Curator

Release 2.0

Gagan Kumar

Oct 26, 2021

CONTENTS

Curator (/kyoorādr/) is an infrastructure consumption analysis project for OCP.

Check out the *About* section for further information, including how to *Install Curator* the project.

Note: This project is under active development.

CHAPTER

ONE

CONTENTS

1.1 About

1.1.1 Project Curator

Project Curator is an infrastructure consumption analysis project for the OpenShift Platform. Project Curator retrieves infrastructure utilization for the OpenShift Platform using koku-metrics-operator and provides users the ability to query the infrastructure utilization based on time period and namespaces. Users can generate periodic and custom reports of the infrastructure utilization and get them delivered through emails.

Project Curator also provides APIs to query the information utilization data and can be used to feed data to any infrastructure billing system or business intelligence system.

You need to have administrator access to an OpenShift v.4.5+ cluster to deploy Project Curator. Once deployed, all the authorized users and systems will be able to track the infrastructure utilization of OpenShift.

To submit an issue or a feature request, please raise an issue here.

1.1.2 Persona

With Project Curator,

- Administrators can
- know the entire OpenShift cluster utilization.
- know the utilization grouped by projects.
- know the utilization trend over time.
- End Users can
- know the usage for my projects in OpenShift.
- get weekly, daily and monthly reports.
- get alerts for usage limits.
- Analysts can
- pass the usage to an internal billing system.
- feed the usage data to BI systems.
- export the usage data to my own tools.

1.2 Release

1.2.1 Release Notes

Version curator1.0

Source code

Project Curator is an open-source project to provide infrastructure usage reporting for OpenShift (OCP) clusters. Curator is part of the Operate First initiative to include operational experience as an important contributor to software development. We welcome your comments and feedback in the #curator channel in the Operate First Slack organization or via email to curator@redhat.com.

Features included in the release:

- Easy Installation
- Automated report generation grouped by namespaces
- Daily, Weekly, and Monthly reports
- API Access
- On-demand timeframe data
- On-demand daily, weekly, and monthly data
- On-demand download of raw data collected by Koku-metrics operator
- Email service
- Custom SQL Queries
- Test suite to check code changes
- S3 is now optional
- Zipped data storage in any S3-compatible volume
- Unzipped data storage in any S3-compatible volume

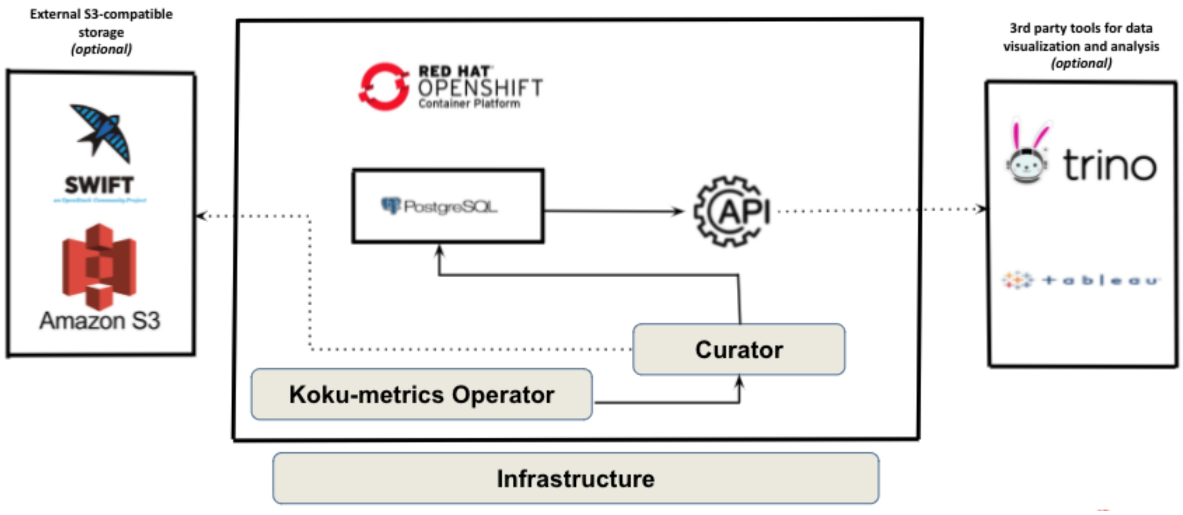
For more information and to know how to use these features, refer the *Feature List* section.

1.2.2 Known Bugs

N/A

1.3 Architecture

1.3.1 System Design



1.4 Features

1.4.1 Feature List

- Easy Installation

  Install the Curator project and the related dependencies by executing just one installation script.

- Automated report generation grouped by namespaces

  Daily, weekly and monthly reports grouped by namespaces will be generated for the infrastructure utilization in the OpenShift Cluster.

- API Access

  Users will be able to query the Curator API to get the infrastructure utilization metrics including the ability to download the raw data from the Koku-metrics operator. Read the *API* section for further information.

- Mailing Service

  Users can get daily, weekly and monthly reports delivered to their email addresses.

- Custom SQL Queries

  Users can manually log in to PostgreSQL and run their own custom SQL queries for specific infrastructure utilization information.

- Test suite to check code changes

  Developers who make changes to the project to meet new requirements can run the test suite that comes with the package to check the project sanity.

- S3 is now optional

  You can now use Curator without any S3-compatible volume. At the time of installation, you will be asked to either opt-in or opt-out of using S3 storage.

- Zipped data storage in any S3-compatible volume

  Infrastructure data in zipped CSV format can be pushed to any S3-compatible volume for archiving.

- Unzipped data storage in any S3-compatible volume

  Infrastructure data in unzipped CSV format can be pushed to any S3-compatible volume. for integration with third-party data analysis tools.

1.5 Requirements

To run Project Curator, the users are expected to meet the prerequisites listed below.

1.5.1 Prerequisite

Admin access to OpenShift 4.5+

Install Koku Metrics Operator

- You need to install koku-metrics-operator either via operator-hub or command line.
- We are currently using koku-metrics-operator version 0.9.8 with koku-metric config specified here.

Install PostgreSQL Image

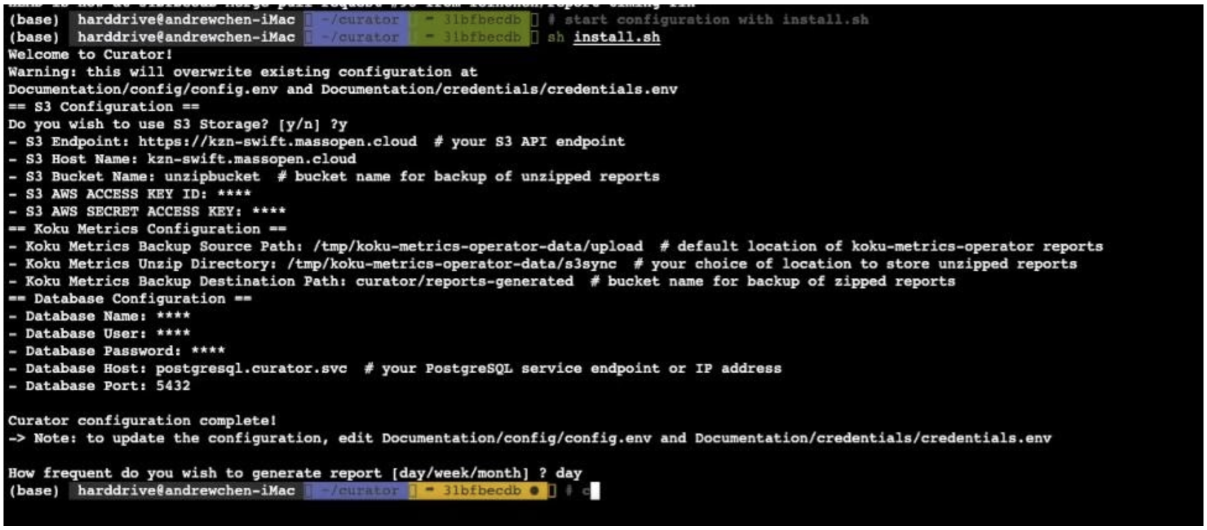
- Need to have postgres database up and running in cluster to store all information.
- We are currently using postgres version 10.X and config file specified here.

1.6 Installation

1.6.1 Install Curator

To install Curator, run `install.sh`. This will first prompt you to opt-in or opt-out of S3 back-up option and then prompt you to enter the configuration variables needed to run the project.

Here is the screenshot of how terminal should look like -



```
(base)  harddrive@andrewchen-iMac  ~/curator  - 31bfbecdb  # start configuration with install.sh
(base)  harddrive@andrewchen-iMac  ~/curator  - 31bfbecdb  sh install.sh
Welcome to Curator!
Warning: this will overwrite existing configuration at
Documentation/config/config.env and Documentation/credentials/credentials.env
== S3 Configuration ==
Do you wish to use S3 Storage? [y/n] ?y
- S3 Endpoint: https://kzn-swift.massopen.cloud  # your S3 API endpoint
- S3 Host Name: kzn-swift.massopen.cloud
- S3 Bucket Name: unzipbucket  # bucket name for backup of unzipped reports
- S3 AWS ACCESS KEY ID: ****
- S3 AWS SECRET ACCESS KEY: ****
== Koku Metrics Configuration ==
- Koku Metrics Backup Source Path: /tmp/koku-metrics-operator-data/upload  # default location of koku-metrics-operator reports
- Koku Metrics Unzip Directory: /tmp/koku-metrics-operator-data/s3sync  # your choice of location to store unzipped reports
- Koku Metrics Backup Destination Path: curator/reports-generated  # bucket name for backup of zipped reports
== Database Configuration ==
- Database Name: ****
- Database User: ****
- Database Password: ****
- Database Host: postgresql.curator.svc  # your PostgreSQL service endpoint or IP address
- Database Port: 5432

Curator configuration complete!
-> Note: to update the configuration, edit Documentation/config/config.env and Documentation/credentials/credentials.env

How frequent do you wish to generate report [day/week/month] ? day
(base)  harddrive@andrewchen-iMac  ~/curator  - 31bfbecdb ●  # c
```

Note: At any time you can change the configuration and credentials by editing `Documentation/config/config.env` and `Documentation/credentials/credentials.env`

If you change the value for variable `HAS_S3_ACCESS` later, collected files will be either pushed or not depending on the value for variable. You might lose some earlier data.

Optionally, you may set:

- MC_GLOBAL_FLAGS – flags passed to all invocations of the mc command
- MC_MIRROR_FLAGS – flags passed only to the mc mirror command

1.6.2 Deploy Curator

Run `oc apply -k .` to deploy this application into the namespace defined by the `namespace:` setting in `kustomization.yaml`.

If you have Kustomize installed separately , run: `kustomize build | oc apply -f`

1.6.3 Delete Curator

To delete the application from OpenShift, run: `kustomize build | oc delete -f-`

1.7 API

API Deployment

Deploy the API to Openshift

1. Install CRD

   This part will be generated using kubebuilder, which requires Go v1.16+.

   To use a prebuilt image, follow the below steps:

   ```
   cd apis/report
   make install
   make deploy IMG=quay.io/operate-first/curator-crd
   cd ../..
   ```

   If you would like to build CRD from scratch or you made changes to the apis/report source code, then follow the steps:

   ```
   cd apis/report
   make install
   make docker-build docker-push IMG=<some-registry>/<project-name>:tag
   make deploy IMG=<some-registry>/<project-name>:tag
   cd ../..
   ```

2. Deploy the HTTP API

```
kustomize build apis | oc apply -f-
```

3.1 Download raw Koku-Metric-Operator reports for given time frame

```
oc port-forward $(oc get pods -l=app=curator-api -o name) 5000:5000
curl -XGET "http://localhost:5000/download?start=2021-08-22%2003:00:00.
↪000000&end=2021-08-22%2004:00:00.000000"
```

start and end parameters will be cast to PostgreSQL timestamp. Therefore multiple formats are supported.

Downloaded report follows similar structure of a Koku-Metrics-Operator uploaded report.

```
<start>-<end>-koku-metrics.tar.gz
├── <start>-<end>-koku-metrics.0.csv
├── <start>-<end>-koku-metrics.1.csv
├── <start>-<end>-koku-metrics.2.csv
└── <start>-<end>-koku-metrics.3.csv
```

3.2 Create a sample `Report` by defining the parameters

- reportingEnd: Required, RFC 3339 Datetime.
- reportingStart: Optional, RFC 3339 Datetime.
- reportPeriod: Optional, String. One of Day, Week, Month. Report period N = 1, 7, 30 days.
- namespace: Optional, String. Show report for namespace only. If omitted, show report for all namespace.

Method 1: Time frame report

Provide parameter for both `reportingStart` and `reportingEnd`. (`reportPeriod` will be ignored if provided)

Result report contains raw CPU and memory metrics for time frame [`reportingStart`, `reportingEnd`) in project `namespace` (if provided).

```
# apis/report/config/samples/batch_v1_report.yaml
apiVersion: batch.curator.openshift.io/v1
kind: Report
namespace: report-system
metadata:
  name: timeframe-report-sample
spec:
  reportingStart: "2021-10-25T00:00:00Z"
  reportingEnd: "2021-10-26T00:00:00Z"  # prevents Reports targeting␣
↪future time
  namespace: koku-metrics-operator
```

Method 2: Standard daily, weekly, monthly report

Provide parameter for both `reportPeriod` and `reportingEnd`.

Result report contains raw CPU and memory metrics for the past N days until reportingEnd (including reportingEnd) in project `namespace` (if provided).

```
# apis/report/config/samples/batch_v1_report.yaml
apiVersion: batch.curator.openshift.io/v1
kind: Report
namespace: report-system
metadata:
  name: daily-report-sample
spec:
  reportingEnd: "2021-08-26T00:00:00Z"  # prevents Reports targeting␣
↪future time
  reportPeriod: Day
  namespace: koku-metrics-operator
```

Create one of the two Reports above you just defined:

```
oc project report-system
# Using project "report-system" on server ...
oc apply -f apis/report/config/samples/batch_v1_report.yaml
```

Access the Report by identifying Report by name and namespace it was created. For example, to access `daily-report-sample` on namespace `report-system`:

```
oc port-forward $(oc get pods -l=app=curator-api -o name) 5000:5000
curl -XGET "http://localhost:5000/report?reportName=daily-report-
↪sample&reportNamespace=report-system"
```

1.8 Testing

Before deploying the application, you can run `verify_connection.sh` to test your S3 bucket and PostgreSQL database connectivity.

To run database/S3 connectivity check: (OC cluster access required)

```
sh verify_connection.sh
```

To run test case with provided example test data: (Testing at local python environments. Some package required)

```
cd testing; python -m unittest curator-test.py
```