


 [fluxcd](#) / [flagger](#) Public

Progressive delivery Kubernetes operator (Canary, A/B Testing and Blue/Green deployments)

[fluxcd.io/flagger/](#)

 Apache-2.0 license

 **4.1k** stars  **610** forks

 Star



 Notifications

 **Code**  Issues 178  Pull requests 20  Actions  Security  Insights

main

Go to file



ayan9600 Merge pull request #1338 from wwadge/add-pro...



8 hours ago



2,618

[View code](#)



README.md

Releases 91

flagger

 **v1.27.0** Latest

on Dec 16, 2022

+ 90 releases

release v1.27.0




openssf best practices passing

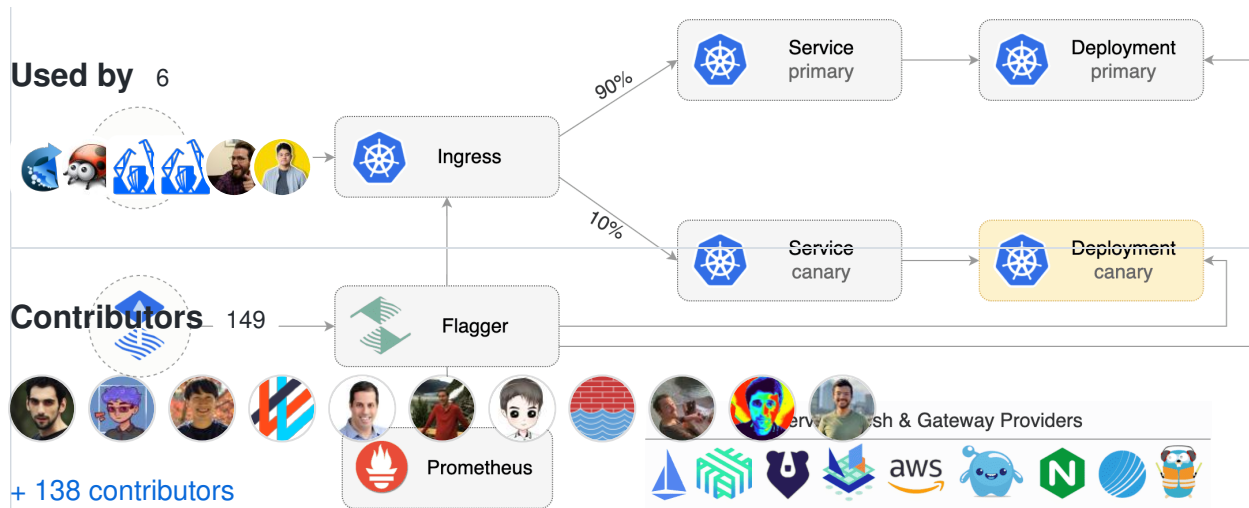
go report A+

license scan passing

Artifact Hub flagger

CLOMonitor Report A

- ## Packages 4
- Flagger is a progressive delivery tool that automates the release process for applications running on Kubernetes. It reduces the risk of introducing a new software version in production by gradually shifting traffic to the new version while measuring application health and running conformance tests.
-  **flagger**
 -  **flagger-admission**
 -  **charts/flagger**



Flagger implements several deployment strategies (Canary releases, A/B testing, Blue/Green mirroring) and integrates with various Kubernetes ingress controllers, service mesh, and monitoring solutions.

Languages

- Go 90.9%
- Shell 8.7%
- Other 0.4%

Flagger is a [Cloud Native Computing Foundation](#) project and part of the [Flux](#) family of GitOps tools.

Documentation

Flagger documentation can be found at fluxcd.io/flagger.

- Install
 - [Flagger install on Kubernetes](#)
- Usage
 - [How it works](#)
 - [Deployment strategies](#)
 - [Metrics analysis](#)
 - [Webhooks](#)
 - [Alerting](#)
 - [Monitoring](#)
- Tutorials
 - [App Mesh](#)
 - [Istio](#)
 - [Linkerd](#)
 - [Open Service Mesh \(OSM\)](#)
 - [Kuma Service Mesh](#)
 - [Contour](#)

- [Gloo](#)
- [NGINX Ingress](#)
- [Skipper](#)
- [Traefik](#)
- [Gateway API](#)
- [Kubernetes Blue/Green](#)

Adopters

Our list of production users has moved to <https://fluxcd.io/adopters/#flagger>.

If you are using Flagger, please [submit a PR to add your organization](#) to the list!

Canary CRD

Flagger takes a Kubernetes deployment and optionally a horizontal pod autoscaler (HPA), then creates a series of objects (Kubernetes deployments, ClusterIP services, service mesh, or ingress routes). These objects expose the application on the mesh and drive the canary analysis and promotion.

Flagger keeps track of ConfigMaps and Secrets referenced by a Kubernetes Deployment and triggers a canary analysis if any of those objects change. When promoting a workload in production, both code (container images) and configuration (config maps and secrets) are being synchronized.

For a deployment named *podinfo*, a canary promotion can be defined using Flagger's custom resource:

```
apiVersion: flagger.app/v1beta1
kind: Canary
metadata:
  name: podinfo
  namespace: test
spec:
  # service mesh provider (optional)
  # can be: kubernetes, istio, linkerd, appmesh, nginx, skipper, contour
  # for SMI TrafficSplit can be: smi:v1alpha1, smi:v1alpha2, smi:v1alpha3
  provider: istio
  # deployment reference
  targetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: podinfo
  # the maximum time in seconds for the canary deployment
  # to make progress before it is rollback (default 600s)
```

```
progressDeadlineSeconds: 60
# HPA reference (optional)
autoscalerRef:
  apiVersion: autoscaling/v2beta2
  kind: HorizontalPodAutoscaler
  name: podinfo
service:
  # service name (defaults to targetRef.name)
  name: podinfo
  # ClusterIP port number
  port: 9898
  # container port name or number (optional)
  targetPort: 9898
  # port name can be http or grpc (default http)
  portName: http
  # add all the other container ports
  # to the ClusterIP services (default false)
  portDiscovery: true
  # HTTP match conditions (optional)
  match:
    - uri:
        prefix: /
  # HTTP rewrite (optional)
  rewrite:
    uri: /
  # request timeout (optional)
  timeout: 5s
# promote the canary without analysing it (default false)
skipAnalysis: false
# define the canary analysis timing and KPIs
analysis:
  # schedule interval (default 60s)
  interval: 1m
  # max number of failed metric checks before rollback
  threshold: 10
  # max traffic percentage routed to canary
  # percentage (0-100)
  maxWeight: 50
  # canary increment step
  # percentage (0-100)
  stepWeight: 5
  # validation (optional)
  metrics:
    - name: request-success-rate
      # builtin Prometheus check
      # minimum req success rate (non 5xx responses)
      # percentage (0-100)
      thresholdRange:
        min: 99
      interval: 1m
    - name: request-duration
      # builtin Prometheus check
```

```
# maximum req duration P99
# milliseconds
thresholdRange:
  max: 500
interval: 30s
- name: "database connections"
# custom metric check
templateRef:
  name: db-connections
thresholdRange:
  min: 2
  max: 100
interval: 1m
# testing (optional)
webhooks:
- name: "conformance test"
  type: pre-rollout
  url: http://flagger-helmtester.test/
  timeout: 5m
  metadata:
    type: "helmv3"
    cmd: "test run podinfo -n test"
- name: "load test"
  type: rollout
  url: http://flagger-loadtester.test/
  metadata:
    cmd: "hey -z 1m -q 10 -c 2 http://podinfo.test:9898/"
# alerting (optional)
alerts:
- name: "dev team Slack"
  severity: error
  providerRef:
    name: dev-slack
    namespace: flagger
- name: "qa team Discord"
  severity: warn
  providerRef:
    name: qa-discord
- name: "on-call MS Teams"
  severity: info
  providerRef:
    name: on-call-msteams
```

For more details on how the canary analysis and promotion works please [read the docs](#).

Features

Service Mesh

Feature	App Mesh	Istio	Linkerd	Kuma	OSM	Kubernetes CNI
Canary deployments (weighted traffic)	✓	✓	✓	✓	✓	—
A/B testing (headers and cookies routing)	✓	✓	—	—	—	—
Blue/Green deployments (traffic switch)	✓	✓	✓	✓	✓	✓
Blue/Green deployments (traffic mirroring)	—	✓	—	—	—	—
Webhooks (acceptance/load testing)	✓	✓	✓	✓	✓	✓
Manual gating (approve/pause/resume)	✓	✓	✓	✓	✓	✓
Request success rate check (L7 metric)	✓	✓	✓	✓	✓	—
Request duration check (L7 metric)	✓	✓	✓	✓	✓	—
Custom metric checks	✓	✓	✓	✓	✓	✓

Ingress

Feature	Contour	Gloo	NGINX	Skipper	Traefik	Apache APISIX
Canary deployments (weighted traffic)	✓	✓	✓	✓	✓	✓

Feature	Contour	Gloo	NGINX	Skipper	Traefik	Apache APISIX
A/B testing (headers and cookies routing)	✓	✓	✓	—	—	—
Blue/Green deployments (traffic switch)	✓	✓	✓	✓	✓	✓
Webhooks (acceptance/load testing)	✓	✓	✓	✓	✓	✓
Manual gating (approve/pause/resume)	✓	✓	✓	✓	✓	✓
Request success rate check (L7 metric)	✓	✓	—	✓	✓	✓
Request duration check (L7 metric)	✓	✓	—	✓	✓	✓
Custom metric checks	✓	✓	✓	✓	✓	✓

Networking Interface

Feature	Gateway API	SMI
Canary deployments (weighted traffic)	✓	✓
A/B testing (headers and cookies routing)	✓	—
Blue/Green deployments (traffic switch)	✓	✓
Blue/Green deployments (traffic mirroring)	—	—
Webhooks (acceptance/load testing)	✓	✓
Manual gating (approve/pause/resume)	✓	✓
Request success rate check (L7 metric)	—	—
Request duration check (L7 metric)	—	—

Feature	Gateway API	SMI
Custom metric checks	✓	✓

For all [Gateway API](#) implementations like [Contour](#) or [Istio](#) and [SMI](#) compatible service mesh solutions like [Nginx Service Mesh](#), [Prometheus MetricTemplates](#) can be used to implement the request success rate and request duration checks.

Roadmap

GitOps Toolkit compatibility

- Migrate Flagger to Kubernetes controller-runtime and [kubebuilder](#)
- Make the Canary status compatible with [kstatus](#)
- Make Flagger emit Kubernetes events compatible with Flux v2 notification API
- Integrate Flagger into Flux v2 as the progressive delivery component

Integrations

- Add support for ingress controllers like HAProxy, ALB, and Apache APISIX
- Add support for Knative Serving

Contributing

Flagger is Apache 2.0 licensed and accepts contributions via GitHub pull requests. To start contributing please read the [development guide](#).

When submitting bug reports please include as many details as possible:

- which Flagger version
- which Kubernetes version
- what configuration (canary, ingress and workloads definitions)
- what happened (Flagger and Proxy logs)

Communication

Here is a list of good entry points into our community, how we stay in touch and how you can meet us as a team.

- Slack: Join in and talk to us in the [#flagger](#) channel on [CNCF Slack](#).
- Public meetings: We run weekly meetings - join one of the upcoming dev meetings from the [Flux calendar](#).

- Blog: Stay up to date with the latest news on [the Flux blog](#).
- Mailing list: To be updated on Flux and Flagger progress regularly, please [join the flux-dev mailing list](#).

Subscribing to the flux-dev calendar

To add the meetings to your e.g. Google calendar

1. visit the [Flux calendar](#)
2. click on "Subscribe to Calendar" at the very bottom of the page
3. copy the iCalendar URL
4. open e.g. your Google calendar
5. find the "add calendar" option
6. choose "add by URL"
7. paste iCalendar URL (ends with `.ics`)
8. done