




# One Recipe, Five Clouds, One Query

## Portable GPU Configuration with a Queryable Audit Layer

Christopher Haar (Upbound) & Yuan Chen (NVIDIA)



THE PROBLEM

# Running GPU clusters reliably is hard

KubeCon CloudNativeCon North America 2026

Services × GPUs × operating systems × workloads: more combinations than one team can validate.

### Version compatibility

Kernels, drivers, runtimes, operators, and Kubernetes must align.

### Tribal knowledge

Known-good settings live in runbooks and internal pipelines.

### Expensive failures

One mismatch can disrupt a GPU fleet and take days to diagnose.

| | |
|---|---|
| **300+** | configuration values in a fully tuned training cluster |
| **16+** | recipe components (training example) |
| **5** | component changes, 20 new values, and 2 value changes: training → inference |
| **2 + 20** | component + value changes, H100 → GB200 on GKE (TCPXO → RDMA networking) |
| **4 + 13** | component + value changes, EKS → GKE (GB200 training) |

# What is AICR?

KubeCon CloudNativeCon North America 2026

**AI Cluster Runtime (AICR): an open-source engine and CLI for reproducible GPU cluster configurations.**



### Optimized

Tuned for specific hardware, service, OS, and workload



### Validation built in

Constraint checks and functional validation



### Reproducible

Pinned release + same inputs → repeatable configuration



### Composable

Layered overlays: one desired state, public or private cloud



### Verifiable

Release provenance, SBOMs, and optional bundle signatures



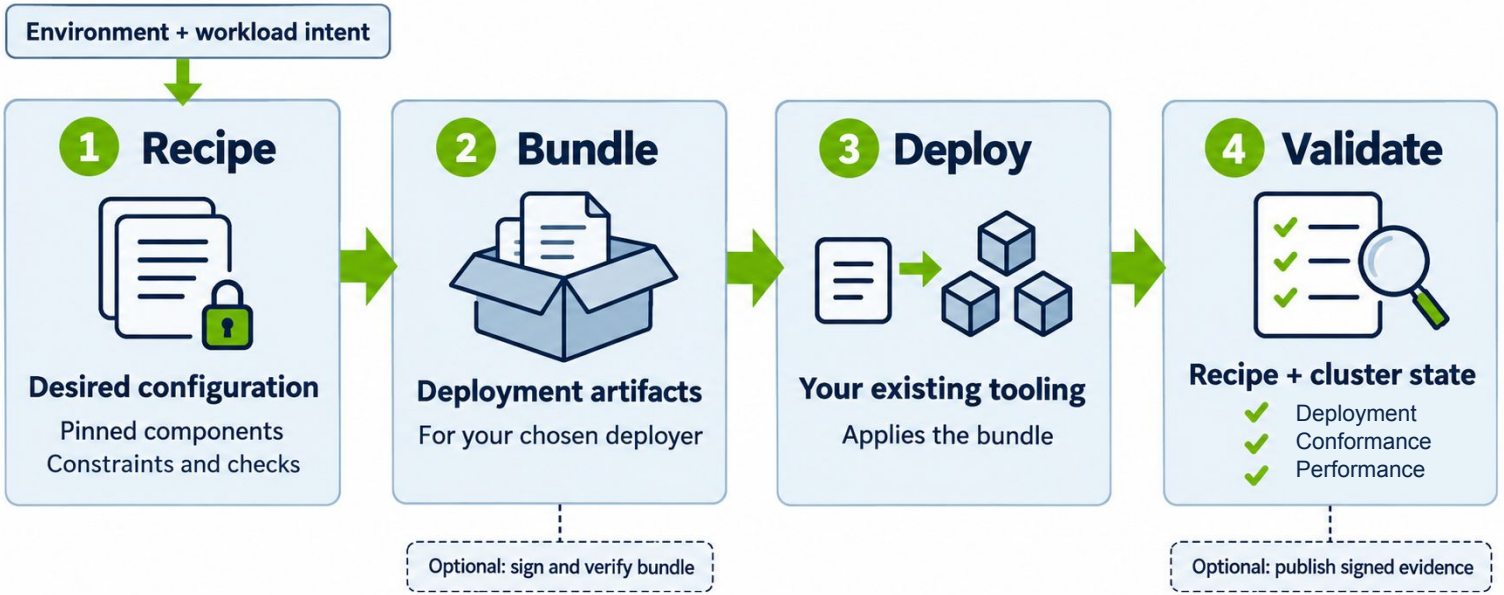
### Fits your tooling

Helm, Argo CD, Flux, Helmfile, Terraform, Fleet + REST API

# AICR workflow



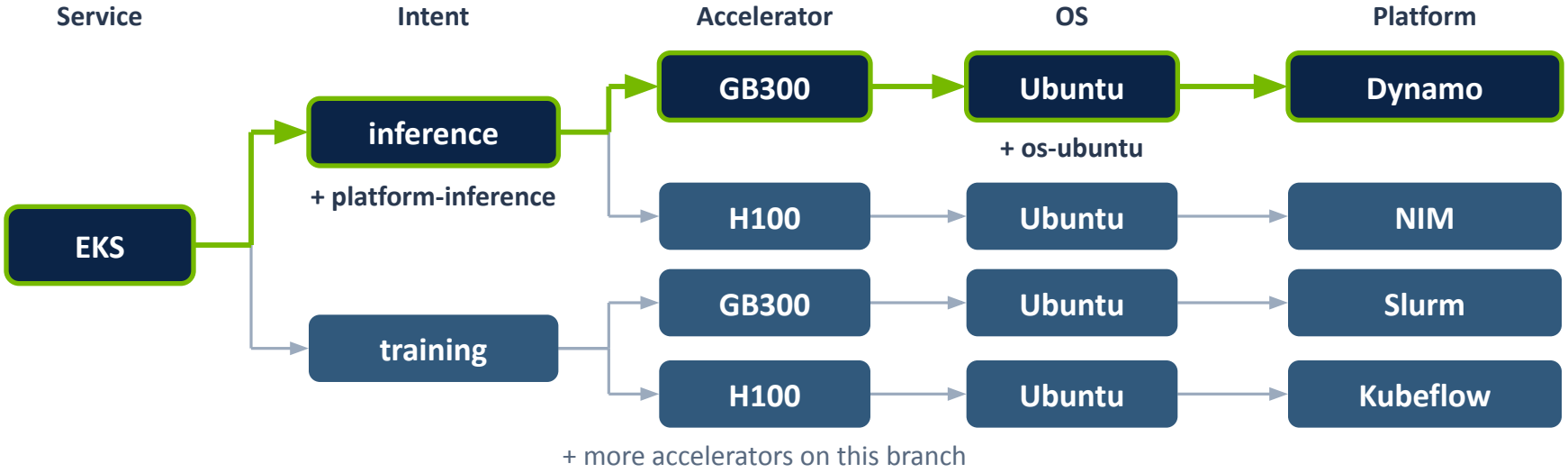
Use criteria or aicr snapshot for environment facts. Choose intent and an optional platform.



# Overlay hierarchy: the EKS branch



Inherit a base, then specialize. The green path is the GB300 inference demo.




**Demo overlay**

**`gb300-eks-ubuntu-inference-dynamo.yaml`**

`spec.base → gb300-eks-ubuntu-inference`

Selected branches; shared base layers, matching overlays, and mixins also contribute.

# From cluster facts to a recipe

KubeCon CloudNativeCon North America 2026

Example: EKS · GB300 · Ubuntu · inference · Dynamo

## Snapshot · observed

```yaml
fingerprint:
  service:
    value: eks
  accelerator:
    value: gb300
  os:
    value: ubuntu
```

**User choices**

inference · Dynamo

## Compose shared knowledge

Base
↓
EKS
↓
EKS inference
↓
GB300
↓
Ubuntu
↓
Dynamo

+ 3 matching overlays · 2 mixins

## Recipe · desired

```yaml
criteria:
  service: eks
  accelerator: gb300
  os: ubuntu
  intent: inference
  platform: dynamo
componentRefs:
- name: gpu-operator
  version: v26.7.2
```

**19 components**

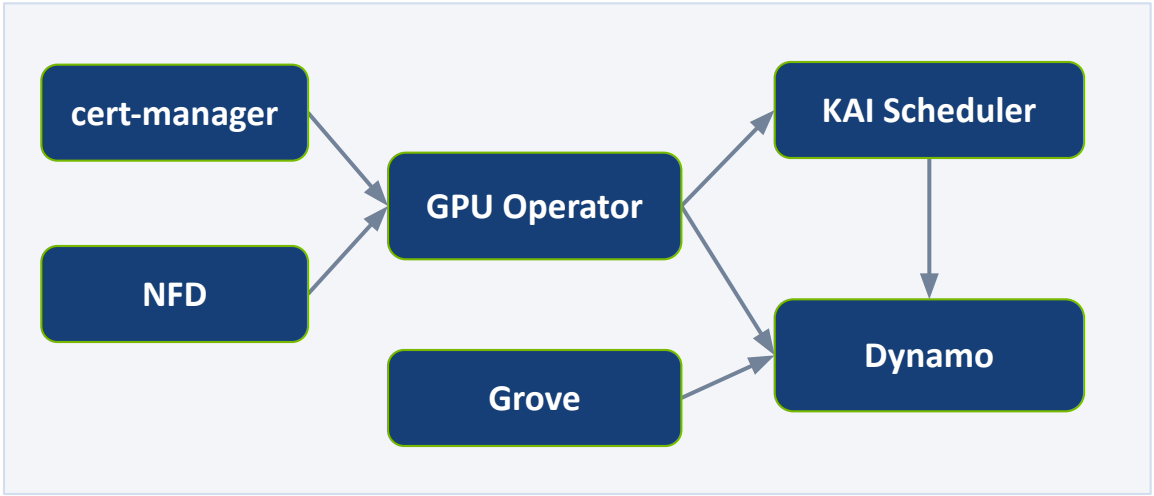
Shared layers compose one configuration for this workload.

# From recipe to bundle



aicr bundle turns the recipe into deployable artifacts, in dependency order.

## Recipe dependencies




## Bundle · deployable

**Helm example**

```
bundle/
├── recipe.yaml
├── checksums.txt
├── deploy.sh
├── 015-gpu-operator/
│   └── values.yaml
⋮
├── 017-kai-scheduler/
├── 018-dynamo-platform/
⋮
```

**Dependency ordering · Checksums · Optional signing**

Also renders for Argo CD · Flux · Helmfile · Terraform · Fleet

# From bundle to deployment



Verify the bundle, then deploy it with your tool, in the recipe's dependency order.

## Deploy with the tool you already use

| | |
|---|---|
| **Helm** | `./deploy.sh, in dependency order` |
| **Helmfile** | `helmfile apply` |
| **Argo CD** | `app-of-apps from Git or OCI` |
| **Flux** | `HelmReleases + kustomization.yaml` |
| **Terraform** | `terraform apply; depends_on keeps order` |
| **Fleet** | `gitrepo.yaml or helmops.yaml` |

## Deployment · ordered

**Helm example**

```
$ aicr verify ./bundle
Trust level: verified
$ ./deploy.sh
══ Pre-flight checks
══ Deploying components
├─ ✓ cert-manager
├─ ✓ gpu-operator
⋮
└─ ✓ prometheus-adapter
✓ 21/21 installed
```

**Verify first · Dependency order in every deployer · Your tool owns the rollout**

Next: validate the deployed cluster

# Is the cluster ready and deployed?

KubeCon CloudNativeCon North America 2026

Check prerequisites first, then confirm the recipe's components are installed and healthy.

**Readiness gate** — Snapshot readings meet recipe prerequisites
Kubernetes · OS · kernel

## Deployment

**Are the recipe's components installed and healthy?**

| | |
|---|---|
| `operator-health` | GPU Operator pods running and healthy |
| `expected-resources` | Declared resources exist and are healthy |
| `gpu-operator-version` | Version meets the recipe constraint |
| `check-nvidia-smi` | Driver and GPU access on available GPU nodes |

Next: Conformance

A failed or unmeasurable prerequisite stops validation before any validator Job runs.
**Then the phases run in order: deployment → conformance → performance.**

# Is the cluster AI-conformant?



CNCF AI Conformance checks exercise the capabilities AI workloads need.

## CNCF AI Conformance

| | |
|---|---|
| **GPU access** | `dra-support · secure-accelerator-access · driver-runtime-management` |
| **Scheduling** | `gang-scheduling` |
| **Observability** | `accelerator-metrics · ai-service-metrics` |
| **Scaling** | `pod-autoscaling · cluster-autoscaling` |
| **Serving** | `inference-gateway` |
| **Operator health** | `robust-controller · platform-health · gpu-operator-health` |

12 checks on main; the recording ran 11

Next: Performance ➡

CI reports · CNCF submission evidence

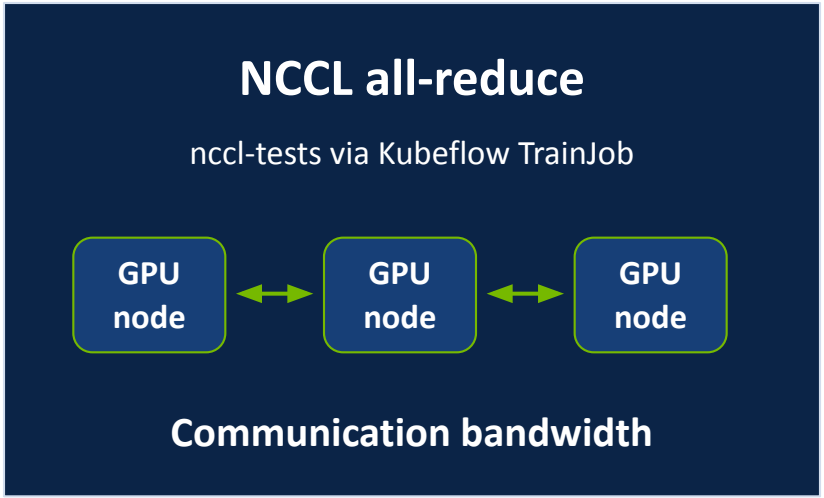
**NVKE 0.22.0 (NVIDIA Kubernetes Engine): CNCF AI Conformant for Kubernetes v1.36 · Validated with AICR recipes and tools**

# Does the cluster perform?

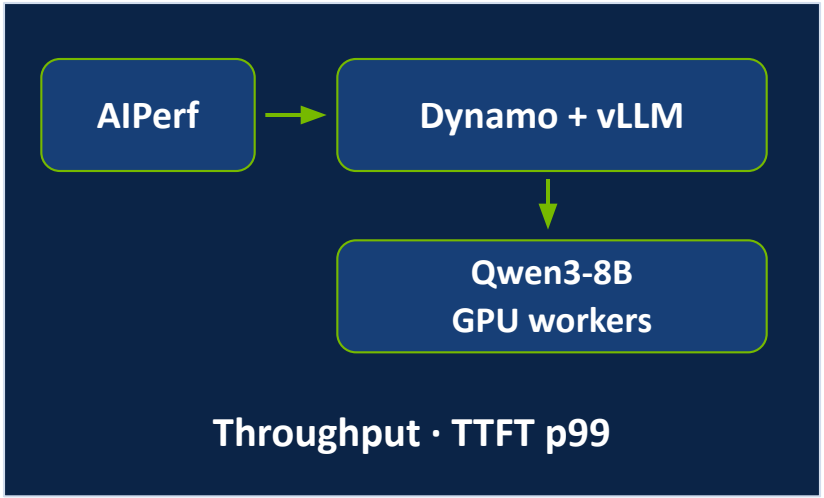


Run representative workloads against recipe-defined expectations.

## Training infrastructure




## Inference serving




**Configurable inference: model · concurrency · routing strategy**

# Day 2: the AICR Operator

KubeCon CloudNativeCon North America 2026

Opt-in coordination for existing GPU clusters, using the same recipes and bundles.

## Why an operator

**Changing cluster facts**
Re-resolve after cluster or AICR changes.

**Current-revision readiness**
Wait for the revision just applied to be ready.

**Upgrade control**
Approval when required; blocked moves stop.

**Deployment evidence**
Bind results to the rollout when enabled.

## Who does what

**AICR**
Discover, resolve, render, assess, and validate.

**AICR Operator**
Plan, request approval, and order steps.

**Argo CD**
Install charts and reconcile their resources.

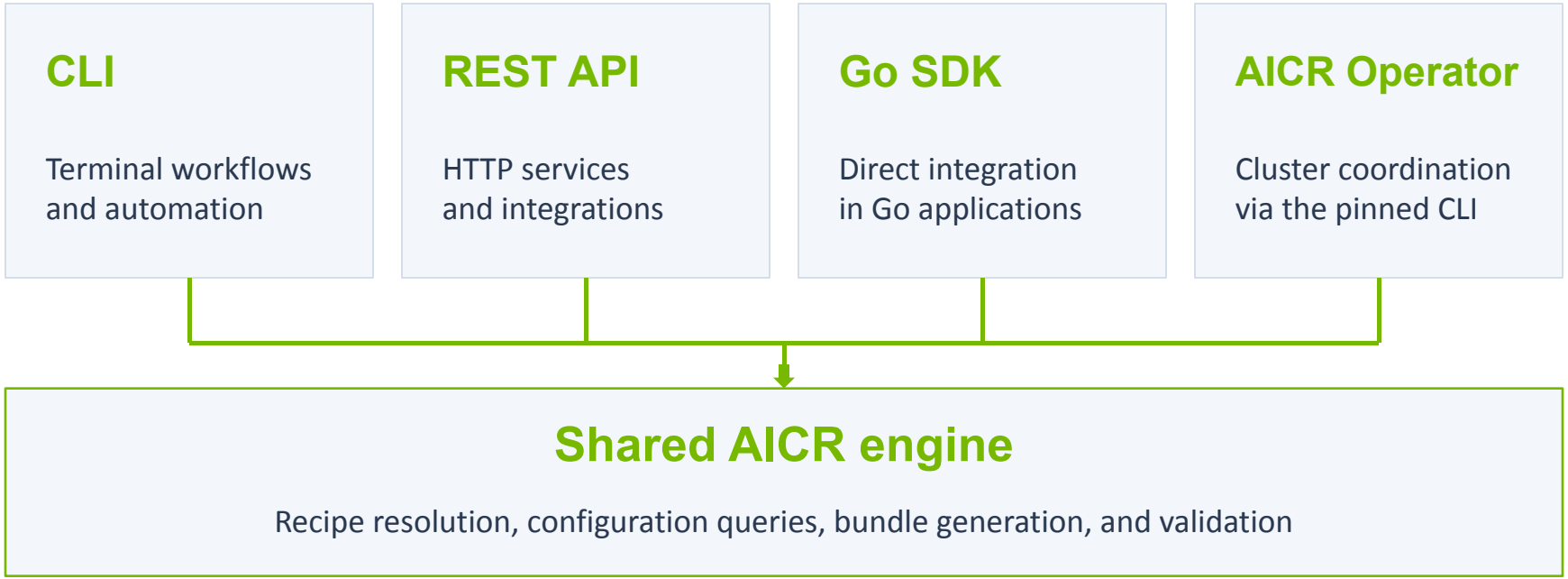
**Component operators**
Manage devices and component lifecycles.

**ClusterRuntime** ⟶ **Plan** ⟶ **RuntimeRollout** · assessed, gated, recorded

# Four ways to use AICR



Terminal, HTTP, Go, and Kubernetes workflows share the same AICR engine.



# Demo: end-to-end workflow



Recorded on a real EKS GB300 cluster. The deployment chapter replays dry-run output; validation was performed against the cluster.

1. **Capture and resolve**
   Snapshot the cluster; choose intent and platform; generate the recipe.

2. **Bundle and deploy**
   Generate and sign a Helm bundle, verify it, then replay deploy.sh dry-run output.

3. **Validate**
   Run deployment, conformance, and performance checks, and produce evidence.

Yuan · recording 2:13 · then live: Nemotron 3 Ultra deployment and chat

1/6  Install aicr

DEMO

# Demo: Nemotron 3 Ultra, live

KubeCon CloudNativeCon North America 2026

Live: NVIDIA Nemotron 3 Ultra, served by Dynamo + vLLM on the AICR-built GB300 stack.




**The model**
Hybrid Mamba–Transformer MoE.

**Total vs. active parameters**
550B total; about 55B active per token.

**Deployment footprint**
~329 GiB NVFP4 weights · 256K served context.

NVIDIA Nemotron 3 Ultra model card · supports up to 1M tokens

Backup video of this demo (52 s)


KubeCon
CloudNativeCon
North America 2026
UP NEXT
Deep dive: the queryable audit layer

AUDIT LAYER

# Four questions for an auditable cluster

KubeCon CloudNativeCon North America 2026

Inspect desired configuration, observed drift, upgrade guidance, and the evidence behind a result.

## What should this cluster run?

```
aicr query
```

Resolve values; compare environments.

## What changed since the baseline?

```
aicr diff
```

Field-by-field drift between two cluster snapshots.

## Is this move safe?

```
aicr upgrade-check
```

Guidance from recorded transitions.

## What can you verify?

```
Origin      cosign / gh attestation
Integrity   aicr verify
Results     aicr evidence verify
```

Signatures attribute results.
Validation checks cluster health.

AUDIT LAYER

# What should this cluster run?

KubeCon CloudNativeCon North America 2026

Resolve configuration for your criteria, then select the value you need.

1. **Describe**
   EKS · GB300 · inference
2. **Resolve**
   Compose the recipe and merge component values.
3. **Select**
   GPU Operator → driver.version

**Resolved answer**

**GPU Operator**
`driver.version`
**580.173.02**
Recipe value ≠ installed driver

```
$ aicr query --service eks --accelerator gb300 --intent inference \
  --selector 'components.gpu-operator.values.driver.version'
```

Resolved configuration · CLI or REST API

AUDIT LAYER

# One query, five clouds

KubeCon CloudNativeCon North America 2026

GB300 inference examples: environment-specific components and GPU ownership.

## EKS

Ubuntu

**GPU Operator installs the driver**

EFA networking
EBS CSI storage

## GKE

COS · gke-default

**GKE supplies driver and device plugin**

Operator driver and plugin disabled

## AKS

azure-managed

**Node image supplies driver and toolkit**

NVIDIA Network Operator

**Also on managed clouds: OKE · LKE**

Other environments: OpenShift · RKE2 · NVKE

Not every combination has a recipe.

AUDIT LAYER

# What changed since the baseline?

KubeCon CloudNativeCon North America 2026

Compare two observed snapshots to reveal added, removed, and modified fields.

**Saved baseline**
Reference cluster state

**Fresh snapshot**
Current observed state

**Selected recorded changes**

**GPU Operator image**
`v26.7.1 → v26.7.2`

**NVSentinel labeler image**
`v1.25.0 → v1.28.0`

**Default StorageClass name**
`Not recorded → ebs-csi-default-sc`

**CI gate: --fail-on-drift**

```
aicr diff --baseline baseline.yaml --target now.yaml --fail-on-drift
```

Validate the changed cluster against the recipe.

# Is this move safe?



Assess the move against recorded transitions and deployer-specific steps.

1. **Compare**
   Current: recipe, bundle, or --from cluster
   Target: recipe or bundle
2. **Assess**
   Match transition records and deployer steps.
3. **Review**
   Read verdicts and required actions before applying.

**GPU Operator**
`v26.7.0 → v26.7.1` **safe**

**NVSentinel**
`v1.20.0 → v1.25.0` **blocked**

No recorded transition starts from NVSentinel v1.20.0.

```
$ aicr upgrade-check --from old.yaml --to new.yaml --deployer helm
```

Any verdict other than safe fails the default gate.

AUDIT LAYER

# Evidence: collect, sign, and submit



Validation writes evidence, a signature attributes it, and a reviewed pointer publishes it.

1. **Collect**
   **`--emit-attestation`** writes a local evidence bundle of the results.

2. **Sign and push**
   **`--push`** signs it with Sigstore and publishes it to an OCI registry.

3. **Submit**
   A reviewed pointer and signer entry put it on validation.aicr.run.

Create, sign, and push to OCI:

```
$ aicr validate -r recipe.yaml \
  --emit-attestation ./out \
  --push ghcr.io/<org>/evidence
```

Only allowlisted signers count. Agreement and disagreement stay visible; results are never averaged.

# Demo: inspect the audit trail



The same real cluster · four commands, four answers.

1. **`aicr query`**
   Read resolved values and compare environments.

2. **`aicr diff`**
   Compare observed snapshots with a saved baseline.

3. **`aicr upgrade-check`**
   Assess recorded transitions and deployer-specific steps.

4. **`aicr evidence verify`**
   Verify local evidence and a published signed pointer.

1/4  What should this cluster run?  aicr query

AUDIT LAYER

# Live: the evidence portal



Live walkthrough: published results, their signers, and whether sources agree.

LIVE WALKTHROUGH

## validation.aicr.run

Follow along on your laptop or phone

KubeCon CloudNativeCon
North America 2026

UP NEXT

# Summary



ENVIRONMENTS

# What's supported today

KubeCon CloudNativeCon North America 2026

Overlay criteria · AICR v1.2.0 + main (November 7, 2026)

## 13 Kubernetes services shown

| | |
|---|---|
| Managed cloud | EKS · GKE · AKS · OKE · LKE · MK8s · CoreWeave CKS* |
| Distributions | OpenShift · RKE2* · k0s* · NVIDIA NVKE* |
| Self-managed | Bare metal · kind (dev/test) |

## 11 GPU accelerators

| | |
|---|---|
| Rubin | VR200* |
| Blackwell | GB300 · GB200 · B300 · B200 · RTX PRO 6000 |
| Hopper | H200 · H100 |
| Ampere / Ada | A100 · L40S · L40 |

## 5 Operating systems

Ubuntu · COS · Oracle Linux · Amazon Linux · SLES

*Talos (mixin; not counted)*

## 6 Deployer families

Helm · Helmfile · Argo CD · Flux · Terraform · Fleet

## 6 Workload platforms

| | |
|---|---|
| Training | Kubeflow · Slurm · Kueue · Run:ai*† |
| Inference | Dynamo · NIM |

** Preview. † On main; next release. Not every combination has a recipe. Argo CD has two formats: seven CLI formats across six deployer families.*

COMMUNITY

# AICR community by the numbers

KubeCon CloudNativeCon North America 2026

GitHub snapshot: November 8, 2026 · 111 human PR/issue authors, excluding bots.

**2,453**
**Pull requests** 2,181 merged

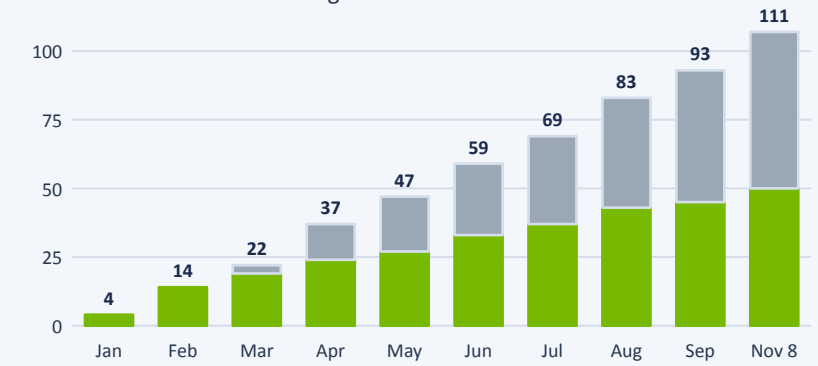
**1,263**
**Issues** 1,084 closed

**111**
**Contributors** PRs & issues

**54%**
**Outside NVIDIA**
GitHub org

## Contributor growth

Cumulative contributors · through Nov 8




**4 → 111** contributors since January

**~190** merged PRs/month, excluding bots

## 25 organizations*

Identified from contributors' GitHub profiles

Akamai · Amazon · CDW · ConfigHub · Cornell University · DaoCloud · Dynamia AI
Elastic · Expanso · F5 · Gats.dev · Microsoft · Mirantis · NVIDIA
Oracle Cloud · PTC · Red Hat · RexCode Digital · Salesforce · Sidero Labs · Solo.io
SUSE · Turf · Upbound · Viettel Cloud

** Self-reported public profiles. Individual contributions do not imply organizational endorsement.*

GET INVOLVED

# Use AICR, then help shape it

KubeCon CloudNativeCon North America 2026

The matrix is bigger than any one team. Use AICR, tell us what's missing, and help fill the gaps.

## Use AICR and tell us what's missing

Report gaps and issues, and share feedback and suggestions.

## Help wanted: coverage and features

- Validate existing recipes and submit signed evidence.
- Cover new platforms and GPUs: recipes, validation, and evidence.
- Build features, tools, and integrations.

## Close open issues

Fix bugs and improve the documentation.

**Explore AICR**

RESOURCES

# References and resources

KubeCon CloudNativeCon North America 2026

Everything shown today, and where to go next. Underlined items are links.

## Get started

- Repo: **github.com/NVIDIA/aicr**
- Docs: **docs.nvidia.com/aicr**
- **End-to-end tutorial**
- `curl -sfL https://get.aicr.run | bash -s --`
- `brew tap NVIDIA/aicr && brew install aicr`

## Validation and provenance

- Evidence portal: **validation.aicr.run**
- **CNCF Kubernetes AI Conformance**
- **Releases, SBOMs, and attestations**

## NVIDIA Technical Blog

- **DSX OS: configure, certify, operate**
- **AICR v1.0: open, stable, verifiable**
- **Vera Rubin: AI-ready on day one**
- **Validated runtime for NVIDIA Dynamo**
- **NVCRE: validate GPU cluster readiness**
- **Layered, reproducible recipes (intro)**

## Community

- **#aicr on Kubernetes Slack** (**slack.k8s.io**)
- **Roadmap**
- **Adopters**
- **Contributing guide**

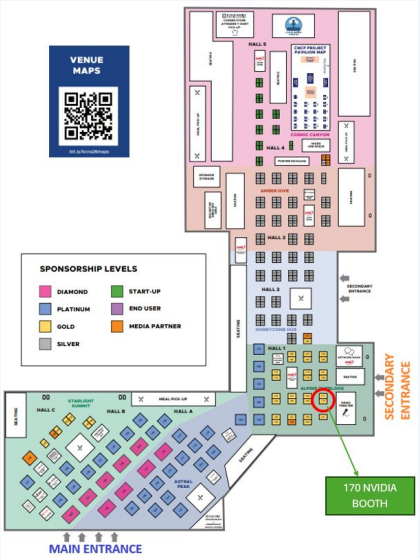
FEEDBACK · VISIT US

# Thank you!

KubeCon CloudNativeCon North America 2026

Come see us at the NVIDIA booth, and scan the code to rate this session.

## Visit NVIDIA at booth 170




### Booth 170

Hall 1, Alpine Overlook

Come by to learn more about AICR and the DSX Kubernetes open-source projects.

## Rate this session

One Recipe, Five Clouds, One Query: Portable GPU Configuration with a Queryable Audit Layer