

Runbook

Kubernetes Cronjob Failure

[← Back to Runbooks](#)

Overview

Kubernetes 52

 Get the Terraform

A Kubernetes Cronjob Failure incident occurs when a scheduled task, or cronjob, in a Kubernetes cluster fails to execute as expected. This may be due to a variety of reasons, such as misconfiguration, resource constraints, or software bugs. The incident requires investigation and debugging to identify the root cause of the failure and resolve the issue to restore normal operation. There is also a Kubernetes limitation that permanently stops a cronjob after too many (e.g. 100) execution errors or failures to schedule.

Parameters

```
1 export CRONJOB_NAME="PLACEHOLDER"
2
3 export POD_NAME="PLACEHOLDER"
4
5 export NAMESPACE="PLACEHOLDER"
6
7 export EXPECTED_SCHEDULE="PLACEHOLDER"
8
9 export PATH_TO_CRONJOB_YAML="PLACEHOLDER"
```



Debug

Check if the cronjob is still active



Check if the pods created by the cronjob are still running

```
kubectl get pods -l job-name=${CRONJOB_NAME}
```



Check the logs of the pods created by the cronjob

```
kubectl logs ${POD_NAME}
```



Check if the cronjob schedule is correct

```
kubectl describe cronjobs/${CRONJOB_NAME}
```



Check if there are any errors in the cronjob events

```
kubectl describe events --field-selector involvedObject.name=${CRONJOB_NAME}
```



Check if the cronjob image exists in the container registry

```
kubectl describe cronjobs/${CRONJOB_NAME} | grep Image:
```



Check the status of the last cronjob run

```
kubectl describe cronjobs/${CRONJOB_NAME} | grep Last Schedule Time:
```



Check if the cronjob is running on the expected node

```
kubectl describe pods ${POD_NAME} | grep Node:
```



Check if the pod has sufficient resources

```
kubectl describe pods ${POD_NAME} | grep -i resource
```



Check if there are any errors in the pod events

```
kubectl describe pods ${POD_NAME} | grep -i events
```




Repair

Check the cronjob configuration to ensure that it is correctly defined and scheduled to run at the intended time.

```
1  #!/bin/bash
2
3
4
5  # Set the namespace and cronjob name
6
7  NAMESPACE=${NAMESPACE}
8
9  CRONJOB_NAME=${CRONJOB_NAME}
10
11
12
13  # Get the cronjob object
14
15  CRONJOB=$(kubectl get cronjob $CRONJOB_NAME -n $NAMESPACE -o json)
16
17
18
19  # Check if the cronjob schedule is correct
20
21  EXPECTED_SCHEDULE=${EXPECTED_SCHEDULE}
22
23  CURRENT_SCHEDULE=$(echo $CRONJOB | jq -r .spec.schedule)
24
25
26
```





```
27 if [ "$CURRENT_SCHEDULE" != "$EXPECTED_SCHEDULE" ]; then
28
29     # Update the cronjob schedule
30
31     kubectl patch cronjob $CRONJOB_NAME -n $NAMESPACE --type='json' -p='[{"op
32
33 fi
```

Check for "100 missed start times" error. Recreate the cronjob if found.



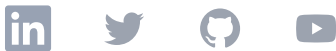
```
1 #!/bin/bash
2
3
4
5 # Set the namespace and cronjob name
6
7 NAMESPACE=${NAMESPACE}
8
9 CRONJOB=${CRONJOB_NAME}
10
11
12
13 # Check for "100 missed start times" error
14
15 if kubectl get cronjob $CRONJOB -n $NAMESPACE | grep -q "100 missed"; then
16
17     # Delete the cronjob
18
19     kubectl delete cronjob $CRONJOB -n $NAMESPACE
20
21
22
23     # Recreate the cronjob
24
25     kubectl apply -f ${PATH_TO_CRONJOB_YAML} -n $NAMESPACE
26
27
28
29     # Verify the new cronjob is scheduled and running
```

```
30
31 kubectl get cronjob $CRONJOB -n $NAMESPACE
32
33 kubectl get pods -l job-name=$CRONJOB -n $NAMESPACE
34
35 fi
```

[Learn more](#)

Related Runbooks

Check out these related runbooks to help you debug and resolve similar issues.



Questions? hello@shoreline.io

Product

- Shoreline for CPUs
- Shoreline for GPUs
- Runbook Library

Resources

- Blog
- Videos
- Podcasts
- Webinars
- Events
- Demos
- How Shoreline.io Works

Support

- Talk to us
- Customers
- Pricing
- Documentation [↗](#)
- Solutions
- Reliability.org [↗](#)

Company

- News
- Our Team
- Careers
- Values
- FAQ

