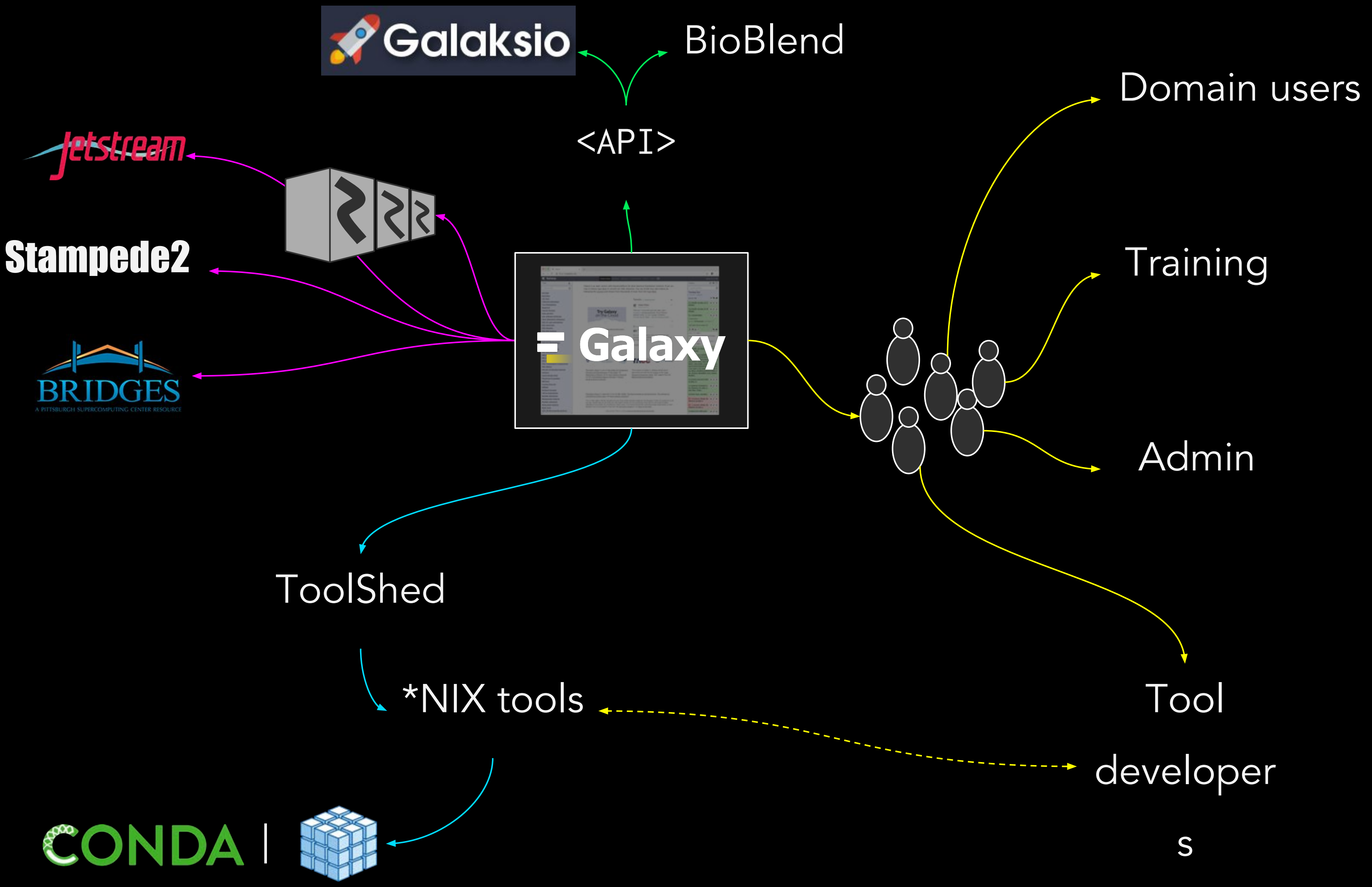
Cloud Resource Federation for Galaxy

Enis Afgan
Galaxy Team
Johns Hopkins University

Jan 23, 2019

Galaxy platform as a science gateway



Galaksio

BioBlend

<API>

Domain users

Jetstream

Stampede2

Training

Galaxy

BRIDGES

A PITTSBURGH SUPERCOMPUTING CENTER RESOURCE

Admin

ToolShed

*NIX tools

Tool
developer
s

CONDA |

130,000

registered users

2PB

user data

20M

jobs run

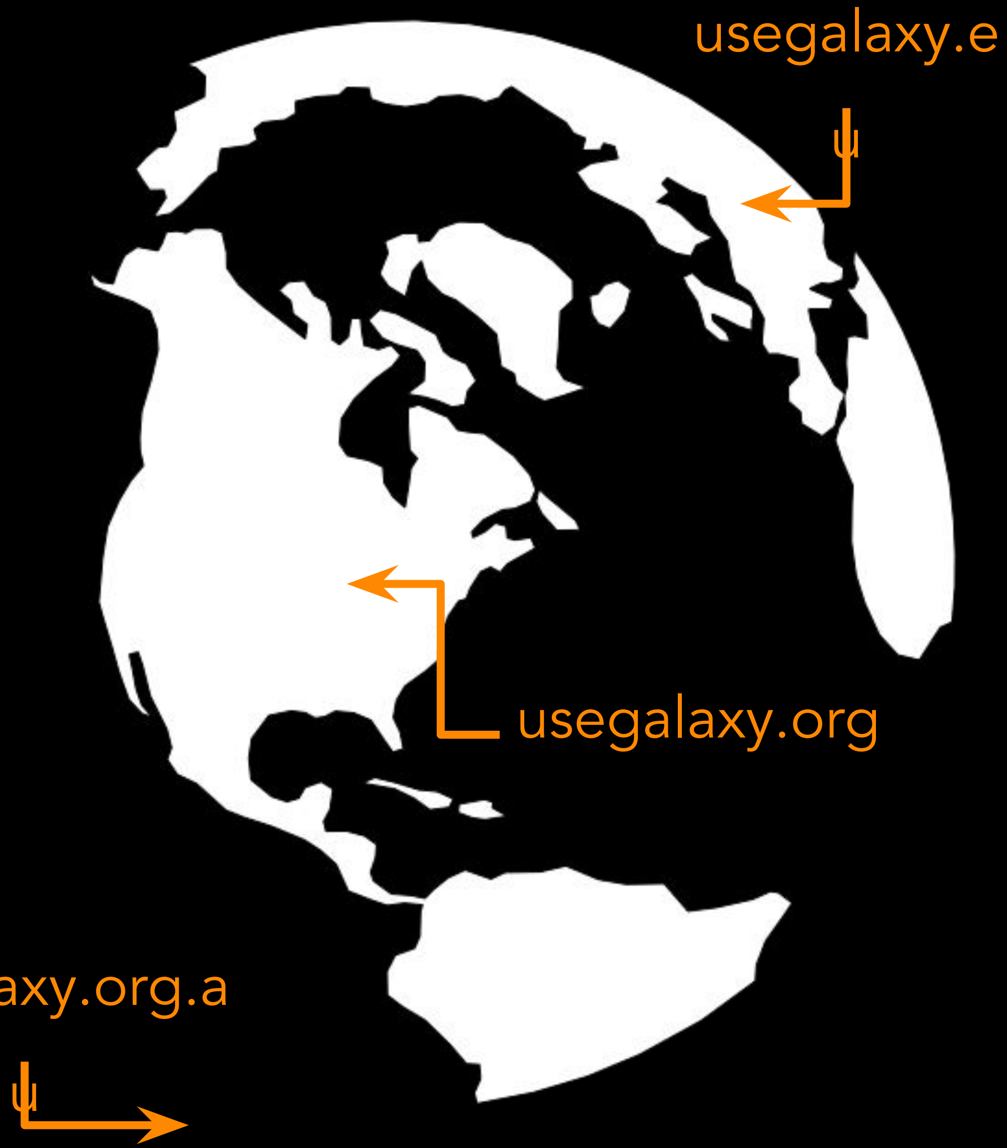
100

training events
(2018 & 2019)

Stats for Galaxy Main (*usegalaxy.org*) in Dec 2018

usegalaxy.* federation - a group of public Galaxy servers

- **Present a similar experience to users no matter which they use**
- Guarantee a minimum service
  - Tools & versions
  - Reference Data
  - Reproducibility
  - Training materials
- Starting with USA, Europe and Australia, more welcome!
- Manage with community assets/repositories
- Don’t prescribe hardware resources



125+ platforms for using Galaxy

Public servers

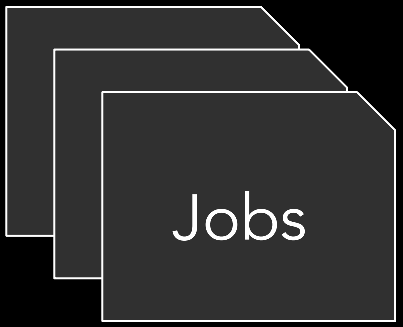
Academic and commercial clouds

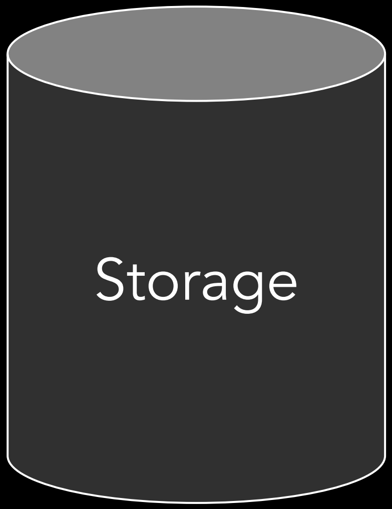
Container images

Virtual Machines

Galaxy is well-adopted by a broad community

Scaling challenges: quotas





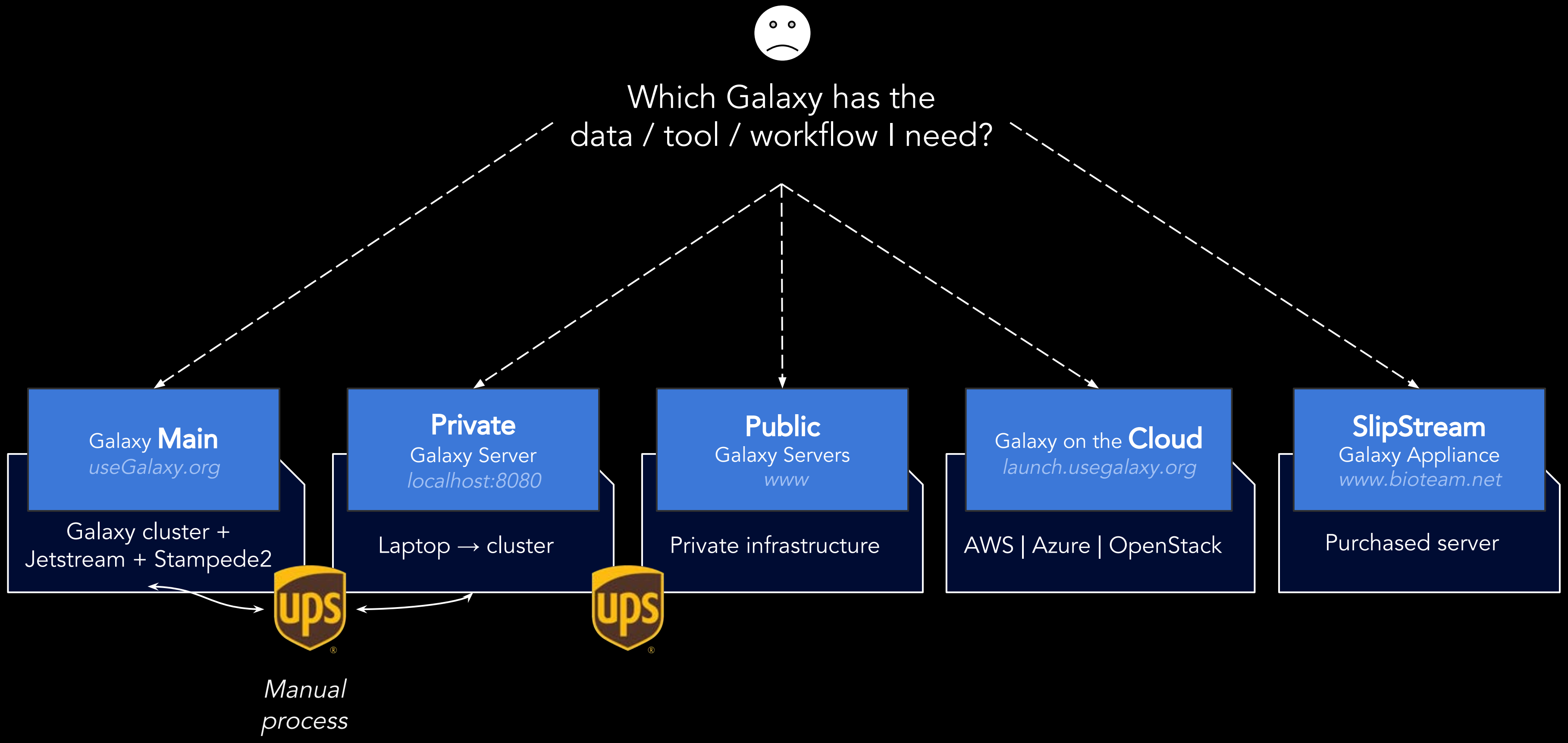
3-4 small jobs

&

2 parallel jobs

250GB

Scaling challenges: silos and fragmentation



☹

Which Galaxy has the data / tool / workflow I need?

Galaxy **Main**
useGalaxy.org
Galaxy cluster + Jetstream + Stampede2

Private
Galaxy Server
localhost:8080
Laptop → cluster

Public
Galaxy Servers
www
Private infrastructure

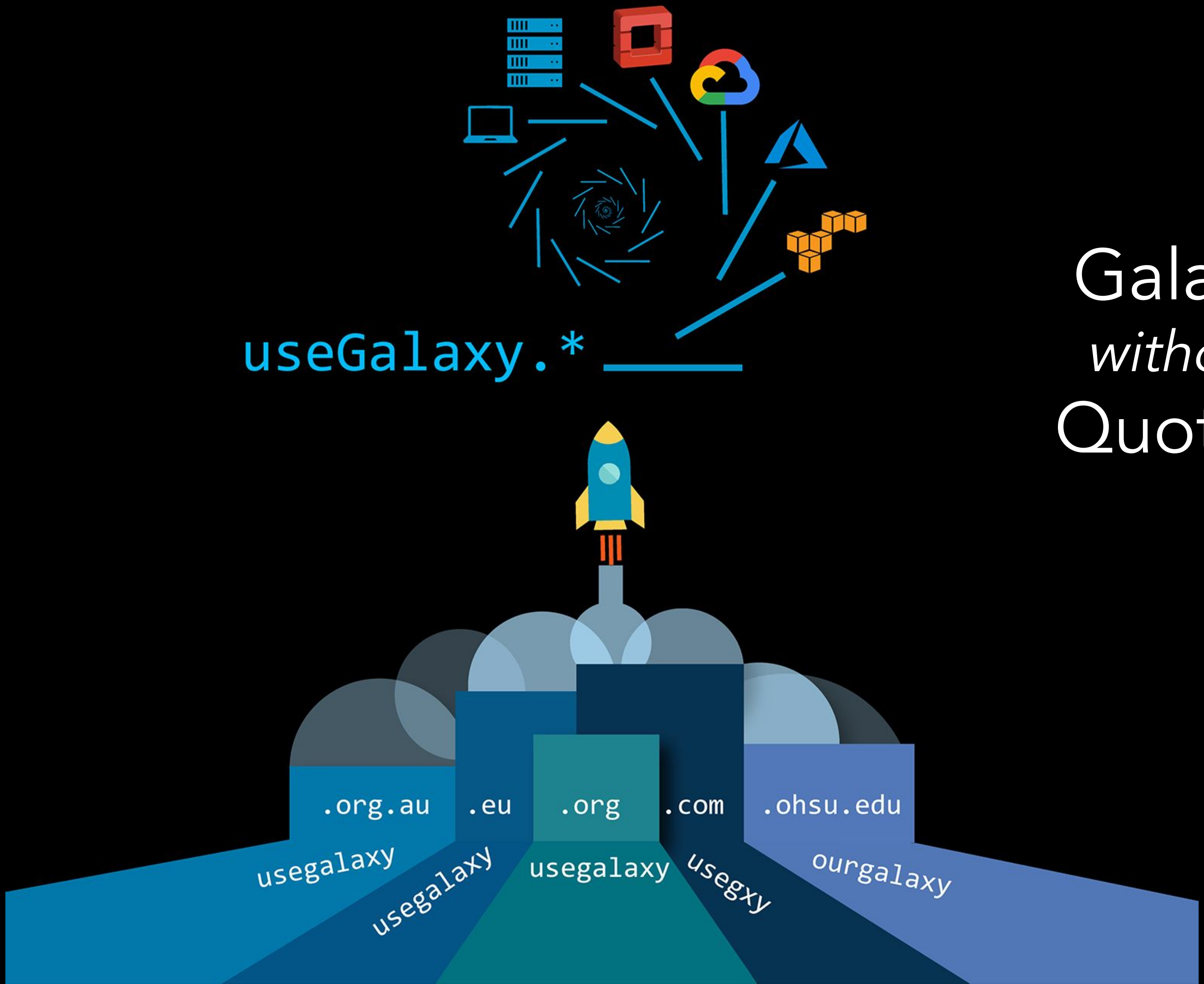
Galaxy on the **Cloud**
launch.usegalaxy.org
AWS | Azure | OpenStack

SlipStream
Galaxy Appliance
www.bioteam.net
Purchased server

Manual process

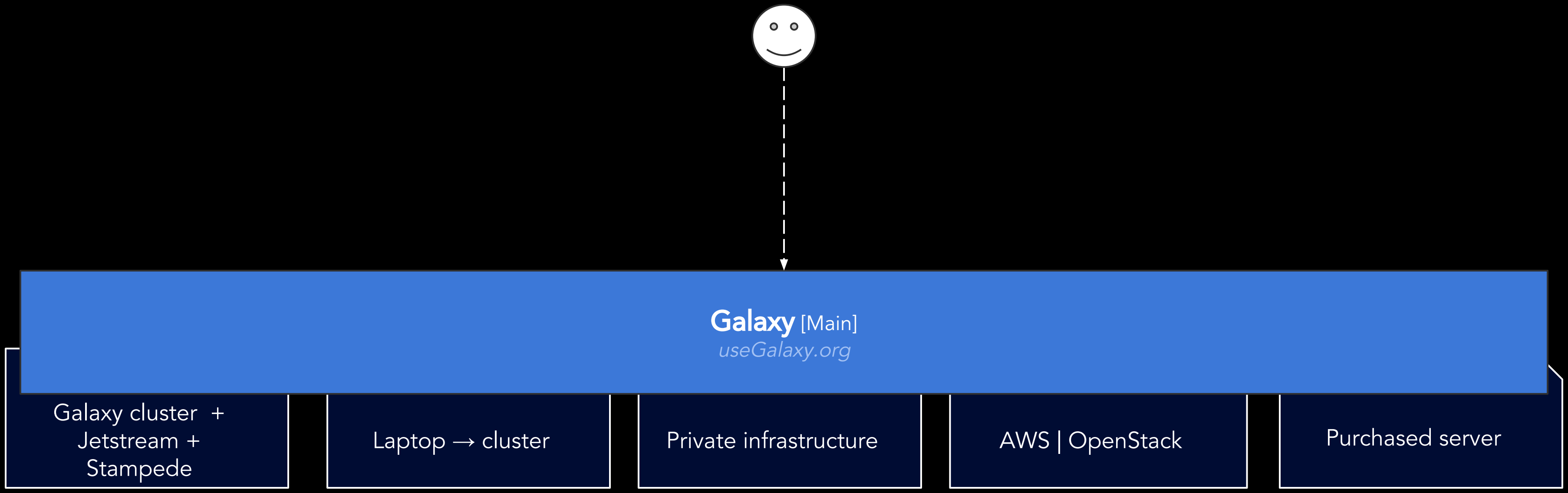
Each server is custom-crafted and centrally administered.

Galaxy-as-a-Service



Galaxy
without
Quotas!

Galaxy-as-a-Service: towards a federated Galaxy



Afgan E, Jalili J, Goonasekera N, Taylor N, Goecks J, "**Federated Galaxy: Biomedical Computing at the Frontier**", IEEE Cloud 2018, San Francisco, July 2018.

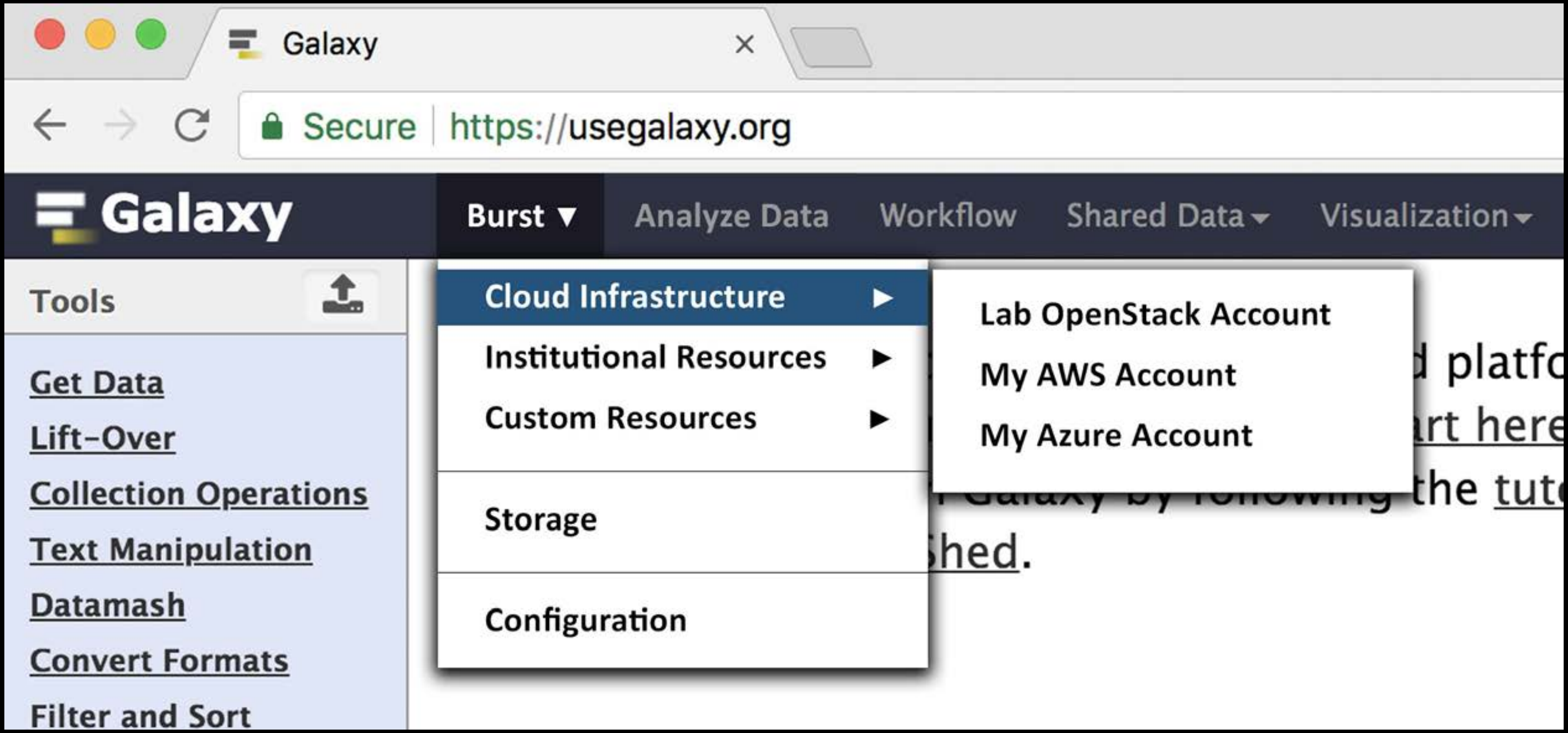
GaaS core components

Compute

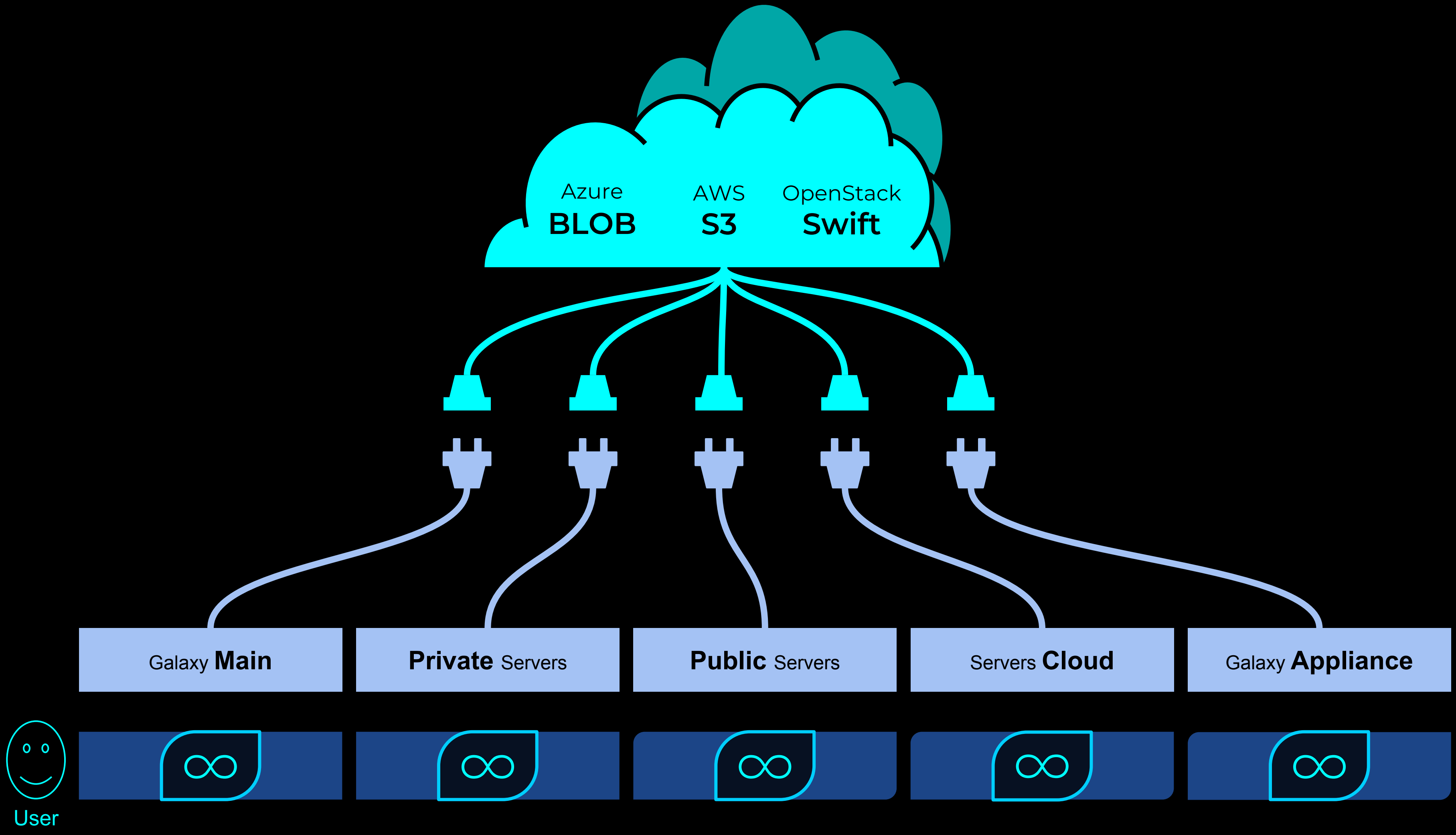
Storage

AuthNZ

Compute: attach compute resources to a session



Storage: allow a user to link to object stores

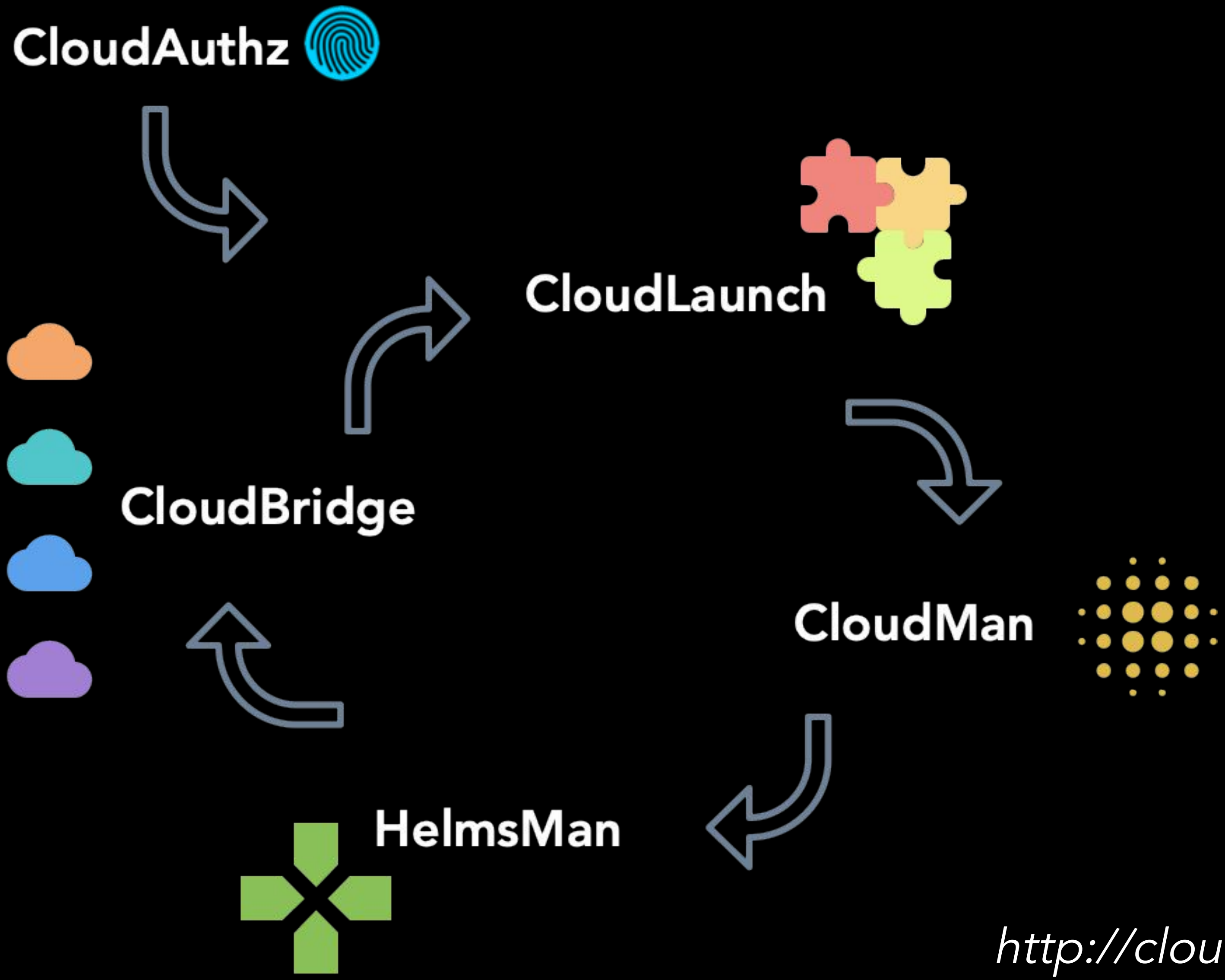


Auth: handle user identity and resource ownership

- Rely on identity that can span Galaxy instances
- Remove, and at least minimize, storing user cloud credentials
- Be compatible with a variety of resource providers

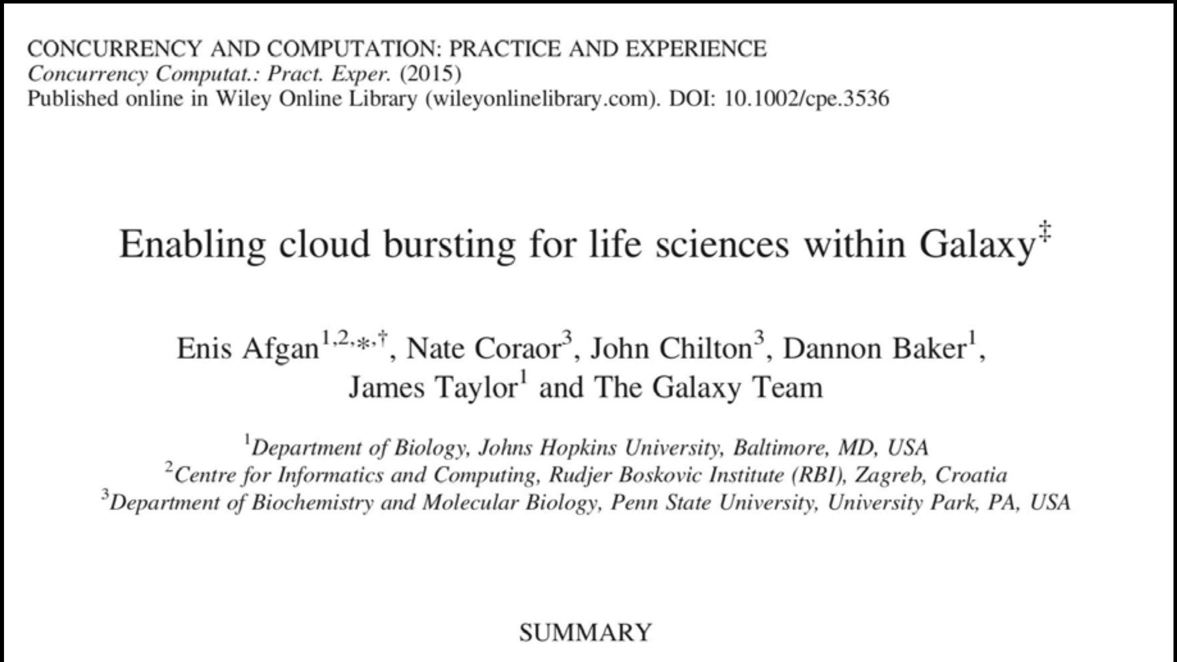
Jalili V, Afgan E, Taylor J, Goecks J, “Cloud Bursting Galaxy: Federated Identity and Access Management”, Biorxiv https://doi.org/10.1101/506238, Dec 2018.

A tool suite for cloud virtual environments: CloudVE



http://cloudve.org

Today: a closer look at compute bursting



CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. (2015)
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/cpe.3536

Enabling cloud bursting for life sciences within Galaxy[‡]

Enis Afgan[1,2,*,†], Nate Coraor[3], John Chilton[3], Dannon Baker[1],
James Taylor[1] and The Galaxy Team

[1]*Department of Biology, Johns Hopkins University, Baltimore, MD, USA*
[2]*Centre for Informatics and Computing, Rudjer Boskovic Institute (RBI), Zagreb, Croatia*
[3]*Department of Biochemistry and Molecular Biology, Penn State University, University Park, PA, USA*

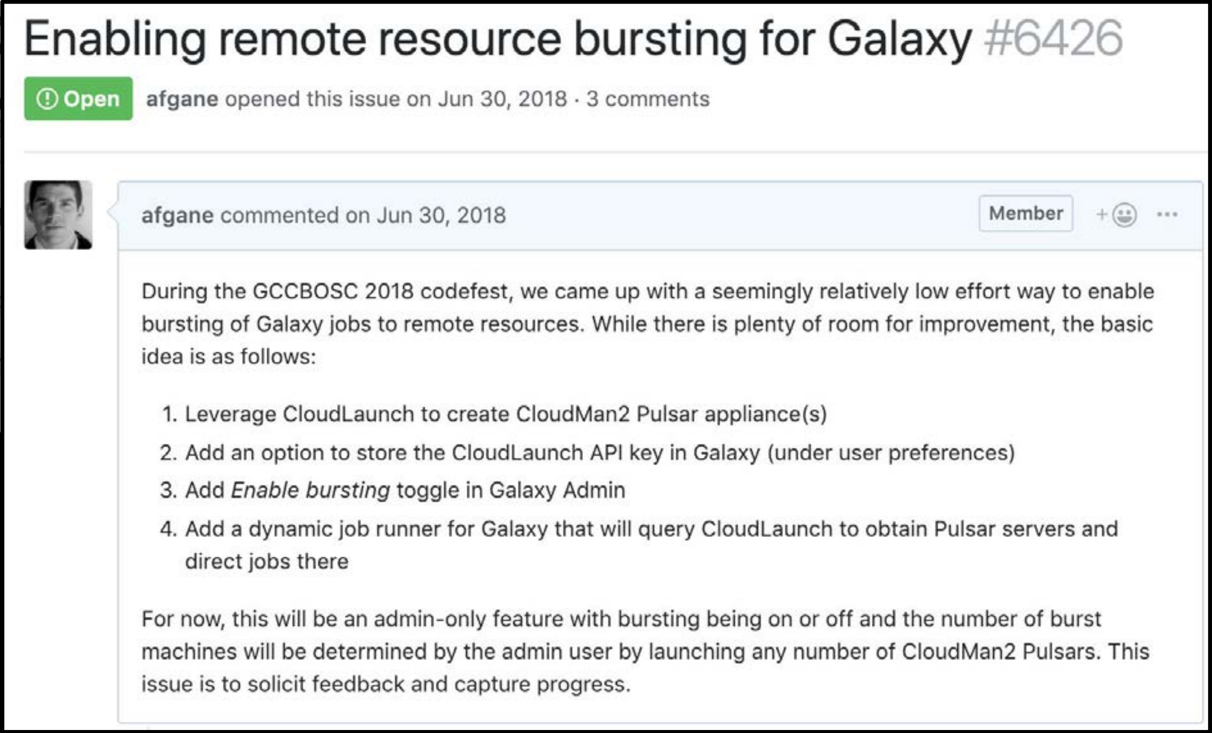
SUMMARY

Fueled by the radically increased capacity to generate data o
research has been constrained by the ability to analyze data. Ga
tion and analysis platform for life science research, has been
However, the scale of data and the scope of tools required h
any monolithic deployment of the Galaxy application. We
approach to utilizing compute and storage resources is necess
in creating a ubiquitous platform capable of simultaneously ut
resources. Specifically, the requirements, process, and an im
detailed. Copyright © 2010 John Wiley & Sons, Ltd.

Received 18 January 2015; Revised 23 March 2015; Accepted 14 Ap

KEY WORDS: cloud computing; cloud bursting; data ar
performance; design; experimentation

2015 proof of concept



Enabling remote resource bursting for Galaxy #6426

afgane opened this issue on Jun 30, 2018 · 3 comments

afgane commented on Jun 30, 2018

During the GCCBOSC 2018 codefest, we came up with a seemingly relatively low effort way to enable bursting of Galaxy jobs to remote resources. While there is plenty of room for improvement, the basic idea is as follows:

1. Leverage CloudLaunch to create CloudMan2 Pulsar appliance(s)
2. Add an option to store the CloudLaunch API key in Galaxy (under user preferences)
3. Add *Enable bursting* toggle in Galaxy Admin
4. Add a dynamic job runner for Galaxy that will query CloudLaunch to obtain Pulsar servers and direct jobs there

For now, this will be an admin-only feature with bursting being on or off and the number of burst machines will be determined by the admin user by launching any number of CloudMan2 Pulsars. This issue is to solicit feedback and capture progress.

2018 implementation plan

2019: GalaxyCloudRunner

- Enables bursting of user jobs to remote compute resources for the Galaxy application
- Integrated with Galaxy 19.01 release but also applicable to older releases
- Enables bursting per Galaxy instance
- Documentation available at *galaxycloudrunner.readthedocs.io*

GalaxyCloudRunner usage

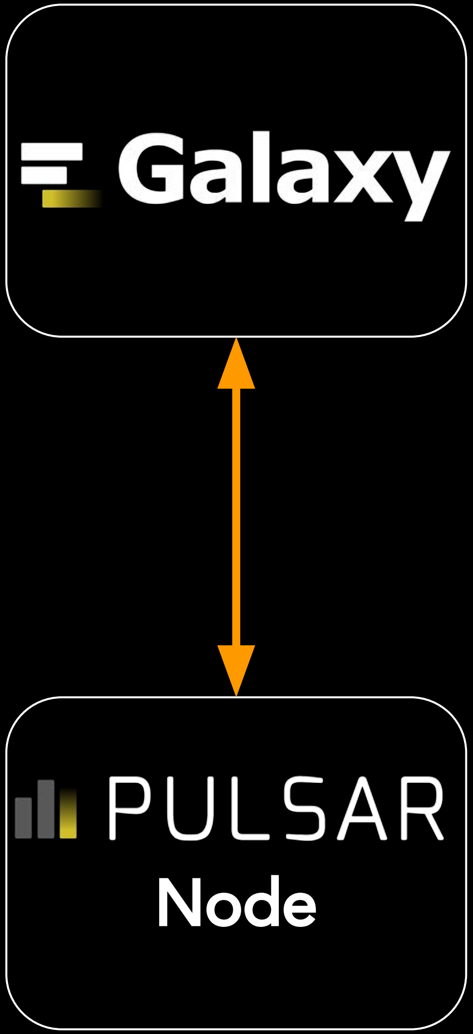
1. Install `galaxycloudrunner` Python library into your Galaxy's virtual environment
2. Add a job rule to Galaxy which will determine the Pulsar node to route to
3. Configure your `job_conf.xml` to use this rule
4. Launch as many Pulsar nodes as you need through CloudLaunch
5. Submit your jobs as usual

What is Pulsar?

- Python server application
- Allows a Galaxy server to run jobs on a remote system
- No shared file system required
- Configurable
- Securable
- Can submit jobs to HPC queueing system
- Automatically handles tool dependency management

https://pulsar.readthedocs.io/

How Pulsar works



1. User clicks “Execute”
2. Galaxy packs up and sends:
   - Data
   - Config files
   - Tool name & version
   - Parameters and other job metadata
3. Pulsar accepts the job
4. Pulsar checks if tool is installed locally
   - If not - Installs tool with Conda or Docker
5. Pulsar submits job to local queue
6. Pulsar waits until job complete
7. Pulsar packs up result and sends it back to Galaxy

What is CloudLaunch?

A gateway for discovering and launching applications on a variety of clouds.

Cloud-agnostic

Backed by CloudBridge, use native cloud capabilities for infrastructure management

Pluggable and extensible

Arbitrary launch process and UI are supported, via an isolated plug-in mechanism

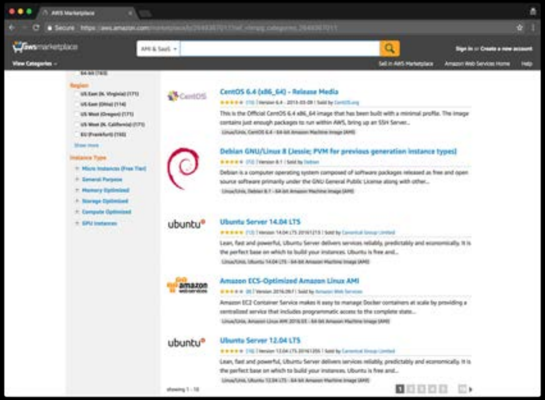
UI and REST API

UI available for end-users but it is all API driven for integration into external apps

Try it at *https://launch.usegalaxy.org/*

Afgan, E., Lonie, A., Taylor, J., Goonasekera, N., "**CloudLaunch: Discover and Deploy Cloud Applications**", Future Generation Computer Systems, June 2018.

Why CloudLaunch?


awsmarketplace

CentOS 6.4 (x86_64) - Release Media

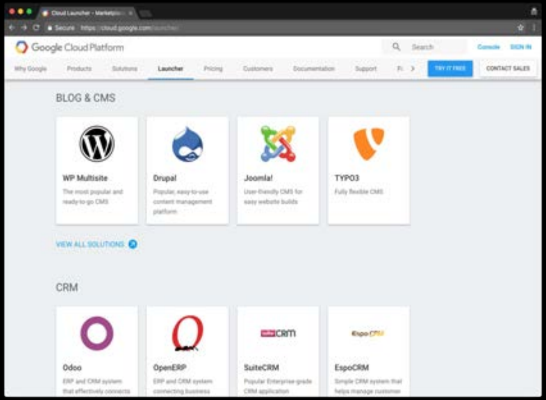
Debian GNU/Linux 8 (Jessie; PVM for previous generation instance types)

Ubuntu Server 14.04 LTS

Amazon ECS-Optimized Amazon Linux AMI

Ubuntu Server 12.04 LTS

AWS Marketplace


Google Cloud Platform

BLOG & CMS

WP Multisite

Drupal

Joomla!

TYPO3

VIEW ALL SOLUTIONS

CRM

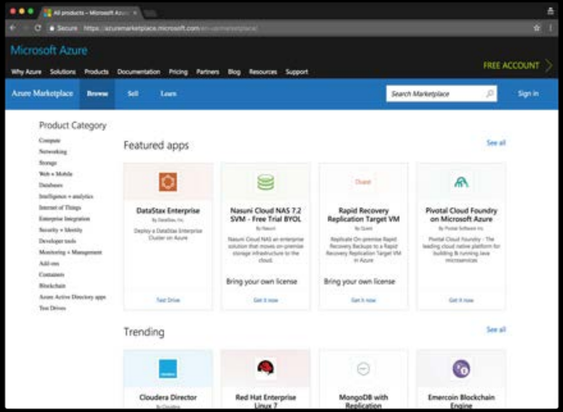
Odoo

OpenERP

SuiteCRM

EspoCRM

GCE Solutions


Microsoft Azure

Azure Marketplace

Product Category

Featured apps

DataStax Enterprise

Nasuni Cloud NAS 7.2 SVM - Free Trial BYOL

Rapid Recovery Replication Target VM

Pivotal Cloud Foundry on Microsoft Azure

Trending

Cloudera Director

Red Hat Enterprise Linux 7

MongoDB with Replication

Emercoin Blockchain Engine

Azure Marketplace

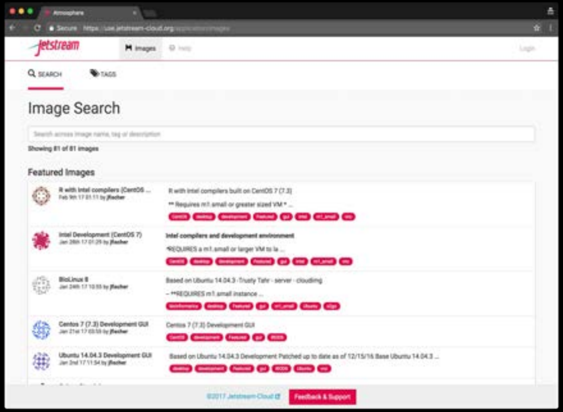
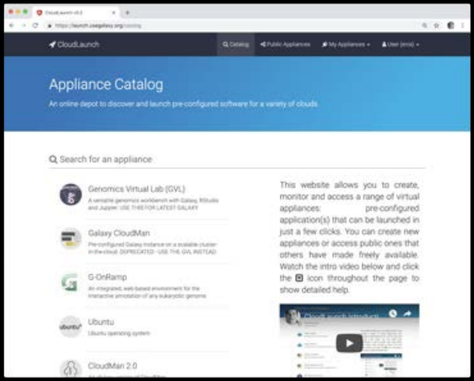

Jetstream

Image Search

Featured Images

Jetstream Atmosphere VMs

CloudLaunch


CloudLaunch

Appliance Catalog

An online depot to discover and launch pre-configured software for a variety of clouds

Search for an appliance

Genomics Virtual Lab (GVL)

Galaxy CloudMan

G-OnRamp

Ubuntu

CloudMan 2.0

This website allows you to create, monitor and access a range of virtual appliances: pre-configured application(s) that can be launched in just a few clicks. You can create new appliances or access public ones that others have made freely available. Watch the intro video below and click the icon throughout the page to show detailed help.

Consistent interface

Single, uniform API

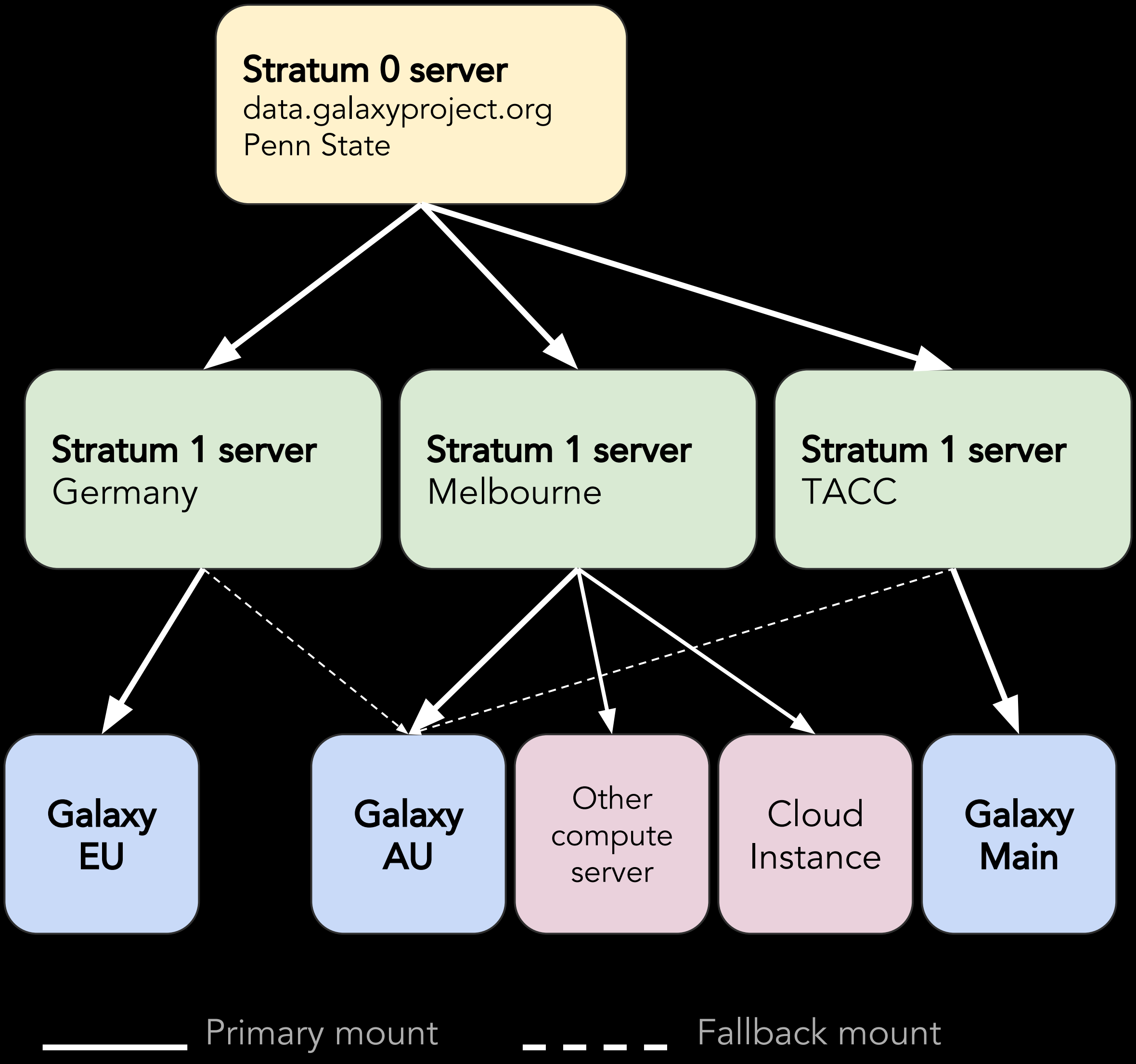
Multi-cloud

Shared global data via CVMFS

Stratum 0: The canonical source
Transactional updates

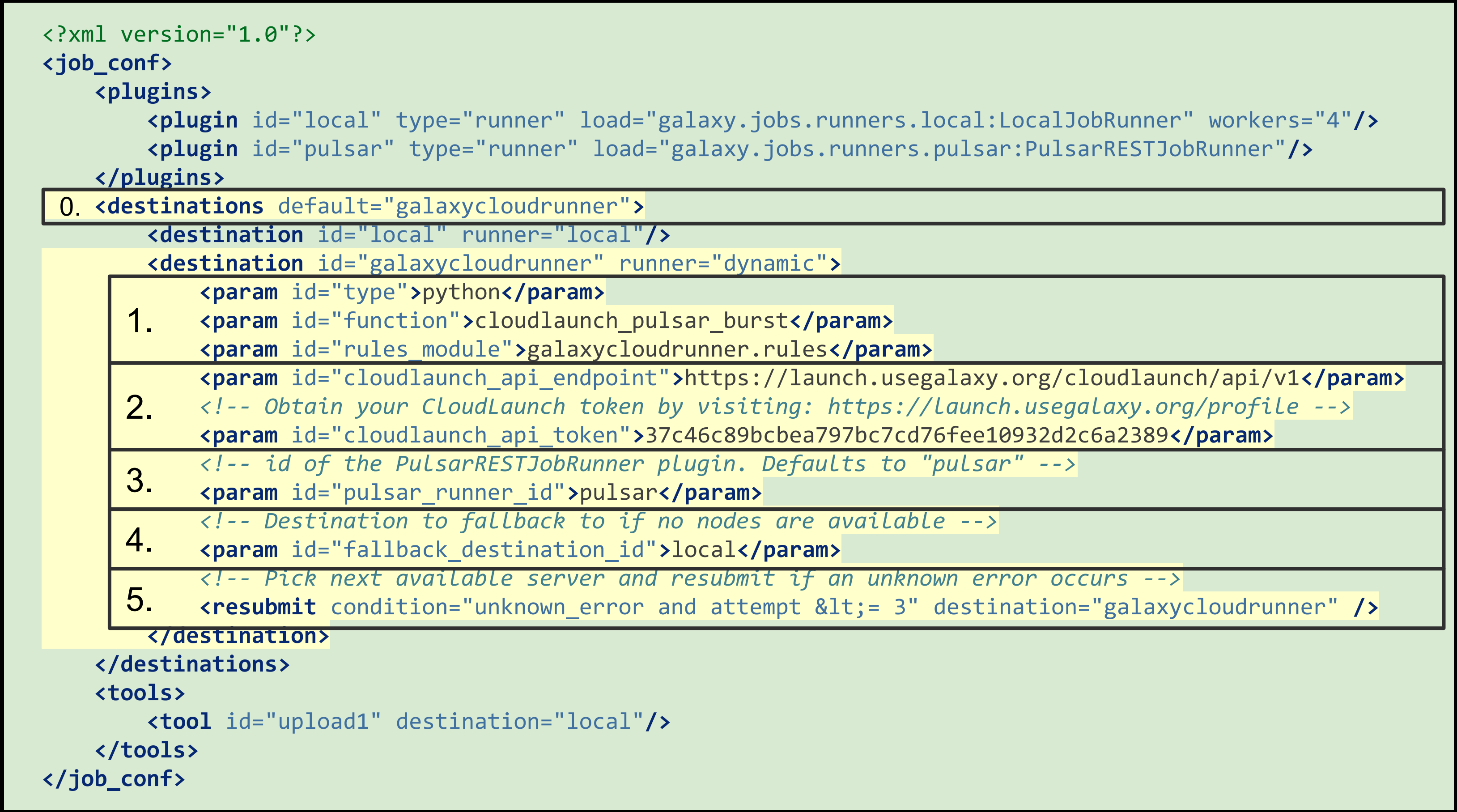
Stratum 1: Multiple servers
Mirrors Stratum 0 server
Continuous updates

User servers: Many multiple servers
Mounts repo from stratum 1
Based on GEO-API
With fallback to other stratum 1s



Configuring Galaxy

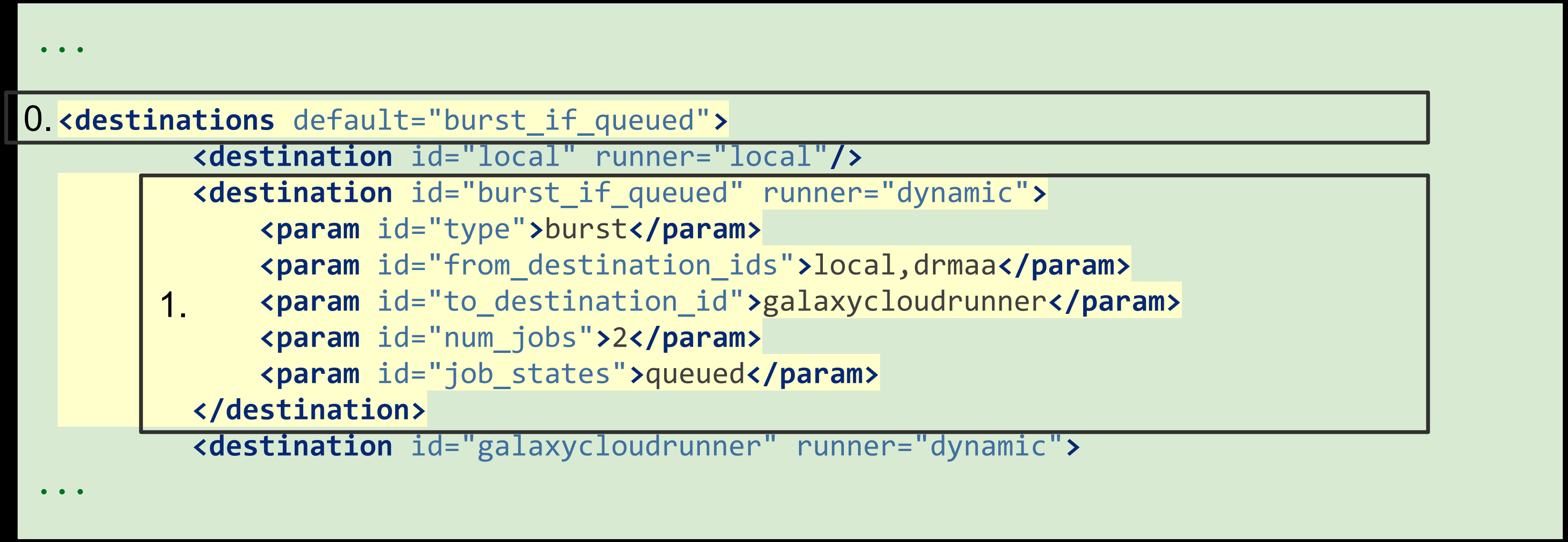
Make use of *dynamic destinations* to define *galaxycloudrunner* as the default destination



```
<?xml version="1.0"?>
<job_conf>
    <plugins>
        <plugin id="local" type="runner" load="galaxy.jobs.runners.local:LocalJobRunner" workers="4"/>
        <plugin id="pulsar" type="runner" load="galaxy.jobs.runners.pulsar:PulsarRESTJobRunner"/>
    </plugins>
0.  <destinations default="galaxycloudrunner">
        <destination id="local" runner="local"/>
        <destination id="galaxycloudrunner" runner="dynamic">
            <param id="type">python</param>
1.          <param id="function">cloudlaunch_pulsar_burst</param>
            <param id="rules_module">galaxycloudrunner.rules</param>
            <param id="cloudlaunch_api_endpoint">https://launch.usegalaxy.org/cloudlaunch/api/v1</param>
2.          <!-- Obtain your CloudLaunch token by visiting: https://launch.usegalaxy.org/profile -->
            <param id="cloudlaunch_api_token">37c46c89bcbea797bc7cd76fee10932d2c6a2389</param>
3.          <!-- id of the PulsarRESTJobRunner plugin. Defaults to "pulsar" -->
            <param id="pulsar_runner_id">pulsar</param>
4.          <!-- Destination to fallback to if no nodes are available -->
            <param id="fallback_destination_id">local</param>
5.          <!-- Pick next available server and resubmit if an unknown error occurs -->
            <resubmit condition="unknown_error and attempt &lt;= 3" destination="galaxycloudrunner" />
        </destination>
    </destinations>
    <tools>
        <tool id="upload1" destination="local"/>
    </tools>
</job_conf>
```

Support for opportunistic bursting

Route jobs to the remote cloud nodes only if the local queue is full.



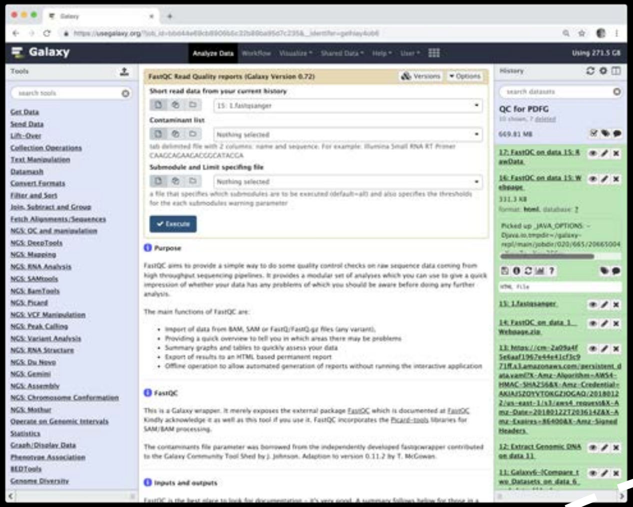
```
...

0.<destinations default="burst_if_queued">
          <destination id="local" runner="local"/>
          <destination id="burst_if_queued" runner="dynamic">
              <param id="type">burst</param>
              <param id="from_destination_ids">local,drmaa</param>
   1.         <param id="to_destination_id">galaxycloudrunner</param>
              <param id="num_jobs">2</param>
              <param id="job_states">queued</param>
          </destination>
          <destination id="galaxycloudrunner" runner="dynamic">
...
```

In addition, can burst based on input file size

GalaxyCloudRunner is extensible so can add your own rules

Galaxy cloud bursting in a picture



FastQC Read Quality reports (Galaxy Version 0.72)

Short read data from your current history: 15: 1.fastqsanger

Contaminant list: Nothing selected

tab delimited file with 2 columns: name and sequence. For example: Illumina Small RNA RT Primer CAAGCAGAAGACGGCATACGA

Submodule and Limit specifing file: Nothing selected

a file that specifies which submodules are to be executed (default=all) and also specifies the thresholds for the each submodules warning parameter

Purpose

FastQC aims to provide a simple way to do some quality control checks on raw sequence data coming from high throughput sequencing pipelines. It provides a modular set of analyses which you can use to give a quick impression of whether your data has any problems of which you should be aware before doing any further analysis.

The main functions of FastQC are:

- Import of data from BAM, SAM or FastQ/FastQ.gz files (any variant),
- Providing a quick overview to tell you in which areas there may be problems
- Summary graphs and tables to quickly assess your data
- Export of results to an HTML based permanent report
- Offline operation to allow automated generation of reports without running the interactive application

FastQC

This is a Galaxy wrapper. It merely exposes the external package FastQC which is documented at FastQC. Kindly acknowledge it as well as this tool if you use it. FastQC incorporates the Picard-tools libraries for SAM/BAM processing.

The contaminants file parameter was borrowed from the independently developed fastqcwrapper contributed to the Galaxy Community Tool Shed by J. Johnson. Adaption to version 0.11.2 by T. McGowan.

Inputs and outputs

FastQC is the best place to look for documentation - it's very good. A summary follows below for those in a

QC for PDFG

669.81 MB

17: FastQC on data 15: RawData

16: FastQC on data 15: Webpage

331.3 KB

format: html, database: ?

Picked up _JAVA_OPTIONS: -Djava.io.tmpdir=/galaxy-repl/main/jobdir/020/665/20665004

HTML file

15: 1.fastqsanger

14: FastQC on data 1: Webpage.zip

13: https://cm-2a09a4f5e6aaf1967e44e41cf3c971ff.s3.amazonaws.com/persistent_data.yaml?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAJSZOYVTOKGZIOGAQ/20180122/us-east-1/s3/aws4_request&X-Amz-Date=20180122T203614Z&X-Amz-Expires=86400&X-Amz-SignedHeaders

12: Extract Genomic DNA on data 11

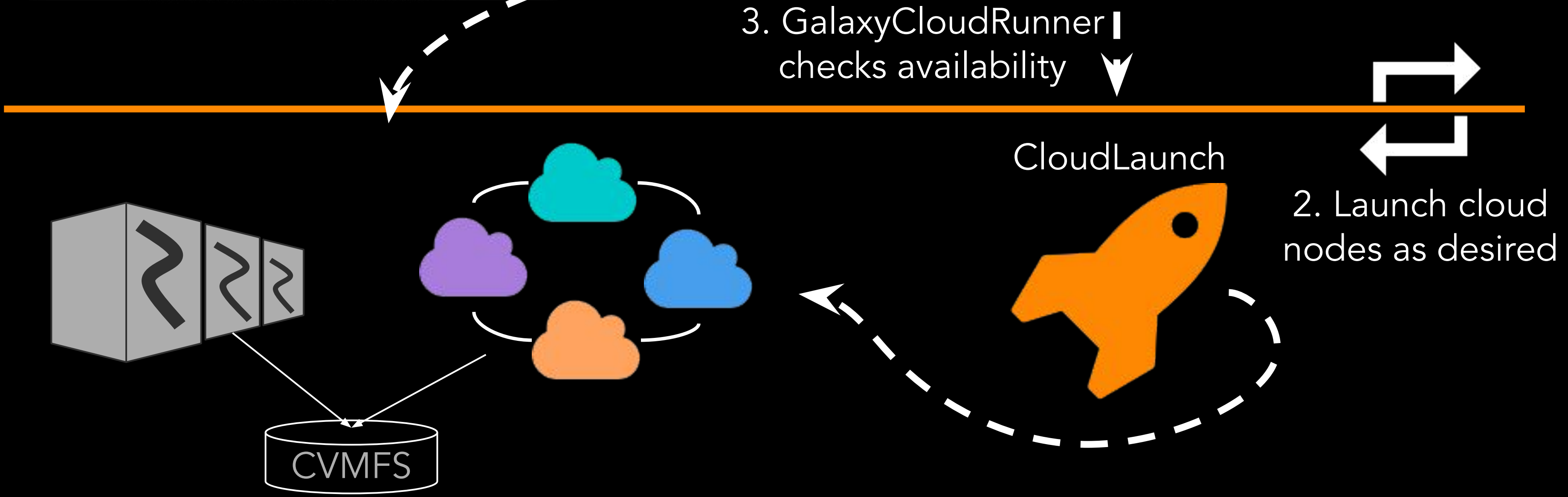
11: Galaxy6-[Compare two Datasets on data 6

job_conf.xml

```
<destination>
…
</destination>
```

1. One-time setup

4. Submit jobs as normal



3. GalaxyCloudRunner checks availability

CloudLaunch

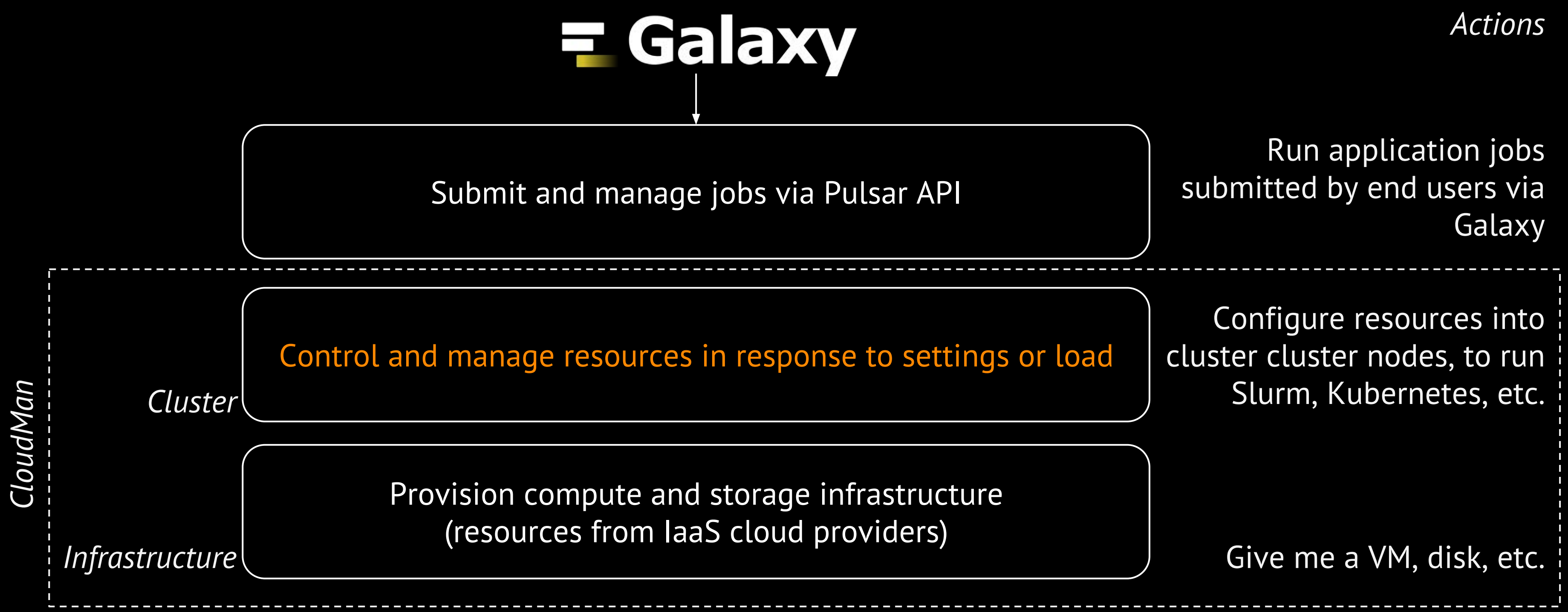
2. Launch cloud nodes as desired

Looking forward and beyond Galaxy

(Auto)-scaling, via CloudMan

Currently, each cloud node is a single, independent resource

Scale can be achieved by adding multiple nodes



Galaxy

Actions

Submit and manage jobs via Pulsar API

Run application jobs submitted by end users via Galaxy

CloudMan

Cluster

Control and manage resources in response to settings or load

Configure resources into cluster cluster nodes, to run Slurm, Kubernetes, etc.

Infrastructure

Provision compute and storage infrastructure (resources from IaaS cloud providers)

Give me a VM, disk, etc.

https://github.com/galaxyproject/cloudman/tree/v2.0

Beyond Galaxy use cases

- CloudBridge is a general-purpose, multi-cloud library for interacting with the IaaS resources

- CloudLaunch leverages CloudBridge and can launch a variety of applications; each appliance is a plugin with custom back-end and front-end components

- CloudMan is a cloud manager for orchestrating a running cloud deployment, primarily focusing on managing Kubernetes clusters for multiple clouds

- HelmsMan is a manager for Helm applications, currently integrated with CloudMan

Acknowledgments

Institutions

JOHNS HOPKINS UNIVERSITY

PennState

OHSU

THE UNIVERSITY OF MELBOURNE

Projects

Galaxy

SGCI Science Gateways Community Institute

genomics VIRTUAL LAB

Infrastructure

Google Cloud

amazon webservices™

XSEDE

Jetstream

Azure

nectar cloud