Authored 8 months ago by **Иван Круглов**

SRE 4 (03.12.2021) | Canary Deployment | Часть 3: Мониторинг canary

Хорошей практикой для Canary является особый мониторинг, когда формируются графики отдельно для стабильной версии и отдельно для канарейки. Все это реализуемо через Argo Rollout.

Для этого необходимо проинструктировать Argo Rollout добавлять специальный лейбл на `stable` и `canary` поды. Редактируем `.helm/templates/deployment.yaml` :

```
...
     canary:
       maxSurge: 1
       maxUnavailable: 1
+      stableMetadata:
+        labels:
+          role: stable
+      canaryMetadata:
+        labels:
+          role: canary
```

Далее нам нужна метрика, которая будет отделять стабильные поды от `canary` подов. Для этого нам понадобятся две метрики:

- `http_server_requests_total` : содержит нужные данные, но на ней нет нужного лейбла. Но есть имя пода.
- `kube_pod_labels` : содержит нужный нам лейбл `label_role` и имя пода.

Их нужно объединить через функцию prometheus group_left через лейбл `pod` :

```
100
* sum(
    rate(http_server_requests_total{app="provider_backend", code!~"5..", city="$city"}[$period])
    * on (pod) group_left()
    kube_pod_labels{label_app="provider_backend", label_role="canary"}
  )
/ sum(
    rate(http_server_requests_total{app="provider_backend", city="$city"}[$period])
    * on (pod) group_left()
    kube_pod_labels{label_app="provider_backend", label_role="canary"}
  )
```

Применяем эту метрику в AnalysisTempalte:

```
       prometheus:
         address: https://prometheus.sre.slurm.io # p8s URL
         query: |
-          100 * sum(irate(http_server_requests_total{app="provider_backend", code!~"5..", city="{{ .Values.city }}"}[1m]
-              / sum(irate(http_server_requests_total{app="provider_backend", city="{{ .Values.city }}"}[1m]))
+          100 * sum(
+              irate(http_server_requests_total{app="provider_backend", code!~"5..", city="{{ .Values.city }}"}[1m])
+              * on (pod) group_left()
+              kube_pod_labels{label_app="provider_backend", label_role="canary"}
+            ) / sum(
+              irate(http_server_requests_total{app="provider_backend", city="{{ .Values.city }}"}[1m])
+              * on (pod) group_left()
+              kube_pod_labels{label_app="provider_backend", label_role="canary"}
+            )
```

ⓘ Полный `analysistemplate.yaml` в приложении.

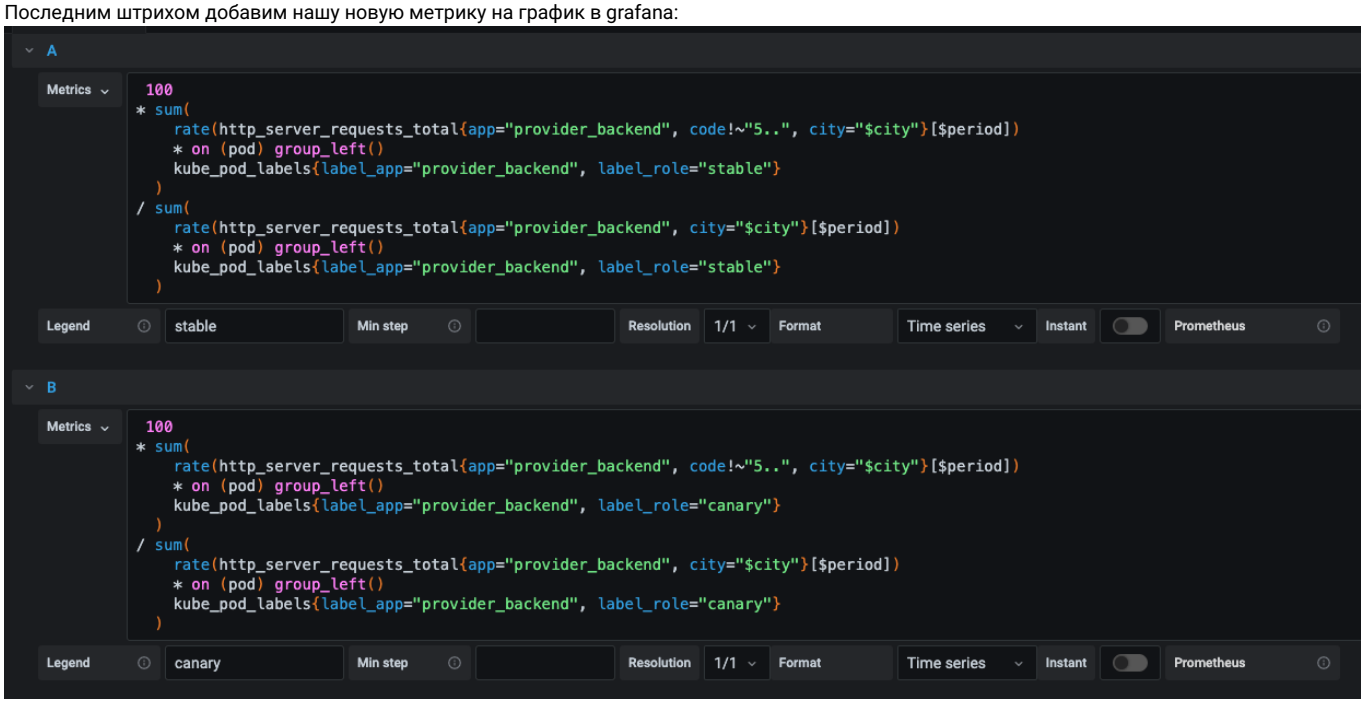
Однако, нам понадобится добавить небольшую задержку перед анализом первого пода. Это необходимо потому, что метрика использует `rate[1m]` , т.е. ей нужны данные как минимум за одну минуту. Которых не будет в самом начале, так как `canary` под только что запустился. Для этого добавим одноминутную паузу вторым шагом релиза.

```
...
       steps:
       - setWeight: 20
+      - pause:
+          duration: 1m
```

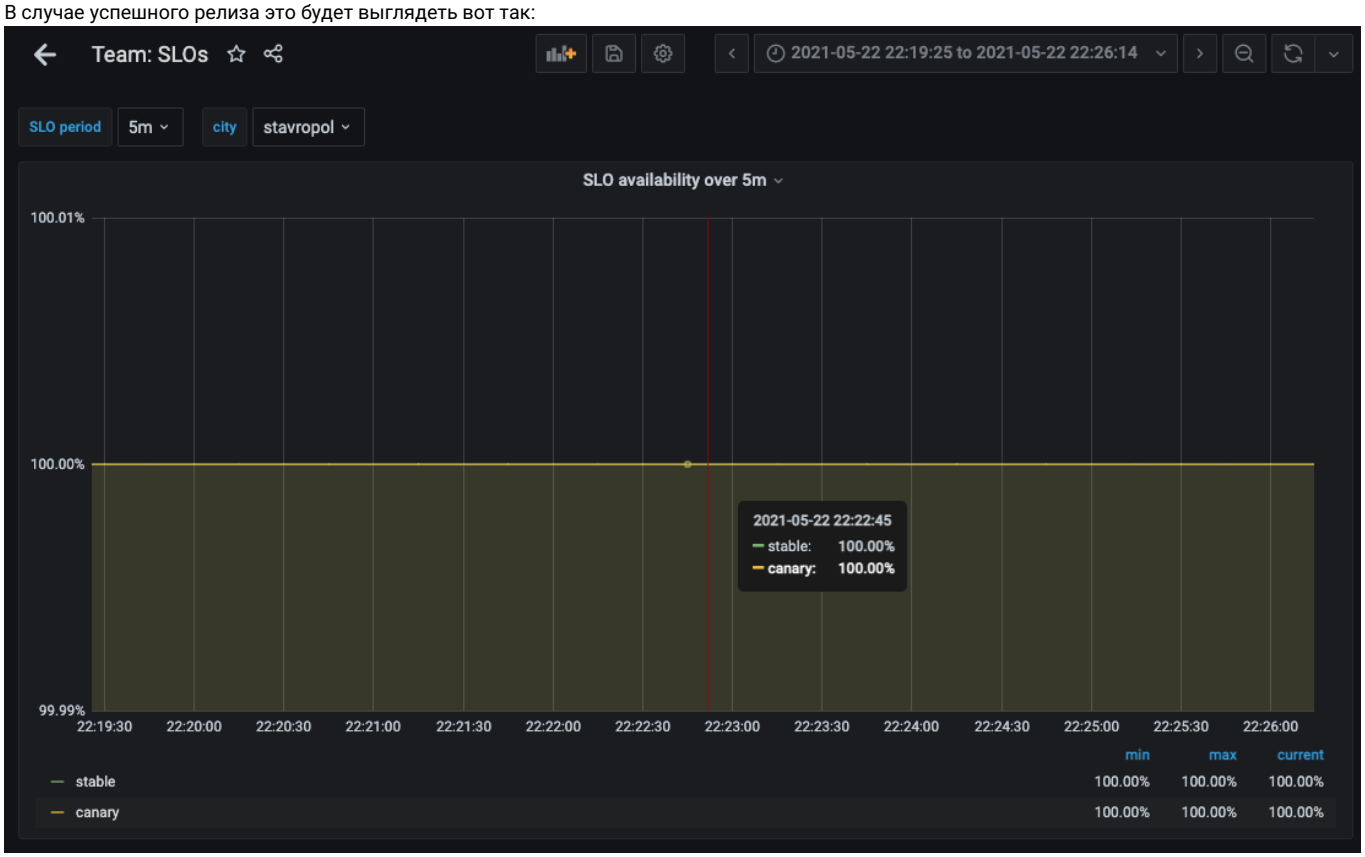
Если все сделано правильно, то во время деплоя мы должны увидеть новые лейблы, которые Argo Rollout будет добавлять и удалять в подах:

```
$ kubectl -n stavropol get pods -l app=provider_backend --show-labels
NAME                          READY   STATUS    RESTARTS   AGE     LABELS
stavropol-7576c99fcf-5hdht    1/1     Running   0          11m     ...,role=stable,...
stavropol-7576c99fcf-glz7d    1/1     Running   0          7m4s    ...,role=stable,...
stavropol-7576c99fcf-hq9m8    1/1     Running   0          13m     ...,role=stable,...
stavropol-7576c99fcf-mjqv6    1/1     Running   0          9m10s   ...,role=stable,...
stavropol-87c8f845d-xvhgm     1/1     Running   0          2m11s   ...,role=canary,... <----- HERE ---------
```

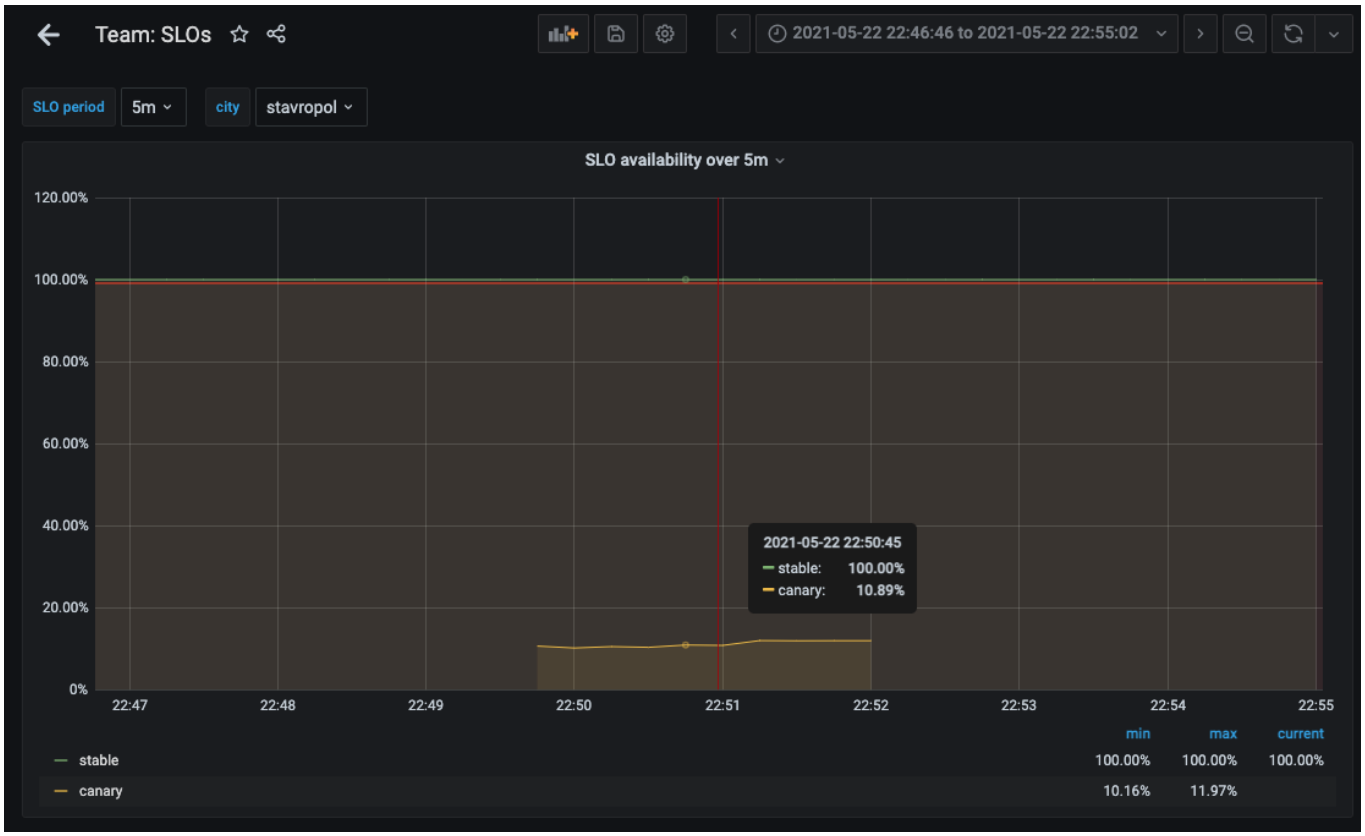
Последним штрихом добавим нашу новую метрику на график в grafana:



В случае успешного релиза это будет выглядеть вот так:



А в случае неуспешного вот так:



Edited 1 month ago

{..} **.helm/templates/analysistemplate.yaml** 1.12 KiB

```yaml
apiVersion: argoproj.io/v1alpha1
kind: AnalysisTemplate
metadata:
  name: {{ .Release.Name }}-success-rate
  labels:
    app: {{ .Chart.Name }}
    chart: "{{ .Chart.Name }}-{{ .Chart.Version }}"
    release: {{ .Release.Name }}
    heritage: {{ .Release.Service }}
spec:
  metrics:
  - name: success-rate
    count: 5 # number of measurments
    interval: 30s # how often to run p8s query
    failureLimit: 3 # how many times the `successCondition` should fail before declare a failure
    successCondition: result[0] >= 99
    provider:
      prometheus:
        address: https://prometheus.sre.slurm.io # p8s URL
        query: |
          100 * sum(
              irate(http_server_requests_total{app="provider_backend", code!~"5..", city="{{ .Values.city }}"}[1m]
              * on (pod) group_left()
              kube_pod_labels{label_app="provider_backend", label_role="canary"}
            ) / sum(
              irate(http_server_requests_total{app="provider_backend", city="{{ .Values.city }}"}[1m])
```

Write a comment or drag your files here…

Markdown is supported