OpenIO

HDF Group
ESRF September 2019
Kita-OIO SDS Integration

Agenda

September 2019

1 **About OpenIO**

2 **OpenIO SDS + KITA**

3 **Demo**

OpenIO ID

Founded in 2015

Quickly growing across geographies and vertical markets

40+

Large customers

Deployments from 3 nodes up to 40
Petabytes and tens of billions of objects

3

Continents

HQ
Lille

Offices
Paris, Tokyo

Teams across EMEA & Japan

40

Employees

Mostly R&D, Support, tech people

Growing fast

3

Investors

elaia

partech

Région Hauts-de-France

Recognition & awards:

PASS FRENCH TECH

bpifrance

EXCELLENCE

Alliancy
LE MAG NUMÉRIQUE ET BUSINESS

OpenIO selected as Cloud start-up to follow closely,
March 2019

OpenIO Vision and Mission

Vision:

We envision a data-centric world where OpenIO is recognized as the universal storage solution for unstructured data

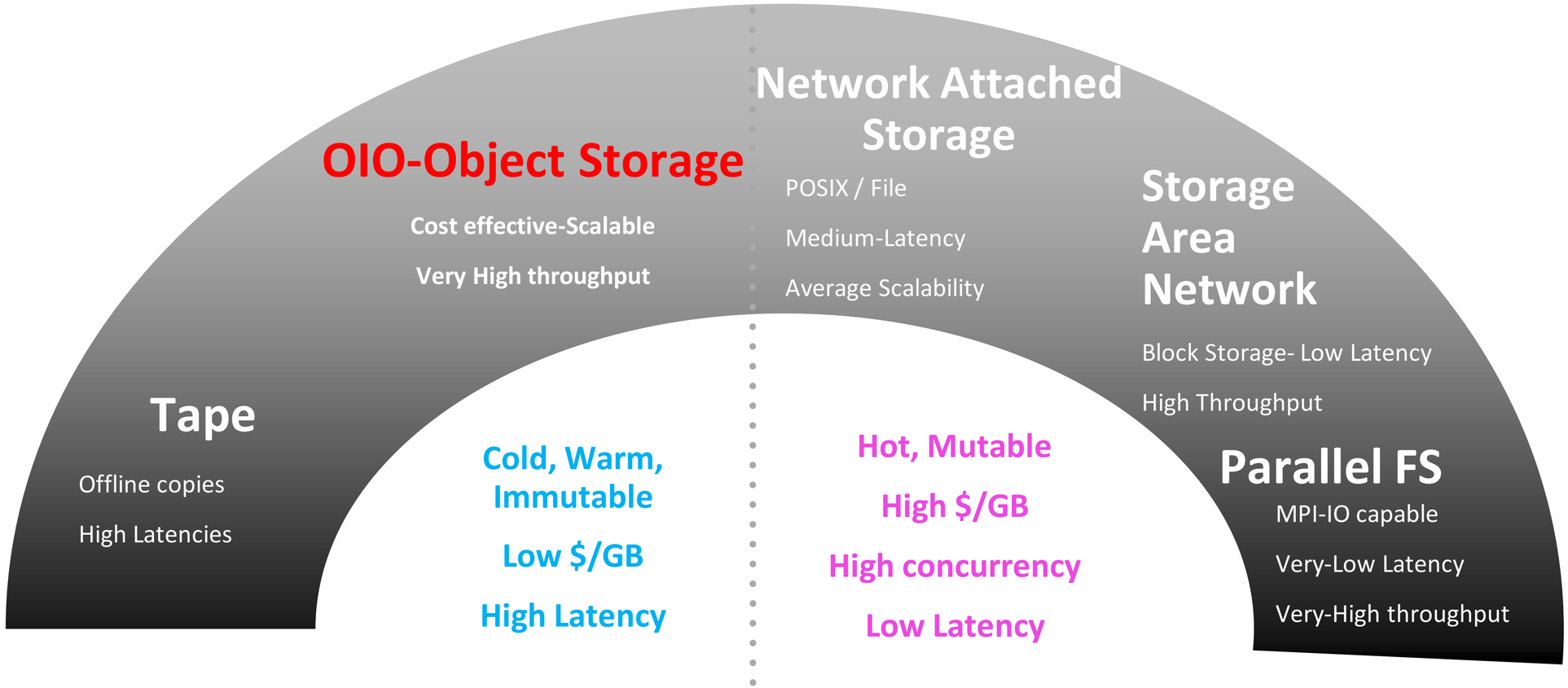
Mission:

OpenIO's mission is to deliver an open source, high performance object storage solution that meets the demanding needs of customers working with **HPC, Big Data** and **AI**

OpenIO SDS and HDF Kita

Jean-François Smigielski
OpenIO- CTO

Storage Landscape in HPC



Why object storage to fill the gap? TCO!

1/ What is Object Storage

- Unstructured Immutable Data + Metadata
- High Parallelism
- 100% Online Dataset
- Cloud-oriented, ideal for large scale
- *De facto* standards: S3 (AWS), Swift (Openstack)

2/ Meanwhile, in HPC...

- S3/Swift are not standards, HDF5 & MPI-IO are
- HDF5 was not designed to work with objects
- Huge mutable datasets
- Lower TCO would be appreciated

3/ How can it integrate?

- Hierarchically, behind a primary fast tier
- Independant tiers with data movements orchestrated

4/ KITA, the necessary middleware

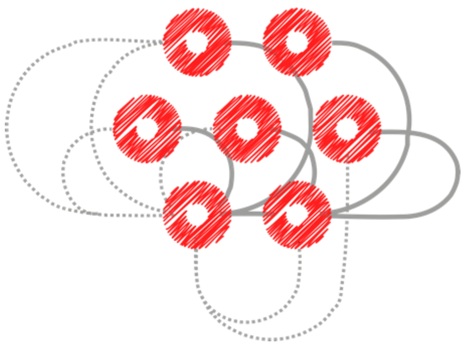
- Persist mutable datasets as immutable objects!
- Independant Object Storage, with HDF5 as an orchestrator (import / export)

Why OpenIO? We Think Different!

ConsciousGrid™ technology

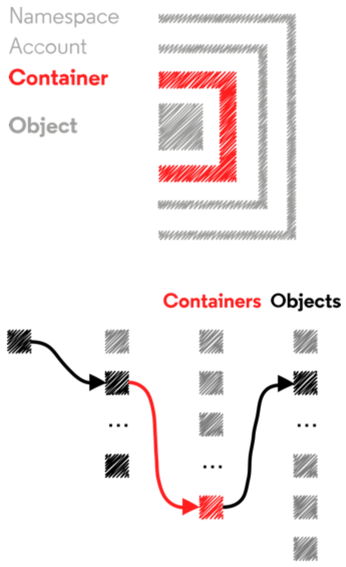
Never rebalance:

Scale up and out in small or large increments and on any hardware that you choose, while maintaining consistent high performance



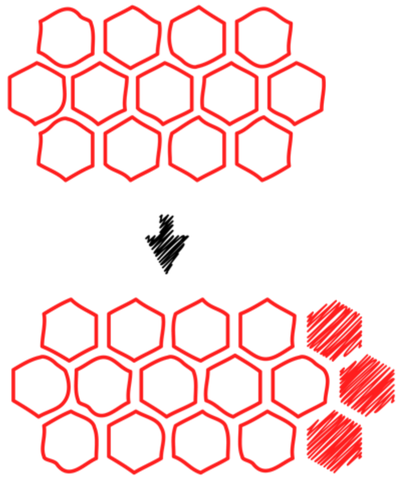
Directory with indirections

Track containers and not objects:



Grid of nodes

Real-time load balancing for optimal data placement, more efficient than a ring-based architecture



Open Source & HW agnostic

Avoid vendor lock-in and keep control of your data

Open source guarantees the continuity of the solution and gives your engineers the opportunity to understand how the tech works

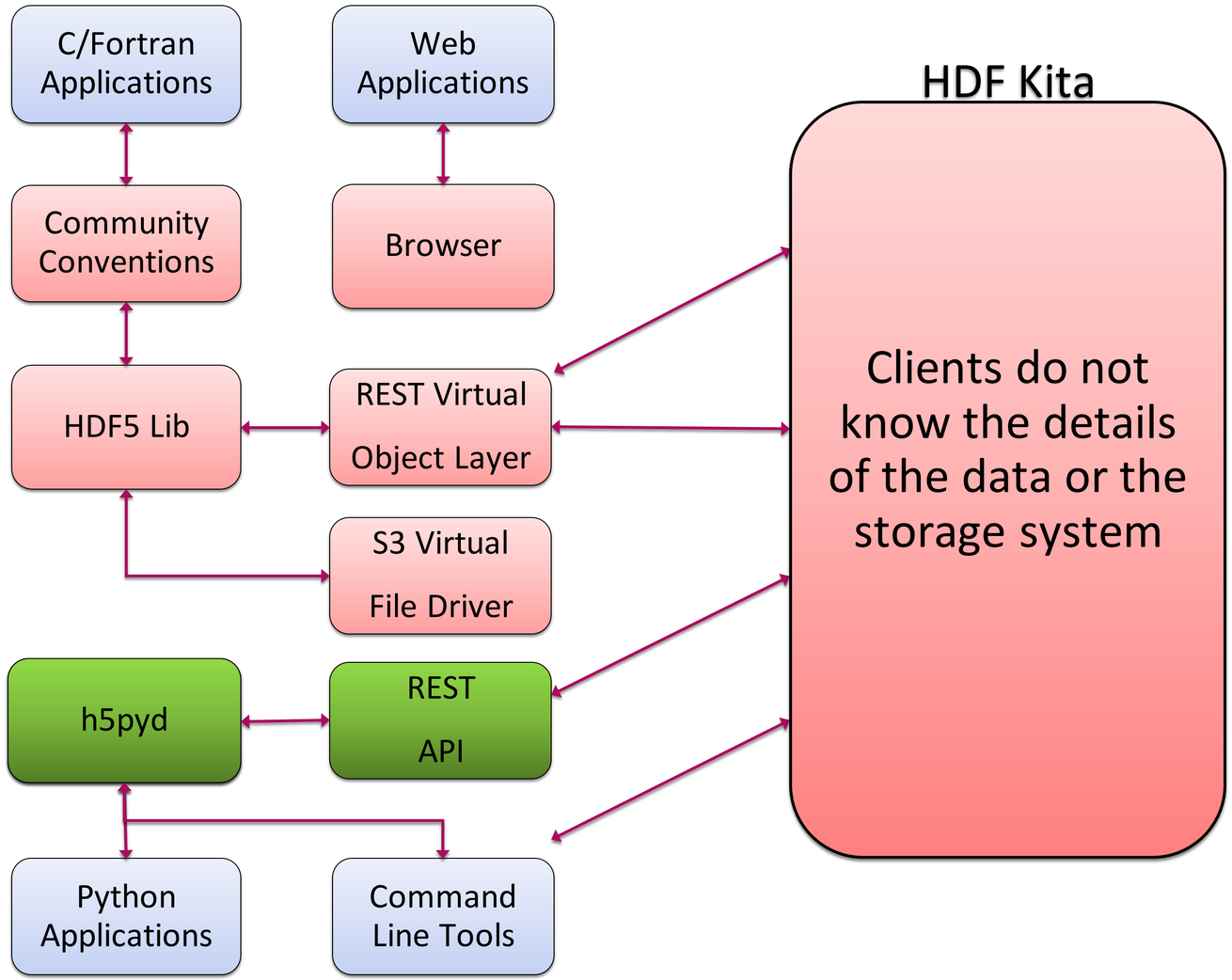
Being hardware agnostic allows better capacity planning and improve your TCO

KITA/OIO Integration Architecture

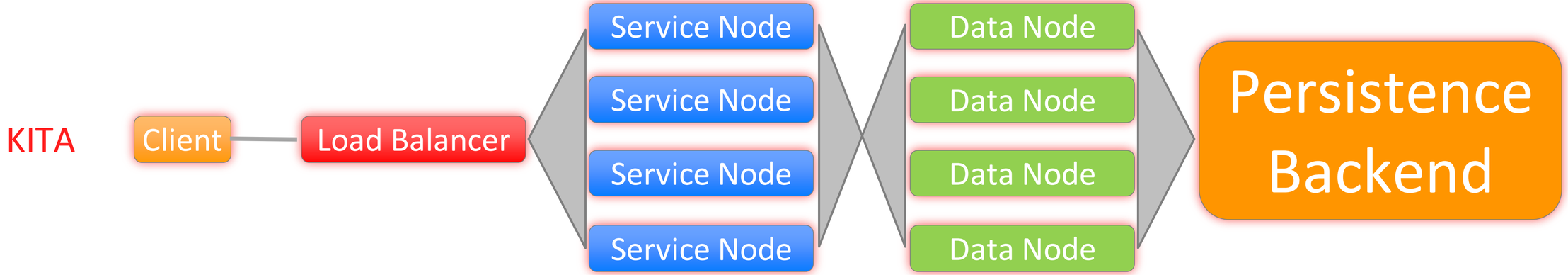
Kita's Architecture

Client SDKs for Python and C are drop-in replacements for libraries used with local files.

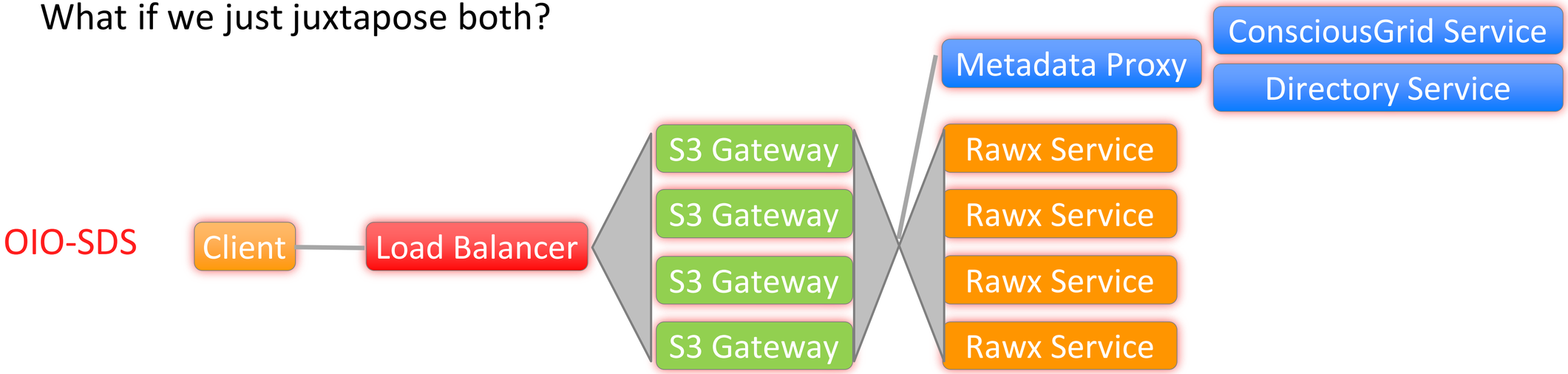
No significant code change to access local and cloud based data.



Kita / OIO Similar Architectures



What if we just juxtapose both?



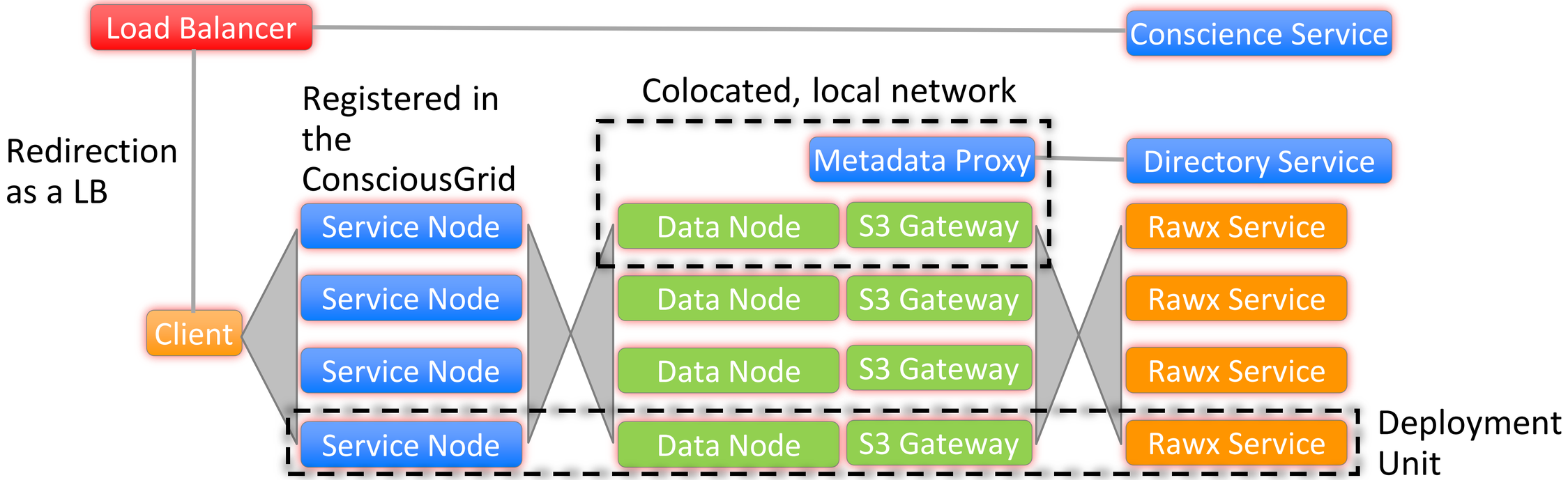
Step 1: Easy Integration

Let's configure a single load-balanced endpoint for the persistence backend

- Many redundant scalability patterns (caching, sharding, load-balancing)
- Huge bandwidth usage: any stream is repeated
- 2 autonomous clusters with deployment patterns
  - Stateless, with K8s or Docker Swarm for Kita
  - Stateful, with Ansible on bare-metal for OpenIO

Barely acceptable for functional validation purposes, as a first step.

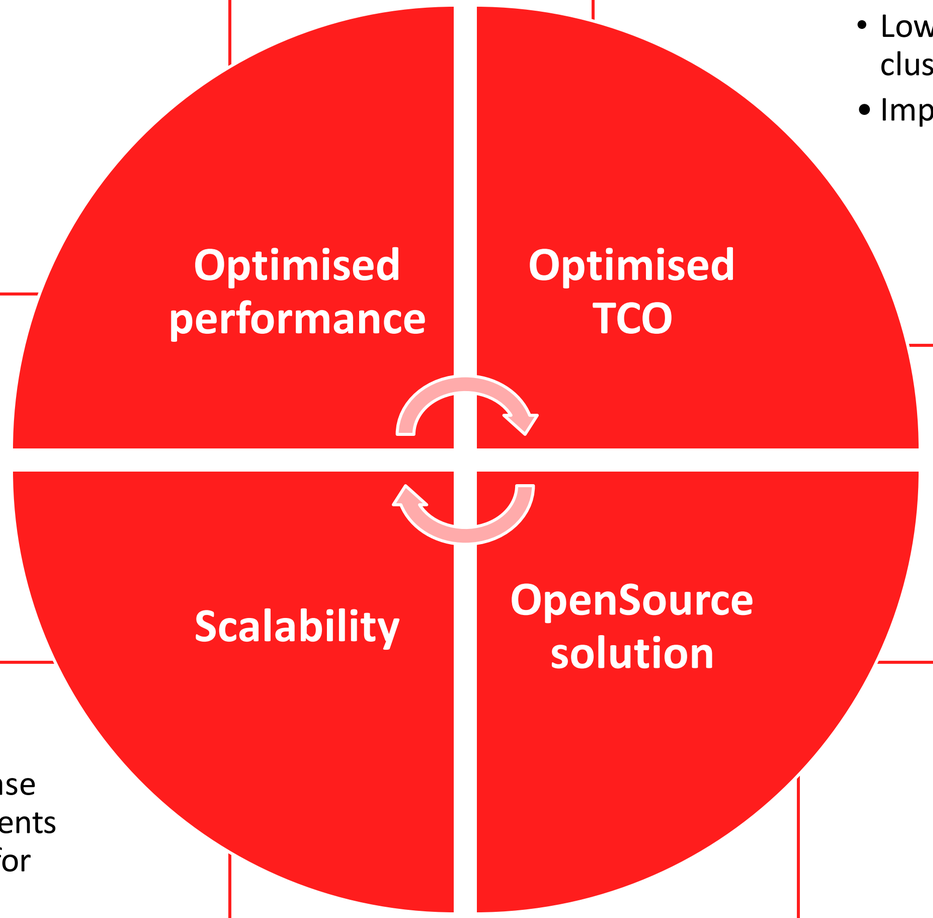
Step 2: Tighter Integration



Kita / OIO: Benefits of a Tight Integration

- Tightly coupled architecture
- Efforts on pragmatism and parcimony

- Lower HW cost & cost/giga
- Lower management cost one cluster to manage rather than two!
- Improved management cycle



- Scaling up increase simultaneous clients and parallelism for HDF access

- Both HDF and OIO are born opensource
- No vendor lock-in
- De facto standards

Kita / OIO: A (not so) Imaginary Use Case

- A scientist visits a research facility and he/she starts a new experiment on an existing run
- The test happens, data are dumped on a fast buffer storage (Parallel FS)
- Soon after, copies are made on the secondary storage (Tape)
- Preliminary validations are performed on the data in the fast buffer
- His/Her long stay comes to an end, he/she returns with a pile of BluRay/DVD
- The data is flushed from the buffer.

- A scientist visits a research facility and he/she starts a new experiment on an existing run
- The test happens, data are dumped on a fast buffer storage (Parallel FS)
- Soon after, copies are made on the secondary storage (Private Cloud) and data is flushed from the buffer
- Preliminary validations are performed from the cloud
- His/Her short stay comes to an end, he/she returns with credentials to the cloud.

Much smaller and cheaper Buffer needed!

Better user experience!

Demo

Want to learn more?

Contacts

- jean-francois.smigielski@openio.io
- marielaure.retureau@openio.io
- dax.rodriguez@hdfgroup.org

Try out Kita for free in our JupyterLab environment:

- See: http://www.hdfgroup.org/hdfkitalab/

Learn more about HDF Kita: https://www.hdfgroup.org/solutions/hdf-kita/

Learn more about OpenIO SDS: https://www.openio.io/product/product-overview