
KubeCon
CloudNativeCon
North America 2024

KubeCon | CloudNativeCon

North America 2024

SIG-Node: Intro and Deep Dive

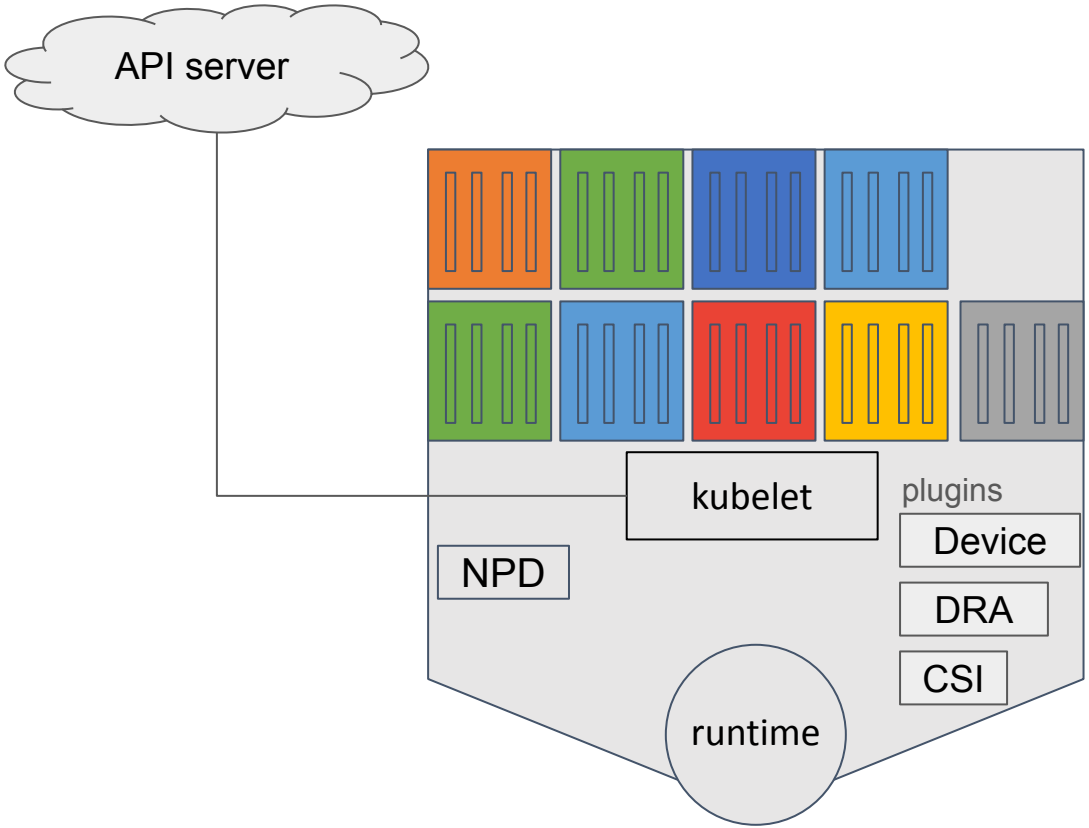
Dawn, Mrunal, Sergey

Agenda

- Introduction
- What's new
  - KEPs
  - Other improvements
- Deep Dive: Pod-level resources
- Future directions
- How to get involved

Introduction

Kubernetes Components



Components on a Node

- **Kubelet**: The foreman on the node
- **Container Runtime**: The Engine Room
- **Resource Management**: The Key to Efficient Workloads
- **Device plugins / DRA**: Assigning the hardware
- **Node Problem Detector**: The Watchdog
- **Storage and Network Components**: CSI, CNI

Together, a Symphony of functionality

- Run containerized workloads efficiently
- Manage pod lifecycles
- Optimize the use of specialized hardware resource
- Preserve node health, enabling self-heal
- Facilitate smooth communication and integration with storage resources

SIG Node

- A virtual but critical team which is responsible for ensuring smooth pod execution on worker machines
  - Primary slack channel #sig-node member count: **4.5k**
  - Primary mailing list member count: **1k**
  - Primary meeting attendee avg participation: **30**
- **11** subprojects, **5** working groups
- KEPs from last year:
  - 1.32 (not quite yet) with **36 tracked** and **15 merged (not quite sure yet)**
  - 1.31 (aug 2024) with **22 tracked** and **16 merged**
  - 1.30 (apr 2024) with **20 tracked** and **13 merged**
- More numbers:
  - Avg # of bugs to triage weekly: ~12
  - Avg # of PRs opened to review weekly: ~20
  - Avg # of active PRs at any given moment: 250+

What’s new

1.32 Feature Updates

1	Introducing Sleep Action for PreStop Hook #3960	Tracked	3 - Stable
2	Support to size memory backed volumes #1967	Tracked	3 - Stable
3	In-Place Update of Pod Resources #1287	Tracked	2 - Beta
4	Add support for a drop-in kubelet configuration directory #3983	Tracked	2 - Beta
5	DRA: Prioritized Alternatives in Device Requests #4816	Tracked	1 - Alpha
6	DRA: Add support for partitionable devices #4815	Tracked	1 - Alpha
7	Add Resource Health Status to the Pod Status for Device Plugin and DRA #4680	Tracked	1 - Alpha
8	KEP-4603: Tune CrashLoopBackoff #4603	Tracked	1 - Alpha
9	Pod level resources #2837	Tracked	1 - Alpha
10	Windows CPU and Memory Affinity #4885	Tracked	1 - Alpha
11	Split stdout and stderr log stream of container #3288	Tracked	1 - Alpha
12	Support PSI based on cgroupv2 #4205	Tracked	1 - Alpha
13	Ensure secret pulled images #2535	Tracked	1 - Alpha
14	Restarting sidecar containers during Pod termination #4438	Tracked	1 - Alpha

🎉 DRA is Beta in 1.32! 🎉

- **Evolution of Device Management:** DRA provides a standardized framework for managing specialized hardware resources like GPUs, TPUs, and network devices.
- **Performance Boost:** Experience up to 16x faster scheduling with the new structured parameters.
- **Enhanced Functionality:**
  - Driver-owned resource claim status enables advanced multi-networking use cases.
  - Seamless integration with the cluster autoscaler for dynamic scaling.
- **Growing Ecosystem:**
  - Example driver and NVIDIA GPU driver are production-ready.
  - CNI and Google TPU drivers are in progress.
- **Future Roadmap:**
  - Prioritized alternatives, partitionable device support, and resource health status are coming soon.

🎉 DRA is Beta in 1.32! (cont) 🎉

- Merged in 1.32
  - Structured parameters (DRA MVP) graduated to **Beta**
  - Faster scheduling (up to 16x)
  - Removal of classic DRA
  - Driver-owned resource claim status (for multi-networking use cases, primarily)
  - Significant progress on autoscaler integration
- Drivers for 1.32 (out-of-tree)
  - Example driver (ready)
  - NVIDIA DRA Driver for GPUs (ready)
  - CNI DRA Driver (in progress)
  - Google TPU Driver (in progress)
- Missed but on track for alpha in 1.33:
  - Prioritized alternatives in device requests
  - Support for partitionable devices
  - DRA resource health status in Pod Status
- Many more to coming …

In-place Pod Resizing Beta

- **Dynamic Scaling:** Scale pod resources (CPU, memory) up or down on-demand.
- **Zero Disruption:** Resize resources without restarting pods or interrupting applications.
- **Benefits:**
  - **Elasticity:** Adapt to fluctuating workloads seamlessly.
  - **Efficiency:** Optimize resource allocation and reduce waste.
  - **Cost Savings:** Avoid over-provisioning and pay only for what you use.
  - **Resilience:** Correct resource misconfigurations without downtime.

In-place Pod Resizing Beta (cont)

- [x] [FG:InPlacePodVerticalScaling] Implement version skew handling for in-place pod resize #117767
- [x] [FG:InPlacePodVerticalScaling] Handle pod CPU resize where caller requests CPU value of 1m #114123
- [x] [FG:InPlacePodVerticalScaling] Resizing pod gets stuck if limit is not configured #126388
- [x] [FG:InPlacePodVerticalScaling] ResourceQuota unresponsive to scale-down #127132
- [x] [FG:InPlacePodVerticalScaling] Emit a events when resize status changes #127172 (remaining work is not beta-blocking)
- [x] [FG:InPlacePodVerticalScaling] Disable in-place resize for guaranteed pods on nodes with a static topology policy #128068
- [x] [FG:InPlacePodVerticalScaling] Container resize policy feature introduced in v1.27 doesn't interrupt CrashLoopBackOff #119838
- [x] [FG:InPlacePodVerticalScaling] Disallow removing requsets & limits during resize #128677

NOTE: The following changes were originally in-scope for beta, but have been moved out:

- [FG:InPlacePodVerticalScaling] Add UpdatePodSandboxResources CRI method #128069 - This was always meant to be a best-effort call to inform the runtime of a resize, but the resize is still handled by the Kubelet. This should move to the GA scope, but we do not have a direct consumer or use case for it now.
- [FG:InPlacePodVerticalScaling] Add kubelet_resize_requests_total metric #128071 - This metric is not sufficiently well defined. With the addition of the `/resize` subresource, we automatically get a metric for total number of resize requests. We will revisit this for GA.
- [FG:InPlacePodVerticalScaling] Implement resize for sidecar containers #128070 - Unfortunately this isn't feasible in v1.32 due to a validation rollback issue (see https://github.com/kubernetes/kubernetes/pull/128367/files#r1834897969). We will make the validation changes in v1.32 to make this feasible in v1.33, but the actual implementation won't land until v1.33.

Pod Level Resources Alpha

- **Dynamic Sharing:** Containers within a pod dynamically share a pool of resources.
- **Simplified Allocation:** Define resource requests and limits for the pod as a whole.
- **Benefits:**
  - **Ease of Use:** Simplify resource allocation for complex applications.
  - **Efficiency:** Optimize resource utilization and reduce waste.
  - **Cost Savings:** Avoid over-provisioning and pay only for what you need.
  - **Flexibility:** Adapt to changing workload demands.

Node Swap Support

- **Controlled Swap Utilization:** Allow nodes to leverage swap memory in a controlled manner.
- **Enhanced Stability:** Improve node stability under memory pressure.
- **Benefits:**
  - **Resilience:** Handle memory spikes and resource fluctuations more effectively.
  - **Flexibility:** Utilize swap for specific workloads or node-level tuning.
  - **Efficiency:** Potentially improve resource utilization and reduce costs.
- **Current Status:**
  - Actively developing toward beta.
  - Undergoing stress tests and eviction manager integration.

Other improvements

- KEP-4369: Allow special characters in environment variables
- KEP-3288: Split Stdout and Stderr Log Stream of Container
- KEP-3857: Recursive read-only (RRO) mounts
- KEP-4540: Add CPUManager policy option to restrict reservedSystemCPUs to system daemons and interrupt processing

Deep Dive: Pod Level Resources

Motivation

- Allow specifying resources at the Pod Level
- Containers can collaborate within the boundaries of the Pod Sandbox
- Container resource peaks may not occur at the same time

Pod Level Resources

- Opt-in
- Support for Memory / CPU
- Cgroups v2 only

Pod Level Resources

- No container limit can exceed the pod limit
- The pod limit must be less than equal to sum of container limit
- Pod requests must be greater than equal to sum of container resources

Pod Level Resources


```yaml
apiVersion: v1
kind: Pod
metadata:
  name: web-pod
spec:
  containers:
  - name: httpd
    image: httpd:2.4-alpine
    resources:
      requests:
        memory: "100Mi"
      limits:
        memory: "100Mi"
  - name: redis
    image: redis:7.4-alpine
    resources:
      requests:
        memory: "150Mi"
      limits:
        memory: "150Mi"
```

Pod Level Resources


```yaml
apiVersion: v1
kind: Pod
metadata:
  name: cool-app-pod
spec:
  resources:
    requests:
      memory: "250Mi"
    limits:
      memory: "250Mi"
  containers:
  - name: httpd
    image: httpd:2.4-alpine
  - name: redis
    image: redis:7.4-alpine
```

Future directions

SIG Node

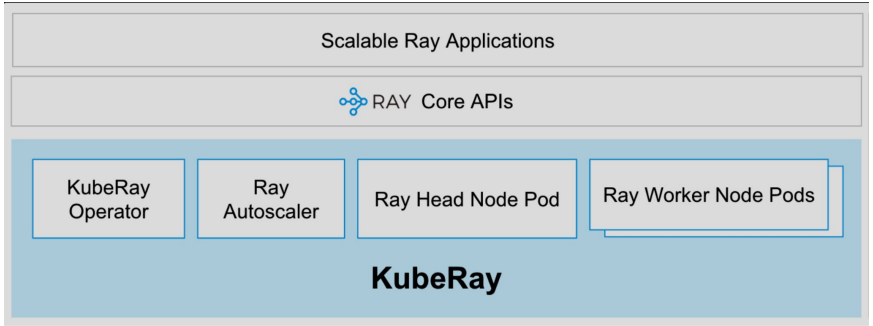
Shifting SIG-Node focus, from infra-centric to workload-centric!

New workload types

Workload changes, SIG Node is changing with it.

KubeRay:

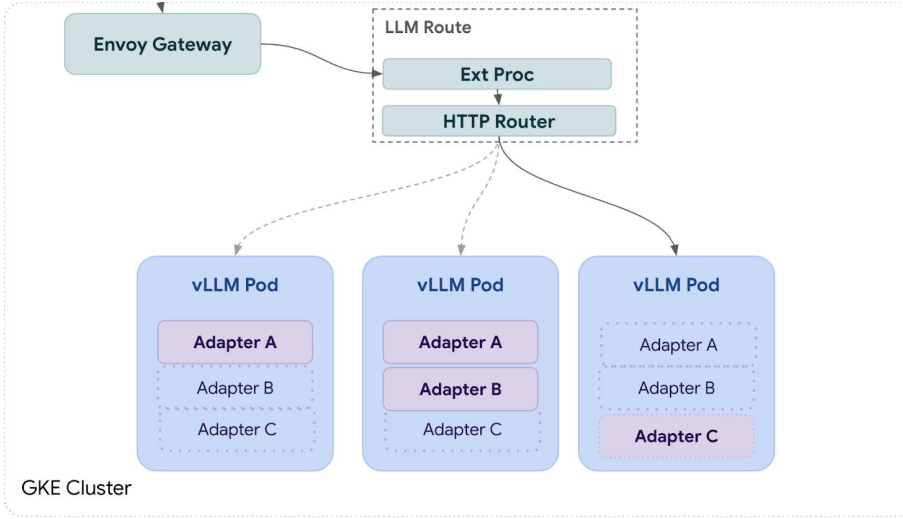
- “Node Pods” is a host for tasks
- K8s has no visibility inside Node Pods



From: Ray on Kubernetes docs

LLM instance Gateway:

- Model server is a new node
- Adapter is a new Pod



From: Kubernetes LLM Instance Gateway PoC Design

Trends

- Pods are not cattle, they are pets
  - Nodes and Pods are expensive to recreate
  - Long and reliable graceful termination
  - Pods are expensive to start
- Pods are dynamic
  - Pods are resizable
  - Flexible resources balancing between containers
  - Some DRA devices are Pod-scoped
- More scenarios for cross-node coordination and orchestration
  - Gang scheduling
  - Pod failure policies with in-place restart

How to get engaged

What is a contribution?

Code Contributions (ordered by priority):

- **Test coverage:** Write unit tests, integration tests, and end-to-end tests.
- **Bug Fixing:** Identify and address reported issues by submitting bug fixes.
- **Feature Implementation:** Contribute new functionalities (via KEPs).
- **Code Reviews:** Review and provide feedback on code changes.

Non-Coding Contributions:

- **Documentation:** Improve existing documentation, create new guides and tutorials.
- **Community Engagement:** Participate in discussions, questions, and help newcomers.
- **User Experience (UX):** Contribute to user experience and overall usability of the project.
- **Translation:** Translate existing documentation and resources into different languages.
- **Event Organization:** Help organize and participate in conferences, meetups, workshops.

Why SIG-Node?

- **Foundational and essential:** Contributing to the core components of Kubernetes builds a strong understanding of the entire ecosystem.
- **Diverse opportunities:** SIG-Node offers a wide range of contributions, from testing to feature development, allowing you to find your niche.
- **Supportive community:** Experienced members in SIG-Node are known for their friendly guidance, making it easier for newcomers to learn and contribute.
- **Direct impact:** By contributing to the foundation of Kubernetes, you directly influence the user experience of this critical technology.
- **Growth potential:** SIG-Node provides opportunities for both beginners and experienced individuals to contribute and grow their skills within the project.

Where to start?

- https://www.kubernetes.dev/
- Kubernetes Contributor Playground (link)
- SIG-Node main page (link)
- Community meetings
  - SIG-Node weekly meeting (link)
  - SIG-Node weekly CI/Triage meeting (link)
- Working groups
  - WG Batch (link)
  - WG Serving (link)
  - WG Policy (link)
  - WG Structured Logging (link)
  - WG Sidecar (link)
- Mentoring (link)

Feedback