Improving latency and scalability of the user runtime job log collecting and exposure in REANA

IRIS-HEP project (link)
Jelizaveta Lemeševa (Vilnius University)
Mentor: Dr. Tibor Šimko (CERN)

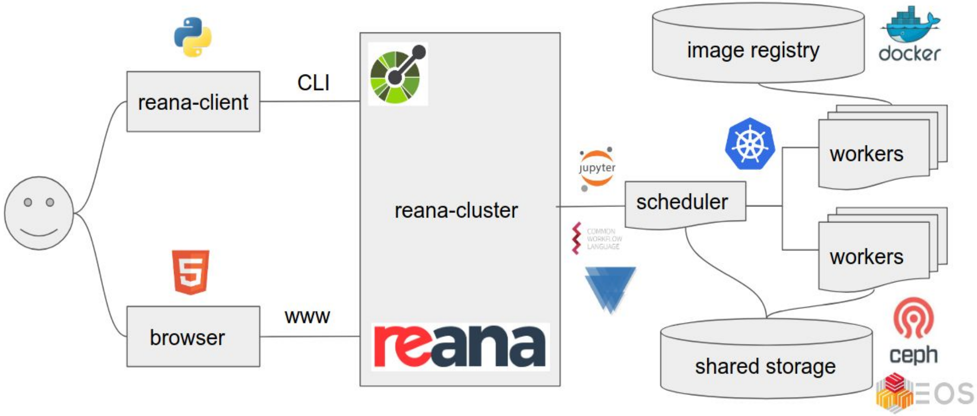
REANA reproducible analysis platform

REANA is a REproducible ANAlysis platform, created for the purpose of reusing and reinterpreting of research data analyses.

The researcher writes his data analysis workflow and submits their analysis pipelines to REANA cluster.

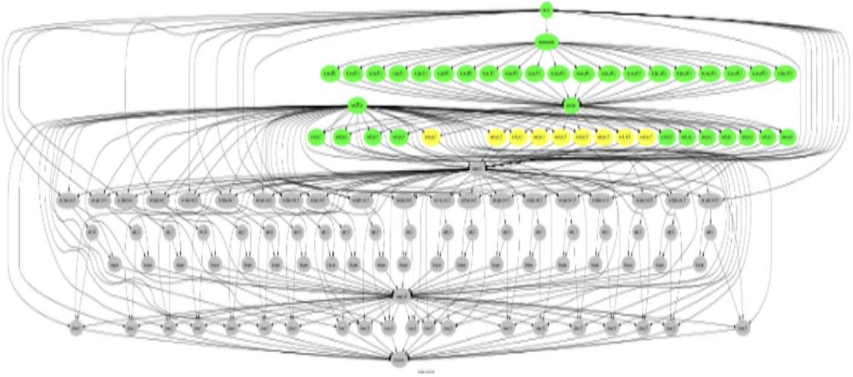
REANA cluster schedules the workflow execution on a computing backend, monitors it, persists the results, and displays them to the user.

REANA keeps track of all workflow runs, assisting in the organisation of analysis development.



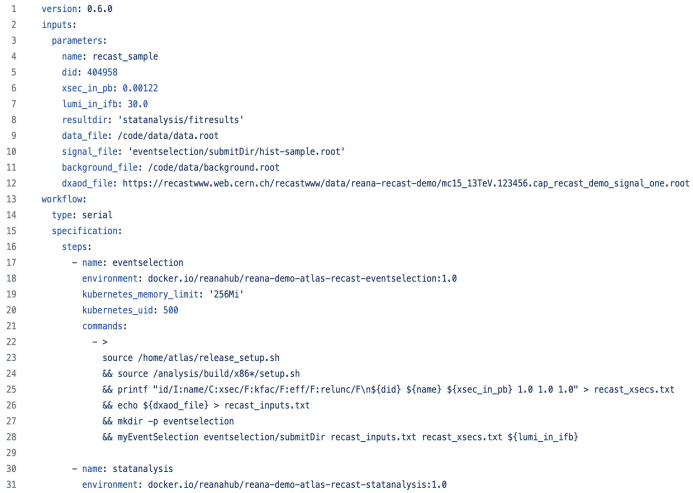
https://docs.reana.io/

Workflow YAML definition



Workflows are defined in YAML files, where it is possible to specify:

- The type of the workflow engine: CWL, Serial, Snakemake or Yadage
- One or multiple steps, which can run sequentially or in parallel
- An environments in which to run the code (Docker images)
- The command to execute in each step
- Additional files: scripts with code, data input and output files
- Custom parameters
- Resource limits for computations (memory)
- etc.



```yaml
version: 0.6.0
inputs:
  parameters:
    name: recast_sample
    did: 404958
    xsec_in_pb: 0.00122
    lumi_in_ifb: 30.0
    resultdir: 'statanalysis/fitresults'
    data_file: /code/data/data.root
    signal_file: 'eventselection/submitDir/hist-sample.root'
    background_file: /code/data/background.root
    dxaod_file: https://recastwww.web.cern.ch/recastwww/data/reana-recast-demo/mc15_13TeV.123456.cap_recast_demo_signal_one.root
workflow:
  type: serial
  specification:
    steps:
      - name: eventselection
        environment: docker.io/reanahub/reana-demo-atlas-recast-eventselection:1.0
        kubernetes_memory_limit: '256Mi'
        kubernetes_uid: 500
        commands:
          - >
            source /home/atlas/release_setup.sh
            && source /analysis/build/x86*/setup.sh
            && printf "id/I:name/C:xsec/F:kfac/F:eff/F:relunc/F\n${did} ${name} ${xsec_in_pb} 1.0 1.0 1.0" > recast_xsecs.txt
            && echo ${dxaod_file} > recast_inputs.txt
            && mkdir -p eventselection
            && myEventSelection eventselection/submitDir recast_inputs.txt recast_xsecs.txt ${lumi_in_ifb}

      - name: statanalysis
        environment: docker.io/reanahub/reana-demo-atlas-recast-statanalysis:1.0
```

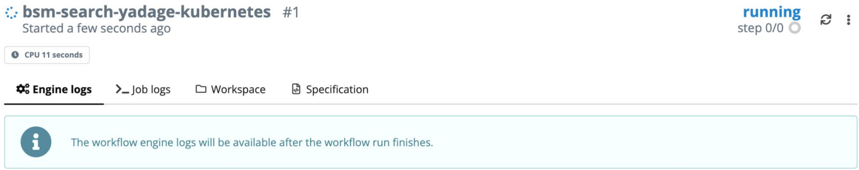
REANA Web UI and CLI

After the workflow submission, the researcher can go to REANA Web UI and inspect workflow execution progress (or alternatively use `reana-client` CLI tool).

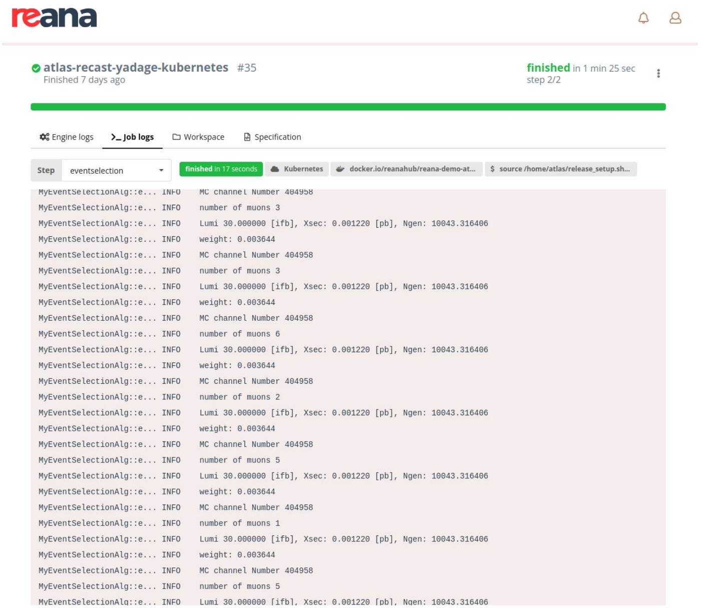
Workflow and job status is reported in real time, but workflow and job logs can only be inspected after execution of each concrete step.

REANA cluster is notified of workflow execution completion by compute backend, and retrieves the logs of the completed job.

This works well for short-running jobs, but some jobs can run for hours and even days, and the user has no feedback of whether the code is executing correctly.



The workflow engine logs will be available after the workflow run finishes.



```
MyEventSelectionAlg::e... INFO    MC channel Number 404958
MyEventSelectionAlg::e... INFO    number of muons 3
MyEventSelectionAlg::e... INFO    Lumi 30.000000 [ifb], Xsec: 0.001220 [pb], Ngen: 10043.316406
MyEventSelectionAlg::e... INFO    weight: 0.003644
MyEventSelectionAlg::e... INFO    MC channel Number 404958
MyEventSelectionAlg::e... INFO    number of muons 3
MyEventSelectionAlg::e... INFO    Lumi 30.000000 [ifb], Xsec: 0.001220 [pb], Ngen: 10043.316406
MyEventSelectionAlg::e... INFO    weight: 0.003644
MyEventSelectionAlg::e... INFO    MC channel Number 404958
MyEventSelectionAlg::e... INFO    number of muons 6
MyEventSelectionAlg::e... INFO    Lumi 30.000000 [ifb], Xsec: 0.001220 [pb], Ngen: 10043.316406
MyEventSelectionAlg::e... INFO    weight: 0.003644
MyEventSelectionAlg::e... INFO    MC channel Number 404958
MyEventSelectionAlg::e... INFO    number of muons 2
MyEventSelectionAlg::e... INFO    Lumi 30.000000 [ifb], Xsec: 0.001220 [pb], Ngen: 10043.316406
MyEventSelectionAlg::e... INFO    weight: 0.003644
MyEventSelectionAlg::e... INFO    MC channel Number 404958
MyEventSelectionAlg::e... INFO    number of muons 5
MyEventSelectionAlg::e... INFO    Lumi 30.000000 [ifb], Xsec: 0.001220 [pb], Ngen: 10043.316406
MyEventSelectionAlg::e... INFO    weight: 0.003644
MyEventSelectionAlg::e... INFO    MC channel Number 404958
MyEventSelectionAlg::e... INFO    number of muons 1
MyEventSelectionAlg::e... INFO    Lumi 30.000000 [ifb], Xsec: 0.001220 [pb], Ngen: 10043.316406
MyEventSelectionAlg::e... INFO    weight: 0.003644
MyEventSelectionAlg::e... INFO    MC channel Number 404958
MyEventSelectionAlg::e... INFO    number of muons 5
MyEventSelectionAlg::e... INFO    Lumi 30.000000 [ifb], Xsec: 0.001220 [pb], Ngen: 10043.316406
```

Project goal

The goal is to enhance the REANA workflow and job logging system with a possibility to capture logs of executing processes “live”.

Considerations:

- Introduce least complexity
- Compatible with Dask on Kubernetes
- Works for both workflow and job logs
- Evaluate reliability and scalability

Log collection alternatives

There are three common alternatives for log collection in Kubernetes:

1. DaemonSet
2. Sidecar
3. Kubernetes API.

Additionally there are multiple log storage options and logging agent options.

The alternatives can be improved further by adding live log result caching to decrease the load on underlying systems in case of numerous user queries, or by redesigning REST API to retrieve only one log at a time instead of logs for the workflow and all its jobs.

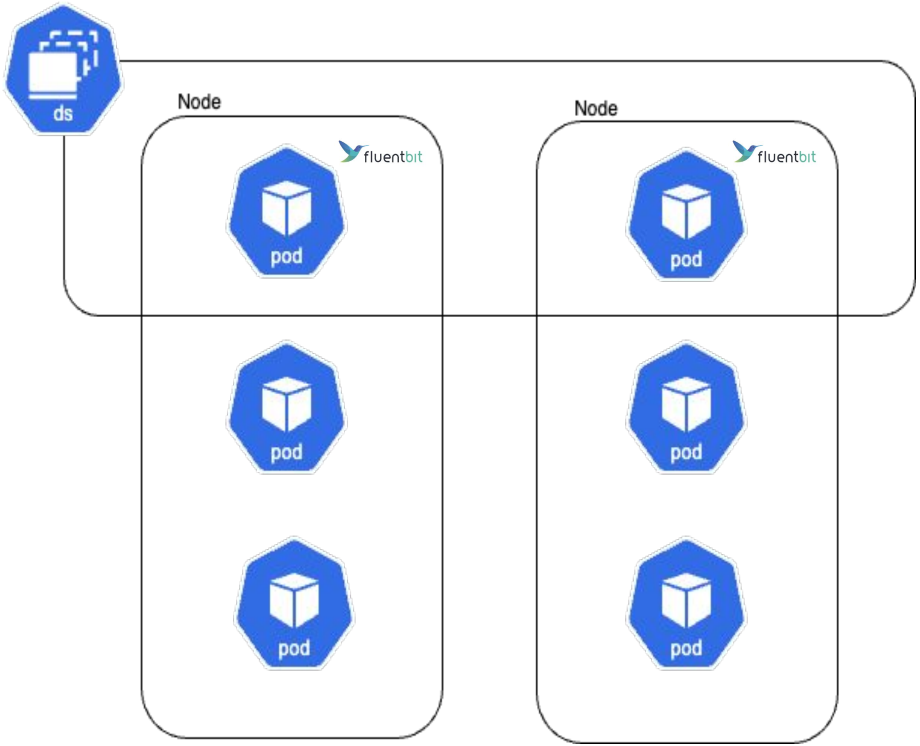
Prototypes for various combinations of setups with different storage and logging agents can be found in jlemesh/reana-demo-logs and REANA forks in jlemesh namespace.

Alternative 1: DaemonSet

A DaemonSet is a set of pods, where at least one pod runs on each Kubernetes node.

A log collection agent collects logs from Kubernetes host container log files and pushes the data to a storage.

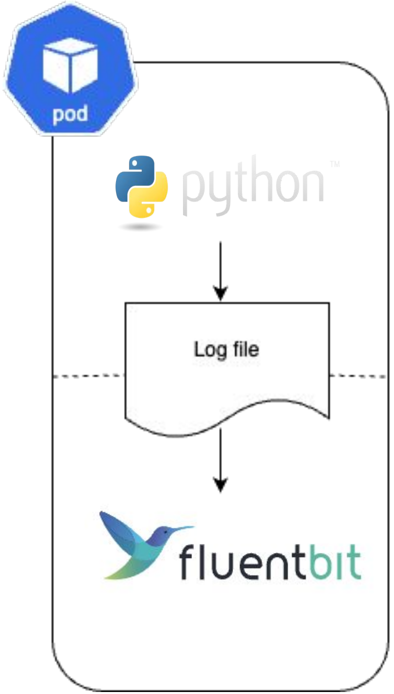
This is the most common technique used for logs collection in Kubernetes.



Alternative 2: Sidecar

A sidecar is a container in a pod that runs alongside some other (main) container and shares network and filesystem resources with it.

Container process writes logs to a file (not stdout as in DaemonSet case) and each pod has a sidecar container that runs a log collection agent process, which reads logs from that file and pushes them to a storage.

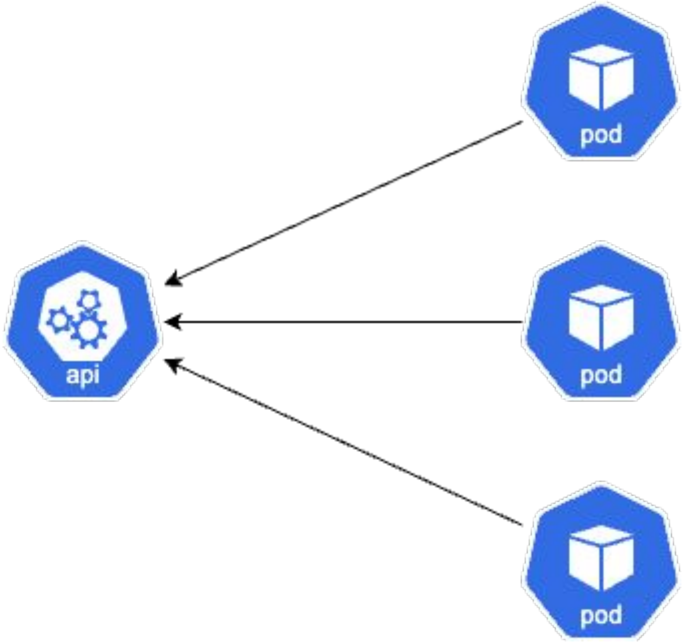


Alternative 3: Kubernetes API

The Kubernetes API server is a front-end to Kubernetes system, and exposes REST HTTP API.

It is possible to collect logs directly from Kubernetes API by making a request to logs endpoint. The logs for the pod are available only while the pod is running, and disappear afterwards, hence should be collected by custom service and pushed to permanent storage.

This is a simplest solution, but risks impacting whole cluster in case the load on Kubernetes API gets too big.



Storage alternatives

PostgreSQL is current REANA database, hence would not require to introduce additional components.

Elasticsearch is a go to option for logs storage.

OpenSearch is an open source fork of Elasticsearch. It is a bit less popular than Elasticsearch, but is recommended at CERN due to its open source licensing.

OpenSearch selected for further tests with DaemonSet and Sidecar setups, and PostgreSQL for Kubernetes API option.

Logging agent alternatives

Fluentd is currently used at CERN to collect logs for Kubernetes clusters, but is slow and there are plans to deprecate it in favor of FluentBit, which is more efficient.

Logstash is commonly used together with Elasticsearch.

Vector is another popular choice, though has complex configuration and incomplete documentation.

FluentBit was selected for further tests with DaemonSet and Sidecar setups.

VECTOR
BY DATADOG

fluentbit

fluentd

Logstash

Log retrieval benchmarking

REANA-TEST Kubernetes cluster was created with 43 nodes (each having 8 VCPU cores and 16 GB of RAM).

We have run benchmarks with 60 parallel workflows, each with 50 parallel steps - 3000 parallel jobs, each job emitting 1000 lines of logs.

Benchmarking flow:

- 60 workflows are submitted to REANA
- the script waits until the pod count in the cluster reaches threshold (60 x 50 = 3000 pods)
- a warmup benchmark is started, which usually returns the slowest results
- actual benchmarks follow (with 1, 2, 5 and 10 requests per second), each lasting 60 seconds, whilst the 3000 jobs are still running

Tool used for benchmarking: tsenart/vegeta.

Benchmarking script and results can be found in jlemesh/reana-demo-logs repository.

Benchmarking results (1 and 2 requests per second)

	1 rps (ms)			**2 rps (ms)**		
	Mean	**p50**	**p95**	**Mean**	**p50**	**p95**
DaemonSet	1132	1096	1360	1349	1332	1640
Sidecar	1054	1053	1209	1191	1172	1375
Kubernetes API	1823	1838	1905	1929	1927	2009
DaemonSet cached	490	145	1219	371	130	1374
Sidecar cached	493	144	1370	368	141	1252
Kubernetes API cached	720	155	1888	501	129	1997
DaemonSet optimized	72	69	133	66	65	89
Sidecar optimized	74	68	127	66	64	89
Kubernetes API optimized	74	68	112	66	64	92

Fastest “plain” setup

Slowest “plain” setup

Caching improves p50, but not p95

Benchmarking results (5 and 10 requests per second)

	5 rps (ms)			10 rps (ms)		
	Mean	**p50**	**p95**	**Mean**	**p50**	**p95**
DaemonSet	2299	1994	3485	21738	27124	30001
Sidecar	1962	1845	2692	17418	17117	30001
Kubernetes API	2457	2303	3207	18411	19062	30001
DaemonSet cached	314	130	1546	425	138	1870
Sidecar cached	316	132	1557	382	141	1580
Kubernetes API cached	441	132	2079	515	137	2355
DaemonSet optimized	55	55	77	55	53	75
Sidecar optimized	55	54	78	58	53	77
Kubernetes API optimized	55	55	85	54	53	79

Requests start timing out at some point

With caching and optimization all setups perform nearly equally well for 1 to 10 rps

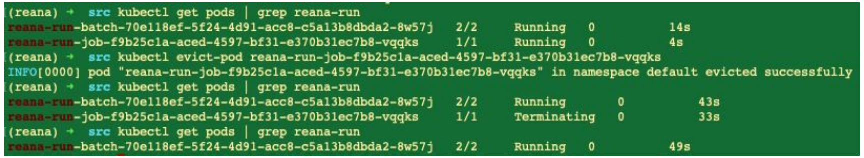
Reliability tests

Tested in a cluster with 3 worker nodes (+1 control plane node).

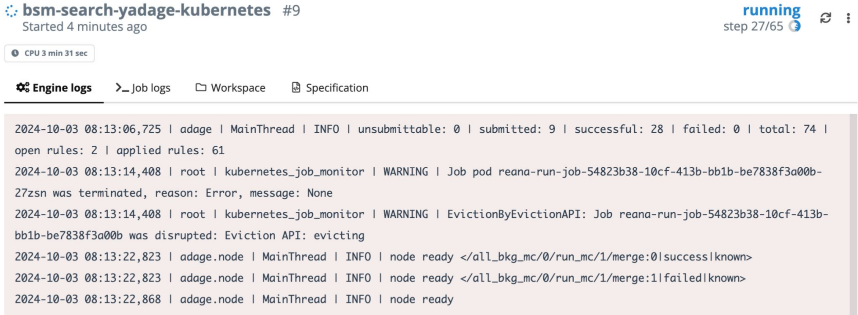
DaemonSet seems to store logs in a most reliable way. In all tested cases except workflow node crash the system collected either job or workflow logs that contain warning/error messages.

Sidecar is the least reliable in terms of logs storage. In multiple cases warnings/errors logs were not collected at all. This is due to FluentBit container being in the same pod as workflow/job, so if the pod crashes, FluentBit is also not working.

The Kubernetes API way of live log retrieval is also quite reliable and collects errors/warnings for the same test cases as DaemonSet, but there is one case when job logs are lost altogether - when workflow node crashes. In this case workflow does not live as long as job pod and has no opportunity to persist job logs in the database.

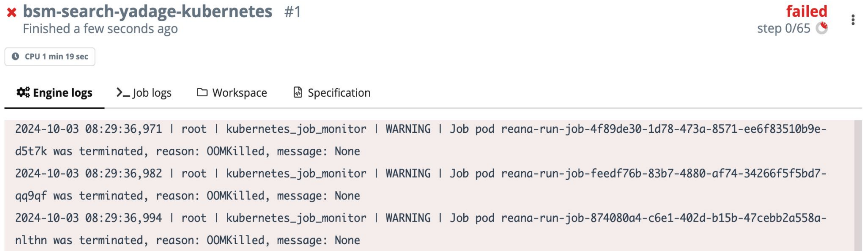


```
(reana) ➜  src kubectl get pods | grep reana-run
reana-run-batch-70e118ef-5f24-4d91-acc8-c5a13b8dbda2-8w57j   2/2     Running   0          14s
reana-run-job-f9b25c1a-aced-4597-bf31-e370b31ec7b8-vqqks     1/1     Running   0          4s
(reana) ➜  src kubectl evict-pod reana-run-job-f9b25c1a-aced-4597-bf31-e370b31ec7b8-vqqks
INFO[0000] pod "reana-run-job-f9b25c1a-aced-4597-bf31-e370b31ec7b8-vqqks" in namespace default evicted successfully
(reana) ➜  src kubectl get pods | grep reana-run
reana-run-batch-70e118ef-5f24-4d91-acc8-c5a13b8dbda2-8w57j   2/2     Running       0          43s
reana-run-job-f9b25c1a-aced-4597-bf31-e370b31ec7b8-vqqks     1/1     Terminating   0          33s
(reana) ➜  src kubectl get pods | grep reana-run
reana-run-batch-70e118ef-5f24-4d91-acc8-c5a13b8dbda2-8w57j   2/2     Running   0          49s
```



bsm-search-yadage-kubernetes #9
Started 4 minutes ago
running
step 27/65
CPU 3 min 31 sec

```
2024-10-03 08:13:06,725 | adage | MainThread | INFO | unsubmittable: 0 | submitted: 9 | successful: 28 | failed: 0 | total: 74 | open rules: 2 | applied rules: 61
2024-10-03 08:13:14,408 | root | kubernetes_job_monitor | WARNING | Job pod reana-run-job-54823b38-10cf-413b-bb1b-be7838f3a00b-27zsn was terminated, reason: Error, message: None
2024-10-03 08:13:14,408 | root | kubernetes_job_monitor | WARNING | EvictionByEvictionAPI: Job reana-run-job-54823b38-10cf-413b-bb1b-be7838f3a00b was disrupted: Eviction API: evicting
2024-10-03 08:13:22,823 | adage.node | MainThread | INFO | node ready </all_bkg_mc/0/run_mc/1/merge:0|success|known>
2024-10-03 08:13:22,823 | adage.node | MainThread | INFO | node ready </all_bkg_mc/0/run_mc/1/merge:1|failed|known>
2024-10-03 08:13:22,868 | adage.node | MainThread | INFO | node ready
```



bsm-search-yadage-kubernetes #1
Finished a few seconds ago
failed
step 0/65
CPU 1 min 19 sec

```
2024-10-03 08:29:36,971 | root | kubernetes_job_monitor | WARNING | Job pod reana-run-job-4f89de30-1d78-473a-8571-ee6f83510b9e-d5t7k was terminated, reason: OOMKilled, message: None
2024-10-03 08:29:36,982 | root | kubernetes_job_monitor | WARNING | Job pod reana-run-job-feedf76b-83b7-4880-af74-34266f5f5bd7-qq9qf was terminated, reason: OOMKilled, message: None
2024-10-03 08:29:36,994 | root | kubernetes_job_monitor | WARNING | Job pod reana-run-job-874080a4-c6e1-402d-b15b-47cebb2a558a-nlthn was terminated, reason: OOMKilled, message: None
```

Reliability tests results

Test name	DaemonSet		Sidecar		Kubernetes API	
	Job log	Workflow log	Job log	Workflow log	Job log	Workflow log
Job eviction	-	+	-	+	+	-
Workflow eviction	-	+	-	-	-	+
Job OOM	-	+	-	+	+	-
Job exception	+	+	+	+	+	-
Job node crash	-	+	-	+	+	-
Workflow node crash	-	-	-	-	—	-
Job node drain	-	+	-	+	+	-
Workflow node drain	-	+	-	-	-	+
Job node pressure eviction	-	+	-	+	+	-

The most reliable

Missing errors in logs

Job logs completely lost

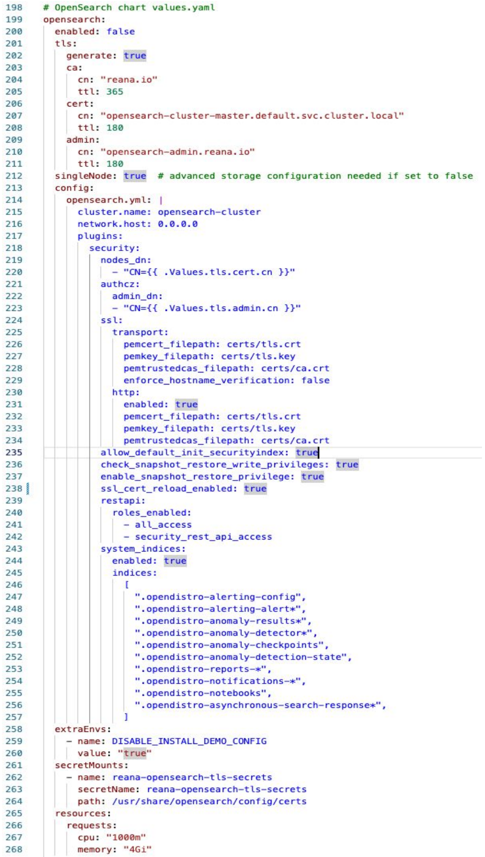
Solution implementation

- Helm chart
- REANA cluster components:
  - reana-job-controller
  - reana-workflow-controller
- REANA clients:
  - reana-client (Python)
  - reana-client (Go)
- Documentation

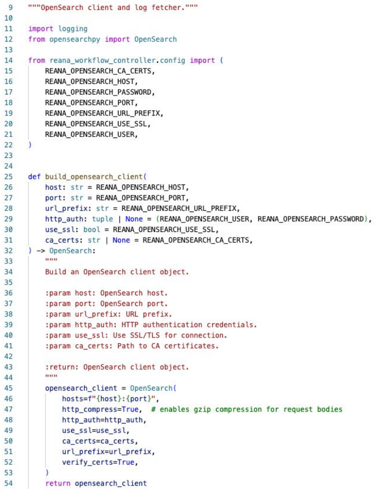
ds

OpenSearch

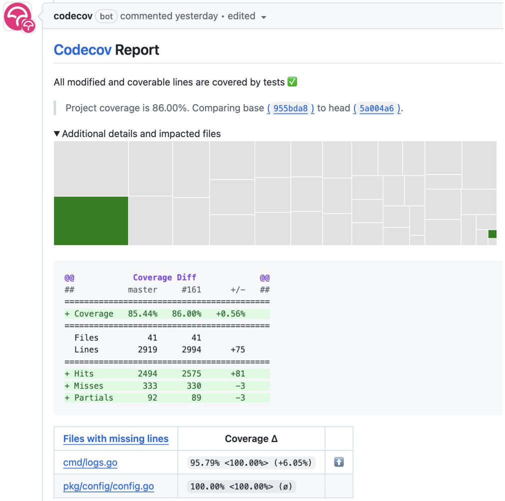
fluentbit



```yaml
# OpenSearch chart values.yaml
opensearch:
  enabled: false
  tls:
    generate: true
    ca:
      cn: "reana.io"
      ttl: 365
    cert:
      cn: "opensearch-cluster-master.default.svc.cluster.local"
      ttl: 180
    admin:
      cn: "opensearch-admin.reana.io"
      ttl: 180
  singleNode: true  # advanced storage configuration needed if set to false
  config:
    opensearch.yml: |
      cluster.name: opensearch-cluster
      network.host: 0.0.0.0
      plugins:
        security:
          nodes_dn:
            - "CN={{ .Values.tls.cert.cn }}"
          authcz:
            admin_dn:
            - "CN={{ .Values.tls.admin.cn }}"
          ssl:
            transport:
              pemcert_filepath: certs/tls.crt
              pemkey_filepath: certs/tls.key
              pemtrustedcas_filepath: certs/ca.crt
              enforce_hostname_verification: false
            http:
              enabled: true
              pemcert_filepath: certs/tls.crt
              pemkey_filepath: certs/tls.key
              pemtrustedcas_filepath: certs/ca.crt
          allow_default_init_securityindex: true
          check_snapshot_restore_write_privileges: true
          enable_snapshot_restore_privilege: true
          ssl_cert_reload_enabled: true
          restapi:
            roles_enabled:
              - all_access
              - security_rest_api_access
          system_indices:
            enabled: true
            indices:
              [
                ".opendistro-alerting-config",
                ".opendistro-alerting-alert*",
                ".opendistro-anomaly-results*",
                ".opendistro-anomaly-detector*",
                ".opendistro-anomaly-checkpoints",
                ".opendistro-anomaly-detection-state",
                ".opendistro-reports-*",
                ".opendistro-notifications-*",
                ".opendistro-notebooks",
                ".opendistro-asynchronous-search-response*",
              ]
  extraEnvs:
    - name: DISABLE_INSTALL_DEMO_CONFIG
      value: "true"
  secretMounts:
    - name: reana-opensearch-tls-secrets
      secretName: reana-opensearch-tls-secrets
      path: /usr/share/opensearch/config/certs
  resources:
    requests:
      cpu: "1000m"
      memory: "4Gi"
```



```python
"""OpenSearch client and log fetcher."""

import logging
from opensearchpy import OpenSearch

from reana_workflow_controller.config import (
    REANA_OPENSEARCH_CA_CERTS,
    REANA_OPENSEARCH_HOST,
    REANA_OPENSEARCH_PASSWORD,
    REANA_OPENSEARCH_PORT,
    REANA_OPENSEARCH_URL_PREFIX,
    REANA_OPENSEARCH_USE_SSL,
    REANA_OPENSEARCH_USER,
)


def build_opensearch_client(
    host: str = REANA_OPENSEARCH_HOST,
    port: str = REANA_OPENSEARCH_PORT,
    url_prefix: str = REANA_OPENSEARCH_URL_PREFIX,
    http_auth: tuple | None = (REANA_OPENSEARCH_USER, REANA_OPENSEARCH_PASSWORD),
    use_ssl: bool = REANA_OPENSEARCH_USE_SSL,
    ca_certs: str | None = REANA_OPENSEARCH_CA_CERTS,
) -> OpenSearch:
    """
    Build an OpenSearch client object.

    :param host: OpenSearch host.
    :param port: OpenSearch port.
    :param url_prefix: URL prefix.
    :param http_auth: HTTP authentication credentials.
    :param use_ssl: Use SSL/TLS for connection.
    :param ca_certs: Path to CA certificates.

    :return: OpenSearch client object.
    """
    opensearch_client = OpenSearch(
        hosts=f"{host}:{port}",
        http_compress=True,  # enables gzip compression for request bodies
        http_auth=http_auth,
        use_ssl=use_ssl,
        ca_certs=ca_certs,
        url_prefix=url_prefix,
        verify_certs=True,
    )
    return opensearch_client
```



codecov bot commented yesterday • edited

Codecov Report

All modified and coverable lines are covered by tests ✅

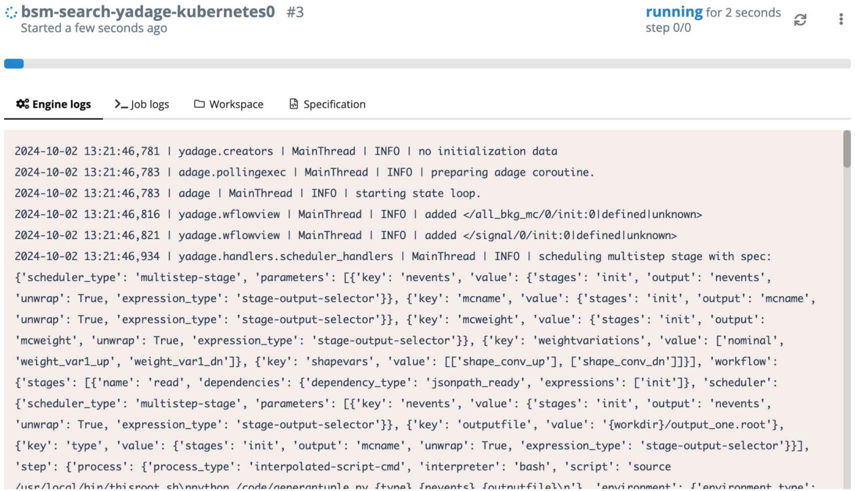
> Project coverage is 86.00%. Comparing base (955bda8) to head (5a004a6).

▼ Additional details and impacted files

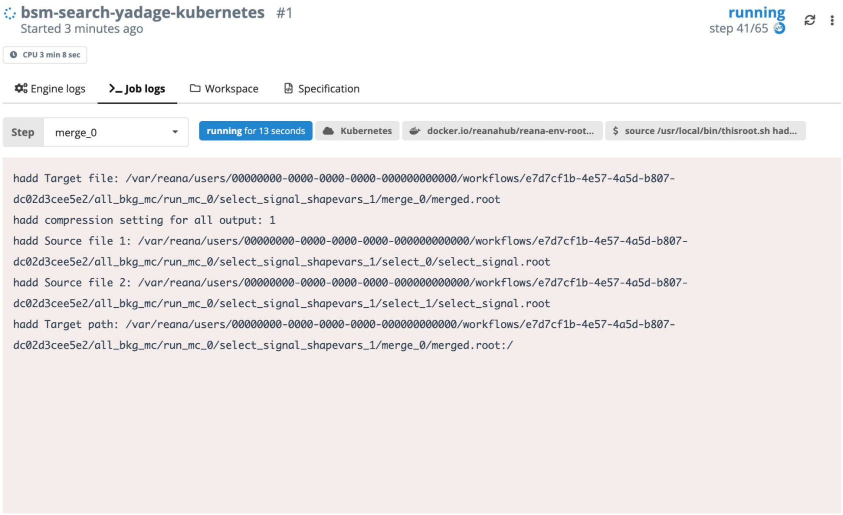
```
@@             Coverage Diff           @@
##           master    #161      +/-   ##
==========================================
+ Coverage   85.44%   86.00%   +0.56%
==========================================
  Files          41       41
  Lines        2919     2994      +75
==========================================
+ Hits         2494     2575      +81
+ Misses        333      330       -3
+ Partials       92       89       -3
```

Files with missing lines	Coverage Δ	
cmd/logs.go	95.79% <100.00%> (+6.05%)	⬆
pkg/config/config.go	100.00% <100.00%> (ø)	

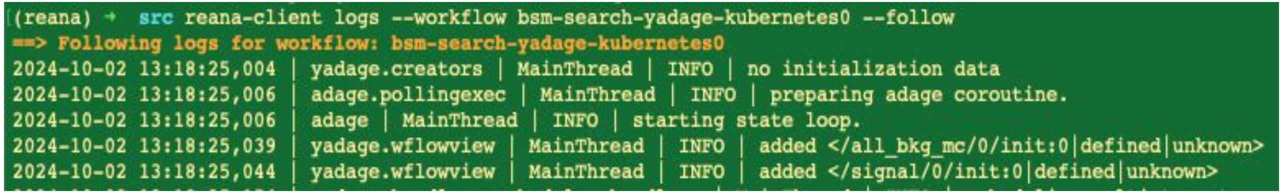
Solution implementation



```
2024-10-02 13:21:46,781 | yadage.creators | MainThread | INFO | no initialization data
2024-10-02 13:21:46,783 | adage.pollingexec | MainThread | INFO | preparing adage coroutine.
2024-10-02 13:21:46,783 | adage | MainThread | INFO | starting state loop.
2024-10-02 13:21:46,816 | yadage.wflowview | MainThread | INFO | added </all_bkg_mc/0/init:0|defined|unknown>
2024-10-02 13:21:46,821 | yadage.wflowview | MainThread | INFO | added </signal/0/init:0|defined|unknown>
2024-10-02 13:21:46,934 | yadage.handlers.scheduler_handlers | MainThread | INFO | scheduling multistep stage with spec:
{'scheduler_type': 'multistep-stage', 'parameters': [{'key': 'nevents', 'value': {'stages': 'init', 'output': 'nevents',
'unwrap': True, 'expression_type': 'stage-output-selector'}}, {'key': 'mcname', 'value': {'stages': 'init', 'output': 'mcname',
'unwrap': True, 'expression_type': 'stage-output-selector'}}, {'key': 'mcweight', 'value': {'stages': 'init', 'output':
'mcweight', 'unwrap': True, 'expression_type': 'stage-output-selector'}}, {'key': 'weightvariations', 'value': ['nominal',
'weight_var1_up', 'weight_var1_dn']}, {'key': 'shapevars', 'value': [['shape_conv_up'], ['shape_conv_dn']]}], 'workflow':
{'stages': [{'name': 'read', 'dependencies': {'dependency_type': 'jsonpath_ready', 'expressions': ['init']}, 'scheduler':
{'scheduler_type': 'multistep-stage', 'parameters': [{'key': 'nevents', 'value': {'stages': 'init', 'output': 'nevents',
'unwrap': True, 'expression_type': 'stage-output-selector'}}, {'key': 'outputfile', 'value': '{workdir}/output_one.root'},
{'key': 'type', 'value': {'stages': 'init', 'output': 'mcname', 'unwrap': True, 'expression_type': 'stage-output-selector'}}],
'step': {'process': {'process_type': 'interpolated-script-cmd', 'interpreter': 'bash', 'script': 'source
/usr/local/bin/thisroot.sh\npython /code/generantuple.py {type} {nevents} {outputfile}\n'}, 'environment': {'environment_type':
```



```
hadd Target file: /var/reana/users/00000000-0000-0000-0000-000000000000/workflows/e7d7cf1b-4e57-4a5d-b807-
dc02d3cee5e2/all_bkg_mc/run_mc_0/select_signal_shapevars_1/merge_0/merged.root
hadd compression setting for all output: 1
hadd Source file 1: /var/reana/users/00000000-0000-0000-0000-000000000000/workflows/e7d7cf1b-4e57-4a5d-b807-
dc02d3cee5e2/all_bkg_mc/run_mc_0/select_signal_shapevars_1/select_0/select_signal.root
hadd Source file 2: /var/reana/users/00000000-0000-0000-0000-000000000000/workflows/e7d7cf1b-4e57-4a5d-b807-
dc02d3cee5e2/all_bkg_mc/run_mc_0/select_signal_shapevars_1/select_1/select_signal.root
hadd Target path: /var/reana/users/00000000-0000-0000-0000-000000000000/workflows/e7d7cf1b-4e57-4a5d-b807-
dc02d3cee5e2/all_bkg_mc/run_mc_0/select_signal_shapevars_1/merge_0/merged.root:/
```



```
(reana) ➜  src reana-client logs --workflow bsm-search-yadage-kubernetes0 --follow
==> Following logs for workflow: bsm-search-yadage-kubernetes0
2024-10-02 13:18:25,004 | yadage.creators | MainThread | INFO | no initialization data
2024-10-02 13:18:25,006 | adage.pollingexec | MainThread | INFO | preparing adage coroutine.
2024-10-02 13:18:25,006 | adage | MainThread | INFO | starting state loop.
2024-10-02 13:18:25,039 | yadage.wflowview | MainThread | INFO | added </all_bkg_mc/0/init:0|defined|unknown>
2024-10-02 13:18:25,044 | yadage.wflowview | MainThread | INFO | added </signal/0/init:0|defined|unknown>
```

Thank you!