Open Infrastructure - Task #9565

Select a CI/CD for deploying helm charts/docker containers etc.

07/26/2021 01:16 PM - Nico Schottelius

Status:	Closed	**Start date:**	07/26/2021
Priority:	Normal	**Due date:**	
Assignee:	Nico Schottelius	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			
PM Check date:			

Description

- Basically: git push && pipeline that does the rest
- Input from your experiences is appreciated

Choices

Flux v2

- Overall nice
- Does not cover the build phase
- Has nice multi cluster support

drone

- Unclear on how / where to store the output
- K8S support seems to be fragile

Jenkins

- The "standard"
- Very heavy (4GB+ memory)

Buildbot

- Old
- Static workers (easy to configure via k8s)
- Seems to be fast and easy to setup
- Python based
- Website and documentation down as of 2021-08-08

GoCD

TL;DR: Does not even start in an IPv6 k8s cluster

- Recommended to me by the buildbot author (haaaaa??)
- Can push to docker registry
- Can be driven by a git repository

Non-working installation:

```
helm upgrade --install --set server.service.type=ClusterIP,server.ingress.enabled=false gocd gocd/gocd
```

Gitlab

TL;DR: has a lot included, maybe too much

- Is heavy to maintain without containers.
- Highly integrated

- Can use k8s workers, can use docker
- Widely deployed
- Huge and tricky to maintain
- Docker:
  - https://hub.docker.com/_/gitlab-community-edition
  - https://docs.gitlab.com/ee/install/docker.html
  - Helm chart support
    - Seems to be "rather native"
    - Registry included -- but no cleanup?
      - Can automatically deleted untagged - might be enough
- Gitlab/k8s seems to be strongly tied to terraform
  - Not suitable for bare metal

ArgoCD**and(?)**argoflow

- Rather complicated / big ecosystem
- Design to be cloud native
- Dependencies nicely solved
  - in order or via DAG

Argo flow

- Output is very S3 centered
  - We could use this, even though it seems overkill
  - This might be a practical requirement
  - Might be able to ignore this feature
- Argo flow tries to access /var/run/docker.sock directly - which does not exist for crio based environments
  - MountVolume.SetUp failed for volume "docker-sock" : hostPath type check failed: /var/run/docker.sock is not a socket file

Flows

DNS Update

Questions:

- Should we create a stand-alone zone repository?
  - Would be very small
  - Can only clone head/last commit
- If using git pull inside the container, we need to pass along credentials
  - possible in a secret

Flow v1

- We change a zone file in git and push it somewhere
- A new helm chart is being created
- (maybe in between) bump the chartversion field?
  - only if knot was able to run it?
- The new helm chart is uploaded to the chartmuseum
- The pods/services are notified about a new version
  - How?
    - Configmap change?
    - git pull?

Flow v2: pull from git repo

- The helm chart is given a git repo (+possible secret)
- The pod tries reloading every minute
  - if checkconf works: restart
  - else: reject
- A webhook in gitea might be used to trigger the DNS server instances
  - Faster deploy
  - Question is where to, whether we have 1 hook per cluster, etc.

Disadvantage: need to build our own container (?)

- In theory a custom container could do that in a pod

Flow v3: push pipeline

- In theory we want every zone change to create a new version number
- Actually we have this already with the git commit

Nothing to be done here.

History

#1 - 07/26/2021 01:31 PM - Nico Schottelius

- Description updated

#2 - 07/26/2021 02:55 PM - Nico Schottelius

- Description updated

#3 - 08/01/2021 09:00 AM - Nico Schottelius

- Description updated

#4 - 08/01/2021 09:24 AM - Nico Schottelius

- Description updated

#5 - 08/01/2021 09:30 AM - Nico Schottelius

- Description updated

#6 - 08/01/2021 06:28 PM - Amal Elshihaby

I prefer using travisCI or circleCI, they are light and easy to maintained.
I think too that they works good with Kubernetes

#7 - 08/02/2021 05:48 AM - Mondi Ravi

I do not have any preference -- I would probably go with Jenkins just because it is widely adopted.

GoCD looks promising also, though I've not used it personally.

#8 - 08/08/2021 01:33 PM - Nico Schottelius

- Project changed from 45 to Open Infrastructure

#9 - 08/08/2021 01:49 PM - Nico Schottelius

- Description updated

#10 - 08/08/2021 01:55 PM - Nico Schottelius

- Description updated

#11 - 08/08/2021 03:24 PM - Nico Schottelius

- Description updated

Argoflow notes

loop:

```
    dag:
      tasks:
        - name: print-message
          template: whalesay
          arguments:
            parameters:
              - name: message
                value: "{{item}}"
          withItems:
            - "hello world"
            - "goodbye world"
```

Sequence

```
    dag:
      tasks:
        - name: print-message
          template: whalesay
          arguments:
            parameters:
              - name: message
                value: "{{item}}"
          withSequence:
            count: 5
```

- A steps template allows you to run a series of steps in sequence.
- A suspend template allows you to automatically suspend a workflow, e.g. while waiting on manual approval, or while an external system does some work.

- *nix atexit support:
  - can submit to somewhere! https://argoproj.github.io/argo-workflows/examples/#exit-handlers

```
apiVersion: argoproj.io/v1alpha1
kind: Workflow
metadata:
  generateName: exit-handler-
spec:
  entrypoint: main
  templates:
    - name: main
      dag:
        tasks:
          - name: a
            template: whalesay
            onExit: tidy-up

    - name: whalesay
      container:
        image: docker/whalesay

    - name: tidy-up
      container:
        image: docker/whalesay
        command: [ cowsay ]
        args: [ "tidy up!" ]
```

Parameters

- Similar to helm values

```
    - name: main
      inputs:
        parameters:
          - name: message
      container:
        image: docker/whalesay
        command: [ cowsay ]
        args: [ "{{inputs.parameters.message}}" ]
```

Chaining in & out via a file:

```
apiVersion: argoproj.io/v1alpha1
kind: Workflow
metadata:
  generateName: parameters-
spec:
  entrypoint: main
  templates:
    - name: main
      dag:
        tasks:
          - name: generate-parameter
            template: whalesay
          - name: consume-parameter
            template: print-message
            dependencies:
              - generate-parameter
            arguments:
              parameters:
```

```
                - name: message
                  value: "{{tasks.generate-parameter.outputs.parameters.hello-param}}"
```

```
    - name: whalesay
      container:
        image: docker/whalesay
        command: [ sh, -c ]
        args: [ "echo -n hello world > /tmp/hello_world.txt" ]
      outputs:
        parameters:
          - name: hello-param
            valueFrom:
              path: /tmp/hello_world.txt
```

```
    - name: print-message
      inputs:
        parameters:
          - name: message
      container:
        image: docker/whalesay
        command: [ cowsay ]
        args: [ "{{inputs.parameters.message}}" ]
```

- artifact: files/blobs
  - Similar to input/output
- Example for a git repository: https://github.com/argoproj/argo-workflows/blob/master/examples/input-artifact-git.yaml

Workflowtemplate

- Basically a workflow stored in k8s that can be reused

CronWorkflow

- as the name says

Webhooks

- Very easy to create w/ input
- Using workflowtemplate

#12 - 08/08/2021 03:47 PM - Nico Schottelius

- Description updated

#13 - 08/08/2021 04:01 PM - Nico Schottelius

GoCD test

```
helm upgrade --install --set server.service.type=ClusterIP,server.ingress.enabled=false gocd gocd/gocd
```

Hangs in creating

```
[16:00] nb3:generic% kubectl describe pods gocd-server-5b8fb6b58f-54qc8
Name:           gocd-server-5b8fb6b58f-54qc8
Namespace:      default
Priority:       0
Node:           server60/2a0a:e5c0:13:0:225:b3ff:fe20:3736
Start Time:     Sun, 08 Aug 2021 15:58:49 +0200
Labels:         app=gocd
                component=server
                pod-template-hash=5b8fb6b58f
                release=gocd
Annotations:    cni.projectcalico.org/podIP: 2a0a:e5c0:13:e1:ddc1:7d11:9a1f:95a5/128
                cni.projectcalico.org/podIPs: 2a0a:e5c0:13:e1:ddc1:7d11:9a1f:95a5/128
Status:         Pending
IP:
IPs:            <none>
Controlled By:  ReplicaSet/gocd-server-5b8fb6b58f
Containers:
  gocd-server:
    Container ID:
    Image:          gocd/gocd-server:v21.2.0
    Image ID:
    Port:           8153/TCP
    Host Port:      0/TCP
```

```
    State:          Waiting
      Reason:       ContainerCreating
    Ready:          False
    Restart Count:  0
    Liveness:       http-get http://:8153/go/api/v1/health delay=90s timeout=1s period=15s #success=1 #failure
=10
    Readiness:      http-get http://:8153/go/api/v1/health delay=90s timeout=1s period=15s #success=1 #failure
=10
    Environment:
      GOCD_PLUGIN_INSTALL_kubernetes-elastic-agents:        https://github.com/gocd/kubernetes-elastic-agents/
releases/download/v3.7.1-230/kubernetes-elastic-agent-3.7.1-230.jar
      GOCD_PLUGIN_INSTALL_docker-registry-artifact-plugin:  https://github.com/gocd/docker-registry-artifact-p
lugin/releases/download/v1.1.0-104/docker-registry-artifact-plugin-1.1.0-104.jar
    Mounts:
      /docker-entrypoint.d from goserver-vol (rw,path="scripts")
      /godata from goserver-vol (rw,path="godata")
      /home/go from goserver-vol (rw,path="homego")
      /preconfigure_server.sh from config-vol (rw,path="preconfigure_server.sh")
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-2m9th (ro)
Conditions:
  Type              Status
  Initialized       True
  Ready             False
  ContainersReady   False
  PodScheduled      True
Volumes:
  config-vol:
    Type:      ConfigMap (a volume populated by a ConfigMap)
    Name:      gocd
    Optional:  false
  goserver-vol:
    Type:       PersistentVolumeClaim (a reference to a PersistentVolumeClaim in the same namespace)
    ClaimName:  gocd-server
    ReadOnly:   false
  kube-api-access-2m9th:
    Type:                     Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:   3607
    ConfigMapName:            kube-root-ca.crt
    ConfigMapOptional:        <nil>
    DownwardAPI:              true
QoS Class:                    BestEffort
Node-Selectors:               <none>
Tolerations:                  node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                              node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type    Reason     Age   From               Message
  ----    ------     ----  ----               -------
  Normal  Scheduled  100s  default-scheduler  Successfully assigned default/gocd-server-5b8fb6b58f-54qc8 to se
rver60
  Normal  Pulled     97s   kubelet            Container image "gocd/gocd-server:v21.2.0" already present on ma
chine
  Normal  Created    97s   kubelet            Created container gocd-server
  Normal  Started    97s   kubelet            Started container gocd-server
```

#14 - 08/08/2021 04:10 PM - Nico Schottelius

- Description updated

#15 - 08/08/2021 04:25 PM - Nico Schottelius

- Description updated

#16 - 08/08/2021 04:57 PM - Nico Schottelius

- Description updated

#17 - 08/08/2021 05:20 PM - Nico Schottelius

- Description updated

#18 - 08/08/2021 06:36 PM - Nico Schottelius

- Description updated

#19 - 08/08/2021 07:14 PM - Nico Schottelius

- Description updated

#20 - 12/18/2021 09:10 PM - Nico Schottelius

- Status changed from In Progress to Closed

We use argocd + argo workflow.