

# Introducing the Acacia Object Storage Service

4 April 2022. Version 1.00



# Focus for this module

At the end of this module, you will be able to:

- Define the terms:
  - Object storage
  - Buckets
  - Objects
- Compare and contrast object storage and filesystems
- Discuss the benefits of using object storage
- Identify Pawsey's new object storage system, called Acacia



The backplane of Acacia, Pawsey's new object storage system for research, exposes the cabling and high speed networking essential to building this new, state of the art infrastructure.

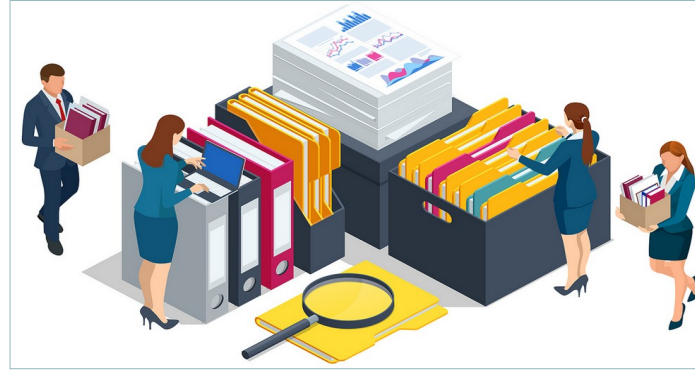
Section 1

# Introduction to Filesystems

# What is a filesystem?

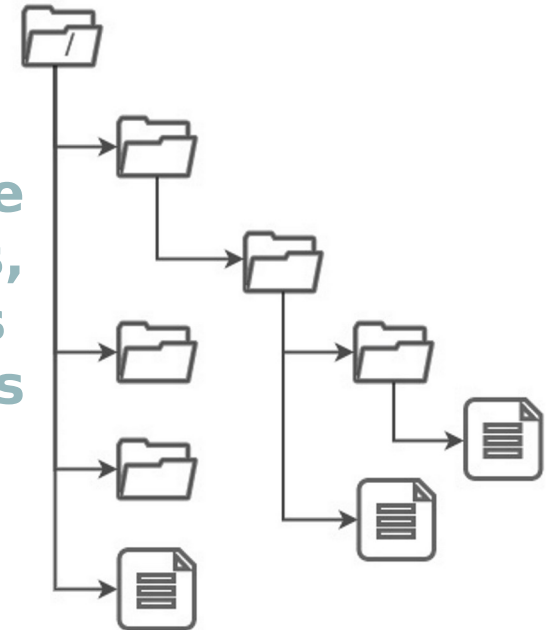
- Simply stated, a **filesystem** is a system or way of grouping and storing files that is hierarchical.
- A filesystem groups data together in **files**. You can interactively perform actions on files (e.g., edit) them right where they are in the filesystem.
- You can organise files into **directories** (or folders). You can use this as a visual aid in grouping together files.

Pawsey supercomputers like **Magnus**, **Zeus**, **Topaz**, and **Setonix** use shared filesystems to store data.



Filing cabinets, folders and files - the 'old' way of organising

Translated online into.... filesystems, directories and files



# More about Filesystems

- A **file** has associated with it metadata, such as file name, timestamp, location of the file in the folder hierarchy, file attributes (read, write), etc.

You use metadata for such actions as searching for a file.

- A **directory** is a hierarchical approach to file management. The hierarchy that you establish is arbitrary, that is, it can be as deep or shallow as you want. E.g., It can have only 1 level or many levels.

## Filesystem Examples

Directories

Files

Directories

Files

3 levels of Hierarchy

Files

Directories



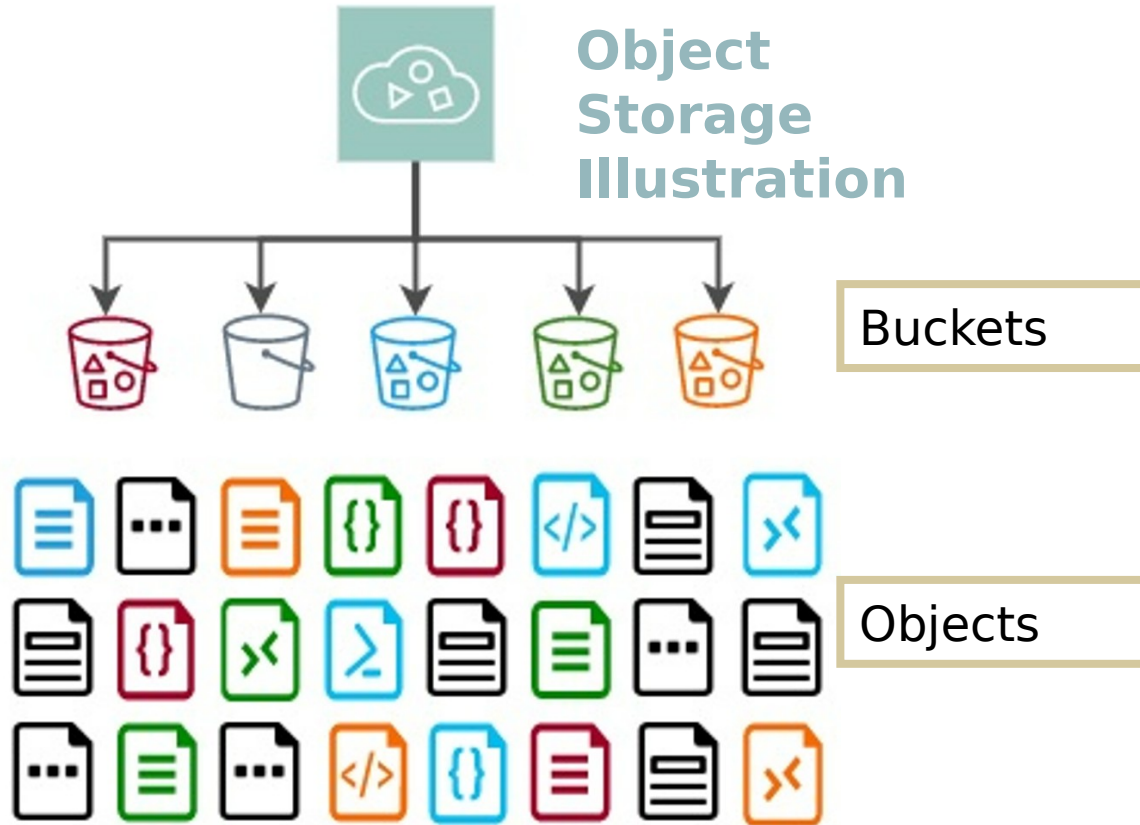
**pawsey**

Section 2

# **Introduction to Object Storage**



# What is object storage?



Pawsey will use object storage (called Acacia) for use with its post-Capital Refresh resources (e.g., Setonix).

- Simply stated, **object storage** is a way of grouping and storing files that is flat (not hierarchical).
- The same functionality of a hierarchy can be achieved by prepending subfolder names to an object name.
- Object storage is a collection of unique **objects**. You cannot interactively perform actions on objects (e.g., edit them) in object storage. You must remove them to do so.
- You can organise objects into **buckets** using object naming. You can use this as an aid in visualising object groupings within a bucket.
- The buckets and objects here are colour coded. E.g., green objects go in green buckets. An object must be in a single bucket.

# More about Object Storage



All objects and buckets are “equivalent” data items with unique keys. There is no hierarchy.



- **Object storage** does not allow you to interactively change objects. To change an object, you must first get it from storage. When you finish, you put it back.



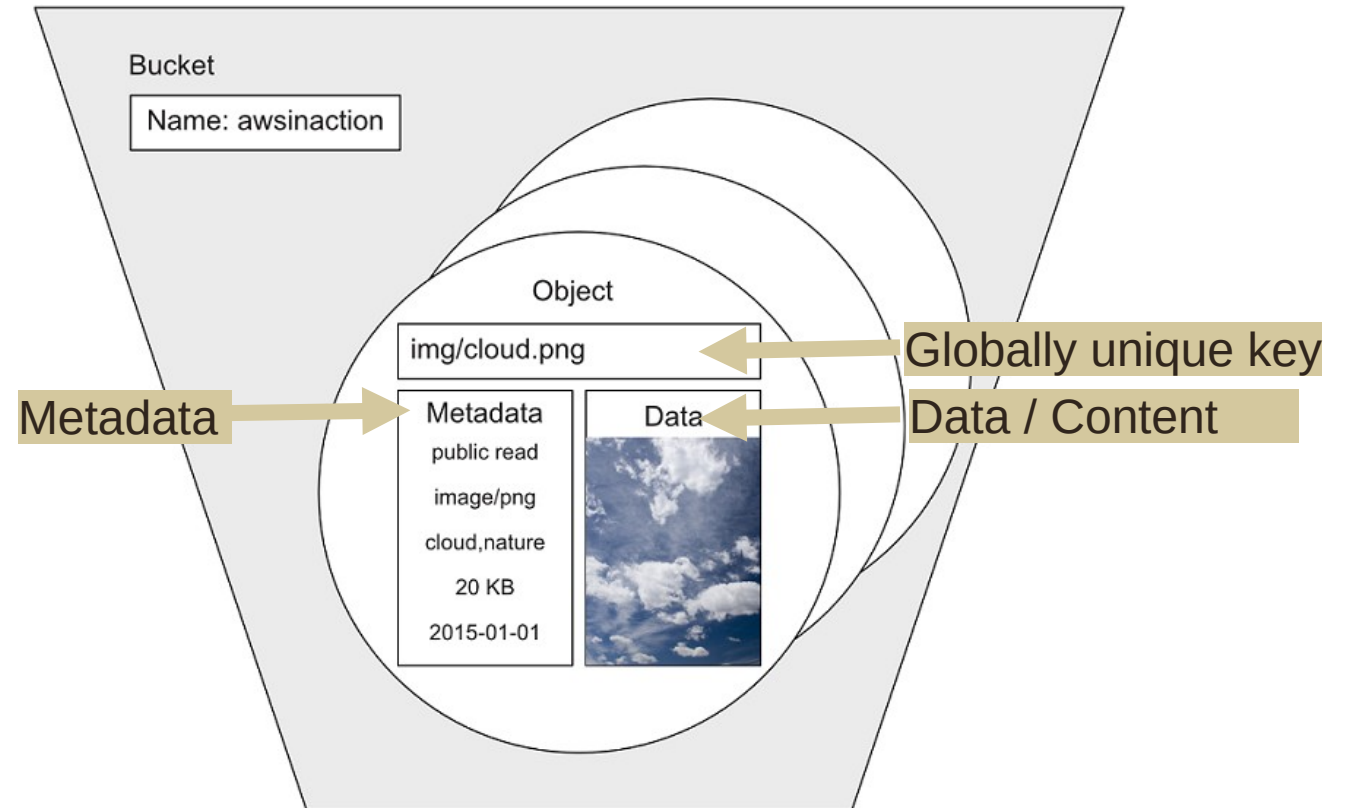
- Since every **object** must have a **globally unique identifier** (key), you will know exactly the object you are getting.
- If you do not know the key, you can search using robust metadata.
- To help you organise your objects, you can use **buckets**. Buckets are for organisational reference only, all objects (including buckets) are simply stored equally as data objects.



# More about Objects

- An **object** has associated with it, three things:
  - Content, or data. This is often a file or an image, but it may also be a part of a file or simply a sequence of bytes.
  - Metadata. This contains information about the data to help with searching.
  - A globally unique identifier (key).
- An object's metadata is stored separate from an object.

## Metadata Example



[Image: Storing your objects](#)



**pawsey**

Section 3

# **Analogy: Filesystems and Object Storage**

# Filing Cabinet vs Bank Vault

## Filing Cabinet

- You can locate, copy, edit, and remove files from the filing cabinet.
- The filing cabinet is easily accessible to you and your colleagues, but difficult to access for people outside the office.
- A filing cabinet can only hold a limited number of files before you need a new filing cabinet.
- It is hard to keep and search notes about the files themselves.



## Bank Vault



- Once you deposit an object in the vault, you cannot do anything to it until you take it out of the vault.
- Anyone, anywhere can access the bank vault if they have the right authorization and key.
- The vault is shared with all the other Bank customers, so objects need unique names.
- Vaults can hold a lot of objects and can be readily expanded by making the vault bigger.
- Notes about objects are easily created when you deposit an object. You can search and sort notes.



pawsey

Section 4

# **Filesystems & Object Storage: A Review**

# Filesystems and Object Storage in Review

Filesystems	Object Storage
<ul style="list-style-type: none"><li>• Stores data in files</li></ul>	<ul style="list-style-type: none"><li>• Stores data and custom metadata in objects</li></ul>
<ul style="list-style-type: none"><li>• Allows you to make changes to a file “in-place” (within filesystem)</li></ul>	<ul style="list-style-type: none"><li>• Does not allow you to make changes to objects (immutable)</li></ul>
<ul style="list-style-type: none"><li>• Has fixed attributes (metadata) for folders and files (e.g., filename, time stamps)</li></ul>	<ul style="list-style-type: none"><li>• Supports custom metadata for buckets and objects to enable easier searching and retrieval</li></ul>
<ul style="list-style-type: none"><li>• Manages data within a file hierarchy, where each file has a unique identifier for the lowest (sub)level of hierarchy</li></ul>	<ul style="list-style-type: none"><li>• Manages data in a flat repository / no hierarchy, where each object has a globally unique identifier (key)</li></ul>
<ul style="list-style-type: none"><li>• Are best suited for simplified access and management of shared files</li><li>• Are best used for maintaining a single, accurately updated version of a file</li></ul>	<ul style="list-style-type: none"><li>• Is best suited for retaining massive amounts of unstructured data</li></ul>
<ul style="list-style-type: none"><li>• Have upper limits for expansion, which put a cap on (unlimited) scalability</li></ul>	<ul style="list-style-type: none"><li>• Is ill-suited for frequently changing, transactional data</li></ul>

Section 5

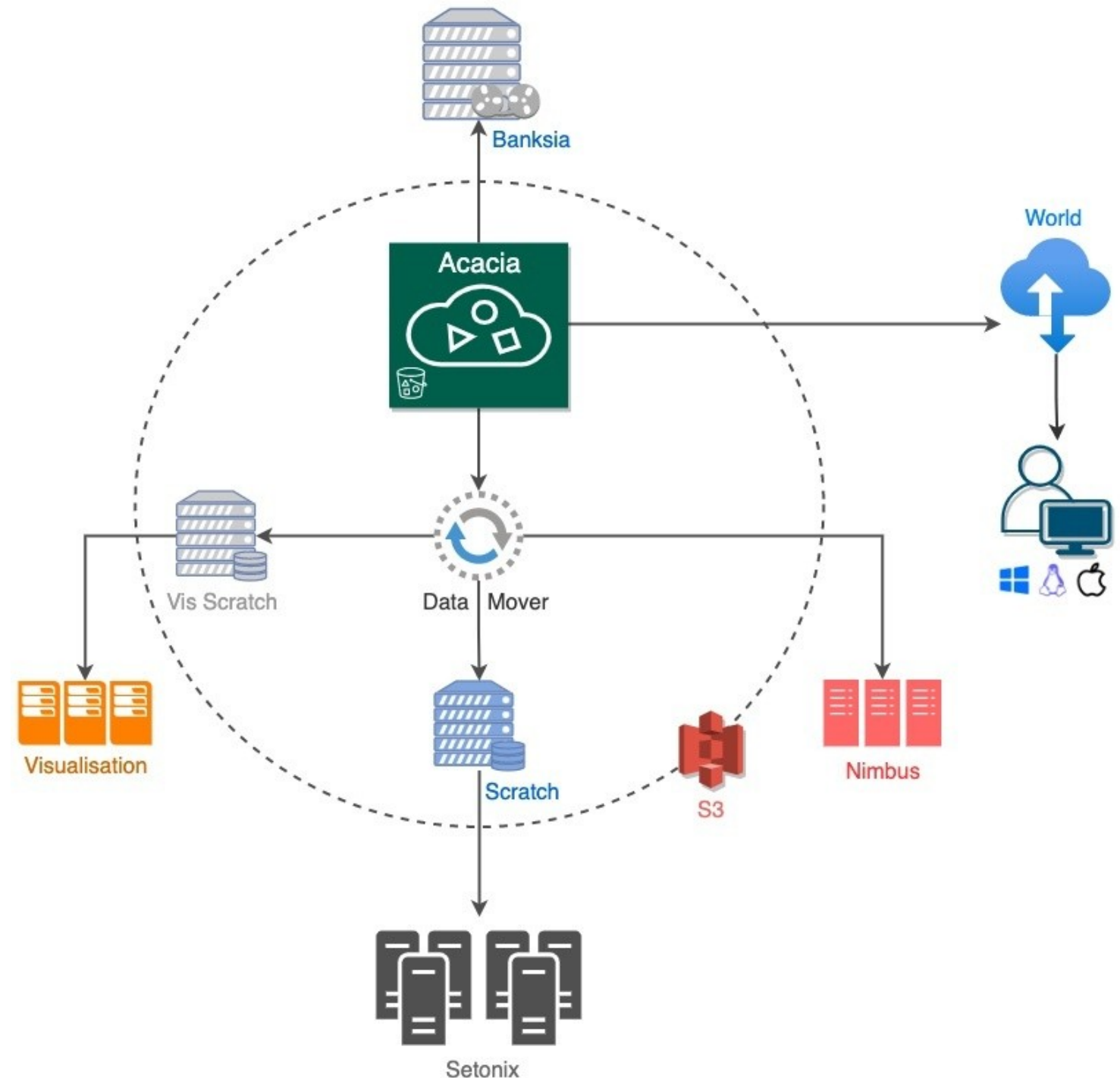
# Introducing Acacia



# Acacia @ Pawsey

As seen here, Acacia:

- Is the central object storage service for Pawsey projects
- Interconnects all Pawsey services
- Connects with Setonix (super compute), Nimbus (Cloud compute) and Banksia (cold storage)
- Is accessible from both inside Pawsey (Pawsey resources) and outside Pawsey (World)



# Introducing Acacia Object Storage

**multi-Tier  
Storage**

**acacia**  
**60PB object  
storage**

Ceph software

5000+ 16TB disks

400+ Nvmes

**BankSia**  
**70PB tape  
storage**


5 PB front-end cache

32 TS1150 high

performance tape library drives

2 TS1160 next generation drives

- Acacia allows for up to 1000 buckets per user.
- Acacia allows for up to 1 million objects per bucket.
- Users are given 100 GB of Acacia storage per user.
- Projects are given 1 TB of Acacia storage per project.
- Users interact with Acacia using S3 clients.



Now we are going to walk through a series of tutorials on how to use Acacia, both from the command line and as part of your HPC workflows.

# Tutorial outline

## 1) Get access to Acacia

- Setting up clients to access Acacia from your computer and Pawsey systems

## 2) Acacia from your computer

- Work with buckets and objects using mock data
- Create publicly accessible links to shared with your colleagues

## 3) Acacia on Pawsey systems

- Mini tutorial on using tar files effectively
- Integrate Acacia into your HPC workflows.