Rucio deployment with k8s for ATLAS and CMS

pre-GDB - kubernetes many faces, 10/12/2019

Thomas Beermann
on behalf of the Rucio team

RUCIO
SCIENTIFIC DATA MANAGEMENT

Rucio

- Distributed data management system principally developed by the ATLAS experiment
- Open source community project now also adopted by CMS for Run-3 and a lot of other bigger and smaller experiments
- The current ATLAS deployment uses separate Openstack VMs for servers and daemon services and split by integration and production:
  - 15 / 2 production / integration server VMs
  - 25 / 7 production / integration daemon VMs
  - 3 haproxy load balancers
  - 2 / 1 production / integration webui servers + a couple of VMs for misc services, e.g., nagios
- Deployment is fully managed with Puppet

Limitation of current deployment (ATLAS)

- It is running stable and we have a lot of experience with the current model but:
  - Regular problems with Python dependencies that are overwritten by distrosync breaking our deployment
  - The puppet deployment grew over time and became quite complicated
  - Adapting the deployment to add or remove new daemons to adapt to different workloads requires manual intervention and is rather slow
  - Setup of a new deployment is complicated and needs a lot of support for the initial installation
  - The VM resources for the ATLAS deployment are highly underutilized because of redundancies and the static deployment model with Puppet
  - Hunting down problems can be tedious sometimes due to the distributed nature of the deployment
- Could benefit a lot of a more dynamic kubernetes deployment

Why Kubernetes

- Containers provide an isolated and minimal environment with only the necessary dependencies needed for the application
- Initial deployment of new services becomes really easy and is quick thanks to Helm charts
- Changes in the deployment and software upgrades are quickly propagated through the system
- Auto-scaling can help in case of spikes in the workload and to better utilize the available resources / better energy efficiency
- Centralized monitoring and logging can make it easier to find problems

Deployment with Helm and Flux

- The Rucio server and daemon services are fully packaged with Helm
- Available in our own repo (https://rucio.github.io/helm-charts/) but will also add it Helm Hub
- Set up of a new Rucio instance is now as simple as adapting a few configuration parameters and installing the Helm chart
- Recently started to look into Flux for automated deployment:
  - Since we have the Helm charts already available it is rather easy to set up
  - Adds accountability which is important for us since there can be multiple people trying to change the deployment
  - Changing the deployment is done by simple git commits, similar to puppet but much quicker
  - Could bridge the gap for of our ops people not having too much experience with Kubernetes, yet

Monitoring and Auto-Scaling

- Will be using Prometheus for all our monitoring needs
- In our current deployment we are using statsd/graphite integrated directly into our core services
- But we are gradually adding prometheus metrics that can then also be used for the auto-scaling
- We have a lot metrics for internal queues and server response time that would be a good fit for auto-scaling
- Spikes in transfer activity or deletion campaigns or server response times could then trigger the temporary deployment of pods

Logging

- CMS is experimenting with the cluster provided fluent setup
- For ATLAS we are currently writing the logs using Filebeat / Logstash in our own Elasticsearch instance
- We are using some Logstash filters to extract important fields from the logs helping us to create useful dashboards
- For our production deployment we want to rely on the logging infrastructure provided by the CERN MonIT team for two reasons:
  - Provides a separate private kibana endpoint just for our Kubernetes logs
  - Automatic backup of all logs to Hadoop for long term storage
- But we currently have problems with slowness of the pipeline what we still need to address

Debugging

- It can sometimes happen that a bug was missed during testing but in our current deployment it could be quickly debugged and fixed directly on the node until the hotfix release is ready
- We still need to gain more experience with the debugging and fixing of such bugs in our Kubernetes deployment
- Currently looking into ephemeral containers to attach debug tools

Plans for ATLAS

- We wanted to have a full integration service with the servers and all important daemons running by the end of the year which has been done
- Now we need to gain more experience in operating this service, finding possible problems and solution how to address them
- If everything goes to plan we want to gradually start to move our production services over by the beginning of 2020

CMS Status

- Started to use Kubernetes as soon as services moved to CERN
- Setup is completely relying on Kubernetes which includes a couple of cron jobs that still run on separate VMs for ATLAS
- CMS and ATLAS working closely together on the common Helm and Kubernetes setups

Issues in the ATLAS deployments

- We have encountered a couple issues so far that still need to be addressed before we can move to production:
  - In the early days we had problem with the networking inside the cluster which needed to service restarts on the node but more recently with the newer cluster this did not show up anymore
  - Our average server response time is much slower in Kubernetes and that is still something that needs to be addressed
  - Some of our daemon pods can put the minion node it's running on into "NotReady" state creating several problems in the cluster
  - Problems with slow logging producer

Summary

- We are on a good way to move all our Rucio deployment into Kubernetes
- It has the potential to help and automate some parts of our daily operations for ATLAS
- New installations for other experiments will be much easier
- We have smaller instances running for a long time now without major problems, e.g., DOMA TPC tests
- We have integration services running for ATLAS and CMS
- But we still need to gain more experience which will come over time as we moving more and more into Kubernetes

More information

Website	http://rucio.cern.ch
Documentation	https://rucio.readthedocs.io
Repository	https://github.com/rucio/
Images	https://hub.docker.com/r/rucio/
Online support	https://rucio.slack.com/messages/#support/
Developer contact	rucio-dev@cern.ch
Publications	https://rucio.cern.ch/publications.html
Twitter	https://twitter.com/RucioData