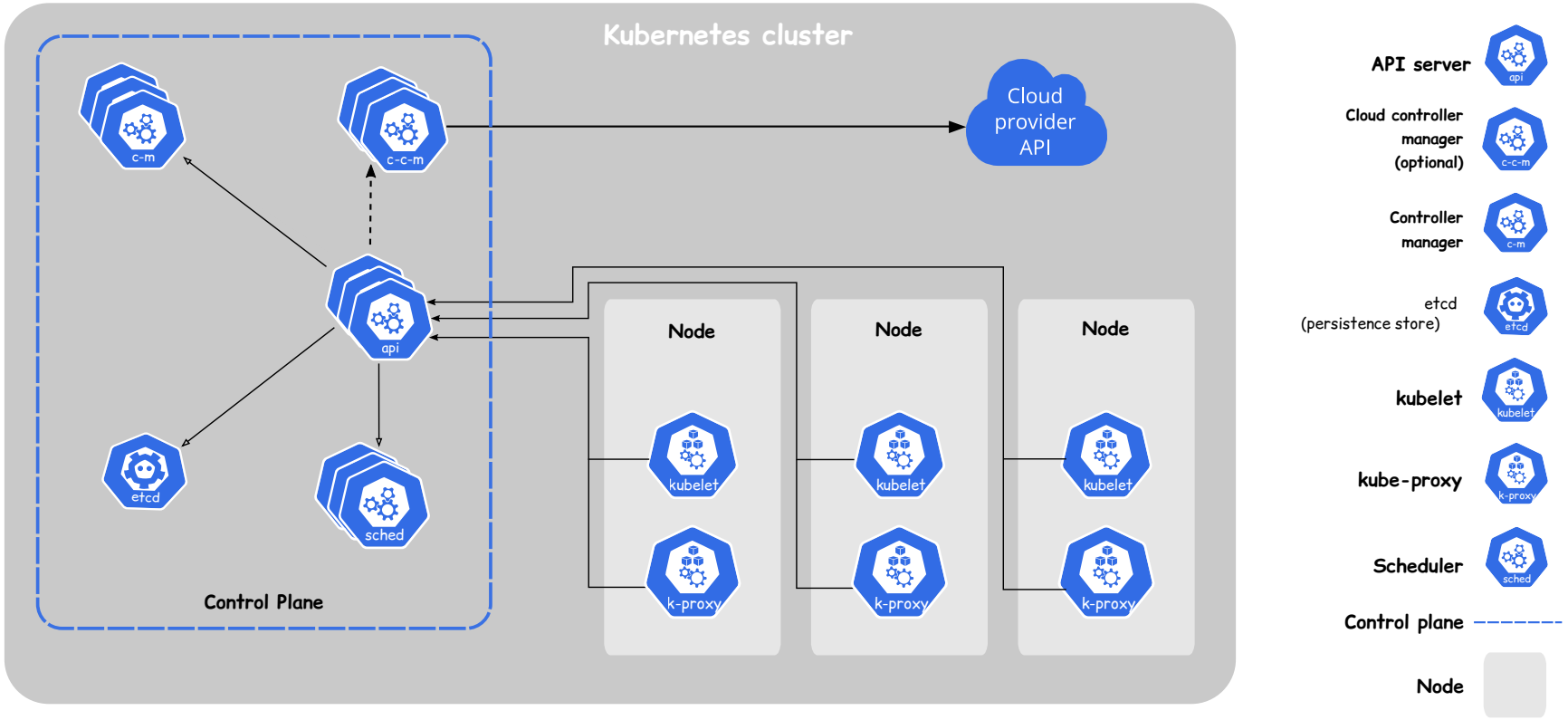
Kubernetes Control Plane in Depth

Ziping Sun

2025-06-06

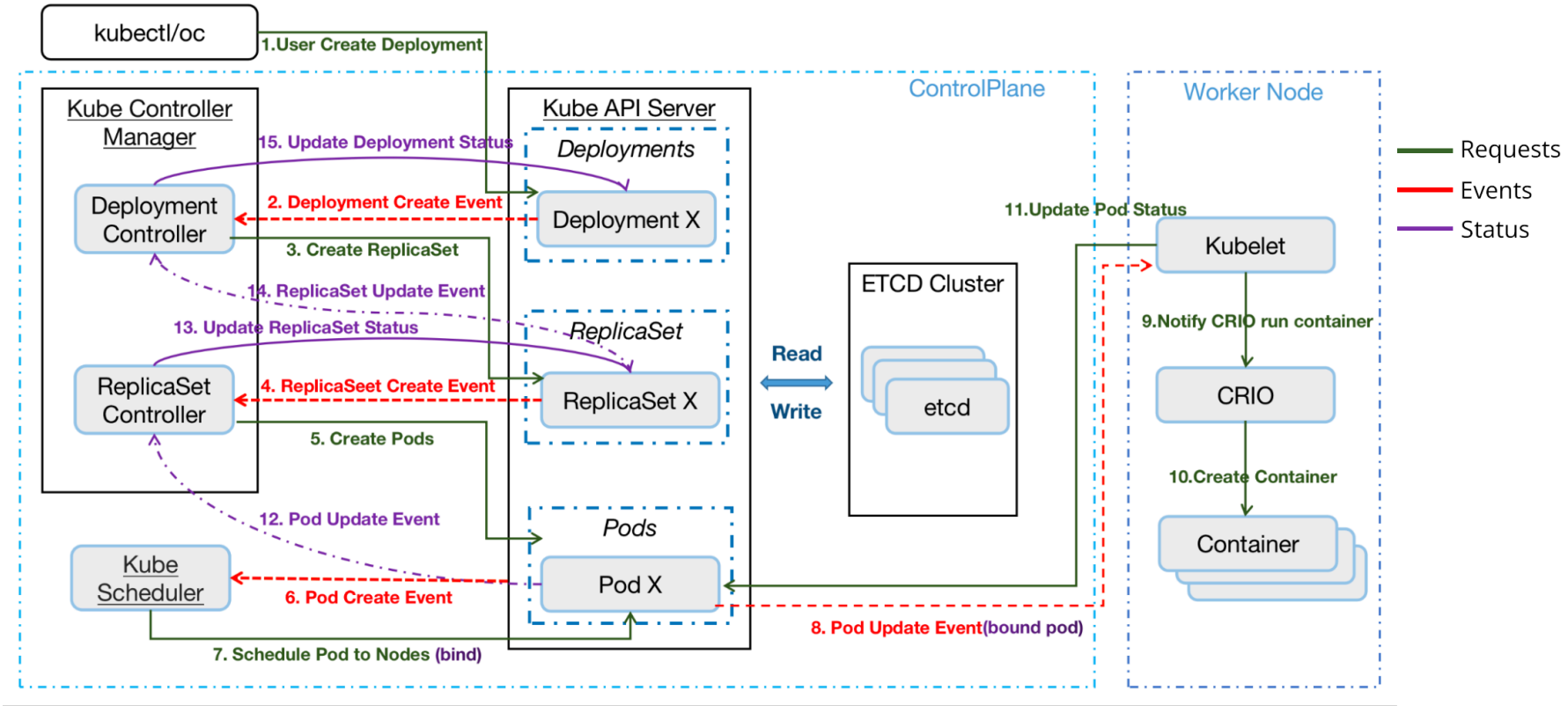
Outline

Overview: Components



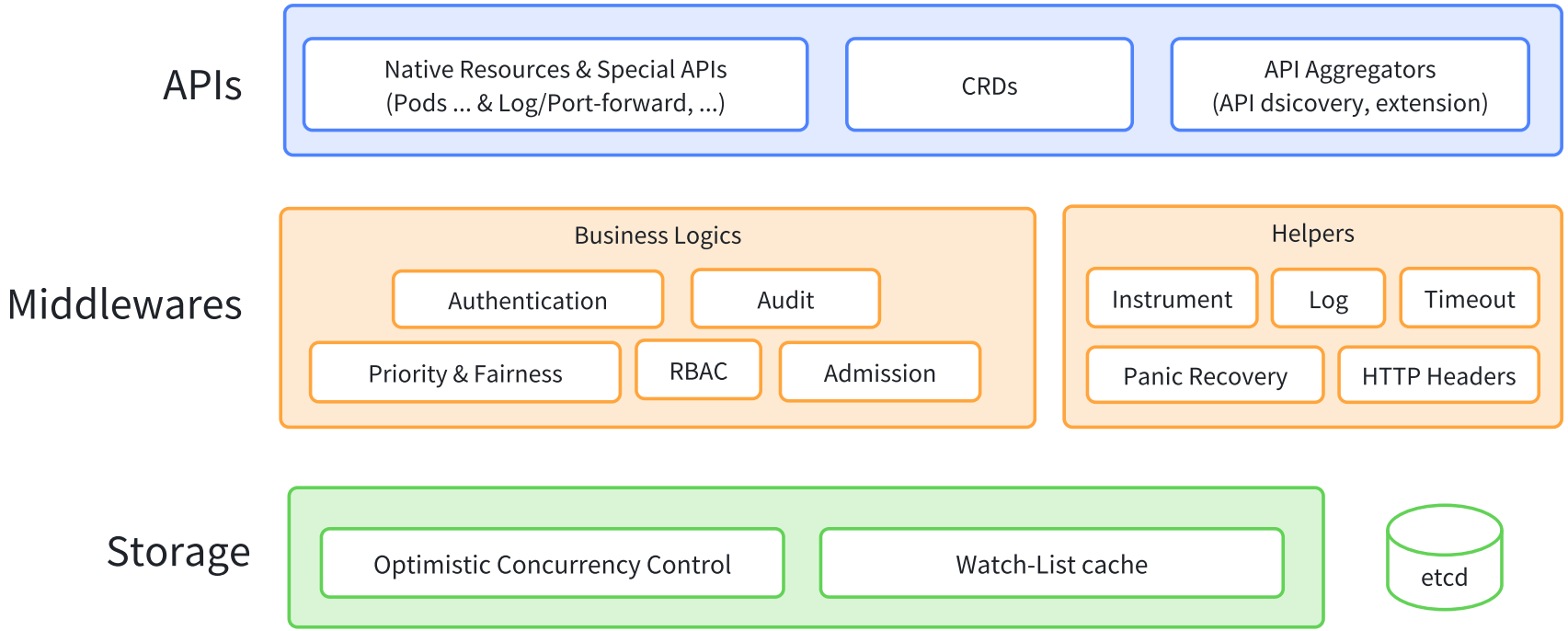
Control Plane: API Server, Controller Manager, Scheduler, etcd

Overview: Hub-and-Spoke Pattern



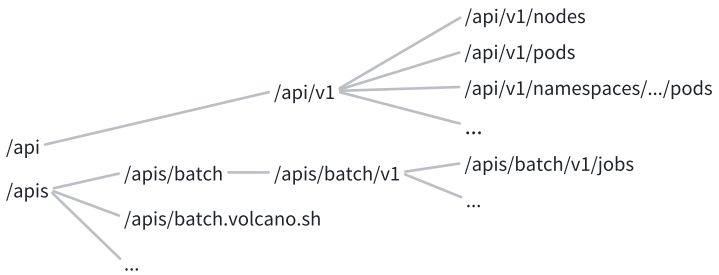
Outline

API Server: Overview



Request Flow: Middlewares (Authn, Audit, Flow Control, RBAC, Admission ⋯) → APIs → Storage

API Server: APIs



API Categories

- **Native** APIs
- CRDs: **batch.volcano.sh**
- Aggregated APIs: **metrics.k8s.io**

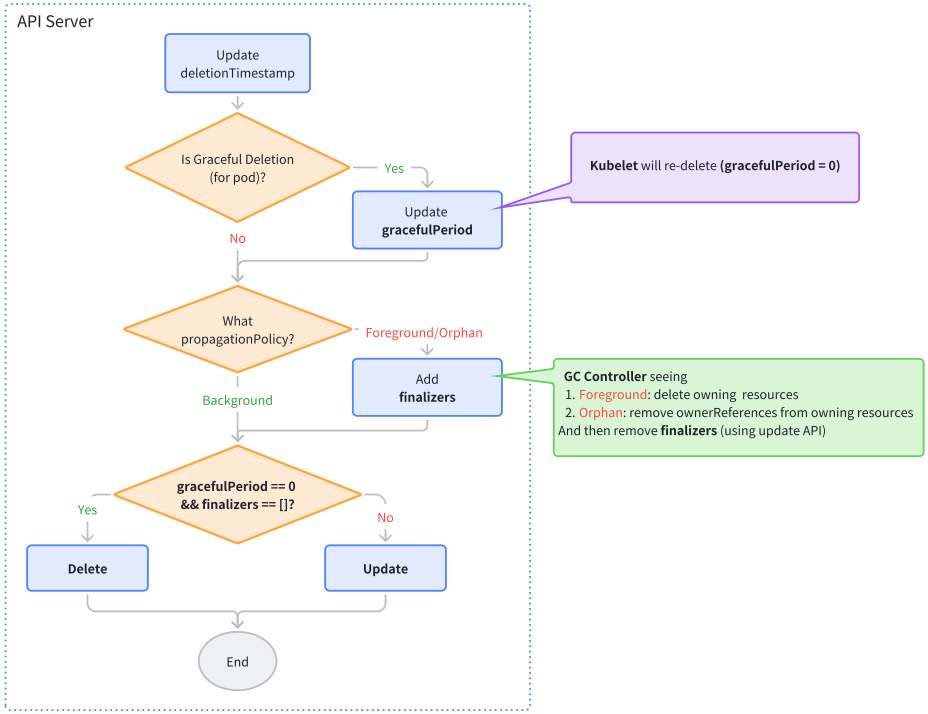
What makes native resources special?

- Special subresources
  - pods **resize/ephemeralcontainers/eviction/binding**
  - ...
  - uncommon "control → data" flow
  - pods **exec/attach/port-forward/logs**
  - service/pod/node **proxy**
- Change API Server behavior: **CRD, API Services** ...
- Virtual resources: **TokenReview** ...

CRUD + Watch Consistency

- Per-resource **linearizability**
- No guarantee for watch

API Server: APIs (deletion flow example)[1]



- Can be asynchronous
- Cascade deletion is implemented by **GC Controller**
- Typical usage
  - **graceful period** Pod
  - **finalizers** PV/PVC

```yaml
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: "2025-06-05T17:02:16Z"
  deletionGracePeriodSeconds: 30
  deletionTimestamp: "2025-06-05T17:04:30Z"
  finalizers:
  - szp.io/example
  labels:
    run: nodejs
  name: nodejs
  namespace: default
  resourceVersion: "2514368"
  uid: 5df0ed31-9bc2-4340-92bd-560bdbcb7330
```

[2] Based on `k8s.io/apiserver/pkg/registry/generic/registry/store.go`

API Server: Middlewares[1]

1. Authentication

- Various methods: **X.509**, **Service Account**, **OIDC** ⋯

2. Audit

- Configured with static Audit Policy

3. Priority and Fairness v1.29

- Controlled by **FlowSchema**

4. Authorization

- Two methods:
  RBAC
  node (hardcoded for kubelet)

5. Admission

- Two forms:
  In-tree Plugin PodSecurity, NamespaceLifecycle ⋯
  Webhooks

[2] Based on `k8s.io/apiserver/pkg/server/config.go`

API Server: Storage

How is resource mapped to KV?

- Native: `/registry/<resources>/<namespace>/<name>`
- CRDs: `/registry/<group>/<resources>/<namespace>/<name>`

Features

- MVCC `resourceVersion` ↔ `mod_revision`
- Encryption at Rest
- Watch Cache

Watch Cache

- cache KVs using etcd **Watch**
- **Stale** ⇔ `resourceVersion` $\neq 0$
- Accelerate requests

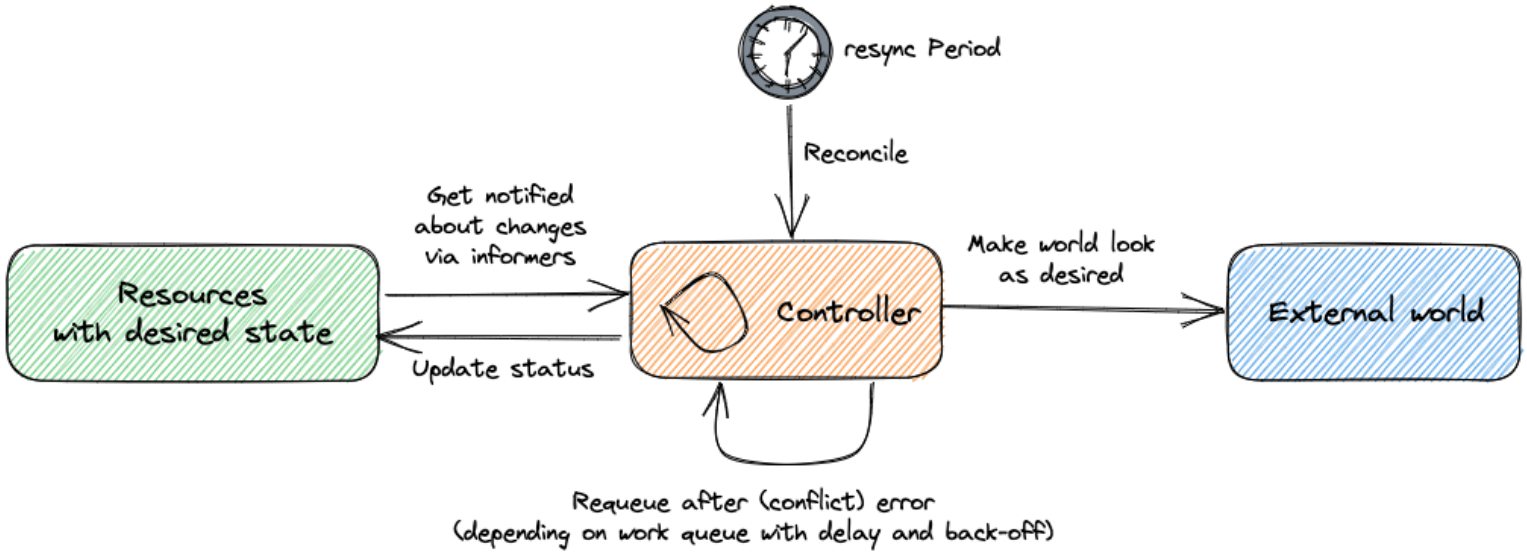
  GET (stale) LIST (stale) < v1.31

  LIST ⩾ v1.31 consistent read

  WATCH
- Note: filter applies after list etcd

Outline

Controller Manager: Overview



Key Patterns

- Reconcile Loop
- Worker Queue
- Leader Election
- ListWatch Cache

Controller Manager: Examples

NamespaceController
- cascade delete namespaced resource

NodeLifecycleController
- taint-based eviction
- node NotReady detector

Job/Deployment Controller …
- workload managerment

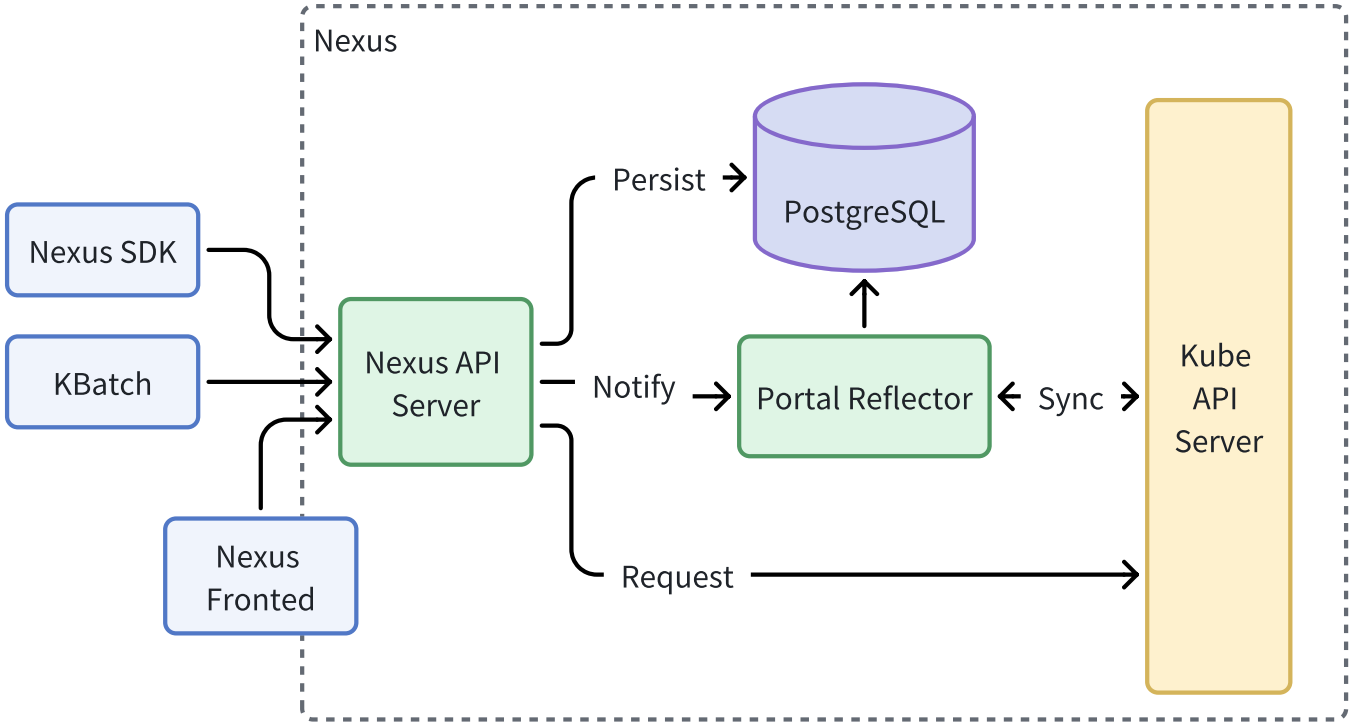
PodGCController
- GC terminated pods if too many
- cascade pod deletion for nonexistent nodes

EndpointSliceController
- create EndpointSlice based on service
- EndpointSlice used by kube-proxy

GC Controller
- cascade deletion
- Orphan, Foreground deletion

Outline

Nexus: Architecture



Nexus API Server

- Handle requests

Portal Reflector

- Sync status
- Handle async operations

Nexus: Debugging the Pod Restart Issue

Issue 1: Too many pods exceeding `terminated-pod-gc-threshold`
→ PodGCController deletes pod
→ VolcanoJobController recreates pod

Issue 2: Node down
→ NodeController adds unreachable & not-ready `NoExecute` taint to node
→ TaintEvictionController (v1.29) deletes pod
→ VolcanoJobController recreates pod

Final solution: Patching VolcanoJobController.

Thanks!