



KubeCon



CloudNativeCon

Europe 2026

#KubeCon #CloudNativeCon

A Tale of Two KEPs: How the Community is Taming Kubernetes' CrashLoopBackoff

Yang Li, Google Cloud





KubeCon



CloudNativeCon

Europe 2026

How many of you have
suffered from
CrashLoopBackOff?



KubeCon



CloudNativeCon

Europe 2026

**Ever wanted to fix
Kubernetes, but didn't know
how to write a KEP?**

back off restarts of crashlooping containers #12648

[New issue](#) Mergednikhiljindal merged 1 commit into [kubernetes:master](#) from [samsabed:crashloop](#) on Aug 25, 2015 Conversation 30 Commits 1 Checks 0 Files changed 14

+505 -34

 Changes from all commits ▾ File filter ▾ Conversations ▾ 

2 docs/user-guide/pod-states.md

<>  ...

The latest 1.0.x release of this document can be found [here](<http://releases.k8s.io/release-1.0/docs/user-guide/pod-states.md>).

RestartPolicy

The possible values for RestartPolicy are Always , OnFailure , or Never . If RestartPolicy is not set, the default value is Always . RestartPolicy applies to all containers in the pod. RestartPolicy only refers to restarts of the containers by the Kubelet on the same node. Failed containers that are restarted by Kubelet, are restarted with an exponential back-off delay, the delay is in multiples of sync-frequency 0, 1x, 2x, 4x, 8x ... capped at 5 minutes and is reset after 10 minutes of successful execution. As discussed in the [pods document](#), once bound to a node, a pod will never be rebound to another node. This means that some kind of controller is necessary in order for a pod to survive node failure, even if just a single pod at a time is desired.

The only controller we have today is [ReplicationController](#) . ReplicationController is only appropriate for pods with RestartPolicy = Always . ReplicationController should refuse to instantiate any pod that has a different restart policy.

Make CrashLoopBackoff timing tuneable, or add mechanism to exempt some exits #57291

[New issue](#)[Open](#)

jgiles opened on Dec 18, 2017

...

Is this a **BUG REPORT** or **FEATURE REQUEST**?: Feature request

/kind feature

What happened:

As part of a development workflow, I intentionally killed a container in a pod with `restartPolicy: Always`. The plan was to do this repeatedly, as a quick way to restart the container and clear old state (and, in Minikube, to load image changes).

The container went into a crash-loop backoff, making this anything but a quick option.

What you expected to happen:

I expected there to be some configuration allowing me to disable, or at least tune the timing of, the CrashLoopBackoff.

How to reproduce it (as minimally and precisely as possible):

Create a pod with `restartPolicy: Always`, and intentionally exit a container repeatedly.

Anything else we need to know?:

I see that the backoff timing parameters are hard-coded constants here:

[kubernetes/pkg/kubelet/kubelet.go](#)

Line 121 in 5f92042

```
121 MaxContainerBackOff = 300 * time.Second
```

[kubernetes/pkg/kubelet/kubelet.go](#)

Line 155 in 5f92042

```
155 backOffPeriod = time.Second * 10
```

One might reasonably expect these to be configurable at least at the kubelet level - say, by a setting like [these](#). That would be

Assignees

No one assigned

Labels

[kind/feature](#)

[lifecycle/frozen](#)

[sig/node](#)

Type

No type

Projects

No projects

Milestone

No milestone

Relationships

None yet

Development

Code with agent mode

No branches or pull requests

Participants



+58



KubeCon

Europe 2026

is:issue state:open sort:reactions-desc



Labels

Milestones

New issue

Open 1,827 Closed 46,929 Author ▾ Labels ▾ Projects ▾ Milestones ▾ Assignees ▾ Types ▾ Reactions ▾

Facilitate ConfigMap rollouts / management area/app-lifecycle area/configmap-api area/declarative-configuration priority/backlog sig/apps sig/service-catalog 296
#22368 · bgrant0607 opened on Mar 2, 2016

Support port ranges or whole IPs in services area/kube-proxy kind/feature priority/backlog sig/network triage/accepted 190
#23864 · adimania opened on Apr 5, 2016

Make CrashLoopBackoff timing tuneable, or add mechanism to exempt some exits kind/feature lifecycle/frozen sig/node 109
#57291 · jgiles opened on Dec 18, 2017

Give the proper reason for killing pods kind/feature needs-triage sig/node 86
#81723 · andreib1 opened on Aug 21, 2019



KubeCon



CloudNativeCon

Europe 2026

Voices from Issue #57291

- **"Success" Exits (Exit Code 0):**
 - Don't punish a good container
- **Early Recovery:**
 - Faster retries for transient errors
- **Late Recovery:**
 - Give us a manual reset button

Workarounds

- **Pod:** Bash wrappers (`while true; do app.py; done`)
- **Cluster:** Custom "Pod Reaper" operators
- **Node:** Forking K8s to patch Kubelet binaries

Typical Modern Workloads

- **Task Isolation:** Fast in-place session resets without Pod rescheduling overhead.
- **Fast Restart on Failure:** Transient, recoverable errors causing massive cascade delays in AI/ML.
- **Critical Sidecars:** Infra kills (e.g., OOMKilled proxies) isolating perfectly healthy main apps.

KEP-4603: Tune Crashloop Backoff

ALPHA

Status: Implementable

Links

- Enhancement Issue #4603
- KEP Documentation

SIGs

Owning SIG:

[SIG Node](#)

Milestones

Alpha v1.32

Feature Gates [↔](#)

This enhancement was designed with opt-in or opt-out at the feature level; there are [feature gates](#) associated with it.

Name	Components
ReduceDefaultCrashLoopBackoffDecay	kubelet

Feature gates for this KEP

People

Authors: [@lauralorenz](#)

Reviewers: [@tallclair](#)

Approvers: [dchen1107](#)

Timeline

Created: 2024-04-29

Latest Milestone: v1.33

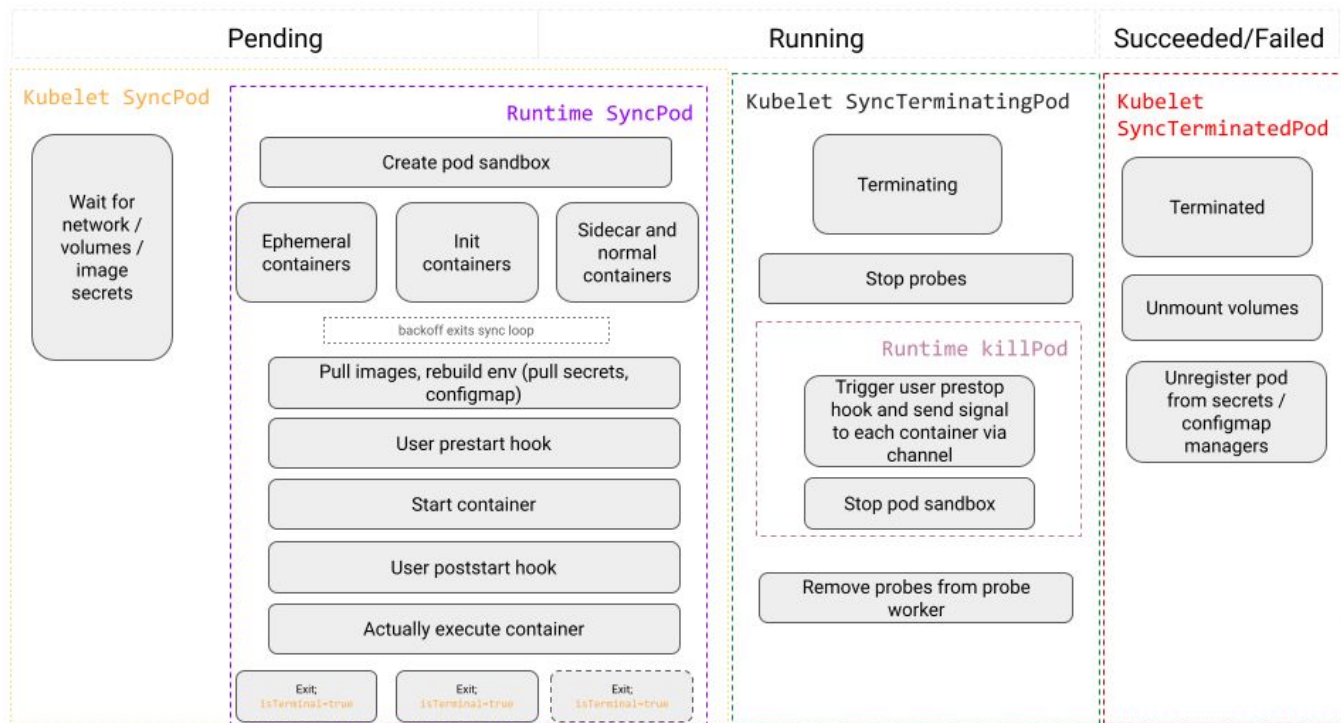
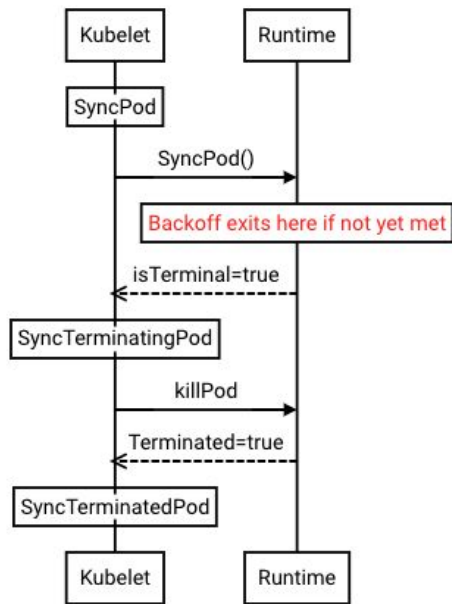
Related KEPs

- [/keps/sig-node/5593-configure-the-max-crashloopbackoff-delay](#)

Alternatives Considered

- Flat-rate restarts for Succeeded Pods
- RestartPolicy: Rapid
- ...

Kubelet Overhead Analysis

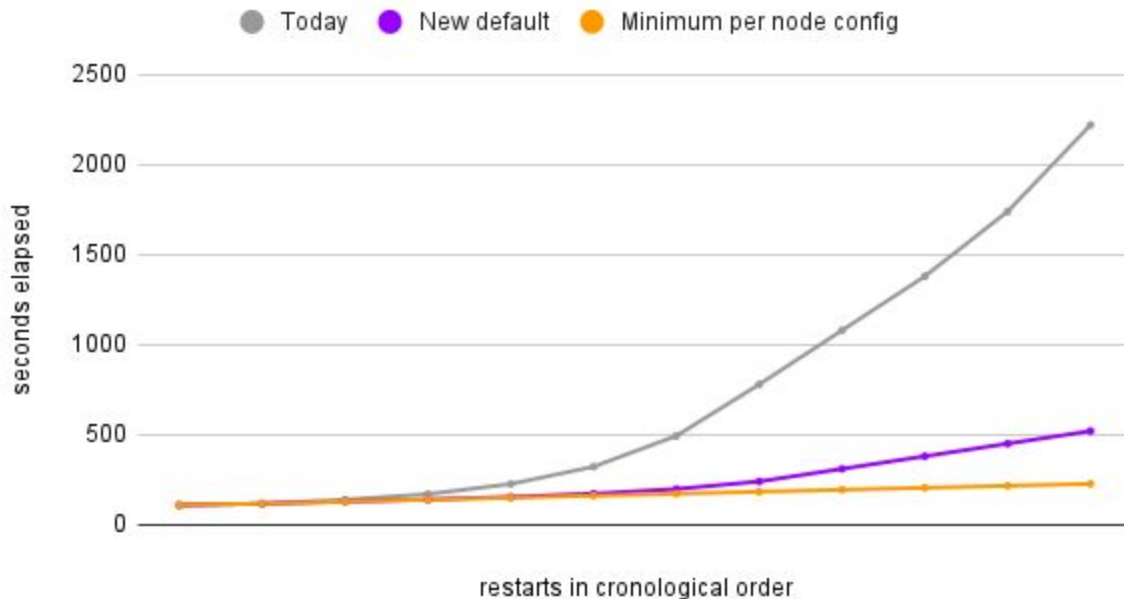


Kubelet Overhead Analysis (cont.)

- **110** Crashing Pods
- **5 QPS** at API Traffic Peak
- **2x** Kubelet CPU

From v1.32 to v1.34: KEP-4603

Today, New default and Minimum per node config



KubeletConfiguration

apiVersion: kubelet.config.k8s.io/v1beta1

kind: KubeletConfiguration

container restart delays will start at 10s, increasing

2x each time they are restarted, to a maximum of 100s

crashLoopBackOff:

maxContainerRestartPeriod: "100s"

apiVersion: kubelet.config.k8s.io/v1beta1

kind: KubeletConfiguration

delays between container restarts will always be 2s

crashLoopBackOff:

maxContainerRestartPeriod: "2s"

KEP-5593: Configure the max CrashLoopBackOff delay

BETA

Status: Implementable

Links

- [Enhancement Issue #5593](#)
- [KEP Documentation](#)

SIGs

Owning SIG:

[SIG Node](#)

Milestones

Alpha	v1.32
Beta	v1.35

Feature Gates

This enhancement was designed with opt-in or opt-out at the feature level; there are [feature gates](#) associated with it.

Name	Components
KubeletCrashLoopBackOffMax	kubelet

Feature gates for this KEP

People

Authors: [@lauralorenz](#) , [@hankfreund](#)

Reviewers: [@talclair](#)

Approvers: [@mrunalp](#)

Timeline

Created: 2025-09-30

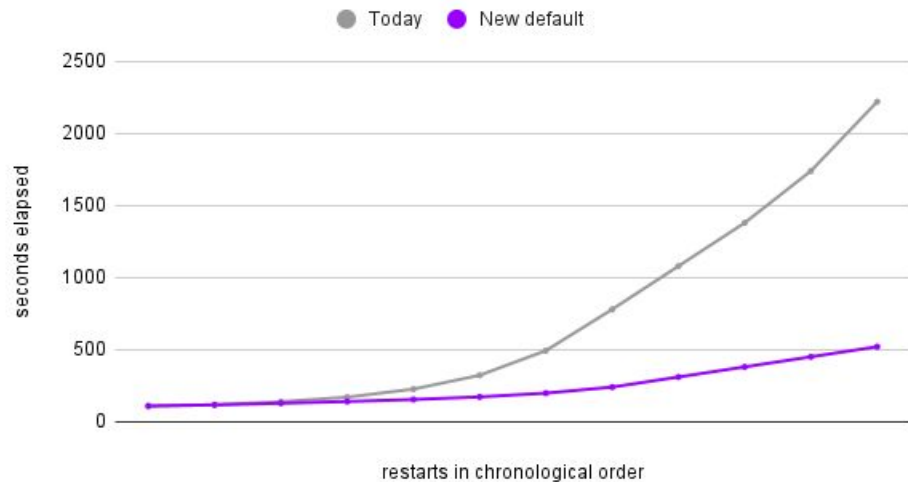
Latest Milestone: v1.35

Related KEPs

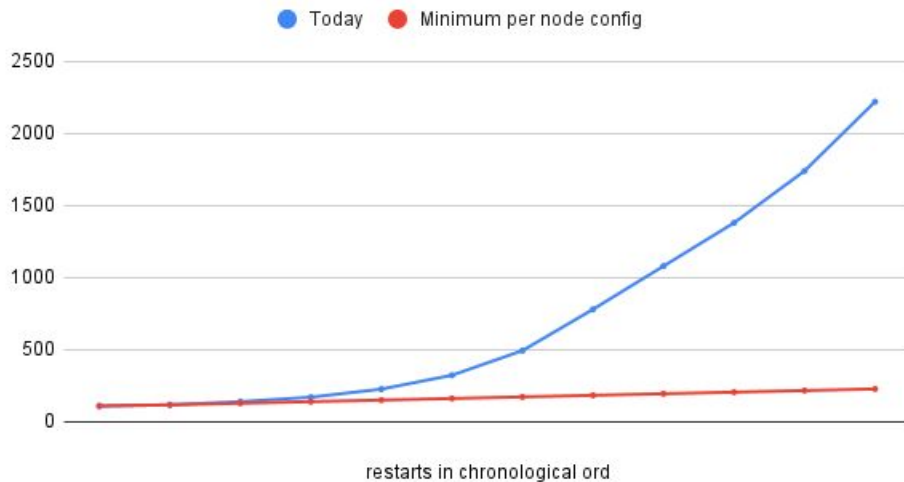
- [/keps/sig-node/4603-tune-crashloopbackoff](#)

v1.35 onwards: KEP-4603 and KEP-5593

Today and New default



Today and Minimum per node config





KubeCon



CloudNativeCon

Europe 2026

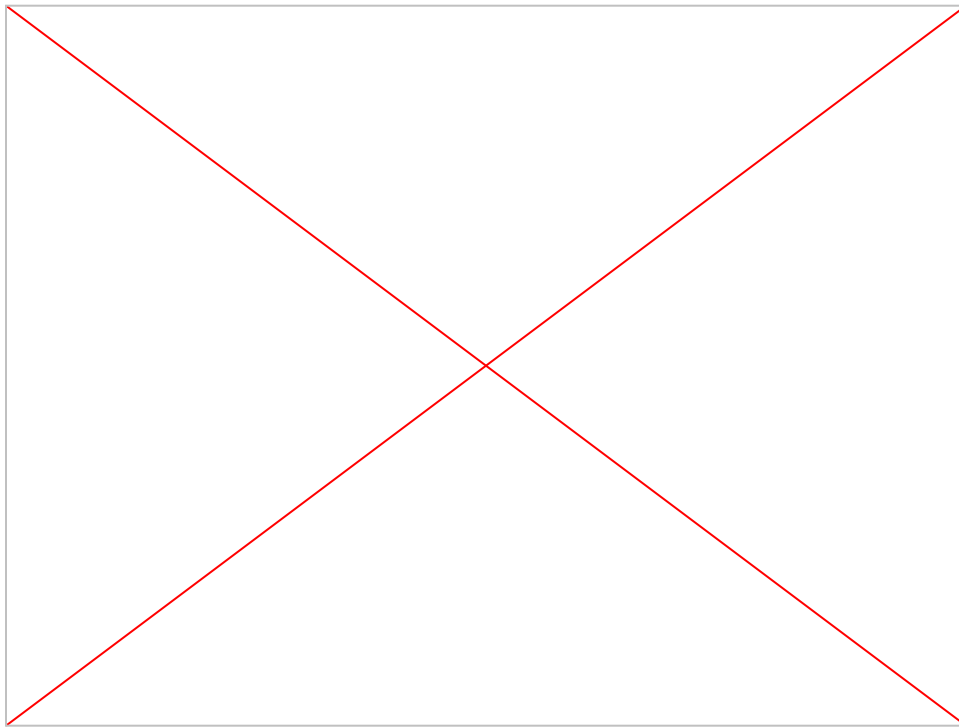
Let's do some demos



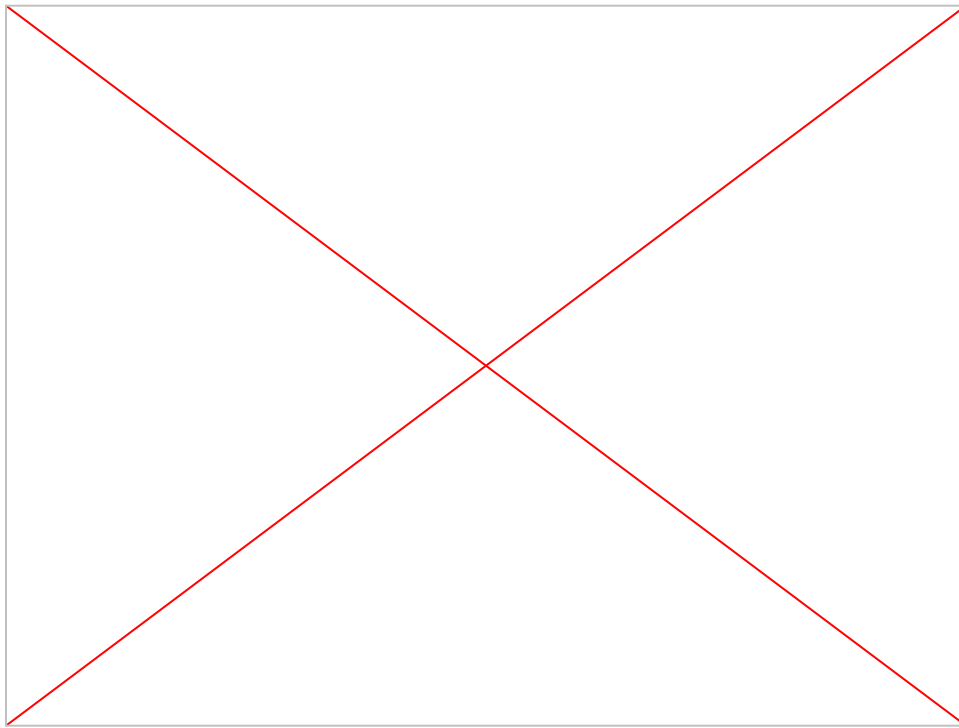
Demo Cluster Setup (v1.35)

- **worker1**
 - **2s** max backoff (KEP-5593 beta with configuration)
- **worker2**
 - **60s** max backoff (KEP-4603 alpha)
- **worker3**
 - **300s** max backoff (pre-v1.35 default)

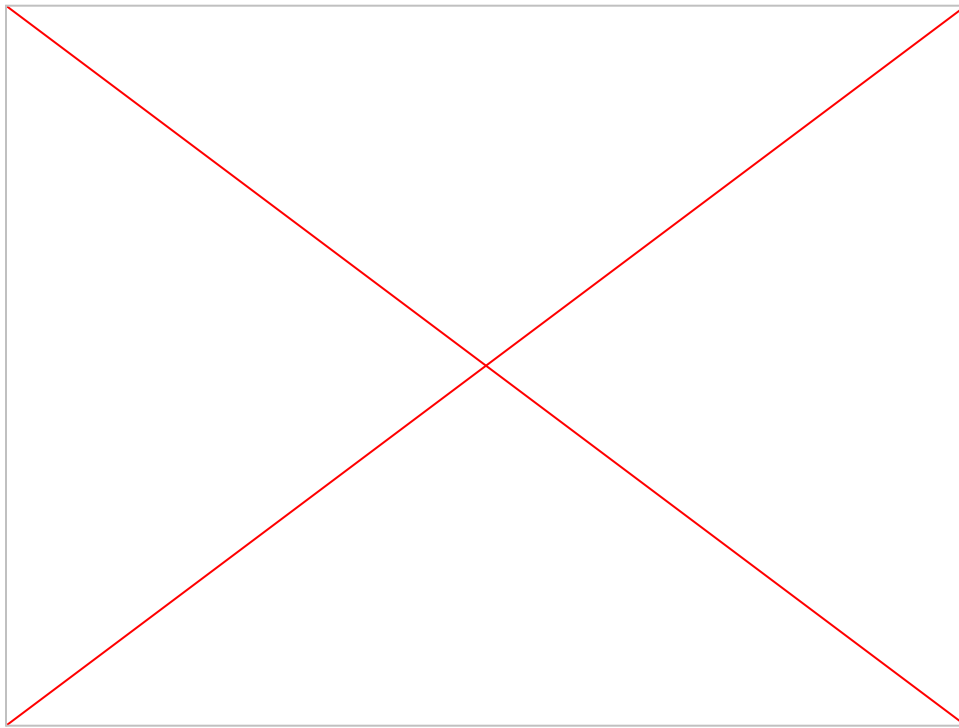
Scenario 1 - Task Isolation



Scenario 2 - Fast Restart



Scenario 3 - Sidecar Restart



Key Takeaways



KubeCon



CloudNativeCon

Europe 2026

- Container restarts are heavy, Kubelet has real physical limits
- KEP-4603 and KEP-5593 gives cluster operators granular, safe control over recovery times
- Open source is a marathon, pragmatic splits unblock years of frustration

Shoutouts



KubeCon



CloudNativeCon

Europe 2026



@lauralorenz



@hankfreund



SIG Node



KubeCon



CloudNativeCon

Europe 2026

Any questions?





KubeCon



CloudNativeCon

Europe 2026

