Deployment Options for Gen3

Gen3 Community Forum
2 July 2025

THE UNIVERSITY OF CHICAGO
CENTER FOR TRANSLATIONAL DATA SCIENCE

OHSU

Australian BioCommons

NeSI
New Zealand eScience Infrastructure

The Agenda

- Community Slides - Conrad Leonard, Australian BioCommons
- Update on Gen3 deployment on Openstack - Claire Rye, Nathalie Giraudon, and Carvin Chen; New Zealand eScience Infrastructure (NeSI)
- Cloud native platform engineering approaches for research and researchers - Colin Griffin, Krumware
- Deploying on-premise Gen3; constraints, plans and opportunities - National Computational Infrastructure (NCI) Australia Team
- Gen3 single node deployment - Platform Engineering Team, Center for Translational Data Science, University of Chicago

Community Slides

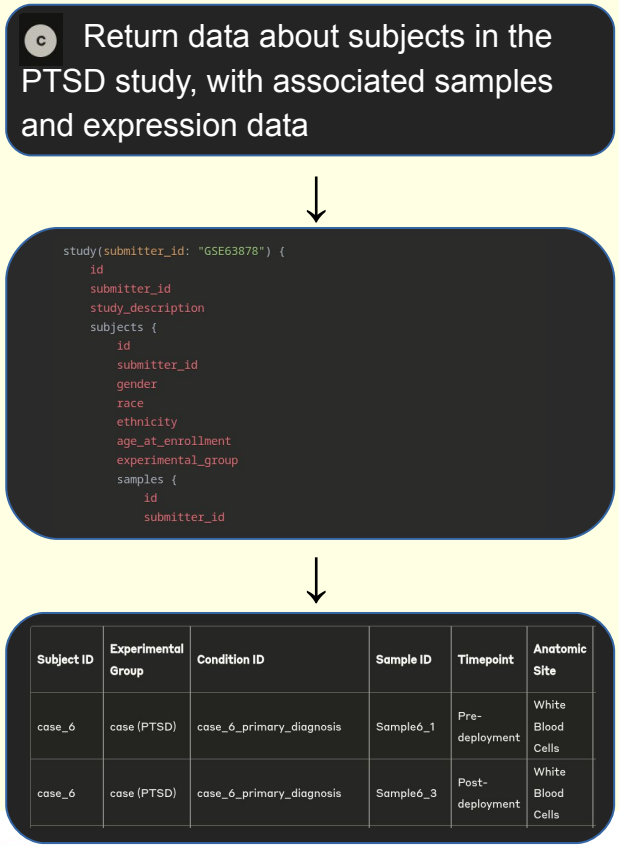
Natural Language Access to Gen3 Commons via MCP

PROTOTYPE

Problem
Gen3 hosts rich, structured research data — but extracting insights often requires schema knowledge and GraphQL expertise, or custom UI widgets

Solution
A Model Context Protocol server that connects an LLM to a Gen3 data commons

Ask
Get involved: try it, give feedback, contribute.
https://github.com/delocalizer/gen3-mcp



Australian BioCommons

Update on Gen3 deployment on Openstack

Claire Rye, Nat Giraudon, and Carvin Chen;
New Zealand eScience Infrastructure (NeSI)

Update on Gen3 deployment on Openstack

New Zealand eScience Infrastructure

About NeSI https://www.nesi.org.nz/

"Driven by the needs of researchers for high-performance productivity, New Zealand eScience Infrastructure (NeSI) designs, builds, and operates a specialised platform of shared high performance computing (HPC) infrastructure and a range of eResearch services."

This drove the need to deploy Gen3 on NeSI own Cloud system… on premise deployment

GEN3 is used for the Aotearoa Genomics Data Repository and Rakeiora project (prototype)

- https://data.agdr.org.nz/
- https://brownse.rakeiora.ac.nz/

genomics aotearoa

A new era for eResearch as NeSI integrates with REANNZ

In 2024, the Minister of Science, Innovation and Technology requested that REANNZ outline how it could integrate the services and capabilities provided by NeSI, into REANNZ.

A draft approach was approved by MBIE in November with the integration of services and transfer of funding to occur at the start of the new financial year, 1 July 2025.

If you want to know more about REANNZ visit www.reannz.co.nz

What happens since the last presentation Jan 2023

- Some staff changes

🧑‍💼 New Arrival

Carvin (Rui) Chen 🆕 Joined fully AGDR in Jan 2025

🪄 Bringing fresh energy!

👋 Farewell

Eirian Perkins 🎈 Off to new adventures

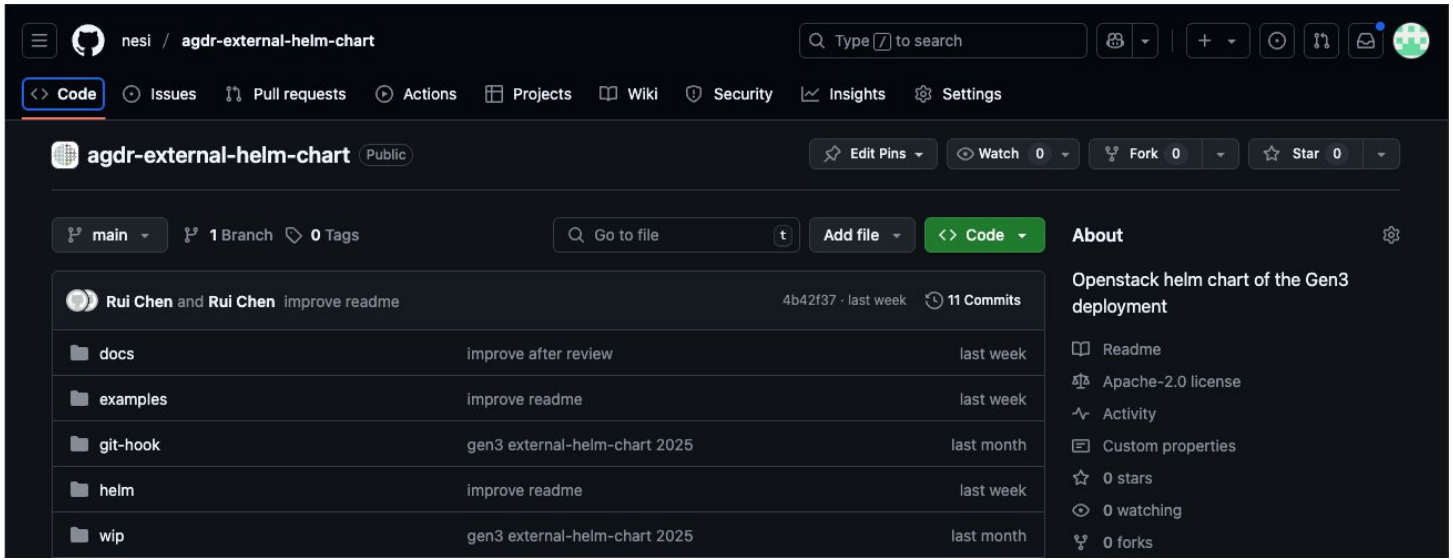
- 🔄 Shifting Gears

Somesh Nistala 🚀 Now exploring other exciting projects still part of the orbit!

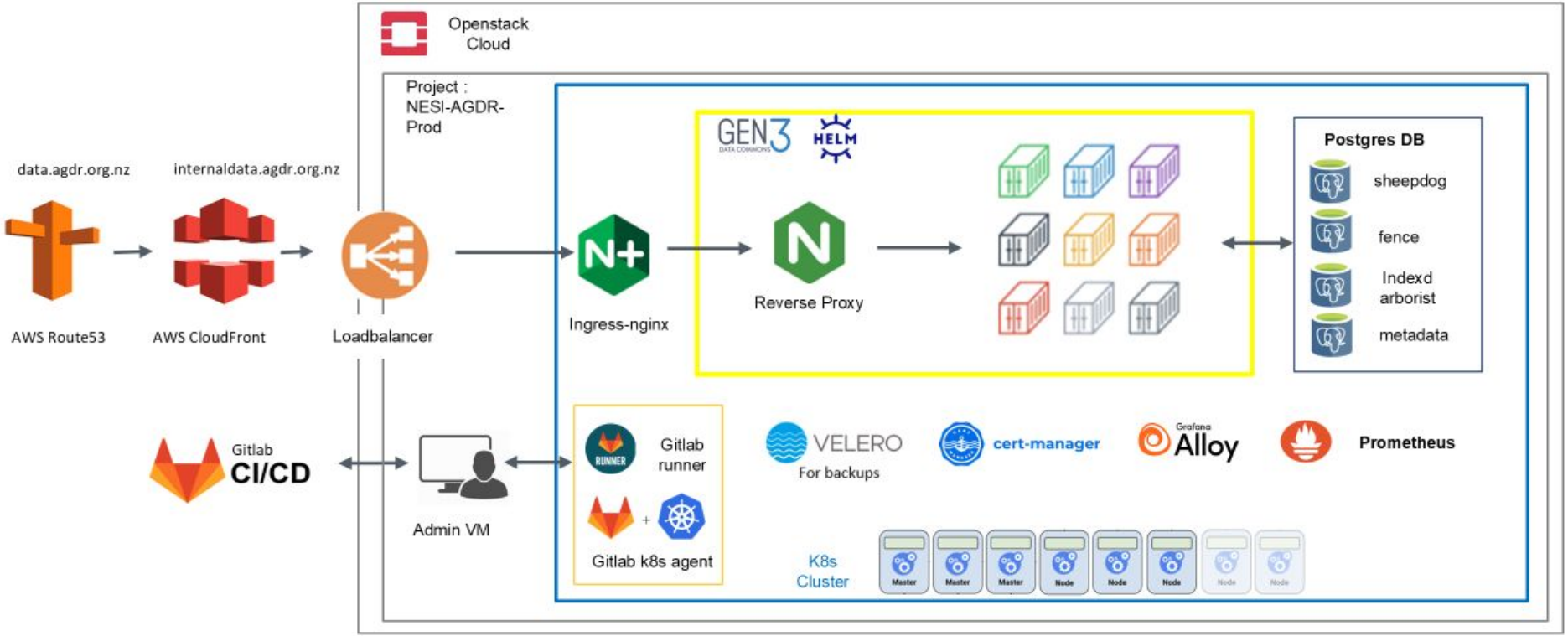
Openstack Helm Chart

📌 Link to the Helm chart:

https://github.com/nesi/agdr-external-helm-chart



AGDR Gen3 Architecture

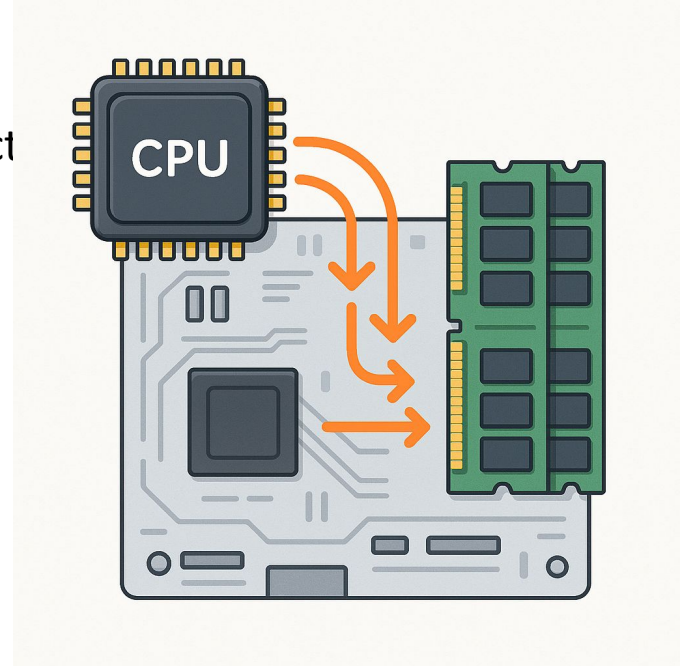


General consideration

The goal is to deploy a stable, reliable Gen3 cluster on the OpenStack cloud and enable highly efficient operations through the coordinated use of Kubernetes, GitLab CI/CD, and GEN3 Helm charts.

Prerequisites

- Has the admin permission of a OpenStack project
- Deployed a Kubernetes cluster into the project



<u>Estimate the resources usage based on workload:</u>

Test cluster:	1 master node (4CPU.8RAM),
	2 worker nodes (8CPU.16RAM)
Production cluster:	3 master nodes(8CPU.16RAM),
	3 worker nodes (16CPU.32RAM)

Node auto scaling is enabled

File structure of work repo (1/2)

`./.gitlab-ci.yml`

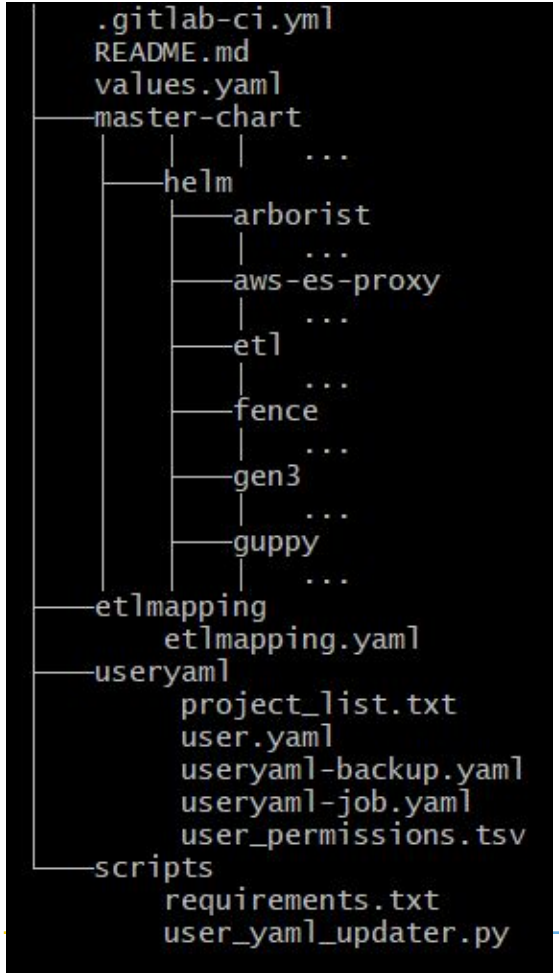
Definition of gitlab pipeline.

`./values.yaml`

Customized configuration of gen3 helm chart, including global configuration, and images, resources, database settings for each sub chart.

`./master-chart`

Helm charts from **uc-cdis/gen3-helm**



File structure of work repo (2/2)

`./etlmapping`

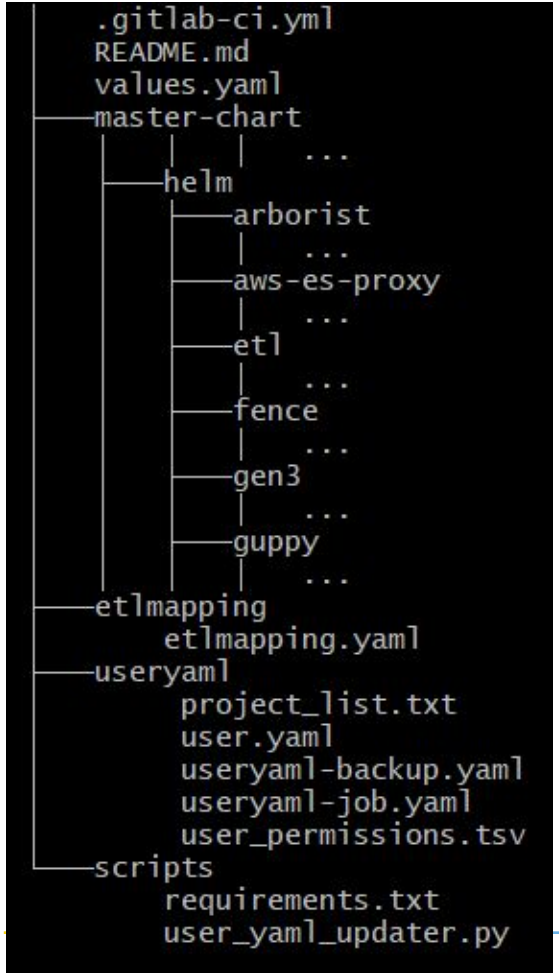
Store the definition file of etlmapping.

`./useryaml`

Store the files for updating user yaml.

`./scripts`

Store the custom scripts used in the pipeline



Experience Sharing Highlights (1/9)

1. Centralized main configuration with a customized values.yaml file

- Global configuration
- The images source and version
- Resources request and limit
- Databases connection settings
- And so on….

./values.yaml:

```yaml
# custom values for gen3.
# This is a YAML-formatted file.
# Declare variables to be passed into your
templates.

# Global configuration
global:
  aws:
    enabled: false
  dev: false
  postgres:
    dbCreate: false
  hostname: "test.agdr.org.nz"
  dictionaryUrl:
"https://dictionary-bucket.s3.amazonaws.com/xxxx.js
on"
  fenceUrl : "https://test.agdr.org.nz/user"
  indexdUrl: "http://indexd-service"
  arboristUrl: "http://arborist-service"
# Dependancy Charts configuration

# -- main configurations for arborist chart
arborist:
  enabled: true
  postgres:
    host:
```

```yaml
# -- main configurations for aws-es-proxy chart
aws-es-proxy:
  enabled: true
  image:
    repository:
docker.elastic.co/elasticsearch/elasticsearch-oss
    pullPolicy: IfNotPresent
    tag: "7.10.2"
  resources:
    requests:
      cpu: 1
      memory: 4Gi
    limits:
      memory: 8Gi
      cpu: 2
  netPolicy:
    ingressApps:
      ....
      - gen3job
    egressApps:
      ....
      - gen3job
  esEndpoint: elasticsearch

# -- main configurations for indexd chart
indexd:
  enabled: true
```

Experience Sharing Highlights (2/9)

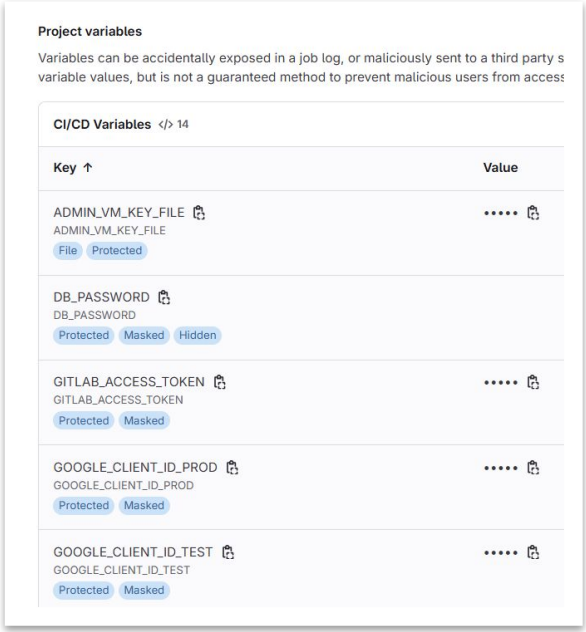
2. Extract large embedded content from subchart values.yaml files into separate files for better readability and management.

- Fence: user.yaml
- ETL: etlmapping.yaml
- Portal: Gitops-logo.png, gitops.css ….

Experience Sharing Highlights (3/9)

3. Manage secrets in the CI/CD variables

We securely store all secrets in GitLab CI/CD variables and reference them directly in our pipeline scripts. This approach makes it easy to manage and use secrets while improving security and reducing the risk of exposing sensitive data.

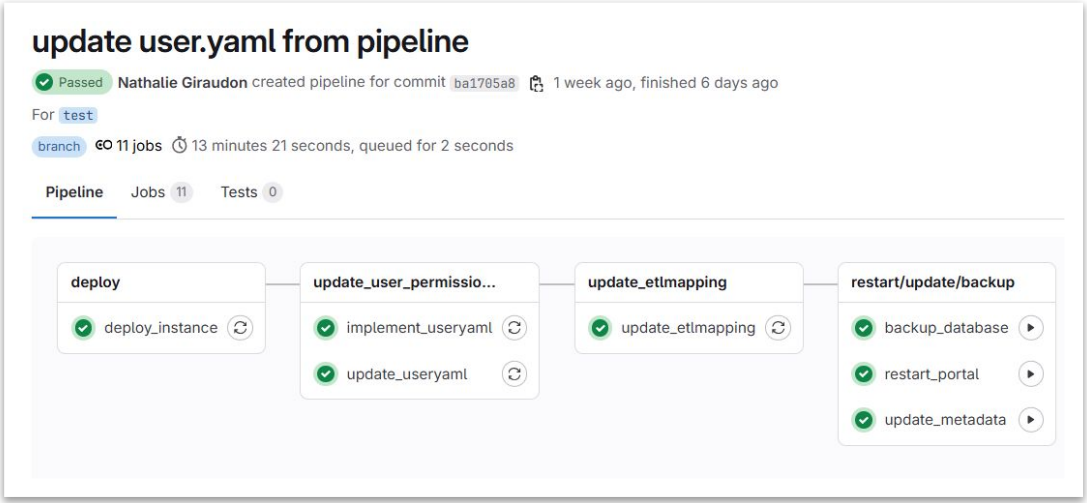


We can also use GCP to store the secrets and referenced by gitlab pipeline

Experience Sharing Highlights (4/9)

4. Automated Maintenance tasks via CI/CD Pipelines 1/2

- Updating user permissions
- Re-indexing elasticsearch
- Updating metadata db
- Restarting service
- Backup database
- ……



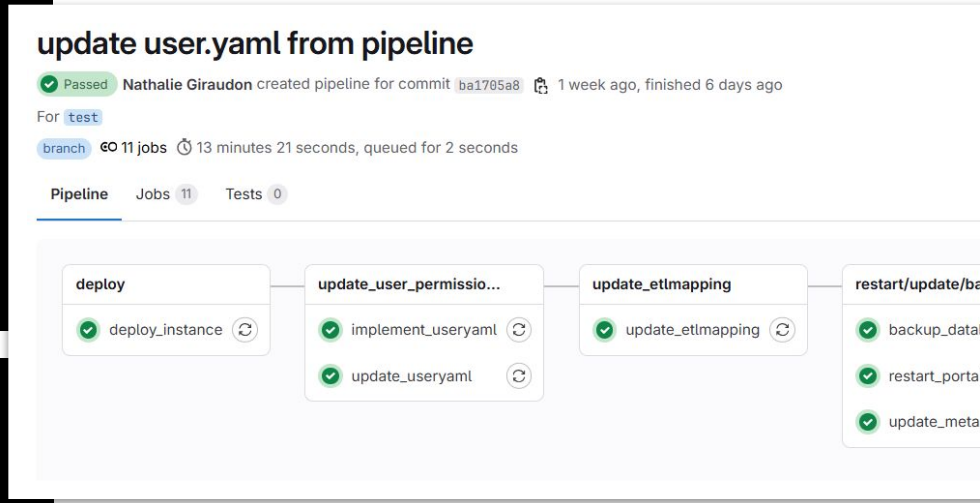
Experience Sharing Highlights (5/9)

4. Automated Maintenance tasks via CI/CD Pipelines (2/2)

- Updating user permissions

```
update_useryaml:
  .....
  script:
    - apk add python3 py3-pip
    - python3 -m venv venv
    - source ./venv/bin/activate
    - pip install -r scripts/requirements.txt
    - python scripts/user_yaml_updater.py
    - kubectl get configmap useryaml -n $KUBE_NAMESPACE
-o yaml > useryaml/useryaml-backup.yaml
```

```
implement_useryaml:
  .....
  script:
    - kubectl delete job useryaml || true
    - kubectl apply -f "useryaml/useryaml-job.yaml"
    - kubectl wait --for=condition=complete
--timeout=300s job/useryaml
```



Experience Sharing Highlights (6/9)

5. Helm chart changes for OpenStack deployment (1/4)

Changes to the helm chart configuration for OpenStack deployment:

(1) Add fenceUrl, arboristUrl, indexdUrl in global section, and attach the value to comfigmap manifest-global via global-manifest.yaml, because those Urls are expected in some sub chart deployment from manifest-global.

./values.yaml:

```yaml
# custom values for gen3.
# This is a YAML-formatted file.
# Declare variables to be passed into your templates.

# Global configuration
global:
  aws:
    enabled: false
  dev: false
  postgres:
    dbCreate: false
  hostname: "test.agdr.org.nz"
  dictionaryUrl:
"https://dictionary-bucket.s3.amazonaws.com/xxxx.json"
  fenceUrl : "https://test.agdr.org.nz/user"
  indexdUrl: "http://indexd-service"
  arboristUrl: "http://arborist-service"
  portalApp: gitops
  publicDataSets: true
  tierAccessLevel: libre
  tierAccessLimit: "1000"
  netPolicy:
    enabled: true
```

\helm\portal\templates\deployment.yaml:

```yaml
            - name: FENCE_URL
              valueFrom:
                configMapKeyRef:
                  name: manifest-global
                  key: fence_url
                  optional: true
            - name: INDEXD_URL
              valueFrom:
                configMapKeyRef:
                  name: manifest-global
                  key: indexd_url
                  optional: true
```

\helm\gen3\templates\global-manifest.yaml:

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: manifest-global
data:
  "environment": {{ .Values.global.environment | quote }}
  "hostname": {{ .Values.global.hostname | quote }}
  "revproxy_arn": {{ .Values.global.revproxyArn | quote }}
  "dictionary_url": {{ .Values.global.dictionaryUrl | quote }}
  "portal_app": {{ .Values.global.portalApp | quote }}
  "public_datasets": {{ .Values.global.publicDataSets | quote }}
  "fence_url": {{ .Values.global.fenceUrl | quote }}
  "indexd_url": {{ .Values.global.indexdUrl | quote }}
  "arborist_url": {{ .Values.global.arboristUrl | quote }}
...
```

Experience Sharing Highlights (7/9)

5. Helm chart changes for OpenStack deployment (2/4)

(2) Resolved the issue when set dbCreate to false, the dbcreated key was missing in these secrets: arborist-dbcreds, fence-dbcreds, indexd-dbcreds, peregrine-dbcreds, sheepdog-dbcreds.

helm/common/templates/_db_setup_job.tpl

```
{{- if not $.Values.postgres.dbCreate }}
  dbcreated: {{ "true" | b64enc | quote }}
{{- end }}
......
```

Experience Sharing Highlights (8/9)

5. Helm chart changes for OpenStack deployment (3/4)

(3) Allow outbound access to port 5432 for PostgreSQL communication across namespaces in the netpolicy.

helm/common/templates/_netpolicy_templates.tpl

```
{{- define "common.db_netpolicy" -}}
  {{- if .Values.global.netPolicy.enabled }}
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: {{ .Chart.Name }}-db-netpolicy
spec:
  ……
  policyTypes:
  - Egress
  egress:
    - to:
        - {}
      ports:
        - protocol: TCP
          port: 5432
```

Experience Sharing Highlights (9/9)

5. Helm chart changes for OpenStack deployment (4/4)

(4) Configurated sub chart aws-es-proxy to use elasticsearch-oss image

Add gen3job to netPolicy – ingressApps/ egressApps for cronjob running

./values.yaml:

```yaml
# -- main configurations for aws-es-proxy chart
aws-es-proxy:
  enabled: true
  image:
    repository:
docker.elastic.co/elasticsearch/elasticsearch-oss
    pullPolicy: IfNotPresent
    tag: "7.10.2"
  netPolicy:
    ingressApps:
      - arranger
      - ...
      - gen3job
    egressApps:
      - arranger
      - ...
```

To GEN3 community,

I began working with Gen3 in early 2025. Over the past few months, I've received a lot of support from Gen3 community. Every question, I posted in the Gen3 community was answered promptly. I'm especially grateful for the help and guidance provided by Fay Booker, Elise Castle, Sara Volk, and Ajo Augustine through the Slack channel.

Thank You!

Any questions?

GEN3

NeSI
New Zealand eScience Infrastructure

GEN3 DATA COMMONS

Cloud Native Platform Engineering Considerations for Research and Researchers

Colin Griffin
Krumware - www.krum.io
Co-Chair of CNCF Platforms WG

What is Platform Engineering

Platform

A collection of capabilities, documentation, and tools that support developing, deploying, operating, and/or managing the delivery of products and services.

Platform Engineering

The design, construction, operation, and evolution of a platform. One way to view the practice is as an empathy-driven approach towards sociotechnical organizational design.

Platform Engineering theory and resources

Cloud Native Computing Foundation

Platform Engineering Whitepaper

https://tag-app-delivery.cncf.io/whitepapers/platforms/

Platform Engineering Maturity Model

https://tag-app-delivery.cncf.io/whitepapers/platform-eng-maturity-model/

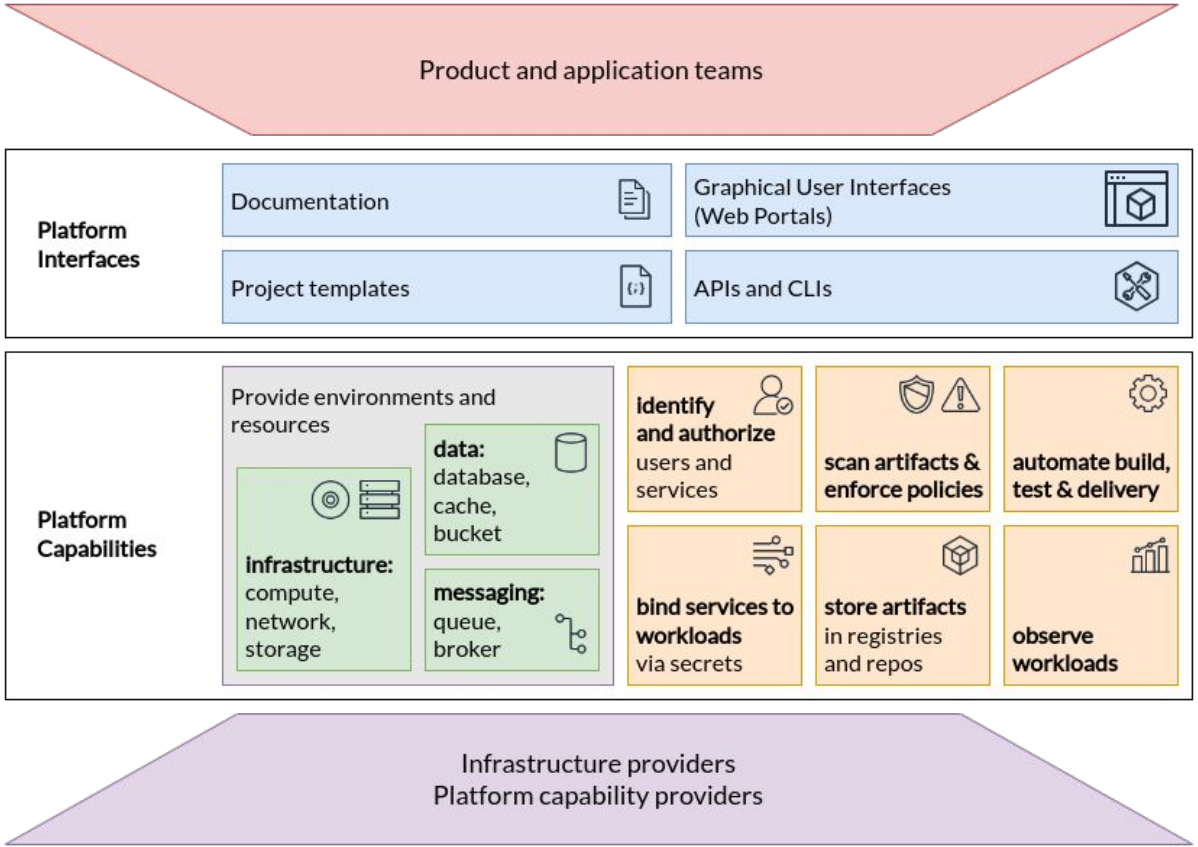
Implementation Challenges

The reality of implementation

Research teams have a significant implementation challenge ahead of them

- Diverse users means no one-size-fits-all solutions
- High compliance industry, data sharing is limited and actively restricted despite the prevailing desire to share.
- Siloed and fragmented research and invention, home-grown solutions.
- Limited IT flexibility and resources
- Risk of hype vs reality. Building data, research, and AI/ML platforms is HARD.

Platform Whitepaper



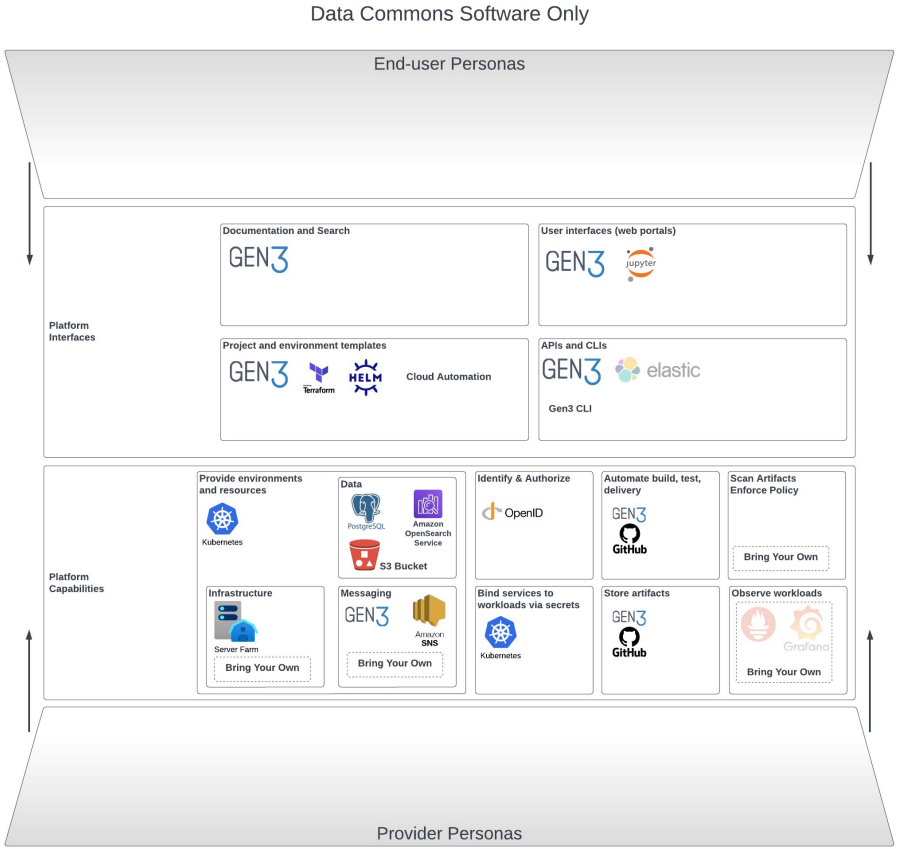
Platform Whitepaper

ASPECT		PROVISIONAL	OPERATIONAL	SCALABLE	OPTIMIZING
Investment	*How are staff and funds allocated to platform capabilities?*	Voluntary or temporary	Dedicated team	As product	Enabled ecosystem
Adoption	*Why and how do users discover and use internal platforms and platform capabilities?*	Erratic	Extrinsic push	Intrinsic pull	Participatory
Interfaces	*How do users interact with and consume platform capabilities?*	Custom processes	Standard tooling	Self-service solutions	Integrated services
Operations	*How are platforms and their capabilities planned, prioritized, developed and maintained?*	By request	Centrally tracked	Centrally enabled	Managed services
Measurement	*What is the process for gathering and incorporating feedback and learning?*	Ad hoc	Consistent collection	Insights	Quantitative and qualitative

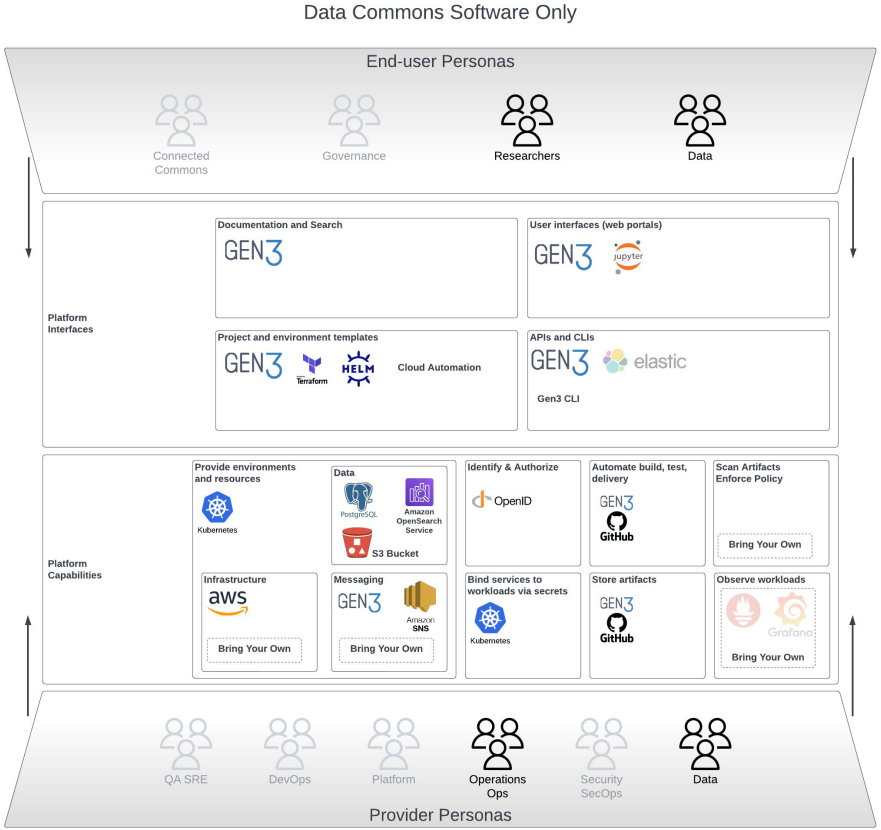
Starting a Platform Strategy

1. Identify the known Platform Components
2. Add expected users AND providers
3. Identify gaps in capabilities
4. Implement new capabilities and enable new customers
    - Bootstrapping
    - Lifecycle Management
    - Platform Integrations
    - Automation

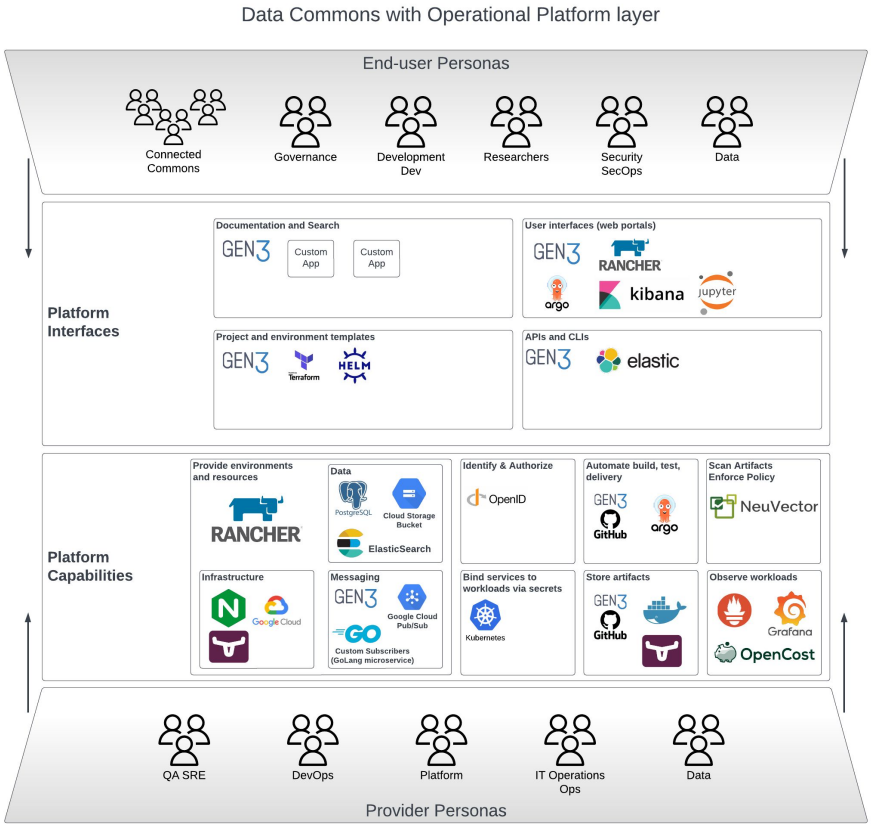
Develop a technical understanding of the platform



Add platform end-users and providers, identify gaps



Add additional capabilities and operationalize the platform



Platforming Mentality

1. Identify a starting point or source application stack
2. Decouple components and enable the stack for “replanting”
3. Layer additional platform components and improve lifecycle management
4. Enable scalable and repeatable Platform Delivery

The App Terrarium

So we have an an application and platform All-In-One, which can be great, but there are limitations and invisible barriers that prevent achieving Production.

Start with the core application

GEN3 DATA COMMONS

To be ready for a platform, next we need to loosen the coupling between the application and its dependent components, and provide Platform basics or bootstraps.

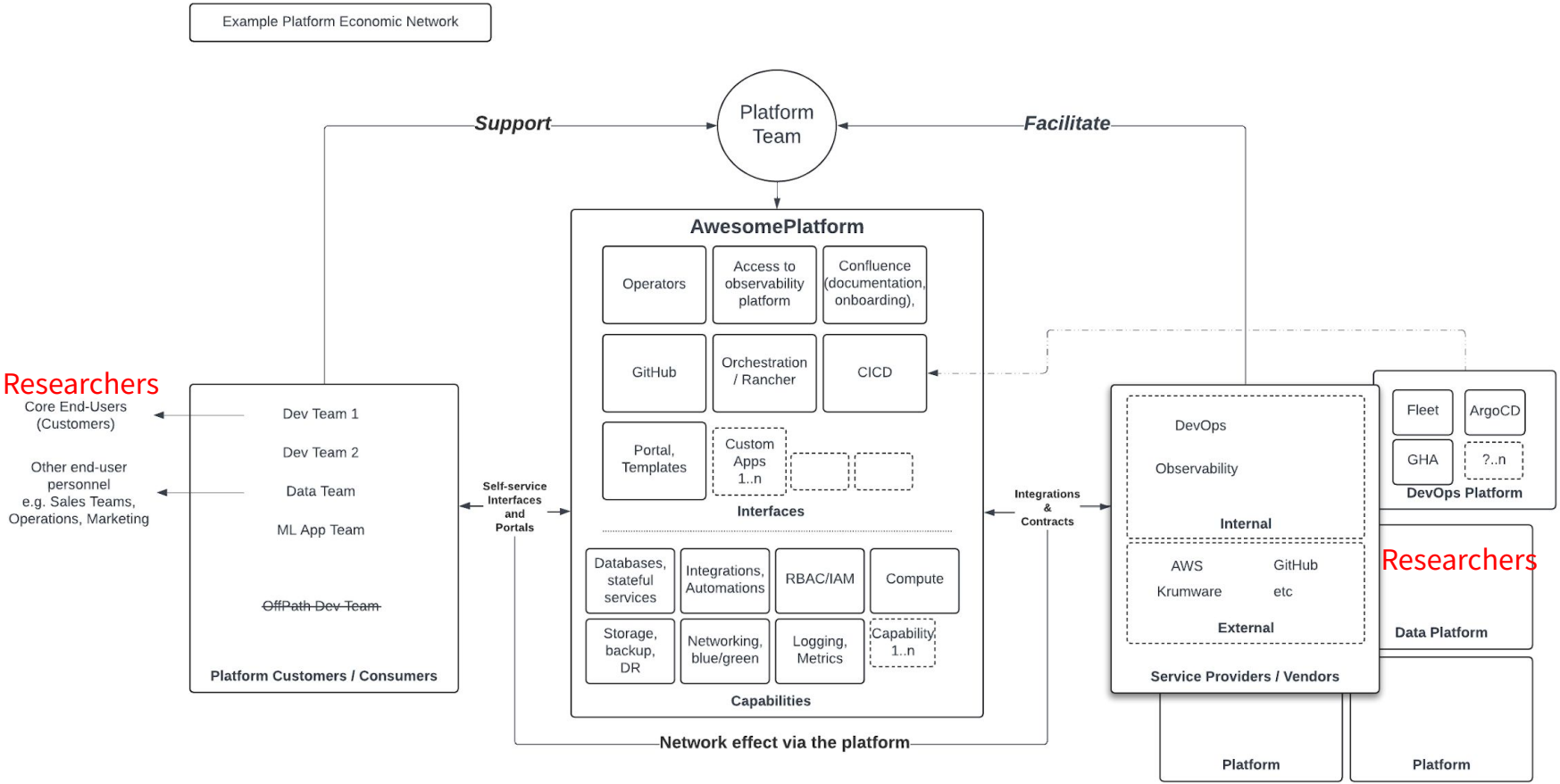
Many apps and tools, one platform to manage them

The platform allows integration of tools to support and allow researchers to focus on research, and supporting developers to deliver more tools.

Template and share the platform

Develop with a templating mentality (golden paths) for other teams, departments, or institutions and enable Platform instances, to amplify capabilities and collaboration.

Platform Economic Network



Questions

Any questions?

Deploying on-premise Gen3; constraints, plans and opportunities

National Computational Infrastructure (NCI) Australia Team

Deploying on-premise Gen3; constraints, plans and opportunities
National Computational Infrastructure

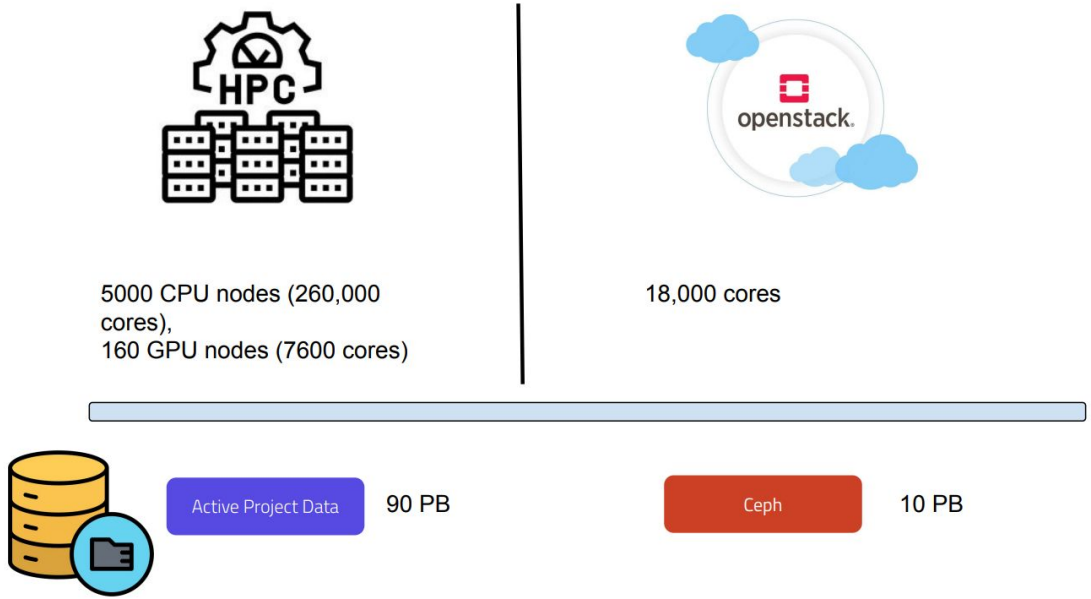
Australian National University

NCI Australia Team
David Monro, David Peters & Warren Kaplan

2 July 2025

National Computational Infrastructure (NCI) Australia

- We're NOT Australia's National Cancer Institute
- NCI is one of Australia's two Tier-1 high-performance computing facilities
- 7,500+ users, 35 universities, major science agencies & industry
- ~27 PB (1276 datasets) of earth sciences, environmental, satellite, and astronomy in 102 collections with ~1600 users



Kubernetes (David Monro)

We are just beginning our K8S journey

- Why deploy K8S on our openstack?
  - Leverage our existing infrastructure and data access
- Constraints in our openstack environment
  - No Magnum
  - No Octavia
- How we plan on making it work
  - Cinder CSI for volume storage
  - Dedicated ingres nodes

Gen3 Deployment (David Peters)

Openstack Cloud Service

- Provides Compute, Object Storage, Volume Storage, Network Services
- Does not provide Database as a Service or Elasticsearch

Openstack and Kubernetes

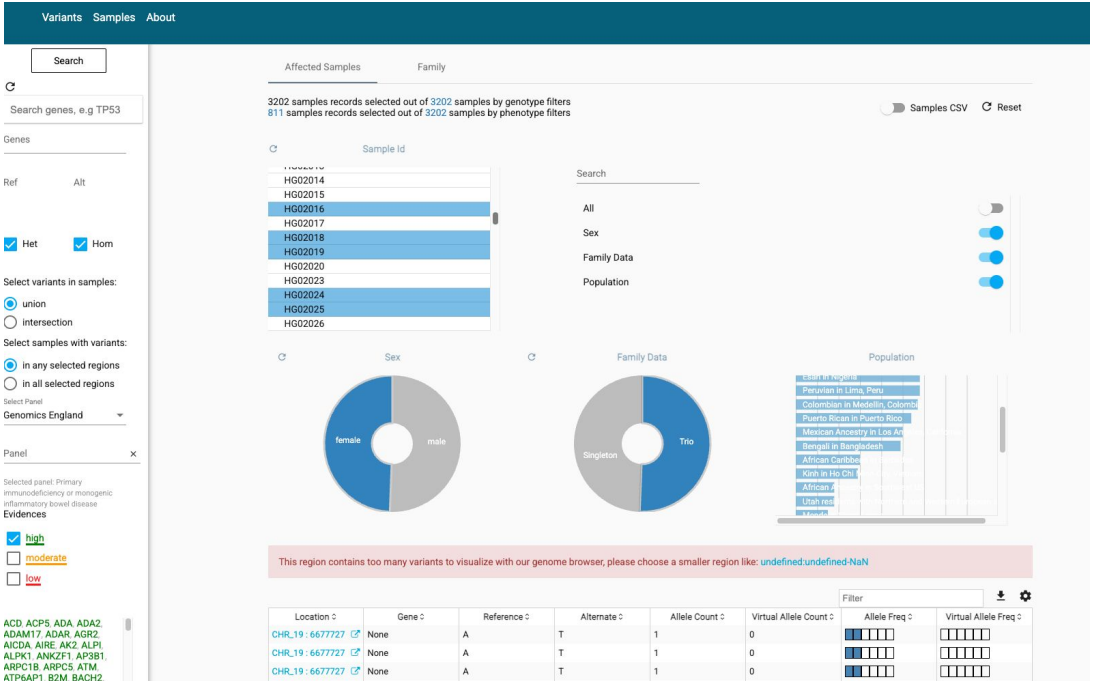
- Most Gen3 components are delivered by a Kubernetes cluster hosted on Openstack compute instances.
- Postgres and Elastic are deployed on Kubernetes during pilot phase however this may change depending on data set, storage and compute requirements.

Authentication and Authorization

- Test Build Environment is using an OIDC to LDAP bridge, ultimately external OIDC provider.
- The method fro managing user authorization has not yet been determined.

Gen3 NCI-Specific Opportunities (Warren Kaplan)

- Retrospective Genome Cohorts
- Genomics Workflows - Gen3 data model will be continuously updated with analysis-ready results.
- Many of our expected users don't code - enriched experience with low latency genomics variant store (dnaerys.org)



Gen3 single node deployment

Platform Engineering Team, Center for Translational Data Science, University of Chicago

Why Single Node Gen3?

- Local testing environment
- Useful for dev, QA, training or demos
- Reduces cost and complexity
- Light weight

Pre-requisites

- 7 CPU (if you're running portal)
- 15GB Ram
- Ample storage for Postgres, Elasticsearch AND the kubernetes pods
- SSL Certificate
  - (we'll show you how to get one)
- Docker (Or other containerization engine)
- Kubernetes Cluster (we'll show some options)
- Ingress Controller
- KubeCTL
- Helm
- K9s (nice to have)

SSL Certificate

- You need to control the DNS of the domain
- Certbot / let’s encrypt based
- `sudo certbot certonly --manual --preferred-challenges=dns -d user.dev-site.net`

- https://docs.gen3.org/gen3-resources/operator-guide/helm/helm-deploy-example/#certbot-to-generate-a-certificate

Kubernetes Options

- Kind
  - https://kind.sigs.k8s.io/docs/user/ingress/#ingress-nginx
- MiniKube
  - https://minikube.sigs.k8s.io/docs/
  - minikube start --cpus=8 --memory=15g
- k3s
  - https://k3s.io

```
cat <<EOF | kind create cluster
--config=-
kind: Cluster
apiVersion: kind.x-k8s.io/v1alpha4
nodes:
  - role: control-plane
    kubeadmConfigPatches:
      - |
        kind: InitConfiguration
        nodeRegistration:
          kubeletExtraArgs:
            node-labels:
"ingress-ready=true"
    extraPortMappings:
      - containerPort: 80
        hostPort: 80
        protocol: TCP
      - containerPort: 443
        hostPort: 443
        protocol: TCP
EOF
```

Ingress Controllers

- Routes external traffic to Gen3 services inside the cluster.
- Quick deployment of nginx-ingress
    - For minikube:
        - minikube addons enable ingress (for minikube)
        - minikube tunnel (to expose services via localhost)
    - Kind Cluster:
        - `kubectl apply -f https://kind.sigs.k8s.io/examples/ingress/deploy-ingress-nginx.yaml`

Preparing a values.yaml

```yaml
global:
  # Deploys bundles postgres/elasticsearch for dev
  dev: true
  hostname: "example.domain.com"
  tls:
    cert: |
      <cert-content>
    key: |
      <key-content>

fence:
  FENCE_CONFIG:
    # if true, will bypass OIDC login, and login a user with username "test"
    # WARNING: DO NOT ENABLE IN PRODUCTION (for testing purposes only)
    MOCK_AUTH: true
  # USER YAML. Passed in as a multiline string.
  USER_YAML: |
    <contents-of-user-yaml>
```

Documentation:
https://docs.gen3.org/gen3-resources/operator-guide/helm/helm-deploy-example/#create-a-minimal-valuesyaml

Data Persistence

GEN3 DATA COMMONS

```yaml
postgresql:
  primary:
    persistence:
      # -- (bool) Option to persist the dbs data.
      enabled: true
```

Deployment

GEN3 DATA COMMONS

```
helm repo add gen3 https://helm.gen3.org
helm repo update
helm upgrade --install gen3 gen3/gen3 -f ./values.yaml
```

Demo

GEN3 DATA COMMONS

Acknowledgements

- Speakers
  - Claire Rye, Nathalie Giraudon, and Carvin Chen; New Zealand eScience Infrastructure (NeSI)
  - Colin Griffin, Krumware
  - National Computational Infrastructure (NCI) Australia Team
  - Platform Engineering Team, Center for Translational Data Science, University of Chicago
- Gen3 Forum Steering Committee
  - Robert Grossman - Center for Translational Data Science, University of Chicago
  - Steven Manos - Australian BioCommons
  - Claire Rye - New Zealand eScience Infrastructure
  - Plamen Martinov - Open Commons Consortium
  - Michael Fitzsimons - Center for Translational Data Science, University of Chicago