


🛡️ Authored 6 months ago by  Иван Круглов

SRE 4 (03.12.2021) | Canary Deployment | Часть 1: Переход на Argo Rollout

Введение в [Argo Rollout](#)

Argo Rollout - это кubernetes контроллер и набор CRDs, которые реализуют продвинутые стратегии деплоя в кubernetes: blue-gree, canary, canary analysis, эксперименты.

В нашем кластере Argo Rollout уже установлен, так что никаких дополнительных действий не требуется. Однако, с процедурой установки можно ознакомиться в [официальной документации](#).

Переход с [Deployment](#) на [Argo Rollout](#)

Меняем `apiVersion` и `kind`

Argo Rollout - это практически полная drop-in замена для `Deployment` объекта в кubernetes. Для этого нужно заменить `apiVersion` и `Kind` на соответствующие значения и задать другую стратегию деплоя. Для этого редактируем `.helm/templates/deployment.yaml`:

```
-apiVersion: apps/v1
-kind: Deployment
+apiVersion: argoproj.io/v1alpha1
+kind: Rollout
```

и ниже редактируем стратегию:

```
strategy:
-   rollingUpdate:
```

```
+   canary:
      maxSurge: 1
      maxUnavailable: 1
```

на самом деле все. Argo Rollout будет вести себя так же как и обычный деплоймент. Полное описание Argo Rollout Spec [доступно в документации](#).

Однако, есть один нюанс. Он заключается в том, что Helm имеет встроенную поддержку Deployment и умеет ждать завершения выкатки. За это отвечает `--wait` или `--atomic` флаги в Helm. Поддержка Argo Rollout еще не интегрирована в Helm. Поэтому этот функционал нам придется реализовать вручную.

❗ Предлагаю подумать вам, какие еще есть последствия того, что Helm не поддерживает Argo Rollout.

wait-argo-rollout.sh

Нам понадобится вспомогательный скрипт `wait-argo-rollout.sh`. Создайте его вручную, используя контент из данного сниппета. Или же скачайте его, используя [следующую ссылку](#).

CI/CD

Далее нам нужно подправить CI/CD. Для этого вносим следующее изменение в самый конец секции `deploy_prod` в `.gitlab-ci.yml`

```
      --atomic
      --debug
      --namespace "$CITY"
+   - /bin/sh ./wait-argo-rollout.sh
```

Деплой

Пушим в репозиторий и смотрим как все отработало. Если все сделано правильно, то должна получиться примерно следующая картина:

```
1276 waiting for argo rollout to complete deployment: 1/600
1277 waiting for argo rollout to complete deployment: 2/600
1278 waiting for argo rollout to complete deployment: 3/600
1279 waiting for argo rollout to complete deployment: 4/600
1280 waiting for argo rollout to complete deployment: 5/600
1281 waiting for argo rollout to complete deployment: 6/600
1282 waiting for argo rollout to complete deployment: 7/600
1283 waiting for argo rollout to complete deployment: 8/600
1284 waiting for argo rollout to complete deployment: 9/600
1285 Deployment successfull
✓ 1286 Running after_script
1287 Running after script...
1288 $ date
1289 Mon May 17 07:20:48 UTC 2021
✓ 1290 Cleaning up file based variables
1291 Job succeeded
```

00:00

00:01

В kubectl появится:

```
$ kubectl -n stavropol get rollout
NAME          DESIRED  CURRENT  UP-TO-DATE  AVAILABLE
stavropol     5        5        2           5
```

А вот Deployment уйдет:

```
$ kubectl -n stavropol get deploy
```

No resources found in stavropol namespace.

Во время деплоя вероятно кратковременная просадка доступности. Это произойдет из-за того, что поды, которые принадлежат Deployment, будут удалены. А новые, принадлежащие Rollout, еще могут быть не созданы. Это единоразовая просадка. В дальнейшем такого не будет.

❶ Есть возможность перехода с Deployment на Rollout без просадки трафика, но это более сложный сценарий, который мы оставим на обсуждение.

Edited just now

 **wait-argo-rollout.sh** 885 bytes

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

```
#!/bin/sh

set -o pipefail
attempts=900
timedout=1

for i in `seq $attempts`; do
    echo "waiting for argo rollout to complete deployment: $i/$attempts"
    kubectl --namespace "$CITY" get rollouts.argoproj.io "$CITY" -o json | jq ".status.stableRS == .status.currentPodHash"
    if [ $? -eq 0 ]; then
        timedout=0
        break
    fi

    kubectl --namespace "$CITY" get rollouts.argoproj.io "$CITY" -o json | jq ".status.abort == true" --exit-status > /dev/null
    if [ $? -eq 0 ]; then
        timedout=0
        break
    fi

    sleep 1
done

if [ $timedout -ne 0 ]; then
    echo "Deployment timed out"
    exit 1
fi

kubectl --namespace "$CITY" get rollouts.argoproj.io "$CITY" -o json | jq ".status.abort == null" --exit-status > /dev/null
if [ $? -eq 0 ]; then
    echo "Deployment successfull"
    exit 0
fi
```

Write a comment or drag your files here...

[Markdown](#) is supported