Horizontal scaling with Galaxy

Enis Afgan
Galaxy Team
Johns Hopkins University

Galaxy Africa
Cape Town
April 5, 2018.
Horizontal scaling: service growth
Horizontal scaling #2: service replication
Scaling Galaxy: overview

- Public servers
- Cloud clusters
- Local installations
Want a local Galaxy?

$ git clone -b release_18.01 https://github.com/galaxyproject/galaxy.git
$ sh run.sh
...
http://localhost:8080
Galaxy beyond the development server

Galaxy needs a complex ecosystem of software to operate effectively:

- Robust database
- Job manager
- FTP server
- Shared data
- Container manager
Automating installations

Automate the process of building each component

Codify knowledge about the system → easier to build, easier to reproduce

We use Ansible as the technology of choice

Roles:
- galaxy-os
- nginx
- postgresql
- postgresql_objects
- galaxy
- interactive-environments
- trackster
- pulsar
- galaxy-tools
- galaxy-extras

Playbooks:
- usegalaxy-playbook
- infrastructure-playbook
- galaxy-cloudman-playbook
- GalaxyKickStart

Control Machine
```
ansible-playbook -i inventory playbook.yml
```

Remote Machine
```
python ~/.ansible-tmp ...
sudo apt-get install ...
```
Closer look at the **Galaxy Main** installation
Galaxy Main requires scale

125,000 registered users

2PB user data

18M jobs run

100 training events (2017 & 2018)

Stats for Galaxy Main (usegalaxy.org) in March 2018
Running into scalability issues...
Decentralizing the installation

- Traditionally, Galaxy was designed for local installation and required a shared file system

- Implement a pluggable interface to compute resources to readily connect to external cluster(s)

- Leverage Pulsar for data staging and Galaxy resource/job manager
Expanding compute capacity

Leveraging National Cyberinfrastructure: Galaxy/XSEDE Gateway

- **Jetstream**
  - PTI IU Bloomington
  - Galaxy Cluster (Rodeo)
    - 256 cores
    - 2 TB memory
    - Corral/Stockyard
    - 20 PB disk

- **Stampede**
  - Stampede
    - 462,462 cores
    - 205 TB memory
  - TACC Austin

- **Pittsburgh Supercomputing Center**
  - PSC, Pittsburgh

- **TACC**
  - Texas Advanced Computing Center

Shared XSEDE resources
Dedicated resources
Integrate with external resources
More than job counts, scaling moved the horizon

Can now run larger jobs, and more jobs:

On Jetstream, 325,000 jobs run on behalf of 12,000 users

Can run new types of jobs:

Galaxy Interactive
Environments: Jupyter, RStudio
Scaling Galaxy: overview

Public servers

Cloud clusters

Individual installations

Flexibility

Usability
100+ public Galaxy Servers

bit.ly/gxyServers
Scaling with consistency

- Multiple Galaxy servers are wonderful for accessibility and versatility
- Globally, they do lead to a good bit of repeated effort → not very scalable

- Focus on usegalaxy.* federation
  - A set of coordinated Galaxy instances with a set of common core tools and reference genomes
  - .org / .eu / .org.au domains exist today
  - Leverage common reference data and, in future, tool and Galaxy binaries
  - Serve as a model for other local instances to reuse installation components
Galaxy federation components

Reference data

- Leverage CernVM file system (CVMFS) as a distributed, read-only file system
- A centrally updated and automatically replicating global set of servers
- Stratum-0: master copy
- Stratum-1: read-only replicas: 3 in the US, 1 in EU, 1 in AU
- Anyone can connect to these: bit.ly/gxyCVMFS

Tools

- Current list of tools (for usegalaxy.eu) is published at bit.ly/gxyEUtools
- Use Ephemeris command line tool to install locally
- Still work in progress; eventually will be able to use CVMFS directly
Scaling Galaxy: overview

Public servers

Cloud clusters

Individual installations

Flexibility

Usability
Galaxy on the cloud

- Launch your own instance of Galaxy on the cloud
  - Within minutes, using a web browser, including the infrastructure and configurations with ability to scale

- Based on CloudMan: a cloud manager for deploying Galaxy on a variety of cloud providers
Genomics Virtual Lab

**GVL:** a middleware layer of machine images, cloud management tools, and online services for cloud bioinformatics

- **A superset of Galaxy-on-the-cloud:** build arbitrarily sized compute clusters on demand, pre-populated with fully configured bioinformatics tools, reference datasets, workflows and visualisation options.
Launch-your-own, via CloudLaunch

Demo using https://launch.usegalaxy.org/
Scaling across clouds
Towards Galaxy-on-the-cloud 2.0

Goals:

- Uniform availability without infrastructure-specific builds
- Well-defined upgrade path for users
- Focus on institutions vs. individual users

Approach:

- Abstract provider differences
- Containerize everything
- Leverage federated infrastructure components (e.g., CVMFS, containers)
How is the launch-your-own enabled?

CloudLaunch.usegalaxy.org

CloudBridge.readthedocs.org

API

cloudman

APPLICATIONS
CloudBridge
A Simple Cross-Cloud Python Library

Multi-cloud computing with CloudBridge

CloudBridge: a simple, open-source Python multi-cloud library.

**Uniform API irrespective of the underlying provider**
No special casing of application code, unlike Apache Libcloud
Simpler code

**Provide a set of conformance tests for all supported clouds**
No need to test against each cloud, unlike Terraform
“Write-once-run-anywhere”

**Supports AWS, Azure, and OpenStack right now**
GCE support is forthcoming

http://cloudbridge.readthedocs.org
provider

compute
  - images
  - vm_types
  - instances
  - regions
    - zones
      - PlacementZone

security
  - keypairs
  - vm_firewalls
    - rules
      - VMFirewallRule

storage
  - volumes
  - snapshots
  - buckets
    - objects
      - BucketObject

networking
  - networks
  - subnets
  - routers
  - gateways
    - floating_ip
      - FloatingIP
CloudLaunch

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

CloudLaunch-as-a-Service

AWS Marketplace

GCE Solutions

Azure Marketplace

Jetstream Atmosphere VMs
**CloudLaunch features**

**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
CloudLaunch demo #2: multi-cloud, multi-app, API

Demo using https://launch.usegalaxy.org
CloudMan and beyond
Manage deployed infrastructure and applications
Managing deployed infrastructure

- Once deployed, an application needs management, and so does the infrastructure.

**Actions**

- Give me a VM, disk, etc.
- Configure this node to run Slurm, K8S, etc. or format & mount this disk as a FS.
- Run Galaxy, Jupyter, Pulsar, etc. in this environment.
- Run application jobs submitted by end users (e.g., via Galaxy).
- Run application jobs submitted by end users (e.g., via Galaxy).

**CloudMan**

- **Application**
  - Deploy applications and/or services onto available resources.
  - Run Galaxy, Jupyter, Pulsar, etc. in this environment.
- **Cluster**
  - Control and manage resources.
  - Configure this node to run Slurm, K8S, etc. or format & mount this disk as a FS.
- **Infrastructure**
  - Provision compute and storage infrastructure (resources from IaaS cloud providers).
  - Give me a VM, disk, etc.
Services composing the runtime system

- **CloudMan**: Launch appliance units via plugins and provide basic resource management capabilities.
- **CloudLaunch**: Compute, Storage, Clustering, Scaling, Monitoring.
- **HelmsMan**: Manage services: start/stop/update (i.e., wrap Helm capabilities).
- **CloudVE Dashboard**: Wrapper around the environment and project management capabilities.
- **Kubernetes**: Runtime platform, managed by the provider or CloudMan.

Applications from an external registry

*Bold notation* indicates approximate degree of completion.
Looking forward: two models of usage

Virtual laboratory

Native application integration

CloudVE Framework

- CloudMan
- CloudLaunch
- CloudBridge

SaaS
- Jupyter
- RStudio
- Galaxy

PaaS
- CloudVE Framework
  - High availability
  - Automatic scaling
  - Cost estimation
  - Automatic backups
  - Community provided extensions

IaaS
- Amazon Web Services
- Microsoft Azure
- Google Compute Engine
- OpenStack

External application registries

Compute resources

Storage resources

Galaxy

CloudVE env

IaaS resources

Large public datasets
Conclusions

We’ve seen three models of scaling Galaxy:

- Public servers
  - Cumulatively great versatility
  - Most accessible
  - Bringing about reusable components
  - Powerful and readily usable
  - Customizable
  - Requires infrastructure and ongoing maintenance
  - Can grow very large

- Cloud clusters

- Individual installations

Flexibility

Usability