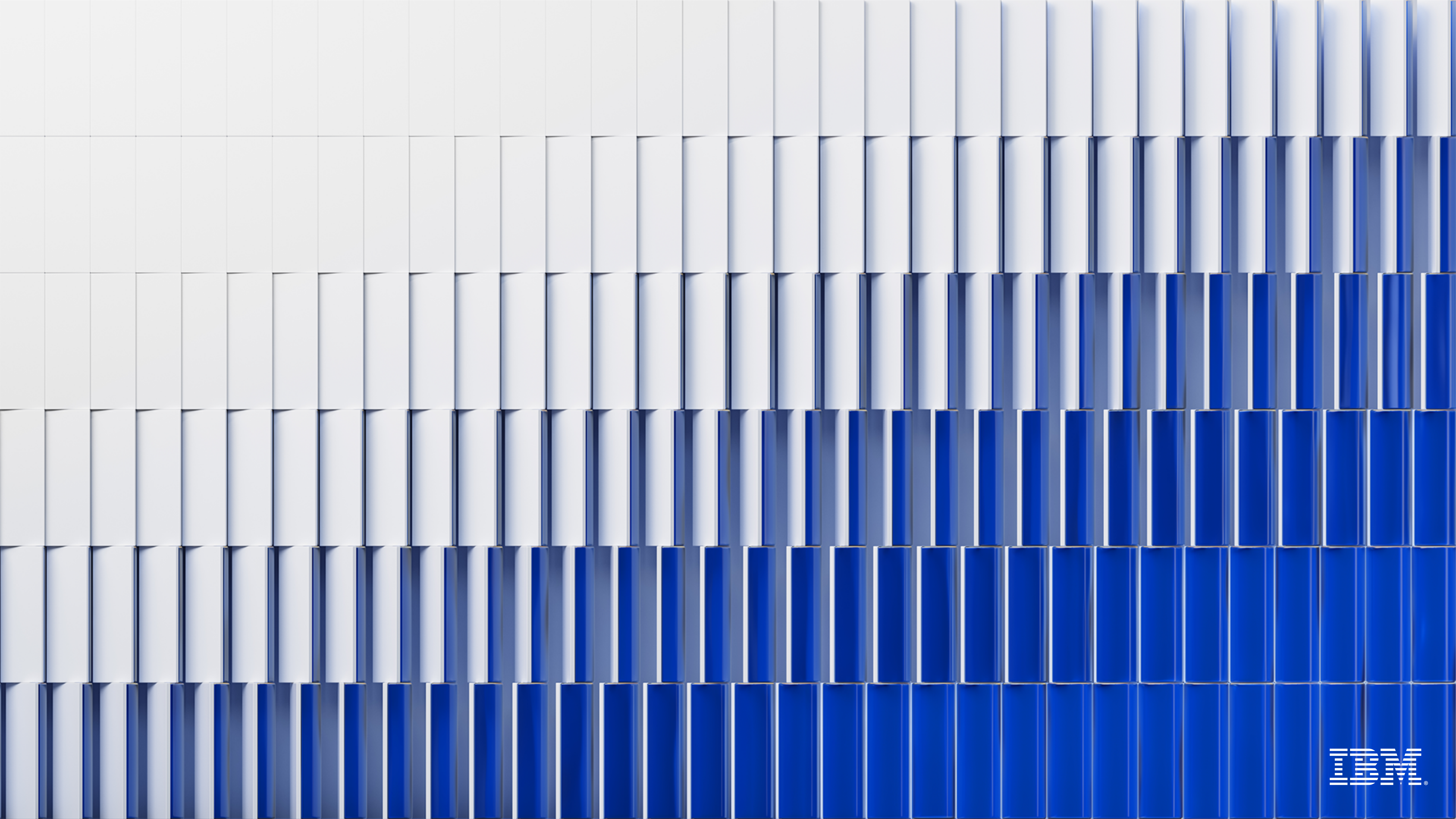




Taming the Wild West of Research Computing: How Policies Saved Us a Thousand Headaches

Alessandro Pomponio
Research Software Engineer

A bit of context

Accelerated Discovery

- 🧬 Accelerated Discovery explores the intersection between *Artificial Intelligence, Hybrid Cloud, and Quantum Computing*.
- 🚀 It aims to *drastically increase how quickly we can discover solutions to tackle today's most urgent problems*.
- 🤝 We collaborate with researchers coming from many different backgrounds: AI, HPC, chemistry, mathematics, and more.
- 🛠️ *My team manages two bare-metal OpenShift clusters for our colleagues to run their workloads on.*



Challenges in Research Computing

Researchers ≠ Kubernetes Experts

Researchers used to HPC environments often create interactive pods to run GPU workloads.

Pods are left running long after training ends, leaving GPUs idle.

Bursty Workloads & Resource Contention

Workloads are bursty, spiking near conference deadlines.

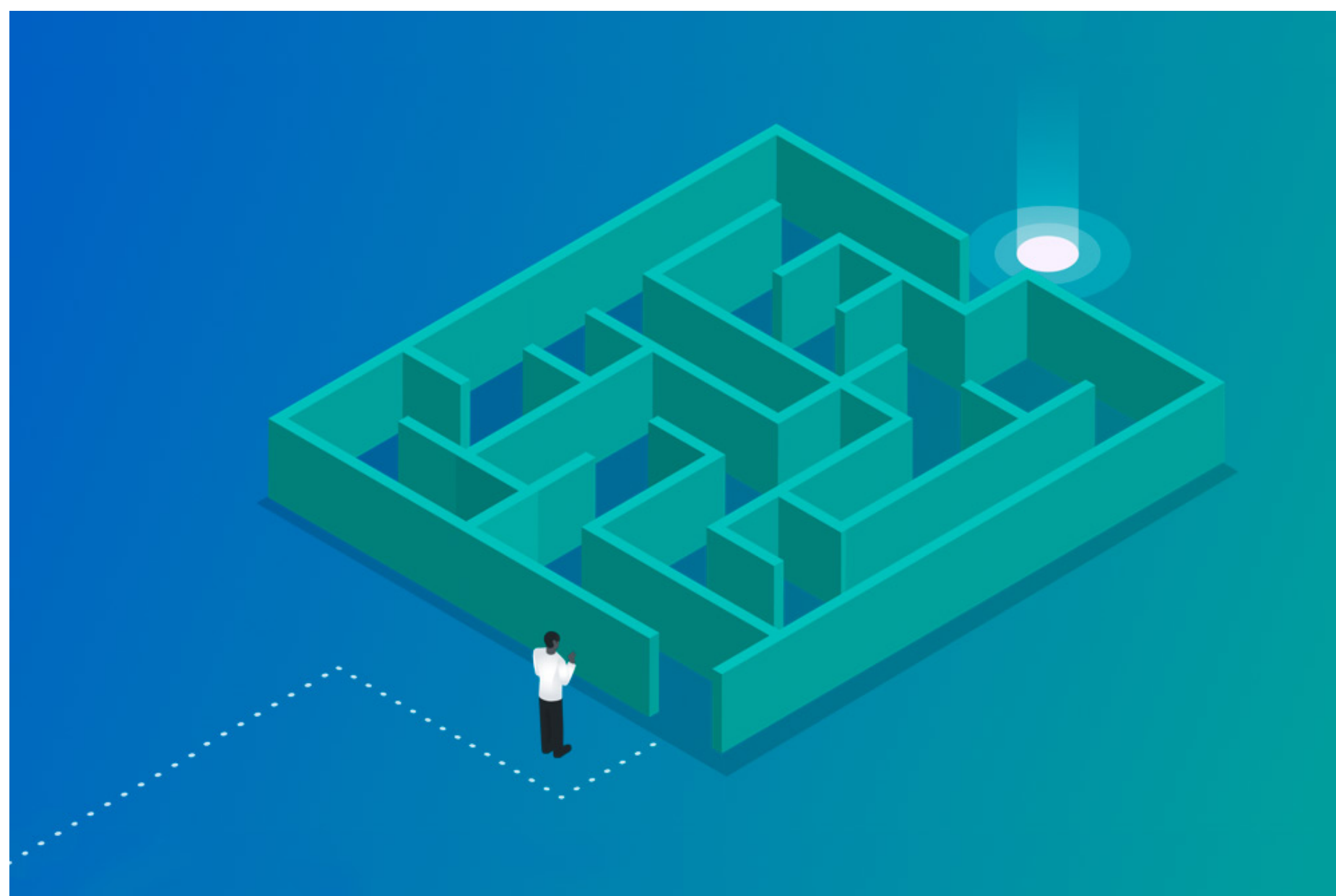
Large CPU jobs often spill over onto GPU nodes, causing scheduling issues.

The side gig problem

Manual processes left admins handling every user request, *twice*.

The volume of requests made it increasingly difficult to focus on our own research.

What's the plan?



Shift namespace-management decisions to users

Introduce automation and tooling that enables self-service namespace management even for non-experts users.

Prevent CPU jobs from running amok

Introduce safeguards to prevent CPU-only workloads from overwhelming the cluster or interfering with GPU scheduling.

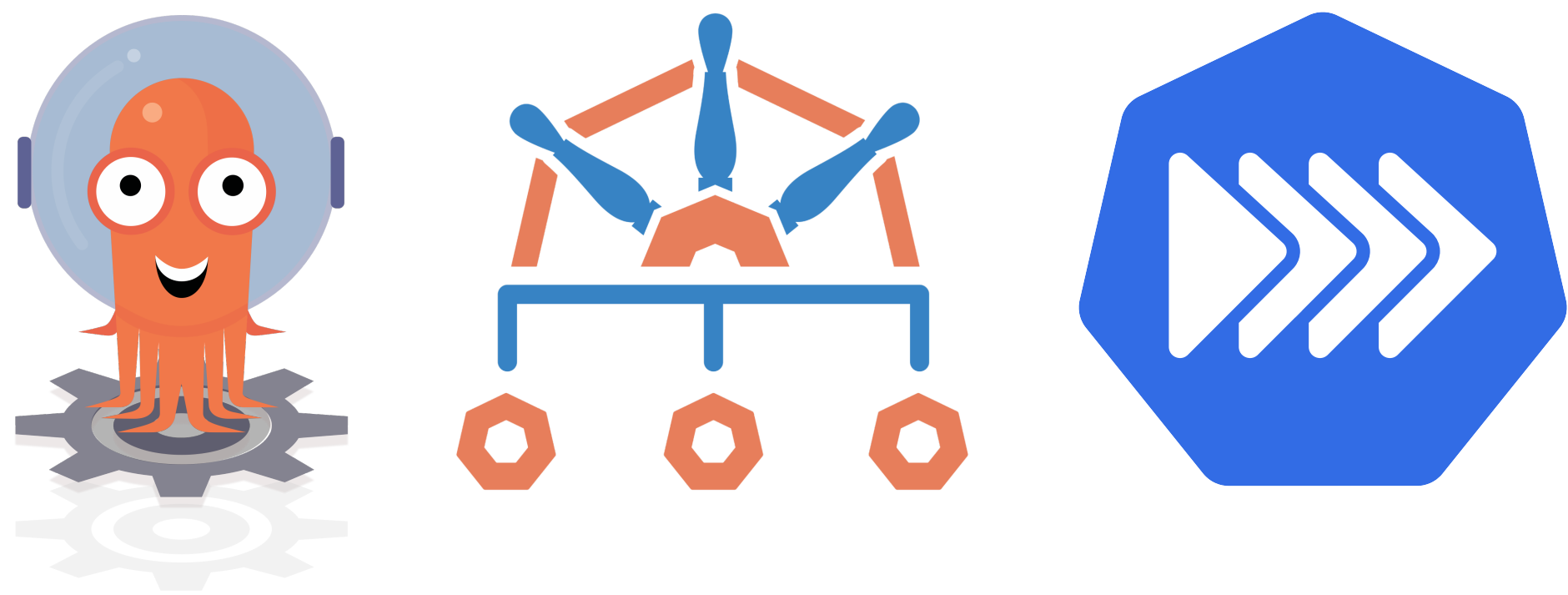
Discourage interactive GPU usage

Prevent the creation of long-lived, interactively accessed pods in favour of more efficient, Job-based workflows that free up (GPU) resources once they're complete.

Make GPU access fair

Improve resource allocation to ensure GPU access is fair, providing a more predictable, HPC-like experience that minimises contention.

What's the plan?



Shift namespace-management decisions to users

Introduce automation and tooling that enables self-service namespace management even for non-experts users.

Prevent CPU jobs from running amok

Introduce safeguards to prevent CPU-only workloads from overwhelming the cluster or interfering with GPU scheduling.

Discourage interactive GPU usage

Prevent the creation of long-lived, interactively accessed pods in favour of more efficient, Job-based workflows that free up (GPU) resources once they're complete.

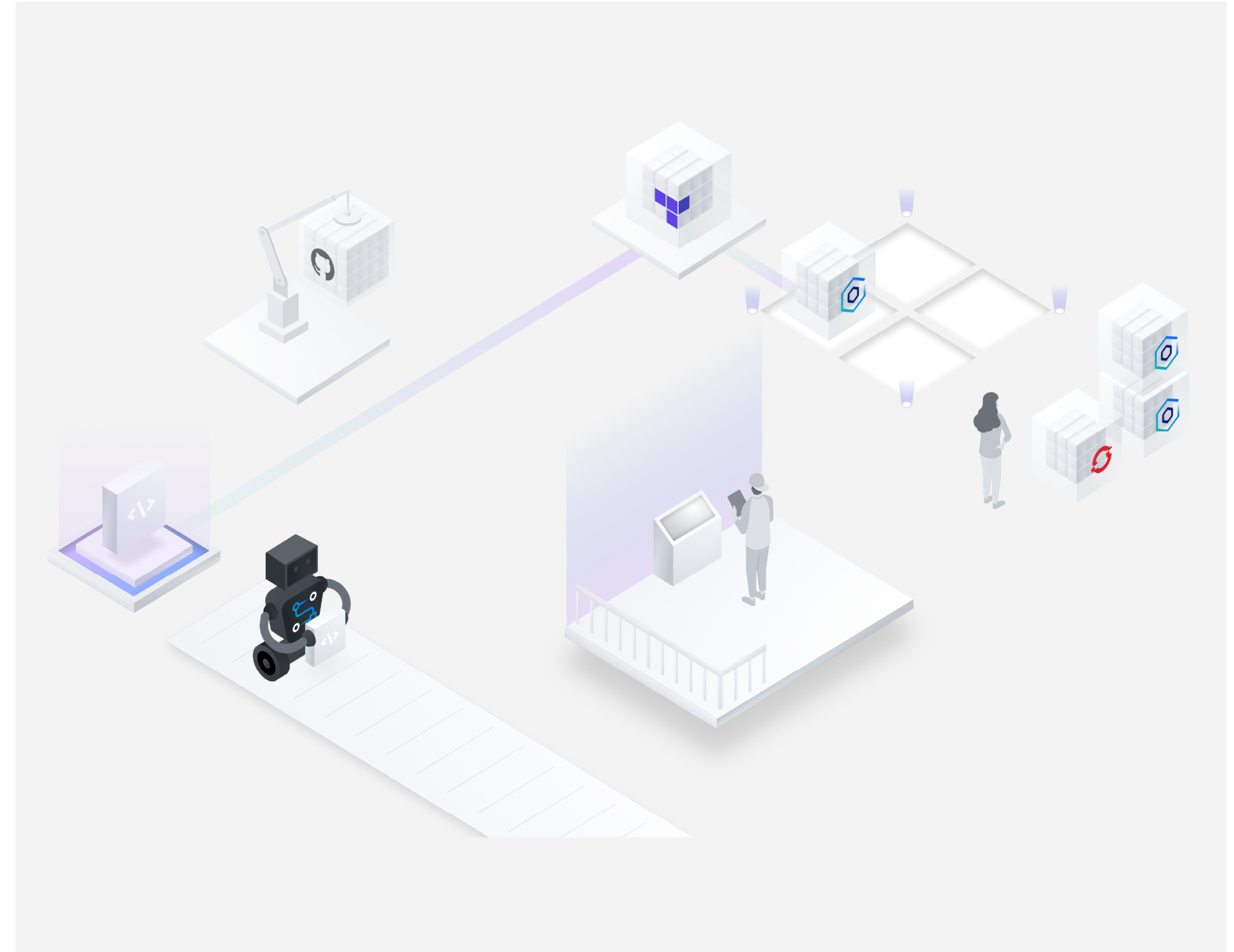
Make GPU access fair

Improve resource allocation to ensure GPU access is fair, providing a more predictable, HPC-like experience that minimises contention.

Automating

Our goals

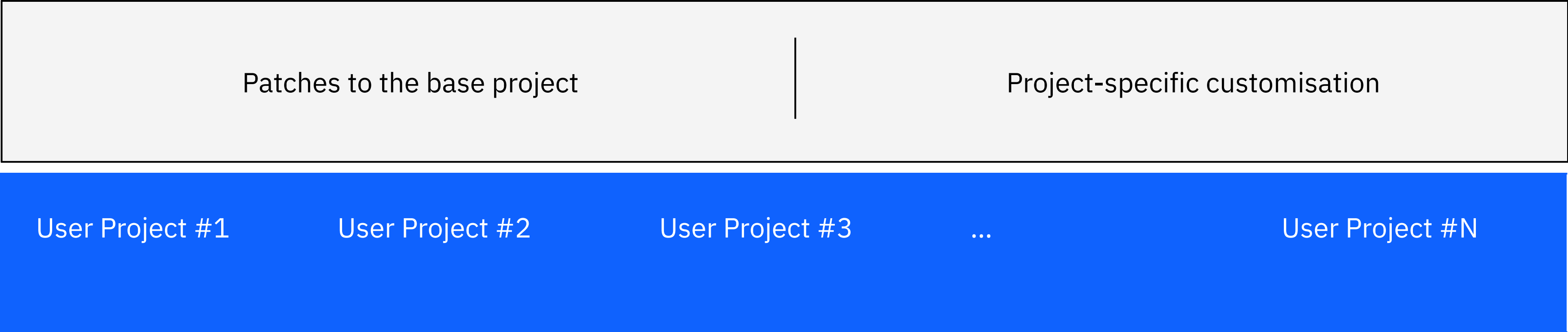
- ***Let users own their namespaces.*** Users submit change requests for admins review and approval.
- ***Eliminate manual deployments.*** Automation saves time, removes guesswork, and guarantees consistency.
- ***Be easy to use.*** It should not impose any toll on non-expert users. Admins should take care of the heavy lifting.
- ***Minimise maintenance overhead.*** Configurations and tooling should be easy to set up, maintain, and extend.



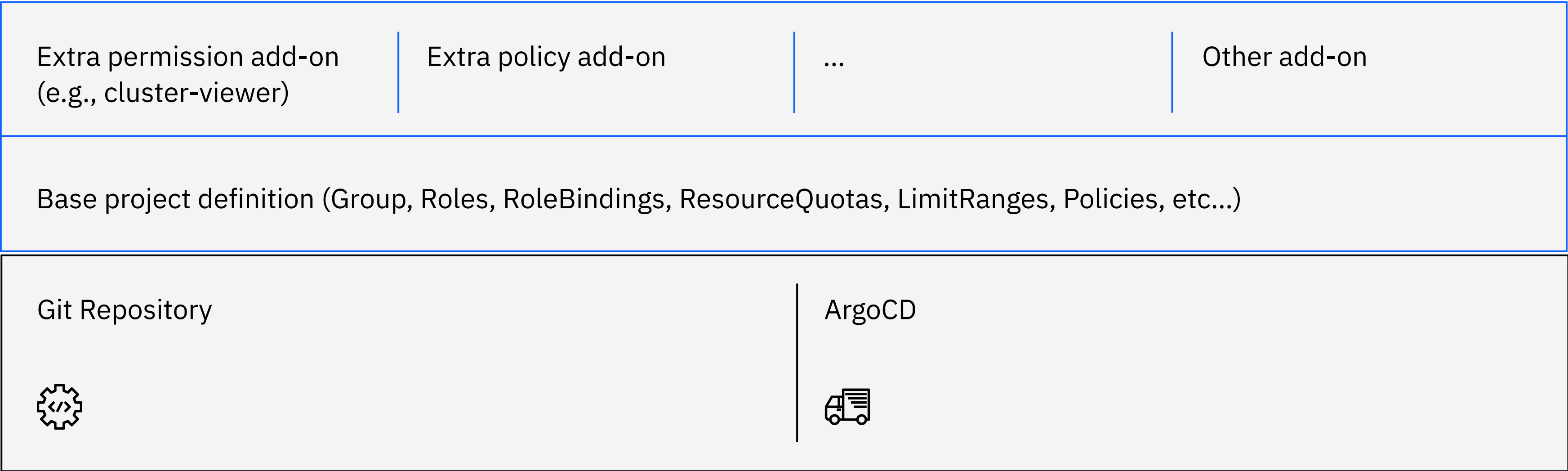
Automating

The idea

Users



Admins



Automating

Minimising maintenance

- Install your GitOps tool of choice in a way that minimises overhead for configuring, patching, etc.

Project: openshift-gitops-operator ▼

Installed Operators > Operator details



Red Hat OpenShift GitOps

1.18.1 provided by Red Hat Inc

Actions ▼

Details

YAML

Subscription

Events

All instances

Argo CD

NamespaceManagement

AnalysisRun

AnalysisTemplate

Application

ApplicationS

Subscription details

Update channel ?

latest 

Update approval ?

Automatic 

Upgrade status

1 installed

0 installing

Up to date



Name

openshift-gitops-operator

Namespace

NS

openshift-gitops-operator

Labels

operators.coreos.com/openshift-gitops-operator.openshift-gitops-operator

Edit 

Created at

Nov 28, 2024, 4:04 PM

Installed version

CSV

openshift-gitops-operator.v1.18.1

Starting version

openshift-gitops-operator.v1.14.2

CatalogSource

CS

redhat-operators

Healthy 

InstallPlan

IP

install-4p594



KyvernoCon

NORTH AMERICA

Automating

Extensibility and ease-of-use

- ***Kustomize is easy for both users and admins.*** Keep it simple and use a LEGO-style approach.
- ***Have everything ready.*** Require users to change as little as possible. **Provide a fully-fledged example to copy/paste from.**
- ***Provide a set of useful add-ons*** that users can use by adding just one line in their kustomization.yaml file.

```
✓ project-provisioning ●
  > argo ●
  ✓ bases
    ✓ addons
      ✓ cluster-reader
        ! clusterrolebinding.yaml
        ! kustomization.yaml
      > codeflare
      > mcad
      > policies
      > ray
    ✓ base-project
      ! group.yaml
      ! job-affinity-rules.yaml
      ! kustomization.yaml
      ! limitrange.yaml
      ! namespace-admin-rolebindin...
      ! resourcequota.yaml
    ✓ replacements
      ! add-namespace-prefix-to-na...
```

Automating

Extensibility and ease-of-use

- *Kustomize is easy for both users and admins.* Keep it simple and use a LEGO-style approach.
- *Have everything ready.* Require users to change as little as possible. **Provide a fully-fledged example to copy/paste from.**
- *Provide a set of useful add-ons* that users can use by adding just one line in their kustomization.yaml file.

```
1  apiVersion: user.openshift.io/v1
2  kind: Group
3  metadata:
4    name: group
5  users: []
```

```
1  apiVersion: v1
2  kind: LimitRange
3  metadata:
4    name: default-resources-for-containers
5  spec:
6    limits:
7      - default:
8          cpu: 1
9          memory: 2Gi
10     defaultRequest:
11       cpu: 500m
12       memory: 512Mi
13     type: Container
```

```
1  resources:
2    - group.yaml
3    - namespace-admin-rolebinding.yaml
4    - resourcequota.yaml
5    - limitrange.yaml
6  # Policies
7    - job-affinity-rules.yaml
```

Automating

Extensibility and ease-of-use

- *Automate the creation of Argo Applications* using [Git Directory Generators](#) and [ApplicationSets](#).
- *Prevent drift* by enabling [self-healing](#) and [automated pruning](#) (⚠).

```
1  apiVersion: argoproj.io/v1alpha1
2  kind: ApplicationSet
3  metadata:
4    name: user-projects
5  spec:
6    goTemplate: true
7    goTemplateOptions: ["missingkey=error"]
8    syncPolicy:
9      automated:
10        prune: true
11        selfHeal: true
12    generators:
13      - git:
14          repoURL: ssh://git@example.com/org/repo
15          revision: HEAD
16          directories:
17            - path: project-provisioning/user-projects/*
18    template:
19      metadata:
20        name: "{{.path.basename}}"
21      spec:
22        project: "provisioning"
23        source:
24          repoURL: ssh://git@example.com/org/repo
25          targetRevision: HEAD
26          path: "{{.path.path}}"
27        destination:
28          server: https://kubernetes.default.svc
29          namespace: "{{.path.basename}}"
30        syncPolicy:
31          syncOptions:
32            - CreateNamespace=true
33          automated:
34            prune: true
35            selfHeal: true
```

Automating

Extensibility and ease-of-use

- *Automate the creation of Argo Applications* using [Git Directory Generators](#) and [ApplicationSets](#).
- *Prevent drift* by enabling [self-healing](#) and [automated pruning](#) (⚠).

```
1  apiVersion: argoproj.io/v1alpha1
2  kind: ApplicationSet
3  metadata:
4    name: user-projects
5  spec:
6    goTemplate: true
7    goTemplateOptions: ["missingkey=error"]
8    syncPolicy:
9      automated:
10        prune: true
11        selfHeal: true
12    generators:
13      - git:
14          repoURL: ssh://git@example.com/org/repo
15          revision: HEAD
16          directories:
17            - path: project-provisioning/user-projects/*
18    template:
19      metadata:
20        name: "{{.path.basename}}"
21      spec:
22        project: "provisioning"
23        source:
24          repoURL: ssh://git@example.com/org/repo
25          targetRevision: HEAD
26          path: "{{.path.path}}"
27        destination:
28          server: https://kubernetes.default.svc
29          namespace: "{{.path.basename}}"
30        syncPolicy:
31          syncOptions:
32            - CreateNamespace=true
33          automated:
34            prune: true
35            selfHeal: true
```

Automating

Extensibility and ease-of-use

- *Automate the creation of Argo Applications* using [Git Directory Generators](#) and [ApplicationSets](#).
- *Prevent drift* by enabling [self-healing](#) and [automated pruning](#) (⚠).

```
1  apiVersion: argoproj.io/v1alpha1
2  kind: ApplicationSet
3  metadata:
4    name: user-projects
5  spec:
6    goTemplate: true
7    goTemplateOptions: ["missingkey=error"]
8    syncPolicy:
9      automated:
10        prune: true
11        selfHeal: true
12    generators:
13      - git:
14        repoURL: ssh://git@example.com/org/repo
15        revision: HEAD
16        directories:
17          - path: project-provisioning/user-projects/*
18      template:
19        metadata:
20          name: "{{.path.basename}}"
21        spec:
22          project: "provisioning"
23          source:
24            repoURL: ssh://git@example.com/org/repo
25            targetRevision: HEAD
26            path: "{{.path.path}}"
27          destination:
28            server: https://kubernetes.default.svc
29            namespace: "{{.path.basename}}"
30          syncPolicy:
31            syncOptions:
32              - CreateNamespace=true
33          automated:
34            prune: true
35            selfHeal: true
```

Automating Results

user-projects

ad-fm-geospatial

agbglobal

ai4spectra

aimc-profiling

ap-testing

! group.yaml

! kustomization.yaml

! resourcequota.yaml

climate-wrf

cotune

cybersecurity

argo

v3.1.8+becb020

Applications

Settings

User Info

Documentation

Application filters

★

Favorites Only

SYNC STATUS

⌚

Unknown

0

✓

Synced

33

⬆

OutOfSync

0

HEALTH STATUS

🔄

Progressing

0

⏸

Suspended

0

♥

Healthy

33

💔

Degraded

0

👾

Missing

0

?

Unknown

0

LABELS

LABELS

CLEAR

PROJECTS

Applications

+ NEW APP

↻ SYNC APPS

🔄 REFRESH APPS

🔍 Search applications...

📄

🗑

Log out

Sort: synchronized ▾ Items per page: all ▾

ad-fm-geospatial

Project: provisioning

Labels:

Status:

♥ Healthy

✓ Synced

Repository: ssh://git@

Target Revi... HEAD

Path: project-provisioning/user-projects/ad-fm-geospa...

Destination: in-cluster

Namespace: ad-fm-geospatial

Created At: 12/11/2024 11:48:59 (a year ago)

Last Sync: 10/21/2025 13:40:40 (11 days ago)

↻ SYNC

🔄 REFRESH

✖ DELETE

slinky

Project: provisioning

Labels:

Status:

♥ Healthy

✓ Synced

Repository: ssh://git@

Target Revi... HEAD

Path: project-provisioning/user-projects/slinky

Destination: in-cluster

Namespace: slinky

Created At: 10/05/2025 10:52:03 (a month ago)

Last Sync: 10/05/2025 10:52:11 (a month ago)

↻ SYNC

🔄 REFRESH

✖ DELETE

discovery-dev

Project: provisioning

Labels:

Status:

♥ Healthy

✓ Synced

Repository: ssh://git@

Target Revi... HEAD

Path: project-provisioning/user-projects/discovery-dev

Destination: in-cluster

Namespace: discovery-dev

Created At: 02/07/2025 10:13:33 (9 months ago)

Last Sync: 08/07/2025 10:43:36 (3 months ago)

↻ SYNC

🔄 REFRESH

✖ DELETE

agbglobal

Project: provisioning

Labels:

Status:

♥ Healthy

✓ Synced

Repository: ssh://git@

Target Revi... HEAD

Path: project-provisioning/user-projects/agbglobal

Destination: in-cluster

Namespace: agbglobal

Created At: 12/11/2024 11:48:59 (a year ago)

Last Sync: 07/04/2025 11:00:16 (4 months ago)

↻ SYNC

🔄 REFRESH

✖ DELETE

digital-twin

Project: provisioning

Labels:

Status:

♥ Healthy

✓ Synced

Repository: ssh://git@

Target Revi... HEAD

Path: project-provisioning/user-projects/digital-twin

Destination: in-cluster

Namespace: digital-twin

Created At: 06/19/2025 16:50:19 (4 months ago)

Last Sync: 06/19/2025 16:50:26 (4 months ago)

↻ SYNC

🔄 REFRESH

✖ DELETE

cybersecurity

Project: provisioning

Labels:

Status:

♥ Healthy

✓ Synced

Repository: ssh://git@

Target Revi... HEAD

Path: project-provisioning/user-projects/cybersecurity

Destination: in-cluster

Namespace: cybersecurity

Created At: 12/11/2024 11:49:00 (a year ago)

Last Sync: 05/22/2025 09:46:56 (5 months ago)

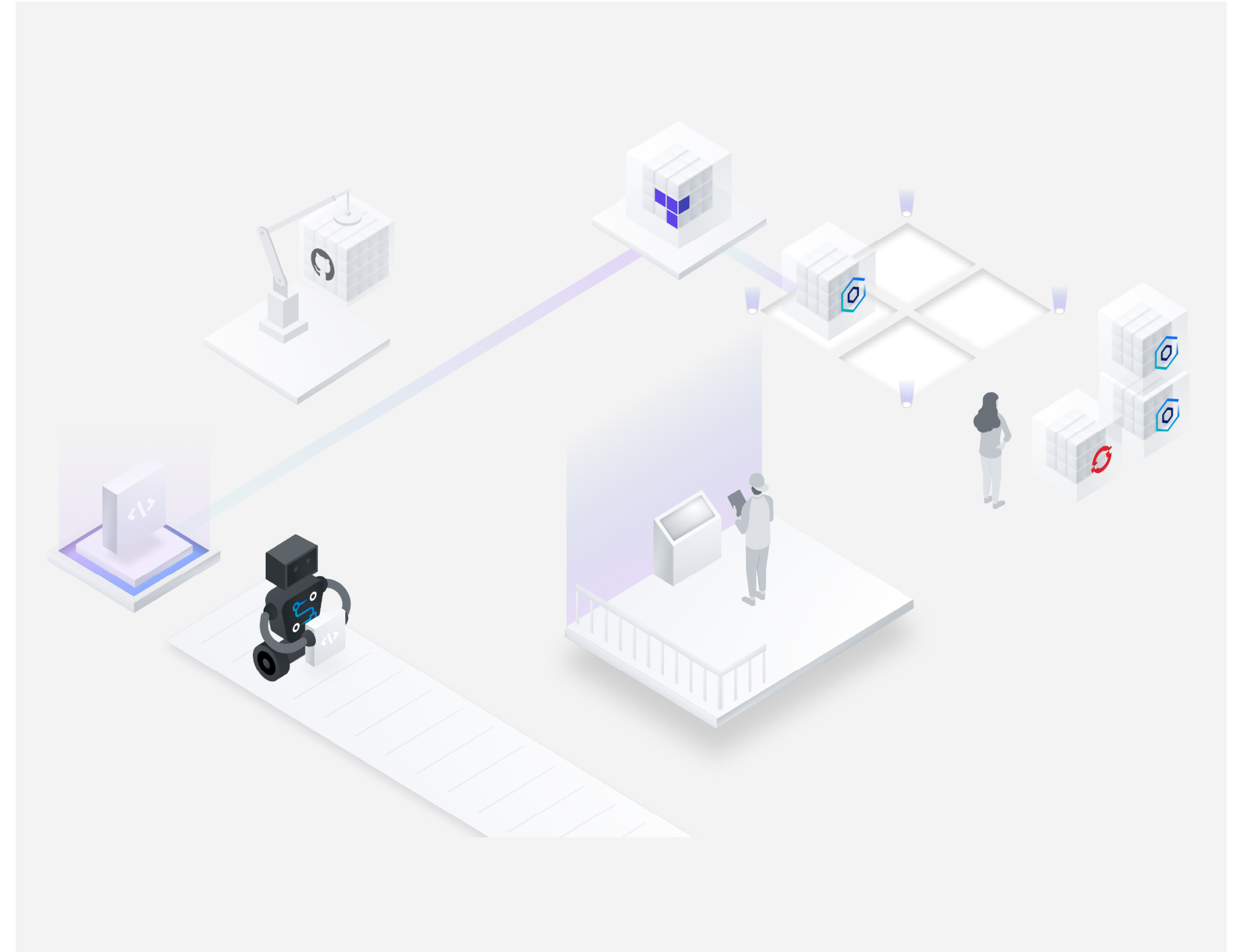
↻ SYNC

🔄 REFRESH

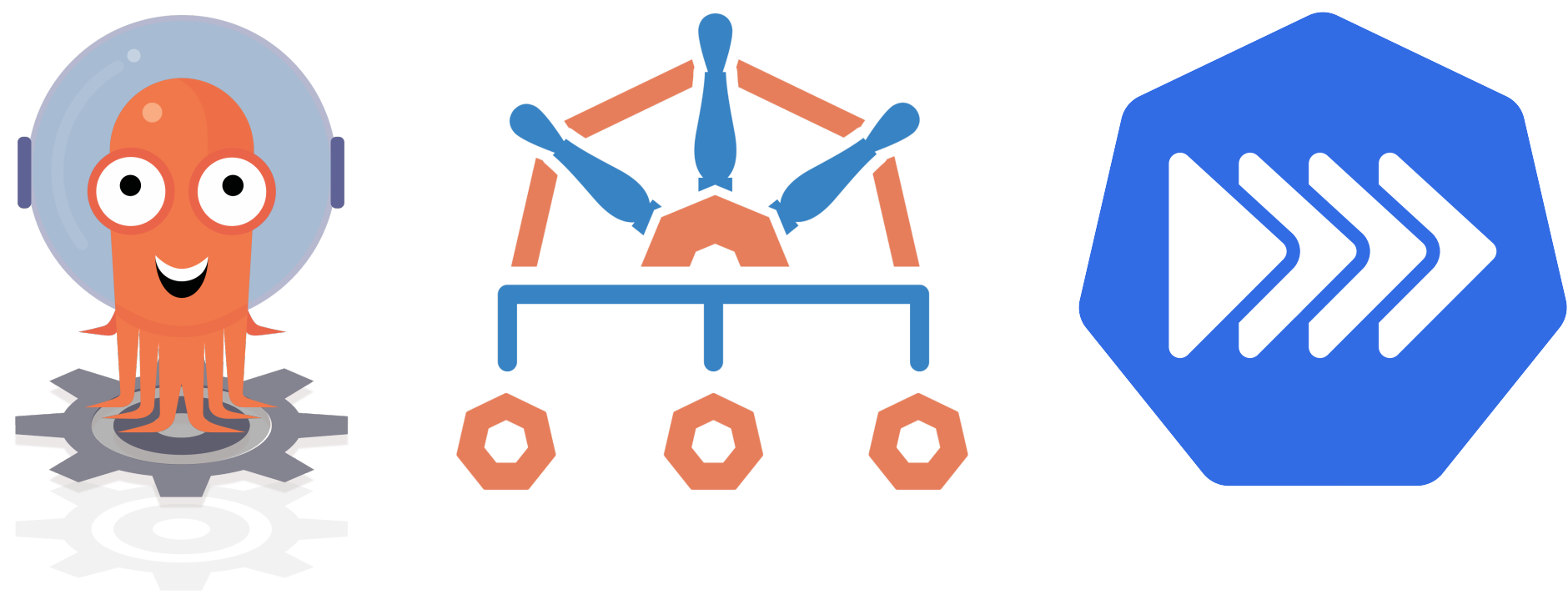
✖ DELETE

Results

- ***Users manage their namespaces.*** They submit PRs that require just a review and changes are deployed instantly. Kustomize is intuitive for both admins and users.
- ***Compliance is guaranteed.*** A shared project definition guarantees compliance and additions there cascade to all namespaces. Self-healing prevents manual changes from persisting.
- ***Maintenance effort is minimised.*** Manual deployments are gone, ArgoCD auto-updates. Slack requests for simple changes have effectively been zeroed.



What's the plan?



✓ Reduce the need for admin intervention

Introduce automation and tooling that enables self-service namespace management even for non-experts users.

Prevent CPU jobs from running amok

Introduce safeguards to prevent CPU-only workloads from overwhelming the cluster or interfering with GPU scheduling.

Discourage interactive GPU usage

Prevent the creation of long-lived, interactively accessed pods in favour of more efficient, Job-based workflows that free up (GPU) resources once they're complete.

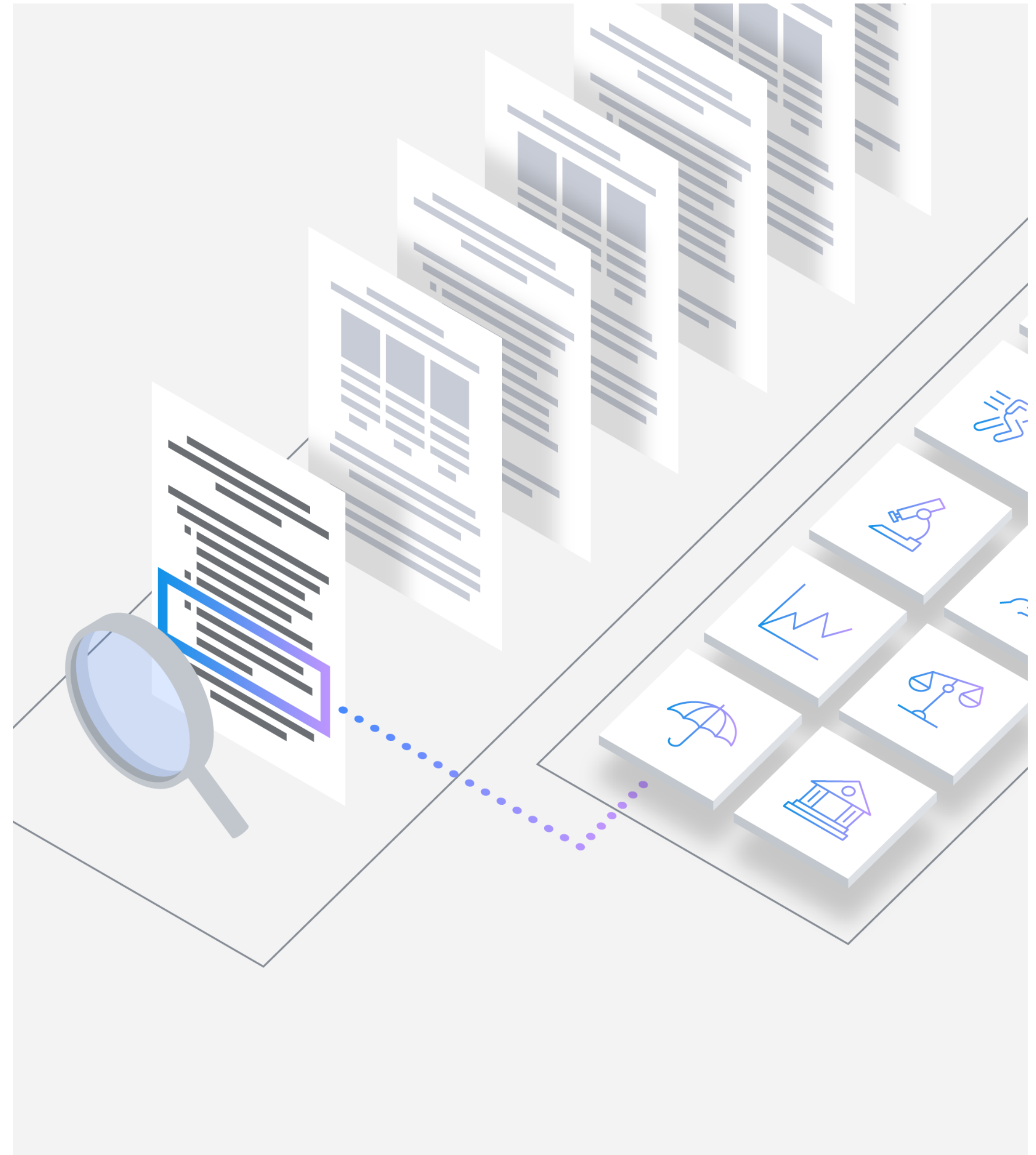
Make GPU access fair

Improve resource allocation to ensure GPU access is fair, providing a more predictable, HPC-like experience that minimises contention.

Policing

Our goals

- ***Fix the root causes of our issues.*** Make sure we help the scheduler start pods on appropriate nodes and prevent interactive GPU pods.
- ***Not be overly disruptive.*** Our policies should not do more harm than good. Exception mechanisms and observability are a must.
- ***Get up and running quickly.*** Avoid having to learn a new language, do as much copy/pasting as possible.



Policing

Handling batch Jobs

How to deal with large CPU-only Jobs jailing GPUs on GPU nodes?

Do not schedule their pods on them:

→ Automatically inject (anti-)affinity rules!

```
1  apiVersion: kyverno.io/v1
2  kind: Policy
3  metadata:
4    name: job-affinity-rules
5  spec:
6    background: false
7    rules:
8      - name: avoid-gpu-nodes-for-jobs-with-no-gpus
9        match:
10         any:
11           - resources:
12             kinds:
13               - Job
14         context:
15           - name: gpu_requests
16             variable:
17               # Use || [0] to return an array with just a zero if the field does not exist.
18               # This is required because the sum function in Kyverno requires at least one element
19               # in the array, which would otherwise be empty.
20               jmesPath: 'request.object.spec.template.spec.containers[].resources.requests."nvidia.com/gpu" || [0]'
21               default: [0]
22           preconditions:
23             all:
24               - key: "{{ sum(gpu_requests) }}"
25                 operator: Equals
26                 value: 0
27         mutate:
28           patchStrategicMerge:
29             spec:
30               template:
31                 spec:
32                   affinity:
33                     nodeAffinity:
34                       requiredDuringSchedulingIgnoredDuringExecution:
35                         nodeSelectorTerms:
36                           - matchExpressions:
37                             - key: nvidia.com/gpu.product
38                               operator: DoesNotExist
39
```

Policing

Handling batch Jobs

How to deal with large CPU-only Jobs jailing GPUs on GPU nodes?

Do not schedule their pods on them:

→ Automatically inject (anti-)affinity rules!

```
1  apiVersion: kyverno.io/v1
2  kind: Policy
3  metadata:
4    name: job-affinity-rules
5  spec:
6    background: false
7    rules:
8      - name: avoid-gpu-nodes-for-jobs-with-no-gpus
9        match:
10         any:
11           - resources:
12             kinds:
13               - Job
14         context:
15           - name: gpu_requests
16             variable:
17               # Use || [0] to return an array with just a zero if the field does not exist.
18               # This is required because the sum function in Kyverno requires at least one element
19               # in the array, which would otherwise be empty.
20               jmesPath: 'request.object.spec.template.spec.containers[].resources.requests."nvidia.com/gpu" || [0]'
21               default: [0]
22         preconditions:
23           all:
24             - key: "{{ sum(gpu_requests) }}"
25               operator: Equals
26               value: 0
27         mutate:
28           patchStrategicMerge:
29             spec:
30               template:
31                 spec:
32                   affinity:
33                     nodeAffinity:
34                       requiredDuringSchedulingIgnoredDuringExecution:
35                         nodeSelectorTerms:
36                           - matchExpressions:
37                             - key: nvidia.com/gpu.product
38                               operator: DoesNotExist
39
```

Policing

Handling interactive pods

How to deal with interactive pods running indefinitely?

- Deny *sleep infinity*?
- Deny *tail -f /dev/null*?

→ Make it impossible to use them!

Bonus: started off from [one of the example policies](#)!

A screenshot of a web browser displaying a Kyverno policy definition. The browser's address bar shows the URL "https://kyverno.io/policies/other/bloc...". The page has an orange header with the text "Like Kyverno? Star on GitHub to follow and support". The main heading is "Policy Definition" with a link icon. Below it is the policy name: "/other/block-pod-exec-by-namespace-label/block-pod-exec-by-namespace-label.yaml". The policy definition is shown in a light gray box with a "YAML" label in the top right corner. The YAML content is as follows:

```
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: deny-exec-by-namespace-label
  annotations:
    policies.kyverno.io/title: Block Pod Exec by Namespace Label
    policies.kyverno.io/category: Sample
    policies.kyverno.io/minversion: 1.6.0
    policies.kyverno.io/subject: Pod
    policies.kyverno.io/description: >-
      The `exec` command may be used to gain shell access, or run other commands. While this can be useful for troubleshooting purposes, it could represent an attack.
      This policy blocks Pod exec commands based upon a Namespace label `exec`.
spec:
  validationFailureAction: Enforce
  background: false
  rules:
    - name: deny-exec-by-ns-label
      match:
        any:
          - resources:
              kinds:
                - Pod/exec
      context:
        - name: nslabelexec
          apiCall:
            urlPath: "/api/v1/namespaces/{{request.namespace}}"
            jmesPath: "metadata.labels.exec || ''"
      preconditions:
        all:
          - key: "{{ request.operation || 'BACKGROUND' }}"
            operator: Equals
            value: CONNECT
      validate:
        message: Executing a command in a container is forbidden for Pods run
        deny:
```

Policing

Handling interactive pods

How to deal with interactive pods running indefinitely?

- Deny *sleep infinity*?
- Deny *tail -f /dev/null*?

→ Make it impossible to use them!

Bonus: started off from [one of the example policies](#)!



```
1  apiVersion: kyverno.io/v1
2  kind: ClusterPolicy
3  metadata:
4    name: exec-only-from-cluster-admins
5    namespace: kyverno-admin
6  spec:
7    validationFailureAction: Enforce
8    background: false
9    rules:
10     - name: exec-only-from-cluster-admins
11       context:
12         - name: exec-namespace-exceptions
13           configMap:
14             name: exec-namespace-exceptions
15             namespace: kyverno-admin
16       match:
17         any:
18           - resources:
19               kinds:
20                 - Pod/exec
21       preconditions:
22         all:
23           - key: "{{ request.operation || 'BACKGROUND' }}"
24             operator: Equals
25             value: CONNECT
26           - key: "{{ request.clusterRoles.contains(@, 'cluster-admin') }}"
27             operator: NotEquals
28             value: true
29           - key: "{{ request.namespace }}"
30             operator: AnyNotIn
31             value:
32               '{{ "exec-namespace-exceptions".data."exceptions" | parse_json(@) }}'
33       validate:
34         message:
35           Executing a command in a container is forbidden for Pods running in
36           this Namespace. To request an exception, reach out to the admins.
37       deny: {}
```

Policing

Handling interactive pods

How to deal with interactive pods running indefinitely?

- Deny *sleep infinity*?
- Deny *tail -f /dev/null*?

→ Make it impossible to use them!

Bonus: started off from [one of the example policies](#)!

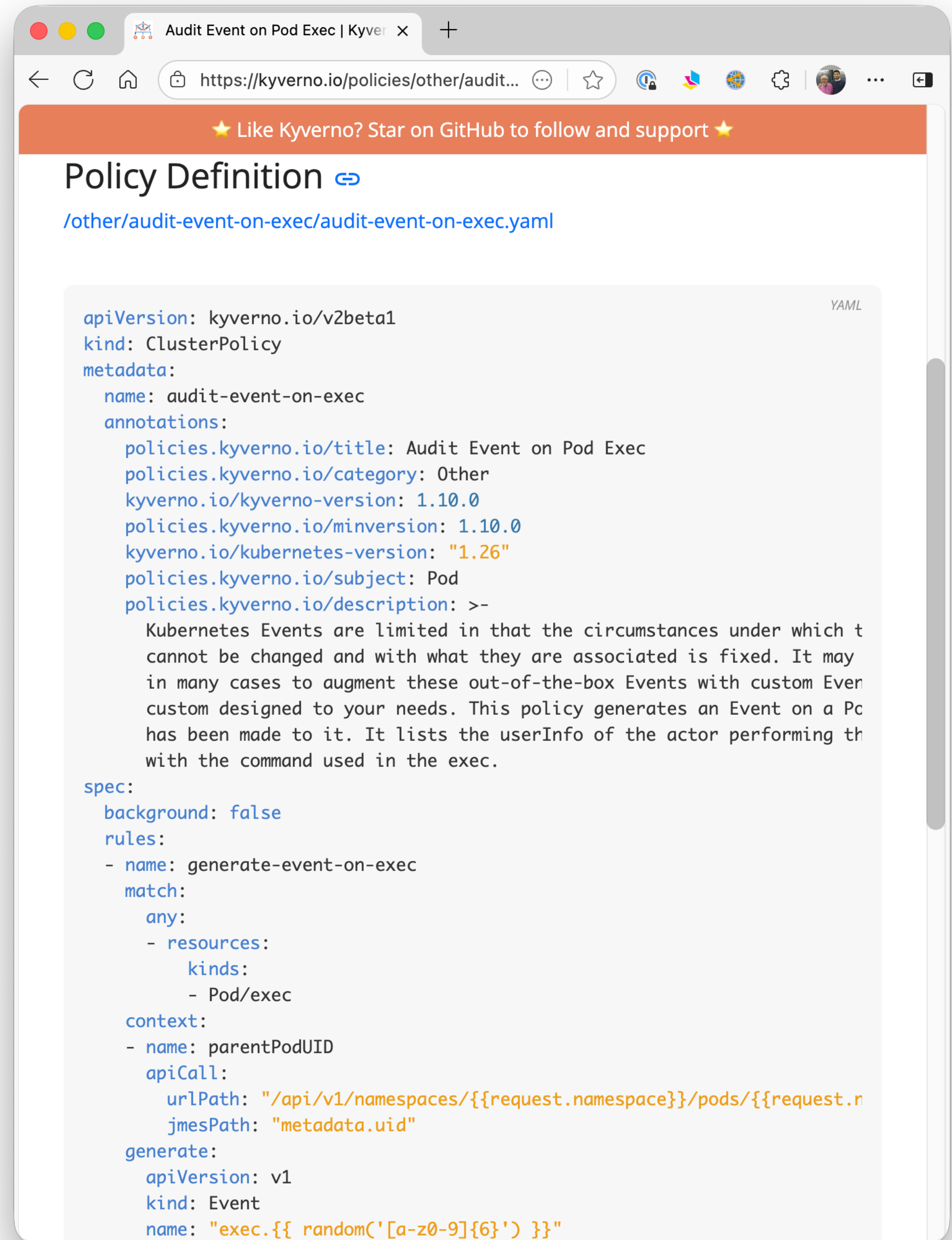
```
1  apiVersion: kyverno.io/v1
2  kind: ClusterPolicy
3  metadata:
4    name: exec-only-from-cluster-admins
5    namespace: kyverno-admin
6  spec:
7    validationFailureAction: Enforce
8    background: false
9    rules:
10     - name: exec-only-from-cluster-admins
11       context:
12         - name: exec-namespace-exceptions
13           configMap:
14             name: exec-namespace-exceptions
15             namespace: kyverno-admin
16       match:
17         any:
18           - resources:
19               kinds:
20                 - Pod/exec
21       preconditions:
22         all:
23           - key: "{{ request.operation || 'BACKGROUND' }}"
24             operator: Equals
25             value: CONNECT
26           - key: "{{ request.clusterRoles.contains(@, 'cluster-admin') }}"
27             operator: NotEquals
28             value: true
29           - key: "{{ request.namespace }}"
30             operator: AnyNotIn
31             value:
32               '{{ "exec-namespace-exceptions".data."exceptions" | parse_json(@) }}'
33       validate:
34         message:
35           Executing a command in a container is forbidden for Pods running in
36           this Namespace. To request an exception, reach out to the admins.
37       deny: {}
```

Policing

Handling interactive pods

But... what if it's too much?

- [Create audit events](#).
- *Make a test in a namespace with power users.*



A screenshot of a web browser displaying the Kyverno Policy Definition page for 'audit-event-on-exec'. The browser's address bar shows the URL 'https://kyverno.io/policies/other/audit...'. The page has an orange header with the text 'Like Kyverno? Star on GitHub to follow and support'. Below the header, the title 'Policy Definition' is followed by a link icon. The URL '/other/audit-event-on-exec/audit-event-on-exec.yaml' is displayed. The main content area shows a YAML configuration for a ClusterPolicy. The YAML includes metadata with name 'audit-event-on-exec', annotations for title, category, version, minversion, subject, and description, and a spec section with background set to false, a rule named 'generate-event-on-exec' matching Pod/exec, and an API call to generate an Event.

```
apiVersion: kyverno.io/v2beta1
kind: ClusterPolicy
metadata:
  name: audit-event-on-exec
  annotations:
    policies.kyverno.io/title: Audit Event on Pod Exec
    policies.kyverno.io/category: Other
    kyverno.io/kyverno-version: 1.10.0
    policies.kyverno.io/minversion: 1.10.0
    kyverno.io/kubernetes-version: "1.26"
    policies.kyverno.io/subject: Pod
    policies.kyverno.io/description: >-
      Kubernetes Events are limited in that the circumstances under which t
      cannot be changed and with what they are associated is fixed. It may
      in many cases to augment these out-of-the-box Events with custom Even
      custom designed to your needs. This policy generates an Event on a Pc
      has been made to it. It lists the userInfo of the actor performing th
      with the command used in the exec.
spec:
  background: false
  rules:
  - name: generate-event-on-exec
    match:
      any:
      - resources:
          kinds:
          - Pod/exec
    context:
    - name: parentPodUID
      apiCall:
        urlPath: "/api/v1/namespaces/{{request.namespace}}/pods/{{request.n
        jsonPath: "metadata.uid"
    generate:
      apiVersion: v1
      kind: Event
      name: "exec.{{ random('[a-z0-9]{6}')} }{"
```

Handling interactive pods

But... what if it's too much?

- Create audit events.
- *Make a test in a namespace with power users.*

ray-disorch-cpuwg-worker-jc84z

Generated from kyverno

NS discovery-dev

Nov 3, 2025, 4:43 PM

An exec was performed by {"extra":{"scopes.authorization.openshift.io":["user:full"]},"groups":["ap-testing-group","discovery-dev-group","gitops-cluster-admins","intelligent-infra-group","system:authenticated:oauth","system:authenticated"],"uid":["redacted"],"username":"Alessandro.Pomponio1@ibm.com"} running commands ["sh","-i","-c","TERM=xterm sh"]

The screenshot shows a web browser window at the URL https://kyverno.io/policies/other/audit... The page title is "Policy Definition" with a link icon. Below the title is the path /other/audit-event-on-exec/audit-event-on-exec.yaml. The main content area contains a YAML code block for a ClusterPolicy named audit-event-on-exec. The policy's annotations include its title, category, version (1.10.0), minimum version (1.10.0), Kubernetes version (1.26), subject (Pod), and a detailed description of its purpose. The spec section shows background set to false and rules starting with apiCall. To the right of the YAML is a "YAML" label. Below the code block is a yellow-bordered box containing a timestamp "Nov 3, 2025, 4:43 PM" and a snippet of a generated Event object, specifically the "exec" rule action. At the bottom, another part of the YAML is visible, showing generate settings like kind: Event and name: "exec.{{ random('a-z0-9') }}".

Policing

Handling interactive pods

It was too much!

- Blocking `oc cp` and `oc rsync`.
- Blocking `exec` on non-GPU pods.

```
1  apiVersion: kyverno.io/v1
2  kind: ClusterPolicy
3  metadata:
4    name: exec-only-from-cluster-admins
5  spec:
6    validationFailureAction: Enforce
7    background: false
8    rules:
9      - name: exec-only-from-cluster-admins
10        context:
11          - name: exec-namespace-exceptions
12            configMap:
13              name: exec-namespace-exceptions
14              namespace: kyverno-admin
15          - name: target_pod_gpus_requested
16            apiCall:
17              urlPath: "/api/v1/namespaces/{{request.namespace}}/pods/{{request.name}}"
18              jmesPath: 'spec.containers[].resources.requests."nvidia.com/gpu"'
19              default: [0]
20        match:
21          any:
22            - resources:
23              kinds:
24                - Pod/exec
25        exclude:
26          any:
27            - clusterRoles:
28              - cluster-admin
29        preconditions:
30          all:
31            - key: "{{ request.operation || 'BACKGROUND' }}"
32              operator: Equals
33              value: CONNECT
34            # Some namespaces are exempted from this rule
35            - key: "{{ request.namespace }}"
36              operator: AnyNotIn
37              value:
38                '{{ "exec-namespace-exceptions".data."exceptions" | parse_json(@) }}'
39            # Allow rsync and tar
40            - key: "{{ request.object.command[0] }}"
41              operator: AllNotIn
42              value:
43                - rsync
44                - tar
45        validate:
46          message:
47            Users in this Namespace are not allowed to execute commands in pods that use GPUs.
48            To request an exception, reach out to the admins.
49        deny:
50          conditions:
51            all:
52              # Deny GPU pods
53              # We need to check the length of the array and ensure it's greater than 0
54              # otherwise, the sum function might complain that it's expecting a non-empty array
55              - key: "{{ length(target_pod_gpus_requested) }}"
56                operator: GreaterThan
57                value: 0
58              - key: "{{ sum(target_pod_gpus_requested) }}"
59                operator: GreaterThan
60                value: 0
```

Policing

Handling interactive pods

It was too much!

- Blocking `oc cp` and `oc rsync`.
- Blocking `exec` on non-GPU pods.

```
1  apiVersion: kyverno.io/v1
2  kind: ClusterPolicy
3  metadata:
4    name: exec-only-from-cluster-admins
5  spec:
6    validationFailureAction: Enforce
7    background: false
8    rules:
9      - name: exec-only-from-cluster-admins
10        context:
11          - name: exec-namespace-exceptions
12            configMap:
13              name: exec-namespace-exceptions
14              namespace: kyverno-admin
15          - name: target_pod_gpus_requested
16            apiCall:
17              urlPath: "/api/v1/namespaces/{{request.namespace}}/pods/{{request.name}}"
18              jmesPath: 'spec.containers[].resources.requests."nvidia.com/gpu"'
19              default: [0]
20        match:
21          any:
22            - resources:
23              kinds:
24                - Pod/exec
25        exclude:
26          any:
27            - clusterRoles:
```

```
rules:
  - name: exec-only-from-cluster-admins
    context:
      - name: exec-namespace-exceptions
        configMap:
          name: exec-namespace-exceptions
          namespace: kyverno-admin
      - name: target_pod_gpus_requested
        apiCall:
          urlPath: "/api/v1/namespaces/{{request.namespace}}/pods/{{request.name}}"
          jmesPath: 'spec.containers[].resources.requests."nvidia.com/gpu"'
          default: [0]
```

operation || 'BACKGROUND' } }"

exempted from this rule
namespace } }"

ce-exceptions".data."exceptions" | parse_json(@) } }'

ject.command[0] } }"

ce are not allowed to execute commands in pods that use GPUs.
on, reach out to the admins.

< the length of the array and ensure it's greater than 0
sum function might complain that it's expecting a non-empty array

```
55 - key: "{{ length(target_pod_gpus_requested) }}"
56   operator: GreaterThan
57   value: 0
58 - key: "{{ sum(target_pod_gpus_requested) }}"
59   operator: GreaterThan
60   value: 0
```



Policing

Handling interactive pods

```
exclude:
  any:
    - clusterRoles:
      - cluster-admin
preconditions:
  all:
    - key: "{{ request.operation || 'BACKGROUND' }}"
      operator: Equals
      value: CONNECT
    # Some namespaces are exempted from this rule
    - key: "{{ request.namespace }}"
      operator: AnyNotIn
      value:
        '{{ "exec-namespace-exceptions".data."exceptions" | parse_json(@) }}'
    # Allow rsync and tar
    - key: "{{ request.object.command[0] }}"
      operator: AllNotIn
      value:
        - rsync
        - tar
validate:
  message:
    Users in this Namespace are not allowed to execute commands in pods that use GPUs.
    To request an exception, reach out to the admins.
deny:
  conditions:
    all:
      # Deny GPU pods
      # We need to check the length of the array and ensure it's greater than 0
      # otherwise, the sum function might complain that it's expecting a non-empty array
      - key: "{{ length(target_pod_gpus_requested) }}"
        operator: GreaterThan
        value: 0
      - key: "{{ sum(target_pod_gpus_requested) }}"
        operator: GreaterThan
        value: 0
```

```
1  apiVersion: kyverno.io/v1
2  kind: ClusterPolicy
3  metadata:
4    name: exec-only-from-cluster-admins
5  spec:
6    validationFailureAction: Enforce
7    background: false
8    rules:
9      - name: exec-only-from-cluster-admins
10        context:
11          - name: exec-namespace-exceptions
12            configMap:
13              name: exec-namespace-exceptions
14              namespace: kyverno-admin
15          - name: target_pod_gpus_requested
16            apiCall:
17              urlPath: "/api/v1/namespaces/{{request.namespace}}/pods/{{request.name}}"
18              jmesPath: 'spec.containers[].resources.requests."nvidia.com/gpu"'
19              default: [0]
20        match:
21          any:
22            - resources:
23              kinds:
24                - Pod/exec
25        exclude:
26          any:
27            - clusterRoles:
28              - cluster-admin
29        preconditions:
30          all:
31            - key: "{{ request.operation || 'BACKGROUND' }}"
32              operator: Equals
33              value: CONNECT
34            # Some namespaces are exempted from this rule
35            - key: "{{ request.namespace }}"
36              operator: AnyNotIn
37              value:
38                '{{ "exec-namespace-exceptions".data."exceptions" | parse_json(@) }}'
39            # Allow rsync and tar
40            - key: "{{ request.object.command[0] }}"
41              operator: AllNotIn
42              value:
43                - rsync
44                - tar
45        validate:
46          message:
47            Users in this Namespace are not allowed to execute commands in pods that use GPUs.
48            To request an exception, reach out to the admins.
49        deny:
50          conditions:
51            all:
52              # Deny GPU pods
53              # We need to check the length of the array and ensure it's greater than 0
54              # otherwise, the sum function might complain that it's expecting a non-empty array
55              - key: "{{ length(target_pod_gpus_requested) }}"
56                operator: GreaterThan
57                value: 0
58              - key: "{{ sum(target_pod_gpus_requested) }}"
59                operator: GreaterThan
60                value: 0
```

Policing

A small bonus

Knowing who created a certain object can be very helpful.

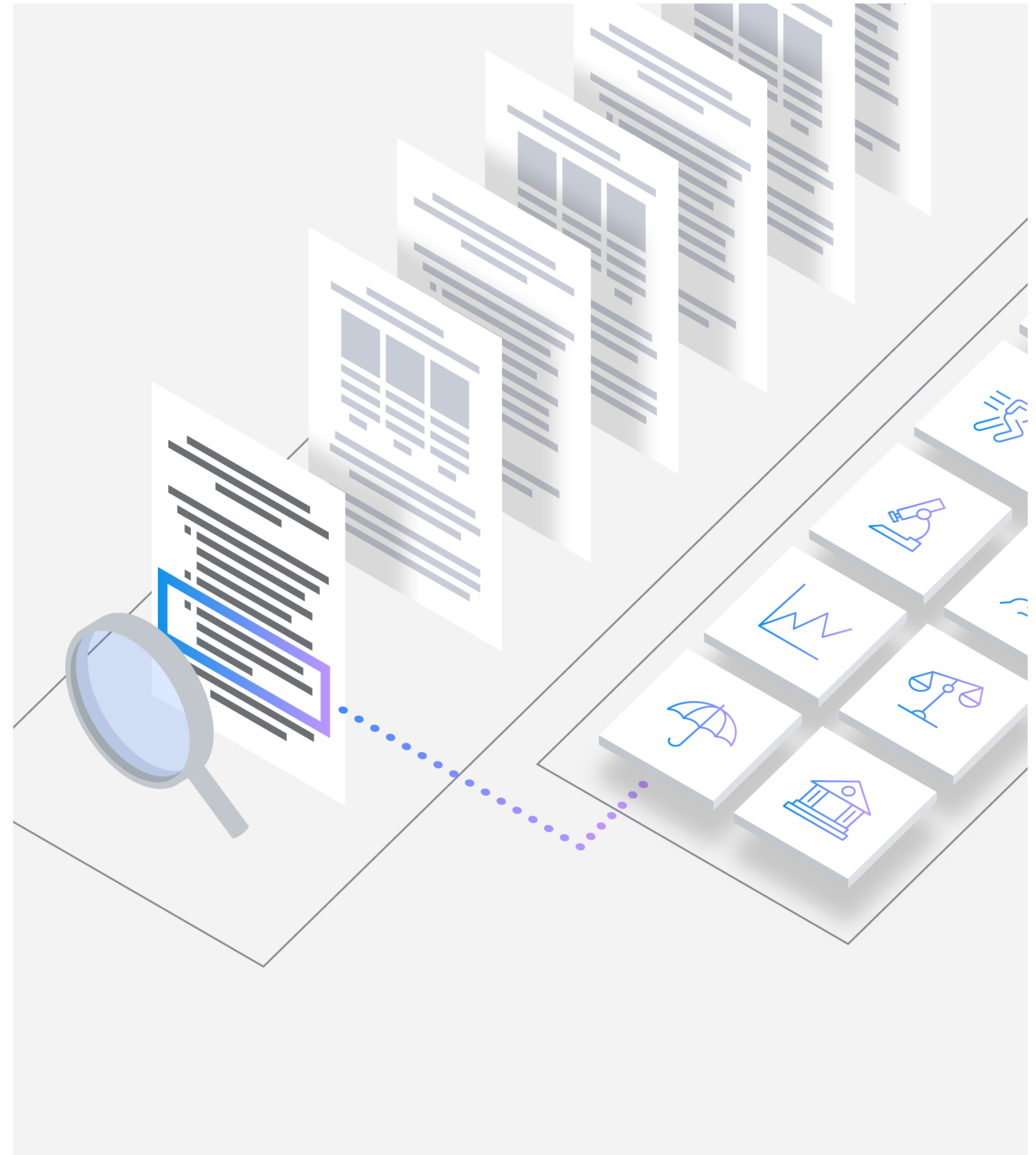
```
1  apiVersion: v1
2  data:
3    .dockerconfigjson: data
4  kind: Secret
5  metadata:
6    annotations:
7      owner: Alessandro.Pomponio1-at-ibm.com
8    creationTimestamp: "2025-03-21T09:21:02Z"
9    name: name
10   namespace: namespace
11   resourceVersion: "619018630"
12   uid: uid
13  type: kubernetes.io/dockerconfigjson
```

```
1  apiVersion: kyverno.io/v1
2  kind: ClusterPolicy
3  metadata:
4    name: add-owner-to-resources
5  spec:
6    background: false
7    rules:
8      - name: add-owner-to-resources
9        match:
10         any:
11           - resources:
12               kinds:
13                 - Job
14                 - Deployment
15                 - DeploymentConfig
16                 - StatefulSet
17                 - DaemonSet
18                 - ReplicaSet
19                 - Pod
20                 - Secret
21                 - ConfigMap
22                 - PersistentVolumeClaim
23
24         preconditions:
25           all:
26             - key: "{{ request.userInfo.username || ' ' }}"
27               operator: NotEquals
28               value: ""
29
30         mutate:
31           patchStrategicMerge:
32             metadata:
33               annotations:
34                 owner:
35                   "{{ replace_all('{{ request.userInfo.username }}', '@', '-at-') }}"
```

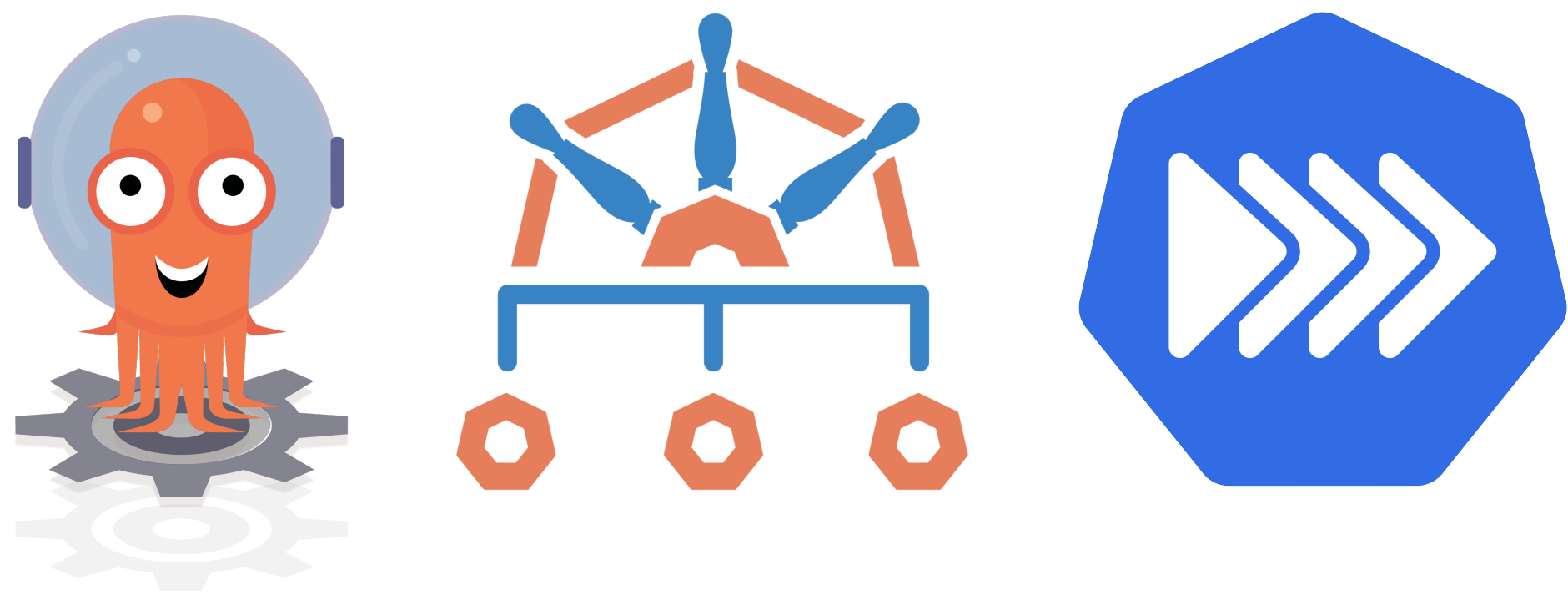
Policing

Results

- ***GPU Jobs have exclusive access to GPU nodes.*** Affinity rules are auto-applied instead of relying on users remembering to add them.
- ***Interactive GPU pods are mostly a thing of the past.*** Working with our colleagues we have identified a few workloads that *really* need this interactivity, but it's now off by default. Other, non-GPU workloads are unaffected, as well as file copying mechanisms.
- ***Users are largely unaware of the policies.*** Governance is transparent and does not hinder their workloads.



What's the plan?



✓ Reduce the need for admin intervention

Introduce automation and tooling that enables self-service namespace management even for non-experts users.

✓ Prevent CPU jobs from running amok

Introduce safeguards to prevent CPU-only workloads from overwhelming the cluster or interfering with GPU scheduling.

✓ Discourage interactive GPU usage

Prevent the creation of long-lived, interactively accessed pods in favour of more efficient, Job-based workflows that free up (GPU) resources once they're complete.

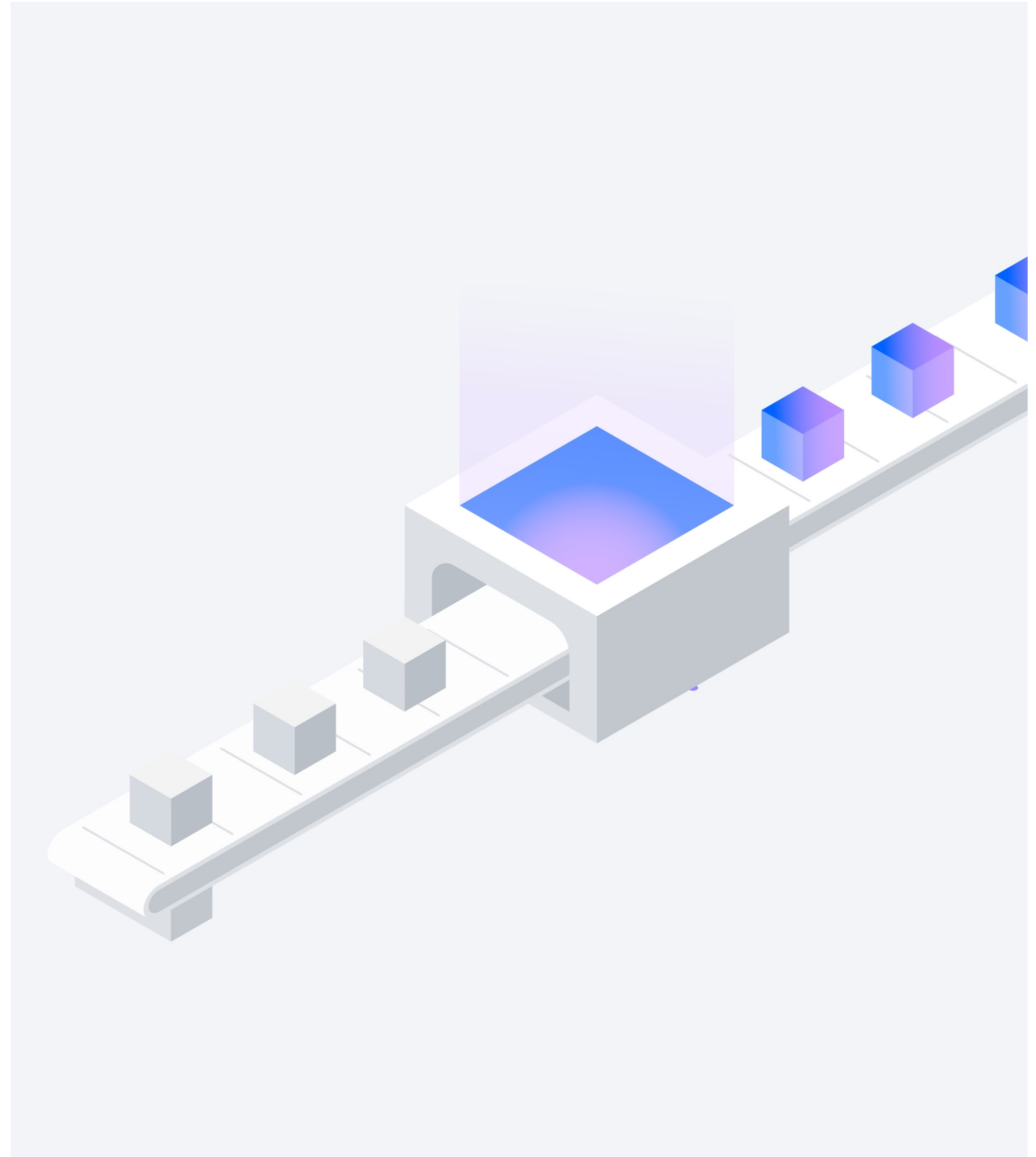
Make GPU access fair

Improve resource allocation to ensure GPU access is fair, providing a more predictable, HPC-like experience that minimises contention.

Queueing

Our goals

- ***Support HPC-like queueing semantics.*** Workloads are often not time-sensitive and researchers just *eventually* want the results.
- ***Support specific GPU requests.*** Workloads often have specific hardware requirements (e.g., at least 80GB of VRAM).
- ***Support generic GPU requests.*** When a generic GPU is sufficient, queue times will be lower.



Queueing

Accessing specific GPUs

Our clusters have multiple GPU types we have acquired over the years:

- T4s
- V100s
- A100s 40GB
- A100s 80GB
- H100s 80GB

```
1  apiVersion: kueue.x-k8s.io/v1beta1
2  kind: ResourceFlavor
3  metadata:
4    name: "nvidia-h100-80gb-pcie"
5  spec:
6    nodeLabels:
7      nvidia.com/gpu.product: NVIDIA-H100-PCIE
8  ---
9  apiVersion: kueue.x-k8s.io/v1beta1
10 kind: ClusterQueue
11 metadata:
12   name: "h100-cluster-queue"
13 spec:
14   cohort: gpu-cohort
15   namespaceSelector:
16     matchLabels:
17       kueue-enable: gpu-cluster-queue
18   resourceGroups:
19     - coveredResources: ["cpu", "memory", "nvidia.com/gpu"]
20       flavors:
21         - name: "nvidia-h100-80gb-pcie"
22           resources:
23             - name: "cpu"
24               nominalQuota: 160
25             - name: "memory"
26               nominalQuota: 400Gi
27             - name: "nvidia.com/gpu"
28               nominalQuota: 4
```

Queueing

Accessing specific GPUs

Our clusters have multiple GPU types we have acquired over the years:

- T4s
- V100s
- A100s 40GB
- A100s 80GB
- H100s 80GB

```
1  apiVersion: kueue.x-k8s.io/v1beta1
2  kind: ResourceFlavor
3  metadata:
4    name: "nvidia-h100-80gb-pcie"
5  spec:
6    nodeLabels:
7      nvidia.com/gpu.product: NVIDIA-H100-PCIe
8  ---
9  apiVersion: kueue.x-k8s.io/v1beta1
10 kind: ClusterQueue
11 metadata:
12   name: "h100-cluster-queue"
13 spec:
14   cohort: gpu-cohort
15   namespaceSelector:
16     matchLabels:
17       kueue-enable: gpu-cluster-queue
18   resourceGroups:
19     - coveredResources: ["cpu", "memory", "nvidia.com/gpu"]
20       flavors:
21         - name: "nvidia-h100-80gb-pcie"
22           resources:
23             - name: "cpu"
24               nominalQuota: 160
25             - name: "memory"
26               nominalQuota: 400Gi
27             - name: "nvidia.com/gpu"
28               nominalQuota: 4
```

Queueing

Accessing specific GPUs

Our clusters have multiple GPU types we have acquired over the years:

- T4s
- V100s
- A100s 40GB
- A100s 80GB
- H100s 80GB

```
1  apiVersion: kueue.x-k8s.io/v1beta1
2  kind: ResourceFlavor
3  metadata:
4    name: "nvidia-h100-80gb-pcie"
5  spec:
6    nodeLabels:
7      nvidia.com/gpu.product: NVIDIA-H100-PCIe
8  ---
9  apiVersion: kueue.x-k8s.io/v1beta1
10 kind: ClusterQueue
11 metadata:
12   name: "h100-cluster-queue"
13 spec:
14   cohort: gpu-cohort
15   namespaceSelector:
16     matchLabels:
17       kueue-enable: gpu-cluster-queue
18   resourceGroups:
19     - coveredResources: ["cpu", "memory", "nvidia.com/gpu"]
20     flavors:
21       - name: "nvidia-h100-80gb-pcie"
22         resources:
23           - name: "cpu"
24             nominalQuota: 160
25           - name: "memory"
26             nominalQuota: 400Gi
27           - name: "nvidia.com/gpu"
28             nominalQuota: 4
```

Queueing

Accessing a generic GPU

Our clusters have multiple GPU types we have acquired over the years:

- T4s
- V100s
- A100s 40GB
- A100s 80GB
- H100s 80GB

```
1  apiVersion: kueue.x-k8s.io/v1beta1
2  kind: ClusterQueue
3  metadata:
4    name: "gpu-cluster-queue"
5  spec:
6    cohort: gpu-cohort
7    namespaceSelector:
8      matchLabels:
9        kueue-enable: gpu-cluster-queue
10   resourceGroups:
11     - coveredResources: ["cpu", "memory", "nvidia.com/gpu"]
12       flavors:
13         - name: "nvidia-a100-80gb-pcie"
14           resources:
15             - name: "cpu"
16               nominalQuota: 0
17             - name: "memory"
18               nominalQuota: 0Gi
19             - name: "nvidia.com/gpu"
20               nominalQuota: 0
21         - name: "nvidia-h100-80gb-pcie"
22           resources:
23             - name: "cpu"
24               nominalQuota: 0
25             - name: "memory"
26               nominalQuota: 0Gi
27             - name: "nvidia.com/gpu"
28               nominalQuota: 0
```

Queueing

Accessing a generic GPU

Our clusters have multiple GPU types we have acquired over the years:

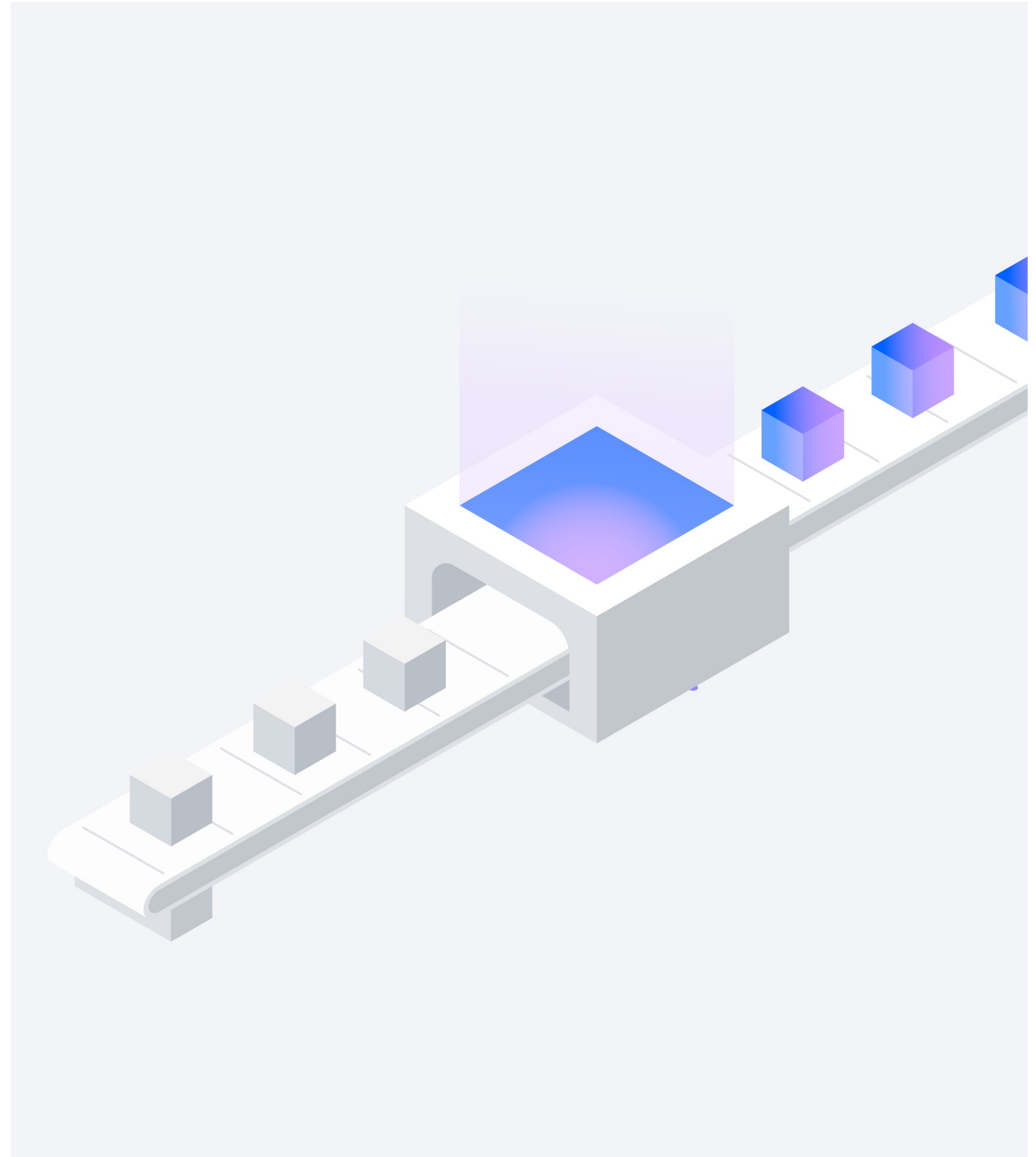
- T4s
- V100s
- A100s 40GB
- A100s 80GB
- H100s 80GB

```
1  apiVersion: kueue.x-k8s.io/v1beta1
2  kind: ClusterQueue
3  metadata:
4    name: "gpu-cluster-queue"
5  spec:
6    cohort: gpu-cohort
7    namespaceSelector:
8      matchLabels:
9        kueue-enable: gpu-cluster-queue
10   resourceGroups:
11     - coveredResources: ["cpu", "memory", "nvidia.com/gpu"]
12       flavors:
13         - name: "nvidia-a100-80gb-pcie"
14           resources:
15             - name: "cpu"
16               nominalQuota: 0
17             - name: "memory"
18               nominalQuota: 0Gi
19             - name: "nvidia.com/gpu"
20               nominalQuota: 0
21         - name: "nvidia-h100-80gb-pcie"
22           resources:
23             - name: "cpu"
24               nominalQuota: 0
25             - name: "memory"
26               nominalQuota: 0Gi
27             - name: "nvidia.com/gpu"
28               nominalQuota: 0
```

Queueing

Results

- *Resource access is now fair.* Workloads are enqueued and eventually execute.
- *Specific and generic queues are provided.* Users can choose to queue for specific GPUs or for a generic one.



“I’m hungry and I dozed off for the past 25 minutes. TL;DR?”

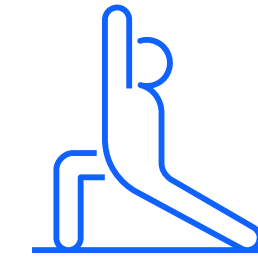
Audience member

To recap...



Admins have less headaches:

- User support queries have basically been zeroed.
- Deployments are not manual anymore. A GitOps approach guarantees visibility, audibility, and compliance.
- Policies automatically solve misconfigurations and prevent problematic behaviour.



Users are happier:

- Namespace-level changes are much quicker.
- Policies greatly reduced resource contention on the clusters.
- GPUs are accessed more fairly.

Thank you



© Copyright IBM Corporation 2025

IBM and the IBM logo are trademarks of IBM Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on ibm.com/legal/copytrade.

This document is current as of the initial date of publication and may be changed by IBM at any time.

THE INFORMATION IN THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT ANY WARRANTY, EXPRESS OR IMPLIED, INCLUDING WITHOUT ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND ANY WARRANTY OR CONDITION OF NON-INFRINGEMENT.

Examples presented as illustrative only. Actual results will vary based on client configurations and conditions and, therefore, generally expected results cannot be provided.

Not all offerings are available in every country in which IBM operates.

Statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

