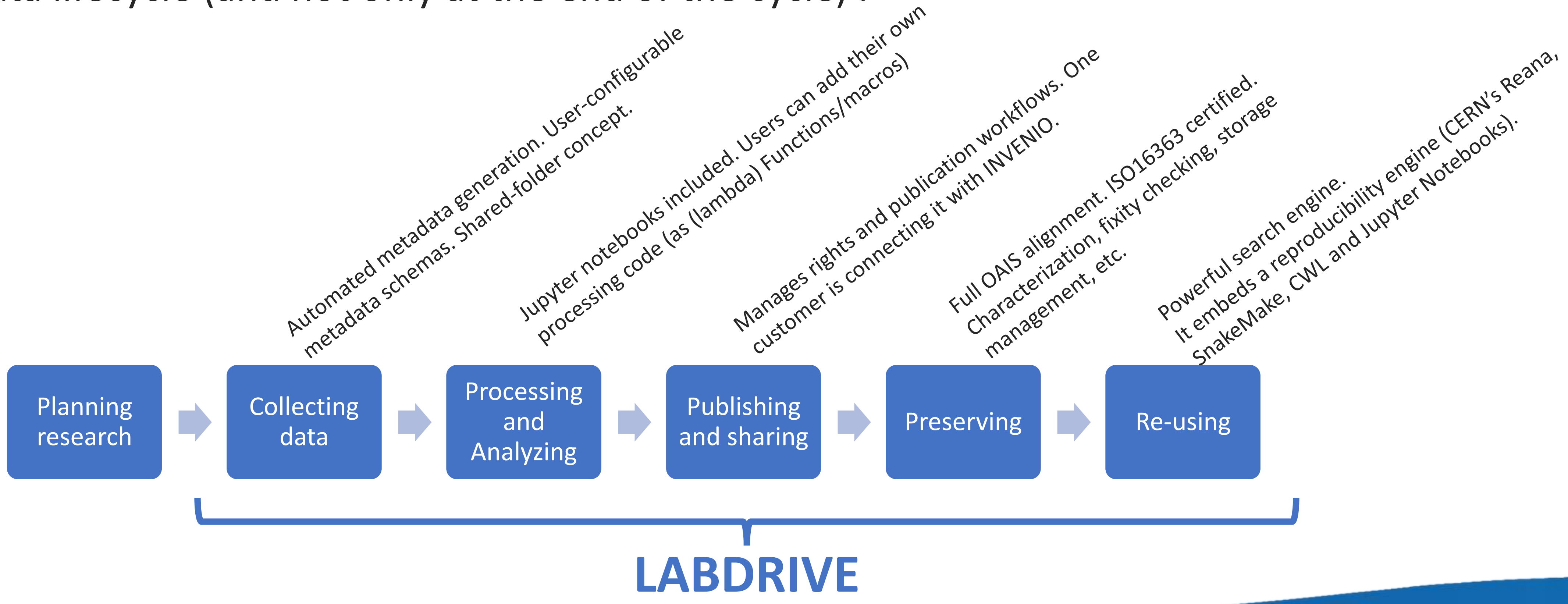


To a **unified repository** that can adapt to the particularities of each dataset, **unifying all content in a single platform**.

- **Risk management**
- **Publishing/rights management**
- **Permissions control**
- **Policies management**
- **Infrastructure management**
- **Cost control**

LABDRIVE is a comprehensive solution:

- Management, not only archiving: It allows organizations to **capture the research** data they produce, helping them to **properly manage, preserve and allow access to it**, during the whole data lifecycle (and not only at the end of the cycle) .


Automated metadata generation. User-configurable metadata schemas. Shared-folder concept.

Jupyter notebooks included. Users can add their own processing code (as (lambda) Functions/macros)

Manages rights and publication workflows. One customer is connecting it with INVENIO.

Full OAIS alignment. ISO16363 certified. Characterization, fixity checking, storage management, etc.

Powerful search engine. It embeds a reproducibility engine (CERN's Reana, SnakeMake, CWL and Jupyter Notebooks).

LABDRIVE is a comprehensive solution:

- Unique approach to package generation: Packages don’t need to be fully defined for ingestion. They can be enriched and modified over time. LABDRIVE manages the complexity.
- Adapts to different needs: Organizations can create working areas with different access methods, metadata schemas, workflows, permissions, rights, storage and cost.
- Scales: 15PB ingestion has been demonstrated (at 500TB/day ingestion rate)

Open

LABDRIVE is based on open standards and has a clear exit path.

- Externally certified alignment to OAIS and ISO16363: ISO16363 is the **gold standard** in digital preservation. LABDRIVE is getting certified in July 2022. Platform allows to create fully supports the OAIS Information model.
- Fully aligned with the TRUST and FAIR principles.
- The platform is certified and externally audited in some of the most demanding security and compliance standards: ISO27001, ISO27017, ISO27018.
- 100% of the information can be retrieved at any time. Can run in the cloud or on-premises.

Recap

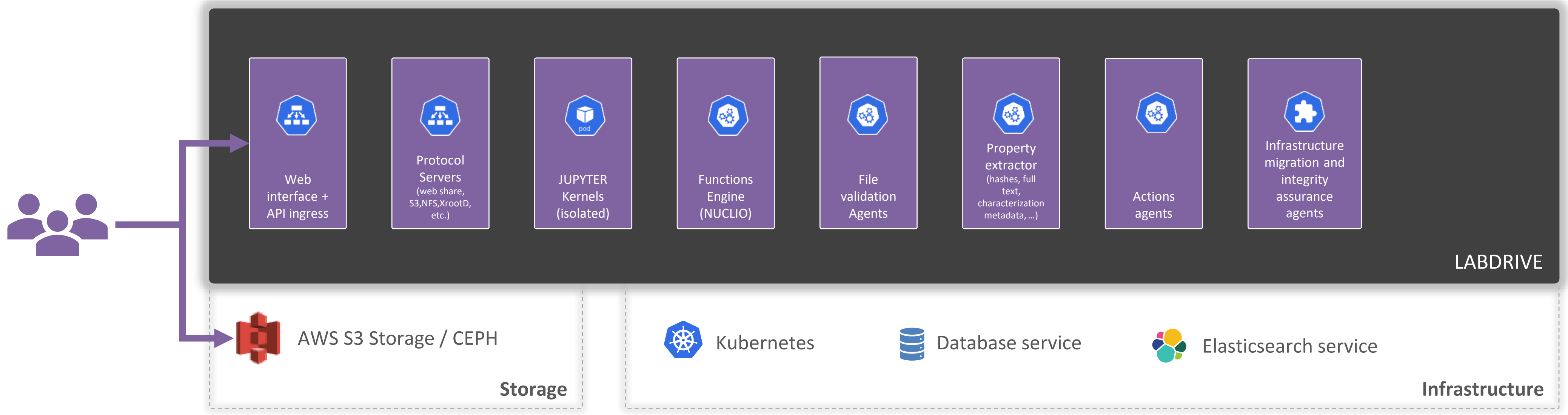
LABDRIVE is a **Research Data Management** and Preservation platform.

Its design principles are:

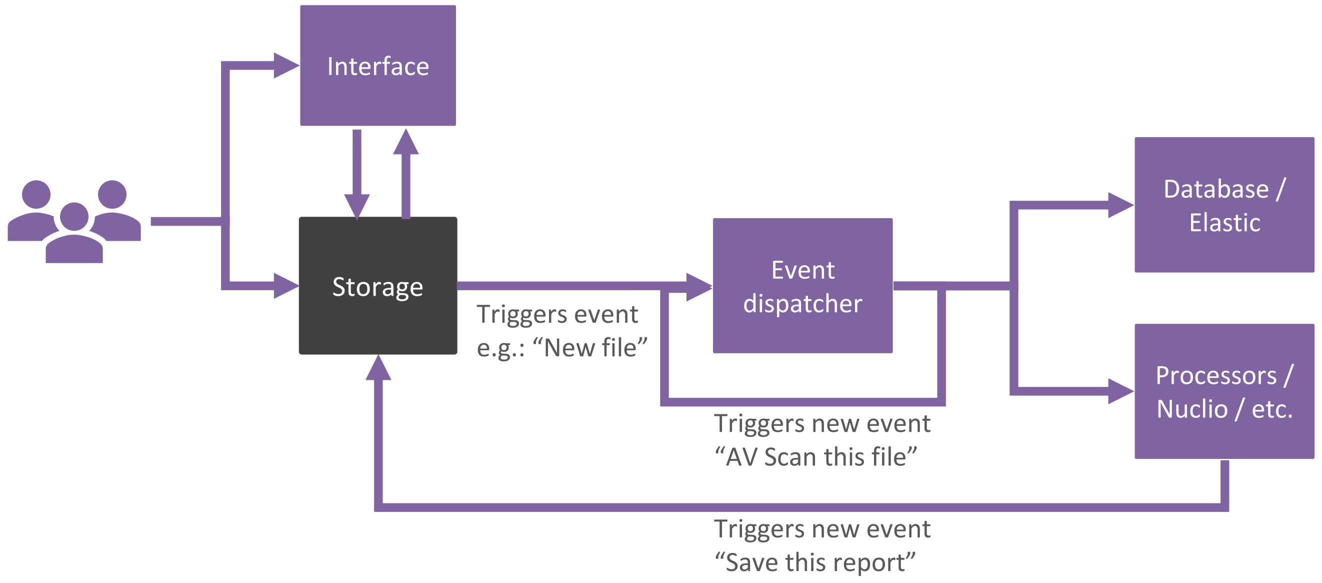
- Unified: It enables organizations to **combine** their research content **in a single, unified, rationalized platform**.
- Comprehensive: It allows organizations to **capture the research** data they produce, helping them to **properly manage, preserve and allow access to it**, during the whole data lifecycle (and not only at the end of the cycle).
- Open: ISO16363, ISO27001, ISO27017, ISO27018 certified. 100% of the information cab be easily exported. No exit barriers.

LABDRIVE Introduction: Architecture

Architecture


Web interface + API ingress
Protocol Servers (web share, S3,NFS,XrootD, etc.)
JUPYTER Kernels (isolated)
Functions Engine (NUCLIO)
File validation Agents
Property extractor (hashes, full text, characterization metadata, ...)
Actions agents
Infrastructure migration and integrity assurance agents
LABDRIVE
AWS S3 Storage / CEPH
Storage
Kubernetes
Database service
Elasticsearch service
Infrastructure

Cloud-native, events-oriented architecture



- Users are able to talk directly and transparently with the AWS S3 storage

  -> The platform becomes massively scalable (up/downloads)

  -> Massively interoperable. Anything that works with S3 can be used to preserve or to interact with preserved data: scripting, SDKs, etc. but also Analytics tools, reporting, reproducibility environments, etc. Thousands of components can be easily connected.

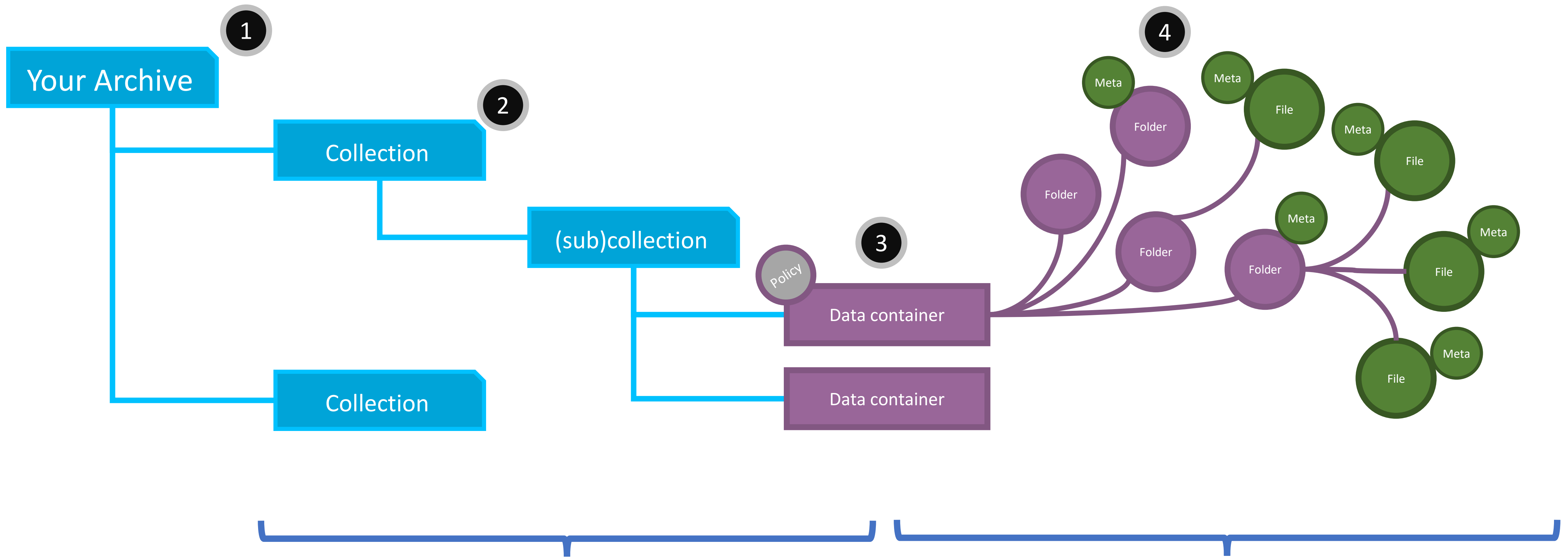
LABDRIVE Introduction:
How research content is to be organized

How research content is to be organized

- LABDRIVE offers a great degree of flexibility on how information is organized when in the platform, and it is able to adapt to any kind of data structure.
- In LABDRIVE, there is **no imposed data structure**: organizations and users are in charge of defining their own data models for their content.
- What are the tools LABDRIVE provide for you to do so?
  - Archival structure/nodes
  - Data containers
  - File/folder structures
  - Per-item metadata
  - Workflows (Functions)
  - Compliance reports

How research content is to be organized

Archival structure/nodes. Data containers. File/folder structures and Per-item metadata

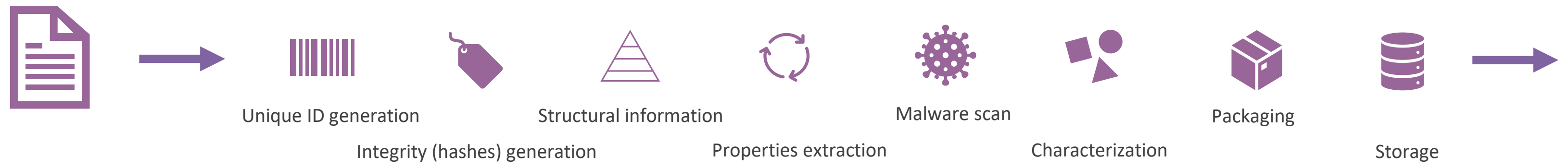


Unit, department, teams, groups, content types…

Objects, experiments, packages, files….

How research content is to be organized

Workflows



- Fine with default (No extended workflow)
- Or things like:
    - Extract TAR then
    - Validate BagIt contents then
    - Verify integrity then
    - Load technical metadata from JSON then
    - Create report

Default workflow

Extended workflows (optional)

How research content is to be organized

Compliance reports

- In general, LABDRIVE adopts the simplicity and flexibility as paradigms.

- For example:
  - Instead of making a metadata field mandatory, we aim at creating a metadata compliance report, that highlights when mandatory fields are not there.
  - Instead of forbidding to upload a certain file format, we aim at using a file format report, that highlights the file formats that should not be used.

  - Instead of forbidding the upload of malware, the platform reports it.

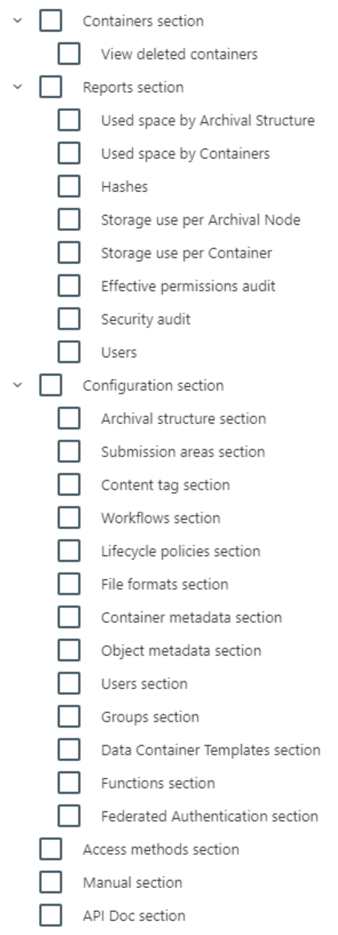
LABDRIVE Configuration

Configuration

- Users and permissions
    - Groups
    - Application permissions
    - Content permissions
    - Federated auth
        - Automatic account creation
        - Automated account configuration
    - Permission audit:
        - Effective permissions
        - Users
    - Security audit

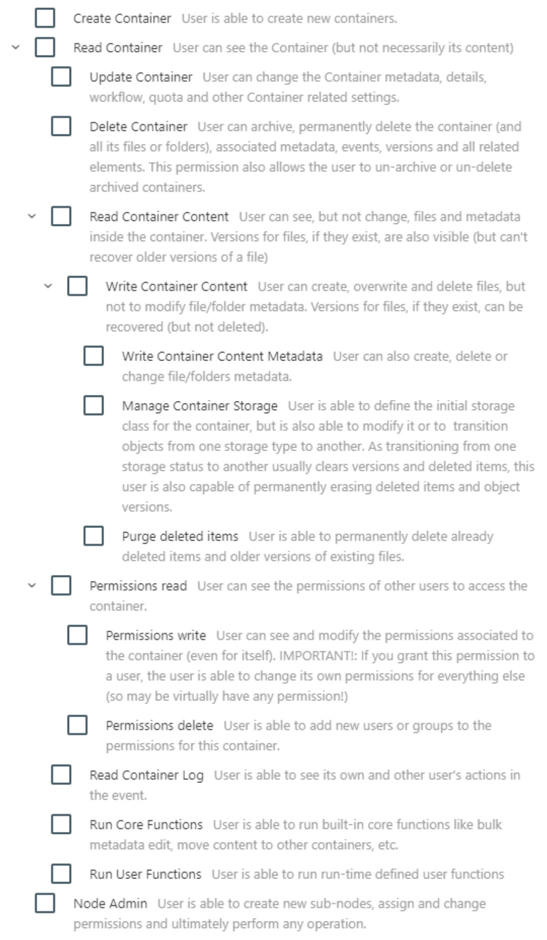
Permissions

BY APPLICATION



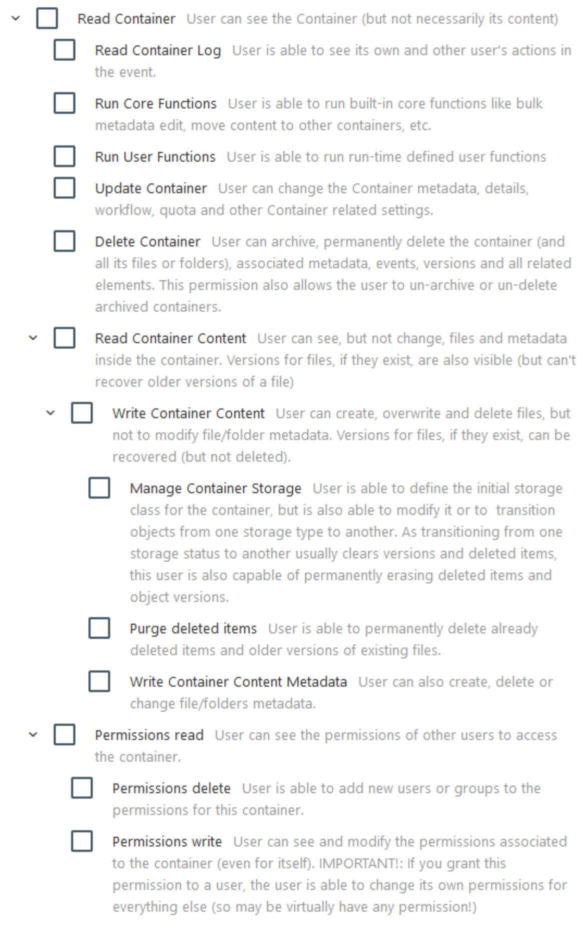
- ☐ Containers section
  - ☐ View deleted containers
- ☐ Reports section
  - ☐ Used space by Archival Structure
  - ☐ Used space by Containers
  - ☐ Hashes
  - ☐ Storage use per Archival Node
  - ☐ Storage use per Container
  - ☐ Effective permissions audit
  - ☐ Security audit
  - ☐ Users
- ☐ Configuration section
  - ☐ Archival structure section
  - ☐ Submission areas section
  - ☐ Content tag section
  - ☐ Workflows section
  - ☐ Lifecycle policies section
  - ☐ File formats section
  - ☐ Container metadata section
  - ☐ Object metadata section
  - ☐ Users section
  - ☐ Groups section
  - ☐ Data Container Templates section
  - ☐ Functions section
  - ☐ Federated Authentication section
- ☐ Access methods section
- ☐ Manual section
- ☐ API Doc section

BY NODE



- ☐ Create Container User is able to create new containers.
- ☐ Read Container User can see the Container (but not necessarily its content)
  - ☐ Update Container User can change the Container metadata, details, workflow, quota and other Container related settings.
  - ☐ Delete Container User can archive, permanently delete the container (and all its files or folders), associated metadata, events, versions and all related elements. This permission also allows the user to un-archive or un-delete archived containers.
  - ☐ Read Container Content User can see, but not change, files and metadata inside the container. Versions for files, if they exist, are also visible (but can't recover older versions of a file)
    - ☐ Write Container Content User can create, overwrite and delete files, but not to modify file/folder metadata. Versions for files, if they exist, can be recovered (but not deleted).
      - ☐ Write Container Content Metadata User can also create, delete or change file/folders metadata.
      - ☐ Manage Container Storage User is able to define the initial storage class for the container, but is also able to modify it or to transition objects from one storage type to another. As transitioning from one storage status to another usually clears versions and deleted items, this user is also capable of permanently erasing deleted items and object versions.
      - ☐ Purge deleted items User is able to permanently delete already deleted items and older versions of existing files.
  - ☐ Permissions read User can see the permissions of other users to access the container.
    - ☐ Permissions write User can see and modify the permissions associated to the container (even for itself). IMPORTANT!: If you grant this permission to a user, the user is able to change its own permissions for everything else (so may be virtually have any permission!)
    - ☐ Permissions delete User is able to add new users or groups to the permissions for this container.
  - ☐ Read Container Log User is able to see its own and other user's actions in the event.
  - ☐ Run Core Functions User is able to run built-in core functions like bulk metadata edit, move content to other containers, etc.
  - ☐ Run User Functions User is able to run run-time defined user functions
- ☐ Node Admin User is able to create new sub-nodes, assign and change permissions and ultimately perform any operation.

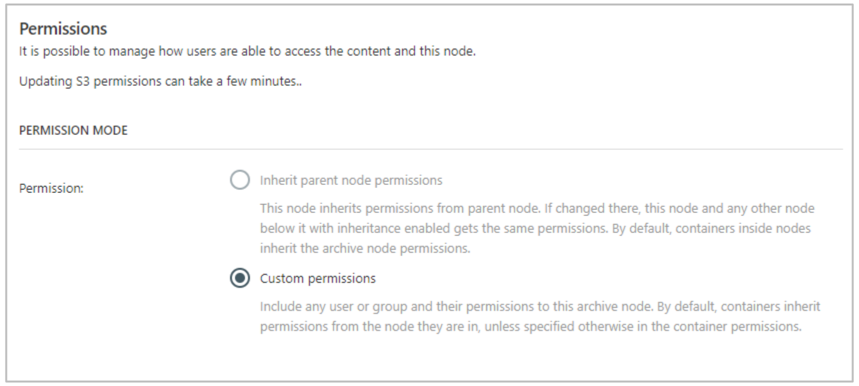
BY CONTAINER



- ☐ Read Container User can see the Container (but not necessarily its content)
  - ☐ Read Container Log User is able to see its own and other user's actions in the event.
  - ☐ Run Core Functions User is able to run built-in core functions like bulk metadata edit, move content to other containers, etc.
  - ☐ Run User Functions User is able to run run-time defined user functions
  - ☐ Update Container User can change the Container metadata, details, workflow, quota and other Container related settings.
  - ☐ Delete Container User can archive, permanently delete the container (and all its files or folders), associated metadata, events, versions and all related elements. This permission also allows the user to un-archive or un-delete archived containers.
  - ☐ Read Container Content User can see, but not change, files and metadata inside the container. Versions for files, if they exist, are also visible (but can't recover older versions of a file)
    - ☐ Write Container Content User can create, overwrite and delete files, but not to modify file/folder metadata. Versions for files, if they exist, can be recovered (but not deleted).
      - ☐ Manage Container Storage User is able to define the initial storage class for the container, but is also able to modify it or to transition objects from one storage type to another. As transitioning from one storage status to another usually clears versions and deleted items, this user is also capable of permanently erasing deleted items and object versions.
      - ☐ Purge deleted items User is able to permanently delete already deleted items and older versions of existing files.
      - ☐ Write Container Content Metadata User can also create, delete or change file/folders metadata.
  - ☐ Permissions read User can see the permissions of other users to access the container.
    - ☐ Permissions delete User is able to add new users or groups to the permissions for this container.
    - ☐ Permissions write User can see and modify the permissions associated to the container (even for itself). IMPORTANT!: If you grant this permission to a user, the user is able to change its own permissions for everything else (so may be virtually have any permission!)

Permissions

PERMISSION MODE



Permissions

It is possible to manage how users are able to access the content and this node.

Updating S3 permissions can take a few minutes..

PERMISSION MODE

Permission:

- [ ] Inherit parent node permissions

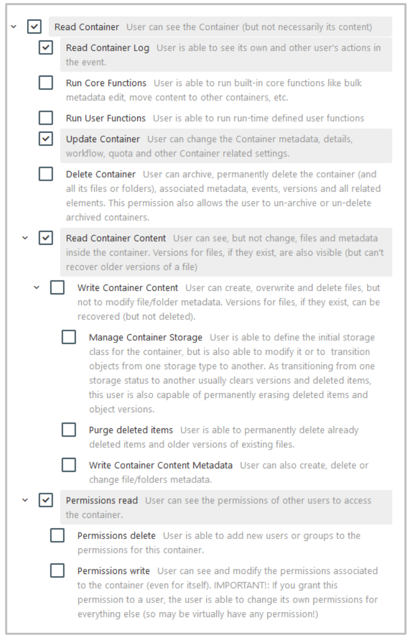
  This node inherits permissions from parent node. If changed there, this node and any other node below it with inheritance enabled gets the same permissions. By default, containers inside nodes inherit the archive node permissions.

- [✓ Custom permissions]

  Include any user or group and their permissions to this archive node. By default, containers inherit permissions from the node they are in, unless specified otherwise in the container permissions.

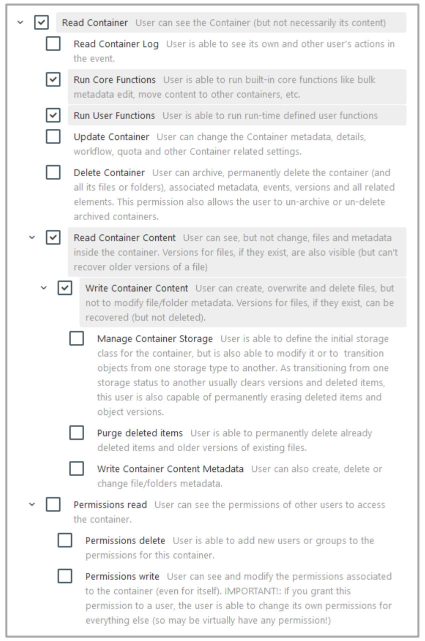
When creating sub-nodes or containers, the user can choose between inheriting the parent permissions or customise. Customised permissions **override** the permissions at a previous level (whether it is a root node, or a sub-node).

USERS AND GROUPS



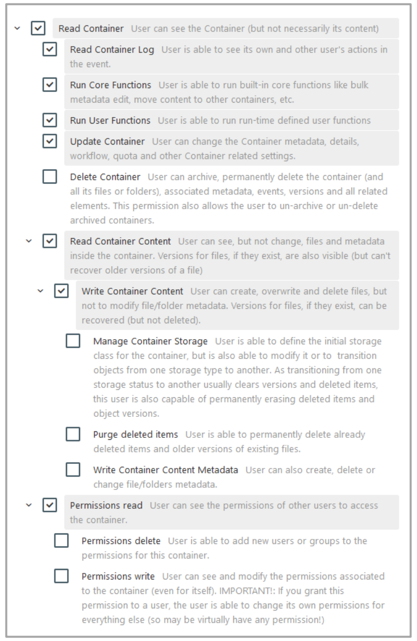
- [x] Read Container User can see the Container (but not necessarily its content)
  - [x] Read Container Log User is able to see its own and other user's actions in the event.
  - [ ] Run Core Functions User is able to run built-in core functions like bulk metadata edit, move content to other containers, etc.
  - [ ] Run User Functions User is able to run run-time defined user functions
  - [x] Update Container User can change the Container metadata, details, workflow, quota and other Container related settings.
  - [ ] Delete Container User can archive, permanently delete the container (and all its files or folders), associated metadata, events, versions and all related elements. This permission also allows the user to un-archive or un-delete archived containers.
  - [x] Read Container Content User can see, but not change, files and metadata inside the container. Versions for files, if they exist, are also visible (but can't recover older versions of a file)
    - [ ] Write Container Content User can create, overwrite and delete files, but not to modify file/folder metadata. Versions for files, if they exist, can be recovered (but not deleted).
      - [ ] Manage Container Storage User is able to define the initial storage class for the container, but is also able to modify it or to transition objects from one storage type to another. As transitioning from one storage status to another usually clears versions and deleted items, this user is also capable of permanently erasing deleted items and object versions.
      - [ ] Purge deleted items User is able to permanently delete already deleted items and older versions of existing files.
      - [ ] Write Container Content Metadata User can also create, delete or change file/folders metadata.
  - [x] Permissions read User can see the permissions of other users to access the container.
    - [ ] Permissions delete User is able to add new users or groups to the permissions for this container.
    - [ ] Permissions write User can see and modify the permissions associated to the container (even for itself). IMPORTANT!: If you grant this permission to a user, the user is able to change its own permissions for everything else (so may be virtually have any permission!)

+



- [x] Read Container User can see the Container (but not necessarily its content)
  - [ ] Read Container Log User is able to see its own and other user's actions in the event.
  - [x] Run Core Functions User is able to run built-in core functions like bulk metadata edit, move content to other containers, etc.
  - [x] Run User Functions User is able to run run-time defined user functions
  - [ ] Update Container User can change the Container metadata, details, workflow, quota and other Container related settings.
  - [ ] Delete Container User can archive, permanently delete the container (and all its files or folders), associated metadata, events, versions and all related elements. This permission also allows the user to un-archive or un-delete archived containers.
  - [x] Read Container Content User can see, but not change, files and metadata inside the container. Versions for files, if they exist, are also visible (but can't recover older versions of a file)
    - [x] Write Container Content User can create, overwrite and delete files, but not to modify file/folder metadata. Versions for files, if they exist, can be recovered (but not deleted).
      - [ ] Manage Container Storage User is able to define the initial storage class for the container, but is also able to modify it or to transition objects from one storage type to another. As transitioning from one storage status to another usually clears versions and deleted items, this user is also capable of permanently erasing deleted items and object versions.
      - [ ] Purge deleted items User is able to permanently delete already deleted items and older versions of existing files.
      - [ ] Write Container Content Metadata User can also create, delete or change file/folders metadata.
  - [ ] Permissions read User can see the permissions of other users to access the container.
    - [ ] Permissions delete User is able to add new users or groups to the permissions for this container.
    - [ ] Permissions write User can see and modify the permissions associated to the container (even for itself). IMPORTANT!: If you grant this permission to a user, the user is able to change its own permissions for everything else (so may be virtually have any permission!)

$\longrightarrow$



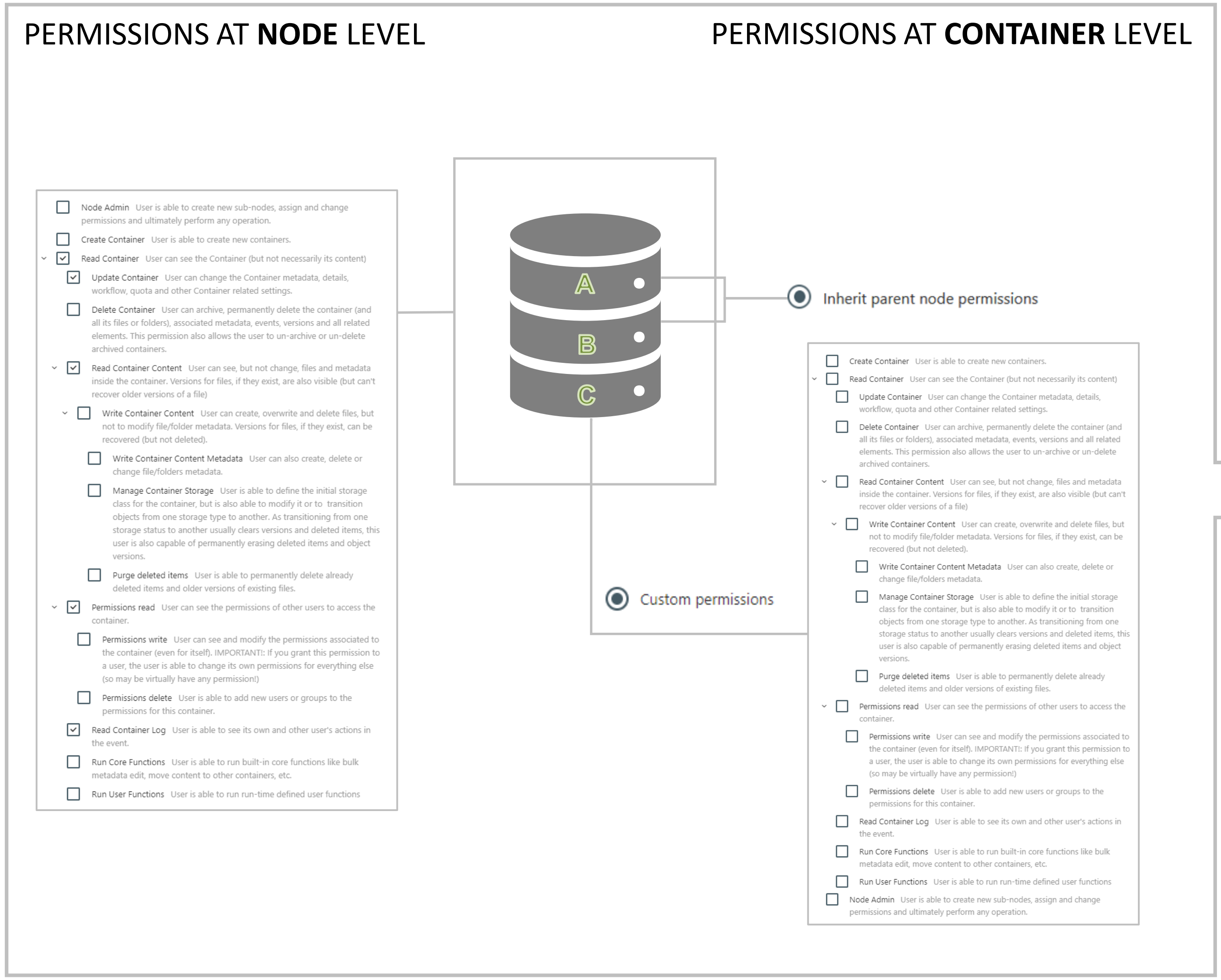
- [x] Read Container User can see the Container (but not necessarily its content)
  - [x] Read Container Log User is able to see its own and other user's actions in the event.
  - [x] Run Core Functions User is able to run built-in core functions like bulk metadata edit, move content to other containers, etc.
  - [x] Run User Functions User is able to run run-time defined user functions
  - [x] Update Container User can change the Container metadata, details, workflow, quota and other Container related settings.
  - [ ] Delete Container User can archive, permanently delete the container (and all its files or folders), associated metadata, events, versions and all related elements. This permission also allows the user to un-archive or un-delete archived containers.
  - [x] Read Container Content User can see, but not change, files and metadata inside the container. Versions for files, if they exist, are also visible (but can't recover older versions of a file)
    - [x] Write Container Content User can create, overwrite and delete files, but not to modify file/folder metadata. Versions for files, if they exist, can be recovered (but not deleted).
      - [ ] Manage Container Storage User is able to define the initial storage class for the container, but is also able to modify it or to transition objects from one storage type to another. As transitioning from one storage status to another usually clears versions and deleted items, this user is also capable of permanently erasing deleted items and object versions.
      - [ ] Purge deleted items User is able to permanently delete already deleted items and older versions of existing files.
      - [ ] Write Container Content Metadata User can also create, delete or change file/folders metadata.
  - [x] Permissions read User can see the permissions of other users to access the container.
    - [ ] Permissions delete User is able to add new users or groups to the permissions for this container.
    - [ ] Permissions write User can see and modify the permissions associated to the container (even for itself). IMPORTANT!: If you grant this permission to a user, the user is able to change its own permissions for everything else (so may be virtually have any permission!)

User and groups permissions add to each other when combined. Therefore, if user A is given individual permissions and then assigned to a group with its own permissions, user A retains their individual permissions and adds the new ones given by the group, regardless of their old ones

Permissions’ Hierarchy

PERMISSIONS AT **NODE** LEVEL

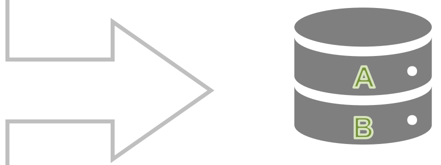
PERMISSIONS AT **CONTAINER** LEVEL



- ☐ Node Admin User is able to create new sub-nodes, assign and change permissions and ultimately perform any operation.
- ☐ Create Container User is able to create new containers.
- ☑ Read Container User can see the Container (but not necessarily its content)
  - ☑ Update Container User can change the Container metadata, details, workflow, quota and other Container related settings.
  - ☐ Delete Container User can archive, permanently delete the container (and all its files or folders), associated metadata, events, versions and all related elements. This permission also allows the user to un-archive or un-delete archived containers.
  - ☑ Read Container Content User can see, but not change, files and metadata inside the container. Versions for files, if they exist, are also visible (but can't recover older versions of a file)
    - ☐ Write Container Content User can create, overwrite and delete files, but not to modify file/folder metadata. Versions for files, if they exist, can be recovered (but not deleted).
      - ☐ Write Container Content Metadata User can also create, delete or change file/folders metadata.
      - ☐ Manage Container Storage User is able to define the initial storage class for the container, but is also able to modify it or to transition objects from one storage type to another. As transitioning from one storage status to another usually clears versions and deleted items, this user is also capable of permanently erasing deleted items and object versions.
      - ☐ Purge deleted items User is able to permanently delete already deleted items and older versions of existing files.
  - ☑ Permissions read User can see the permissions of other users to access the container.
    - ☐ Permissions write User can see and modify the permissions associated to the container (even for itself). IMPORTANT!: If you grant this permission to a user, the user is able to change its own permissions for everything else (so may be virtually have any permission!)
    - ☐ Permissions delete User is able to add new users or groups to the permissions for this container.
  - ☑ Read Container Log User is able to see its own and other user's actions in the event.
  - ☐ Run Core Functions User is able to run built-in core functions like bulk metadata edit, move content to other containers, etc.
  - ☐ Run User Functions User is able to run run-time defined user functions

- ☐ Create Container User is able to create new containers.
- ☐ Read Container User can see the Container (but not necessarily its content)
  - ☐ Update Container User can change the Container metadata, details, workflow, quota and other Container related settings.
  - ☐ Delete Container User can archive, permanently delete the container (and all its files or folders), associated metadata, events, versions and all related elements. This permission also allows the user to un-archive or un-delete archived containers.
  - ☐ Read Container Content User can see, but not change, files and metadata inside the container. Versions for files, if they exist, are also visible (but can't recover older versions of a file)
    - ☐ Write Container Content User can create, overwrite and delete files, but not to modify file/folder metadata. Versions for files, if they exist, can be recovered (but not deleted).
      - ☐ Write Container Content Metadata User can also create, delete or change file/folders metadata.
      - ☐ Manage Container Storage User is able to define the initial storage class for the container, but is also able to modify it or to transition objects from one storage type to another. As transitioning from one storage status to another usually clears versions and deleted items, this user is also capable of permanently erasing deleted items and object versions.
      - ☐ Purge deleted items User is able to permanently delete already deleted items and older versions of existing files.
  - ☐ Permissions read User can see the permissions of other users to access the container.
    - ☐ Permissions write User can see and modify the permissions associated to the container (even for itself). IMPORTANT!: If you grant this permission to a user, the user is able to change its own permissions for everything else (so may be virtually have any permission!)
    - ☐ Permissions delete User is able to add new users or groups to the permissions for this container.
  - ☐ Read Container Log User is able to see its own and other user's actions in the event.
  - ☐ Run Core Functions User is able to run built-in core functions like bulk metadata edit, move content to other containers, etc.
  - ☐ Run User Functions User is able to run run-time defined user functions
- ☐ Node Admin User is able to create new sub-nodes, assign and change permissions and ultimately perform any operation.

FINAL USER/GROUP’S PERMISSIONS



The user or group has the right to see and manipulate containers A and B as allowed by the defined parent node permissions. However, custom permissions have been selected for container C and left blank, which leaves the user unable to see nor manipulate the container.

Configuration

- Archival organization
  - Nodes
  - Node templates

- Containers
  - Concepts
    - Overview
    - Metadata
    - Storage

- Introduction to metadata and metadata configuration
  - How it works
  - Schema
  - Importing/exporting

Operation

Operation

- Create a container
  - GUI
  - API
- Uploading content
  - Browser
  - S3
- Downloading content
  - Browser
  - S3
- Searching
  - GUI
  - API
- Functions and Workflows
  - Overview
  - Sample
  - API
- Storage

Operation - API

(Set your environment variables: API key + S3)
(Offsets and limits)

List Archival Structure
Create an Archival Node
List Archival Node permissions
Edit Archival Node permissions
- List groups
- List permissions
- Assign a permission to a group for an Archival Node

Create a container
- List container metadata schemas
- List object metadata schemas
- List workflows
- Create container

Operation - API

Uploading content

Downloading content

- The S3 way
- The API way
  - Listing objects (file/folder)
  - Downloading

Operation

- Storage
- Reports
- Jupyter notebooks / snakemake / Custom containers

libnova
the most advanced digital preservation platform

Thanks!

contact@libnova.com