

Лабораторная работа №10.

Задание 1.

```
UW PICO 5.09 File: system-info-cron

apiVersion: batch/v1
kind: CronJob
metadata:
  name: system-info-cron
spec:
  schedule: "*/5 * * * *"
  jobTemplate:
    spec:
      template:
        spec:
          containers:
            - name: stat
              image: busybox:latest
              imagePullPolicy: IfNotPresent
              command: ["/bin/sh", "-c", "echo \"Time: $(date), Host: $(hostname), Processes: $(ps aux | wc -l)\""]
              restartPolicy: OnFailure

> kubectl apply -f system-info-cron.yaml
cronjob.batch/system-info-cron configured
> kubectl get cronjob system-info-cron
NAME SCHEDULE TIMEZONE SUSPEND ACTIVE LAST SCHEDULE AGE
system-info-cron */5 * * * * <none> False 0 3m34s 9m4s
> kubectl get jobs
NAME STATUS COMPLETIONS DURATION AGE
system-info-cron-29394150 Complete 1/1 2s 8m42s
system-info-cron-29394155 Complete 1/1 3s 3m42s
> kubectl logs -l job-name=system-info-cron-29394155
/bin/sh: 1: ps: not found
Time: Thu Nov 20 14:35:00 UTC 2025, Host: system-info-cron-29394155-6zfkf, Processes: 0
> kubectl logs -l job-name=system-info-cron-29394150
/bin/sh: 1: ps: not found
Time: Thu Nov 20 14:30:00 UTC 2025, Host: system-info-cron-29394150-6gvb9, Processes: 0
```

Задание 2.

```
UW PICO 5.09

apiVersion: batch/v1
kind: CronJob
metadata:
  name: resource-checker-cron
spec:
  schedule: "*/3 * * * *"
  jobTemplate:
    spec:
      activeDeadlineSeconds: 45
      backoffLimit: 2
      template:
        spec:
          containers:
            - name: cheker
              image: busybox
              imagePullPolicy: IfNotPresent
              command:
                - /bin/sh
                - -c
                - sleep $((random % 60 + 30))
              restartPolicy: OnFailure
```

```

> nano resource-checker-cron.yaml
> kubectl apply -f resource-checker-cron.yaml
cronjob.batch/resource-checker-cron created
> kubectl describe cj resource-checker-cron
Name: resource-checker-cron
Namespace: default
Labels: <none>
Annotations: <none>
Schedule: */3 * * * *
Concurrency Policy: Allow
Suspend: False
Successful Job History Limit: 3
Failed Job History Limit: 1
Starting Deadline Seconds: <unset>
Selector: <unset>
Parallelism: <unset>
Completions: <unset>
Active Deadline Seconds: 45s
Pod Template:
  Labels: <none>
  Containers:
    cheker:
      Image: busybox
      Port: <none>
      Host Port: <none>
      Command:
        /bin/sh
        -c
        sleep $((random % 60 + 30))
      Environment: <none>
      Mounts: <none>
      Volumes: <none>
      Node-Selectors: <none>
      Tolerations: <none>
  Last Schedule Time: <unset>
  Active Jobs: <none>
  Events: <none>
> kubectl get jobs -w
NAME                                STATUS    COMPLETIONS   DURATION   AGE
resource-checker-cron-29394165     Complete  1/1            32s        104s
resource-checker-cron-29394168     Running   0/1            0s         0s
resource-checker-cron-29394168     Running   0/1            0s         0s
resource-checker-cron-29394168     Running   0/1            0s         0s
resource-checker-cron-29394168     Running   0/1            1s         1s
resource-checker-cron-29394168     Running   0/1            1s         1s
resource-checker-cron-29394168     Running   0/1            32s        32s
resource-checker-cron-29394168     Running   0/1            32s        32s
resource-checker-cron-29394168     SuccessCriteriaMet 0/1            33s        33s
resource-checker-cron-29394168     SuccessCriteriaMet 0/1            33s        33s
resource-checker-cron-29394168     Complete  1/1            33s        33s
resource-checker-cron-29394168     Complete  1/1            33s        33s

```

Какие Job's завершаются успешно, а какие с ошибкой?

Ответ: с ошибкой завершаются только те Job'ы, которые не укладываются по времени - превышают лимит 45 секунд.

Задание 3.

UW PICO 5.09

```
apiVersion: batch/v1
kind: CronJob
metadata:
  name: sequential-processor-cron
spec:
  schedule: "*/4 * * * *"
  concurrencyPolicy: Forbid
  jobTemplate:
    spec:
      completions: 4
      template:
        spec:
          containers:
            - name: worker
              image: busybox:latest
              imagePullPolicy: IfNotPresent
              command:
                - /bin/sh
                - -c
                - echo "Processing item $((RANDOM % 100))" && sleep 30
          restartPolicy: OnFailure
```

```
> nano sequential-processor-cron.yaml
> kubectl apply -f sequential-processor-cron.yaml
cronjob.batch/sequential-processor-cron unchanged
> kubectl get cj sequential-processor-cron -o yaml | grep concurrencyPolicy
  {"apiVersion":"batch/v1","kind":"CronJob","metadata":{"annotations":{},"name":"sequential-pro
template":{"spec":{"containers":[{"command":["/bin/sh","-c","echo \"Processing item $((RANDOM % 100
Policy\":\"OnFailure\")}]}}},"schedule":"*/4 * * * *"}
  concurrencyPolicy: Forbid
> kubectl get pods --watch
```

NAME	READY	STATUS	RESTARTS	AGE
sequential-processor-cron-29394176-cddpc	0/1	Pending	0	0s
sequential-processor-cron-29394176-cddpc	0/1	Pending	0	0s
sequential-processor-cron-29394176-cddpc	0/1	ContainerCreating	0	0s
sequential-processor-cron-29394176-cddpc	1/1	Running	0	1s
sequential-processor-cron-29394176-cddpc	0/1	Completed	0	31s
sequential-processor-cron-29394176-cddpc	0/1	Completed	0	32s
sequential-processor-cron-29394176-8l8nf	0/1	Pending	0	0s
sequential-processor-cron-29394176-8l8nf	0/1	Pending	0	0s
sequential-processor-cron-29394176-cddpc	0/1	Completed	0	33s
sequential-processor-cron-29394176-8l8nf	0/1	ContainerCreating	0	0s
sequential-processor-cron-29394176-8l8nf	1/1	Running	0	1s
sequential-processor-cron-29394176-8l8nf	0/1	Completed	0	32s
sequential-processor-cron-29394176-8l8nf	0/1	Completed	0	33s
sequential-processor-cron-29394176-vzhmp	0/1	Pending	0	0s
sequential-processor-cron-29394176-vzhmp	0/1	Pending	0	0s
sequential-processor-cron-29394176-8l8nf	0/1	Completed	0	34s
sequential-processor-cron-29394176-vzhmp	0/1	ContainerCreating	0	0s
sequential-processor-cron-29394176-vzhmp	1/1	Running	0	1s
sequential-processor-cron-29394176-vzhmp	0/1	Completed	0	31s
sequential-processor-cron-29394176-vzhmp	0/1	Completed	0	32s
sequential-processor-cron-29394176-xnzdn	0/1	Pending	0	0s
sequential-processor-cron-29394176-xnzdn	0/1	Pending	0	0s
sequential-processor-cron-29394176-vzhmp	0/1	Completed	0	33s
sequential-processor-cron-29394176-xnzdn	0/1	ContainerCreating	0	0s
sequential-processor-cron-29394176-xnzdn	1/1	Running	0	1s
sequential-processor-cron-29394176-xnzdn	0/1	Completed	0	31s
sequential-processor-cron-29394176-xnzdn	0/1	Completed	0	33s
sequential-processor-cron-29394176-xnzdn	0/1	Completed	0	34s

Задание 4.

```
UW PICO 5.09

apiVersion: batch/v1
kind: CronJob
metadata:
  name: policy-test-cron
spec:
  schedule: "*/2 * * * *"
  concurrencyPolicy: Forbid
  jobTemplate:
    spec:
      template:
        spec:
          containers:
            - name: worker
              image: nginx:latest
              imagePullPolicy: IfNotPresent
              command:
                - /bin/sh
                - -c
                - sleep 150
          restartPolicy: OnFailure
```

```
> nano policy-test-cron.yaml
> kubectl apply -f policy-test-cron.yaml
cronjob.batch/policy-test-cron configured
> nano policy-test-cron.yaml
> kubectl apply -f policy-test-cron.yaml
cronjob.batch/policy-test-cron configured
> kubectl get jobs --sort-by=.metadata.creationTimestamp
NAME                                STATUS    COMPLETIONS   DURATION   AGE
policy-test-cron-29394180           Complete  1/1            2m33s     5m15s
policy-test-cron-29394182           Complete  1/1            2m32s     3m15s
policy-test-cron-29394184           Running   0/1            48s       48s
```



~/BY3/K8S

Контрольные вопросы.

В чем основное отличие CronJob от обычной Job?

Ответ: CronJob имеет расписание и предназначен для задач, выполняемых по определенному графику.

Какой формат используется для задания расписания в CronJob? Что означают 5 полей?

Ответ: Cron-формат: «* * * * *», где «минута час день месяц день_недели».

Какие три политики параллелизма (concurrency policy) существуют у CronJob и в чем их разница?

Ответ: Forbid - новые Job'ы не создаются, если предыдущие выполняются; Allow - новые Job'ы создаются независимо от статуса предыдущих; Replace - незавершенная Job удаляется и создается новая.

Что произойдет, если выполнение задачи, инициированной CronJob, превысит activeDeadlineSeconds?

Ответ: убьется Pod, а Job получит статус Failed.

Где настраиваются параметры parallelism и completions для задач, создаваемых CronJob?

Ответ: в spec.jobTemplate.spec, т.е. в параметрах шаблона Job'ы

Как можно посмотреть логи конкретной Job, которая была создана CronJob?

Ответ: через selector найти Job'у, созданную CronJob и потом с помощью kubectl logs подключиться к ней.

Что произойдет, если CronJob будет запланирована на запуск в момент, когда кластер Kubernetes недоступен?

Ответ: CronJob пропустит выполнение, а когда kuber запуститься - продолжит работу по расписанию.

Какие команды вы бы использовали для диагностики проблем с CronJob, которая не создает Jobs?

Ответ:

kubectl describe cronjob <имя-cronjob>

kubectl get cronjob <имя-cronjob> -o yaml