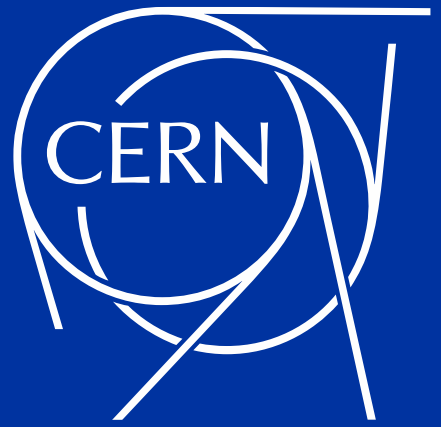


K8s and GitOps for ATLAS Rucio operation

Radu Carpa

27 April 2022

What is Rucio?

- https://rucio.cern.ch/
- **Open-source data-management software**

- Upload files to storage servers
- Group them into datasets
- Enforce replication rules
  - (ex: maintain 2 copies on 2 different continents)
- Recover from replica lost
- Etc.

About Rucio

- **Written in python**
  - Server instances exposing a REST API
  - Client for that API; also, a separate web UI
  - 20+ different daemons (many optional)
    - rule evaluation
    - transfers
    - deletion
    - …
- **Different communities install it differently and with different level of customization**
  - Directly via `pip install`
  - Containers only (provided by the Rucio core team)
  - Helm charts (also provided by the Rucio core team)

About myself

- **Rucio core developer**
  - Mostly working on transfer and deletion workflows
- **Also, in charge of the ATLAS Rucio Kubernetes installation**
  - First production experience with Kubernetes

Atlas Rucio installation

- **1 (small) integration + 3 production clusters (50 nodes in total)**
  - Required capacity: ~ ½ of that. The rest is for comfortable rolling re-installs of clusters
- **Self-managed load-balancer (haproxy 2.5.6) on puppet-managed VMs**
  - In charge of x509-based authentication
  - Migration to LBAAS planned for quite some time. Not sure about the status

Cluster bootstrap

- **Regular rolling reinstalls of the clusters**
  - So, we try to avoid any manual customization
- **Executed via terraform**
  - Initial creation
  - Basic configuration
  - Installation of some helm charts:
    - kube-prometheus-stack;
    - pushgateway;
    - our own chart (some cluster customization)

Due to my love-hate relationship with terraform, I avoid relying “too much” on it

Cluster templates

- **Customized upstream cluster templates (all clusters are now based on 1.22.3-4)**
  - Via a simple python script using the OpenStack SDK
    - configure custom eviction policies
    - disables Prometheus installation

```
kubelet_options.extend([
    '--eviction-soft-grace-period=memory.available=2m',
    '--eviction-max-pod-grace-period=15',
    '--eviction-soft=memory.available<400Mi',
    '--eviction-hard=memory.available<200Mi',
    '--system-reserved=cpu=250m,memory=500Mi,ephemeral-storage=1Gi',
])

template['name'] = template_name
template['flavor_id'] = 'm2.large'
template['master_flavor_id'] = 'm2.large'
template['public'] = False
template['labels'].update({
    'monitoring_enabled': False,  # We'll deploy our own kube-prometheus stack. The default one cannot easily be configured
    'prometheus_adapter_enabled': False,  # Same as for monitoring_enabled, we install our own adapter
    # 'tiller_enabled': False,
    'auto_scaling_enabled': True,
    'min_node_count': 8,
    'max_node_count': 16,
    'eos_enabled': False,
    'kubelet_options': ' '.join(kubelet_options),
})
```

Rucio GitOps

- **Many layers of templating engines***

  - Flux2
  - Sops (to store encrypted secrets in git)
  - Kustomize (required to use sops in flux2, but also used for rucio hot-patching)
  - Helm (managing the Rucio installation and containers)
  - Also, some Jinja2, but we try to get rid of it

  * we are afraid Leonardo DiCaprio will have to come and save us from the Template**Inception**

GitOps examples

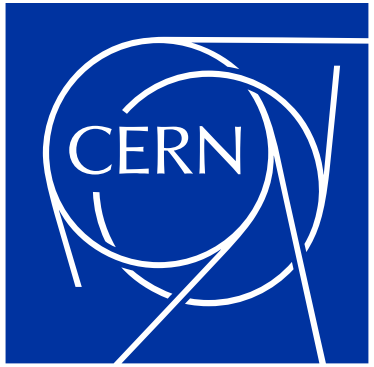
1. managing the Rucio installation
2. hot-patching Rucio
3. secret management

Un-fixed issues

- **Any way to match a PV openstack “share” to the cluster which created it without listing PVs from all clusters?**

- **Recently started hitting the udp “contrack” table limit due to internal dns requests. Any way to increase the limit automatically on cluster installation?**
  - Sysctl, but… contrack module is loaded after sysctl.d execution

- **Any way to disable some alerts in kube-prometheus-stack on installation?**



home.cern