

# All New Kubernetes Monitoring

Kubernetes templates rebuilt for Zabbix 8.0 (and 7.4 a bit...)

Edgar Akhmetshin · Service Delivery Tech Lead & Trainer



# Where we come from: the classic template set

YOU CREATE

Kubernetes nodes

Kubernetes nodes by HTTP

DISCOVERED AUTOMATICALLY

<node name>

Linux by Zabbix agent

x 9



Agent 2  
DaemonSet  
on every  
node

Kubernetes cluster state

Kubernetes cluster state by HTTP

Kubelet <node>

Kubernetes Kubelet by HTTP

x 9

API <node>

Kubernetes API server by HTTP

x 3

Controller manager <node>

Kubernetes Controller manager by HTTP

x 3

Scheduler <node>

Kubernetes Scheduler by HTTP

x 3

Example: 3 control-plane  
+ 6 worker nodes

29

hosts in Zabbix

6

templates to import

2

hosts to create by hand



## Why two hosts?

Nodes and cluster-state templates share LLD keys, so they can't share a host.



## Which host is right?

One pod is collected on up to 3 hosts. A crash loop raises 2 duplicate problems.

# What you told us

Feedback & ZBXNEXT-10349



## So many hosts

A Linux host and a kubelet host per node, plus 3 per control-plane node.



## Component statuses?

Deprecated since Kubernetes 1.19, and it errors on managed clusters like AKS.

# Design goals

## Simpler (KISS principle)

- One host
- No agent on the nodes
- Nodes discovered for you

**Up in minutes.**

## Smarter

- Namespaces and deployments
- Usage vs requests vs limits
- Node pressure (PSI)

**See what matters.**

## Lighter

- 10 hosts instead of 29<sup>1</sup>
- 2 items per pod, not 29+
- No duplicate pod problems

**Scales with you.**

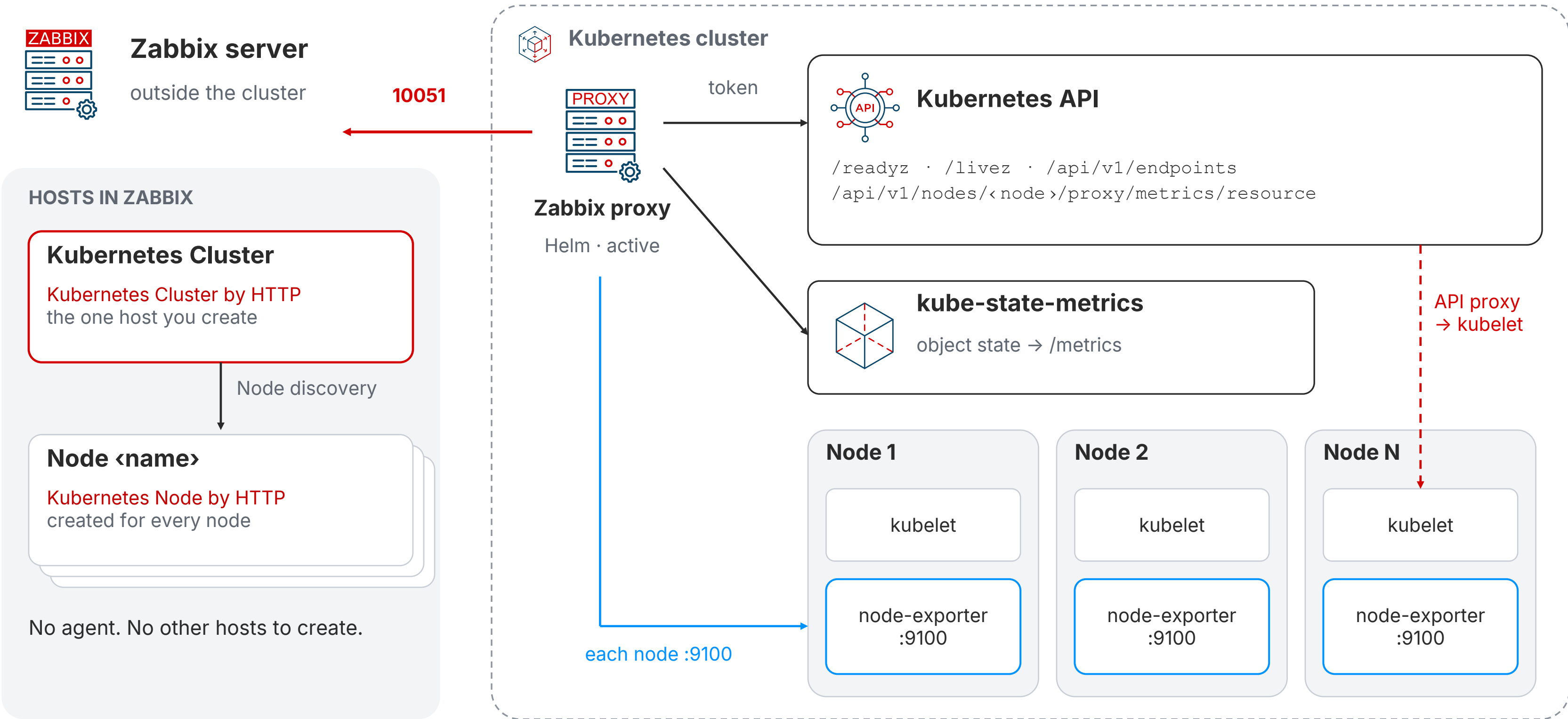
<sup>1</sup> Example: self-managed cluster with 3 control-plane + 6 worker nodes

# How it works

Architecture, Helm chart, setup and data flow



# Architecture at a glance



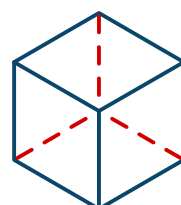
# Helm chart 2.0.0: what gets deployed



ON

## Zabbix proxy

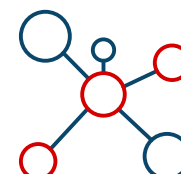
Active mode, SQLite. Collects everything and forwards it to your Zabbix server.



ON · chart 7.2.x

## kube-state-metrics

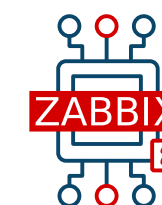
Turns the state of Kubernetes objects into metrics: nodes, pods, deployments, PVCs...



NEW · ON · chart 4.51.x

## node-exporter

DaemonSet on every node: OS metrics on :9100. Replaces the agent for node monitoring.



NOW OFF

## Zabbix agent 2

Not needed by the new templates. Enable only for the classic node hosts.

### Bring your own

Already run kube-state-metrics or node-exporter? Set `kubeStateMetrics.enabled` or `prometheusNodeExporter.enabled` to false and reuse yours.

### Read-only access

A ClusterRole with get/list on cluster objects, plus nodes/proxy for kubelet resource metrics. The token is created for you.

# Up and running in three steps

## 1

### Install the chart

Set ZBX\_SERVER\_HOST in your values, install, then add the proxy (zabbix-proxy) in Zabbix.

```
helm install zabbix . \
  --dependency-update \
  -f zabbix_values.yaml \
  -n monitoring
```

## 2

### Copy the token

The chart creates a service account and a token secret for Zabbix.

```
kubectl get secret \
  zabbix-zabbix-helm-chart \
  -n monitoring \
  -o jsonpath={.data.token} \
  | base64 -d
```

## 3

### Create one host

Link Kubernetes Cluster by HTTP, monitored by the in-cluster proxy. The Host Wizard asks for the token.

```
{ $KUBE.API.TOKEN }
  = <token>
{ $KUBE.API.URL }
  = https://kubernetes
    .default.svc (default)
```

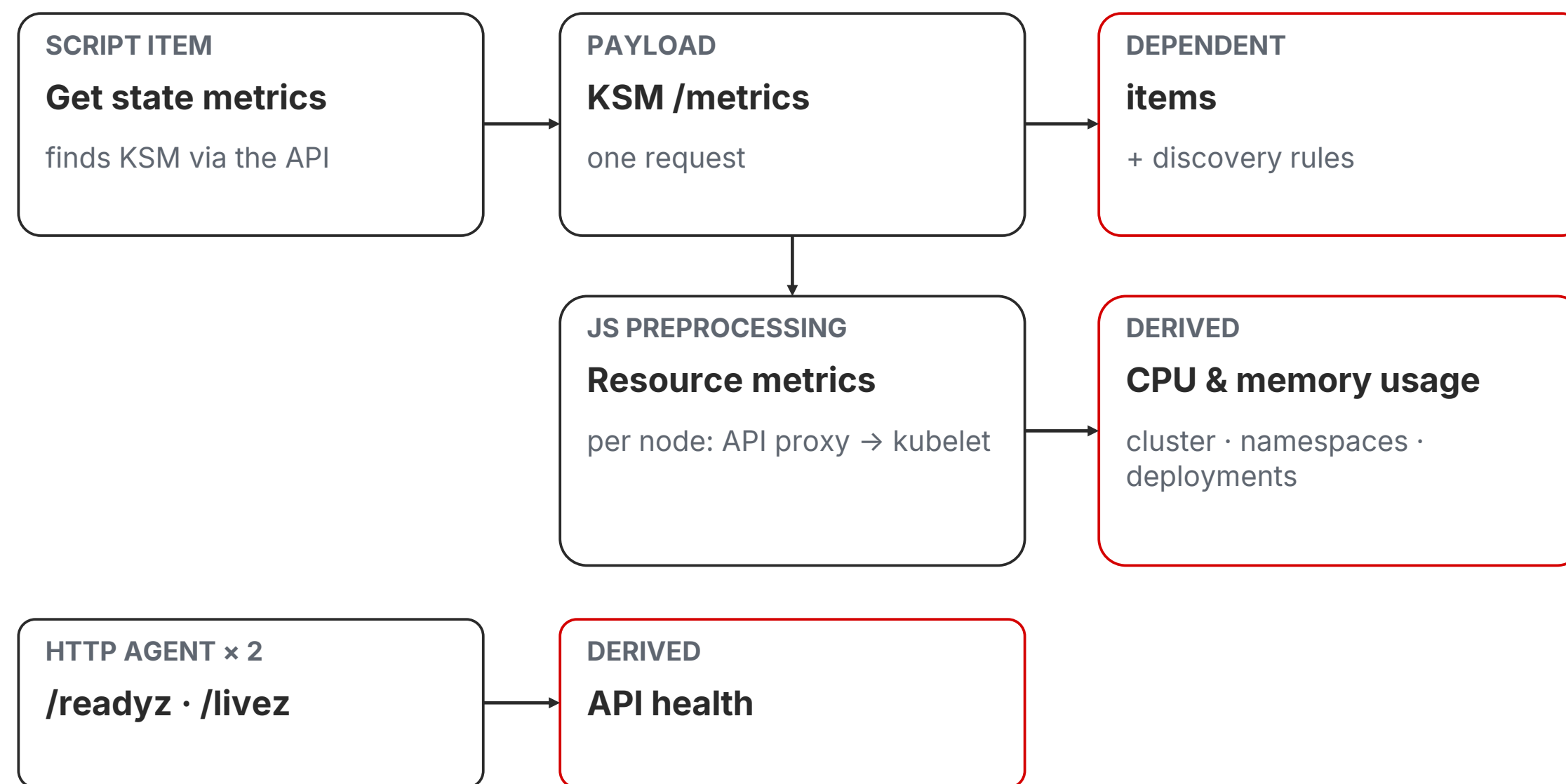


**Result:** node hosts are discovered and created automatically.

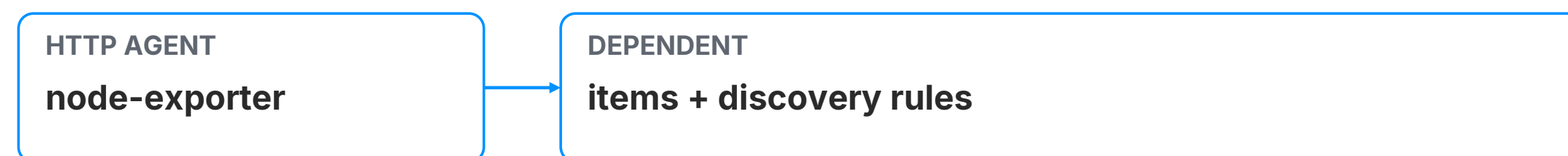
Note: TLS verification is on by default ({ \$HTTP.TLS.VERIFY } = full), so the proxy must trust your cluster's CA certificate.

## Under the hood: collect once, derive everything

### CLUSTER HOST



### NODE HOST (one per node)



### No external scripts

JavaScript runs inside Zabbix.



### One request, many metrics

Dependent items with Prometheus preprocessing.



### Light on the kubelet

/metrics/resource: lean CPU and memory usage.



### History on change

Discard unchanged with heartbeat.

# Macros you'll actually touch

| Macro                                                   | Default                                     | What it does                                    |
|---------------------------------------------------------|---------------------------------------------|-------------------------------------------------|
| <code>{ \$KUBE.API.TOKEN }</code>                       | <code>secret</code>                         | Service-account token — the only required value |
| <code>{ \$KUBE.API.URL }</code>                         | <code>https://kubernetes.default.svc</code> | Works as-is from the in-cluster proxy           |
| <code>{ \$KUBE.LLD.FILTER.NAMESPACE.* }</code>          | <code>. * / CHANGE_IF_NEEDED</code>         | Namespaces to discover                          |
| <code>{ \$KUBE.LLD.FILTER.NODE.* }</code>               | <code>. * / CHANGE_IF_NEEDED</code>         | Nodes that get a host                           |
| <code>{ \$KUBE.CLUSTER.CPU.UTIL.WARN   CRIT }</code>    | <code>70 / 80</code>                        | Cluster CPU utilization, % (memory alike)       |
| <code>{ \$KUBE.NAMESPACE.POD.PENDING.THRESHOLD }</code> | <code>3</code>                              | Pending pods per namespace (context)            |
| <code>{ \$KUBE.REPLICA.MISMATCH.EVAL_PERIOD }</code>    | <code>#5</code>                             | Replica mismatch window (context, regex)        |
| <code>{ \$KUBE.NODE.NUMBER.THRESHOLD }</code>           | <code>2</code>                              | Minimum number of Ready nodes                   |
| <code>{ \$KUBE.NODE.PSI.CPU.WARN   CRIT }</code>        | <code>20 / 50</code>                        | CPU pressure, % (node hosts)                    |
| <code>{ \$HTTP.TLS.VERIFY }</code>                      | <code>full</code>                           | TLS check: none, peer or full                   |

# What Zabbix calls in your cluster

| Request                                         | Why                     | How often                    |
|-------------------------------------------------|-------------------------|------------------------------|
| GET /api/v1/endpoints                           | Find kube-state-metrics | Each interval                |
| GET <kube-state-metrics>/metrics                | Object state            | Each interval                |
| GET /api/v1/nodes/<node>/proxy/metrics/resource | CPU and memory usage    | Per node, each interval      |
| GET /readyz · GET /livez                        | API server health       | Each interval                |
| GET <node IP>:9100/metrics                      | Node OS metrics         | Per node host, each interval |



## Chart ClusterRole (read-only)

get/list: namespaces, pods, services, componentstatuses, nodes, endpoints, events, jobs, cronjobs, deployments, daemonsets, statefulsets · get: nodes/proxy, nodes/metrics, nodes/stats, nodes/spec · URLs: /metrics, /metrics/cadvisor, /readyz, /healthz, /version

# What you get

Cluster → namespaces → workloads → pods → nodes

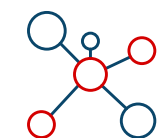


# Cluster: health and capacity at a glance



## Health

API server /readyz and /livez — no more deprecated componentstatuses



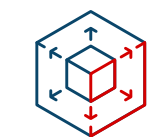
## Nodes

Total · Ready · NotReady



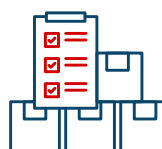
## Capacity

CPU and memory: allocatable · requests · limits · usage → utilization %



## Pods

Running · pending · failed · containers running



## Inventory

15 object counts: namespaces, deployments, StatefulSets, DaemonSets, jobs, services, ingresses, secrets, PVCs...

## Triggers out of the box

High

No Ready nodes

High

Low number of Ready nodes

Average

Readyz / Livez is unhealthy

Average

High CPU / memory utilization > 80%

Warning

High CPU / memory utilization > 70%

Warning

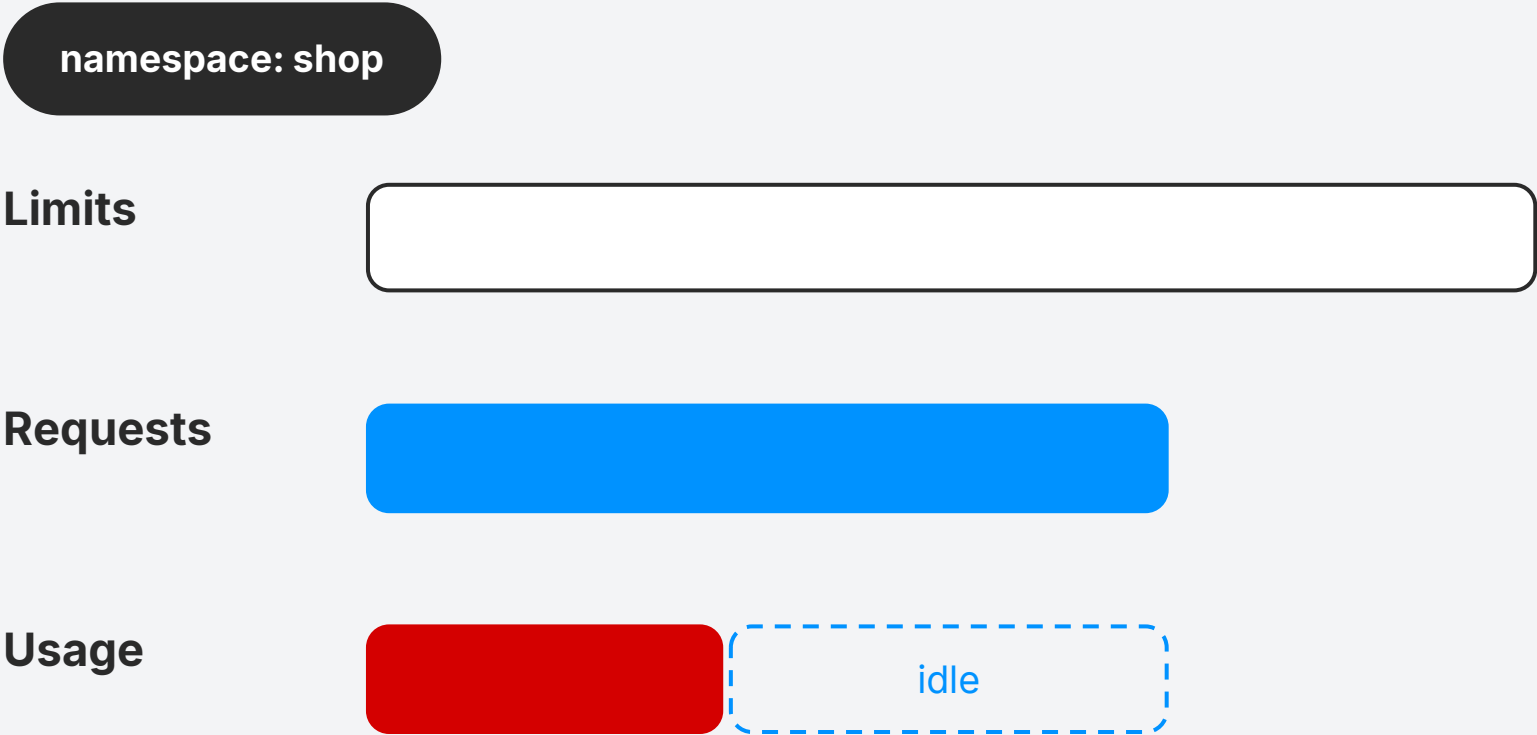
Pod pending high / Pod failed high

Warning

Failed to get kube-state-metrics data

# Namespaces: the missing middle

## One namespace, three numbers



Usage << requests → reserved but idle capacity  
 Usage → limits → CPU throttling, OOM kills  
 Share of cluster = usage ÷ cluster allocatable

## 12 metrics per namespace

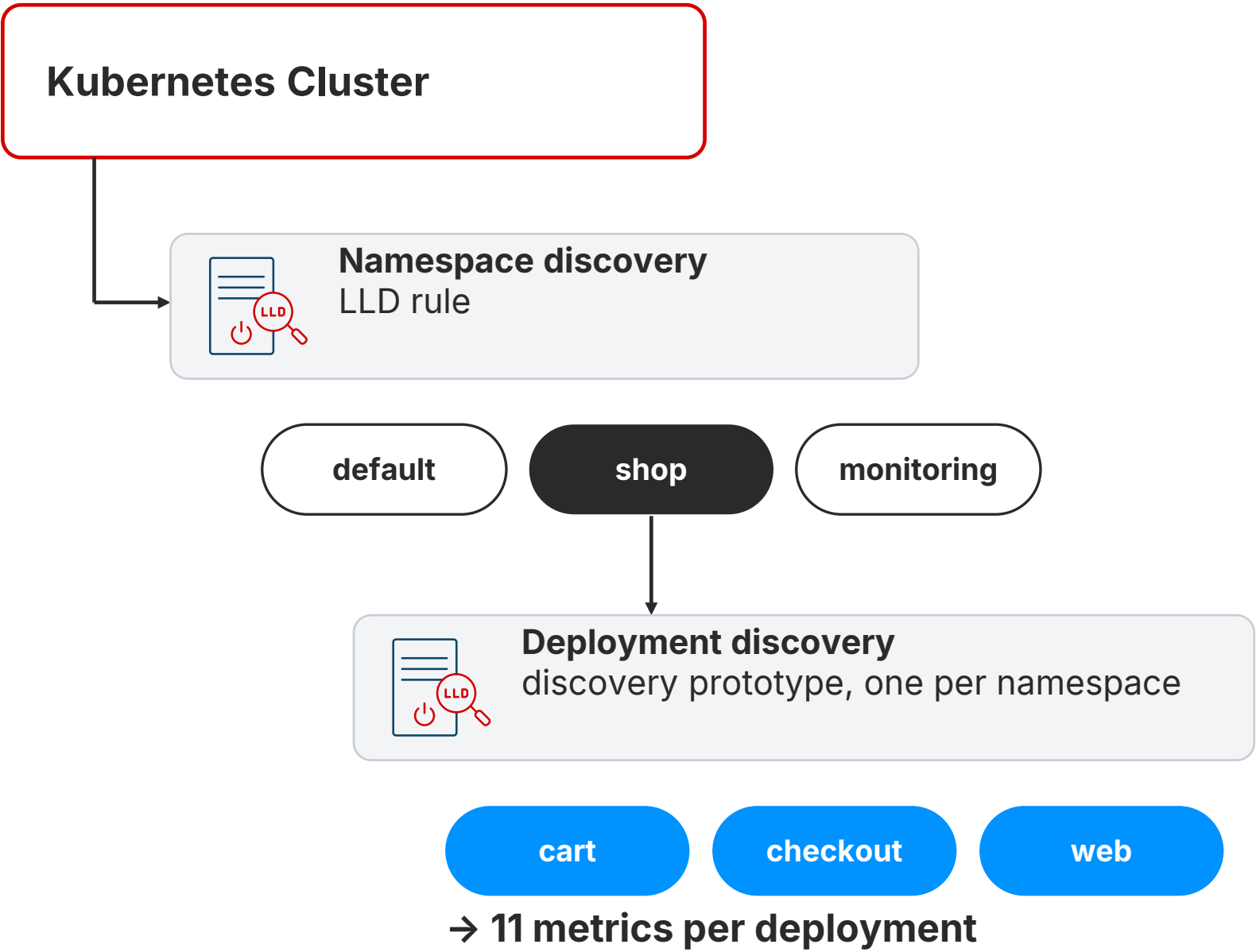
|        |                                              |
|--------|----------------------------------------------|
| CPU    | usage · share of cluster · requests · limits |
| Memory | usage · share of cluster · requests · limits |
| Pods   | running · pending · failed                   |
| Phase  | Active or Terminating                        |

### Alerts

|         |                          |
|---------|--------------------------|
| High    | More than 3 pending pods |
| Warning | Namespace is terminating |

```
Per-namespace threshold:
{ $KUBE.NAMESPACE.POD.PENDING
  .THRESHOLD:"prod"} = 1
```

# Workloads: nested discovery



## Per deployment

|          |                                              |
|----------|----------------------------------------------|
| CPU      | usage · share of cluster · requests · limits |
| Memory   | usage · share of cluster · requests · limits |
| Replicas | desired · available · mismatch               |

Pods are matched via owner info in kube-state-metrics: pod → ReplicaSet → Deployment.

**Warning** Replicas mismatch — window set per object:

|                                                                      |     |
|----------------------------------------------------------------------|-----|
| <code>{ \$KUBE.REPLICA.MISMATCH.EVAL_PERIOD:&lt;context&gt; }</code> |     |
| <code>(default)</code>                                               | #5  |
| <code>"deployment:shop:cart"</code>                                  | #3  |
| <code>regex:".*:default:.*"</code>                                   | 15m |



Discovery prototypes (nested LLD) arrived in Zabbix 7.4

# Pods and storage: lean on purpose

2

## items per pod

Classic: 29 items, plus 6 per container

### What a pod still tells you

Container restarts

Warning

Crash looping

Status phase

Average

Failed

Warning

Pending

Warning

Unknown

### Why so lean?

Pods come and go. Usage is tracked per namespace and deployment; lost pods are removed at once.



### Storage

#### Persistent volumes

capacity · phase · totals per phase

Warning

PV failed

#### Claims (PVC)

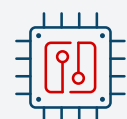
requested size · phase · totals per namespace

Warning

PVC pending

# Nodes without an agent

Kubernetes Node by HTTP: one host per node, fed by node-exporter



## CPU

Utilization · load 1/5/15 · cores



## Memory

Total · available · cache · buffers · swap · utilization



## Pressure (PSI)

NEW

CPU waiting · memory and I/O waiting / stalled



## Network

Per interface: traffic · packets · errors · speed · link



## Filesystems

Size · available · used · utilization per mount point



## Disks

I/O utilization · read/write bytes · IOPS

## Why pressure?

Utilization tells you how busy a node is.

**Pressure stall information (PSI) tells you who is waiting.**

Share of time tasks stall on CPU, memory or I/O (Linux PSI).

### Alerts include

- CPU pressure > 20% / 50%
- CPU or memory > 90%
- Filesystem > 80% / 90%
- Disk I/O ≥ 80%, link down, net errors

# Dashboards out of the box



Kubernetes Cluster · 4 pages

Overview

Nodes, NotReady, namespaces, pods, containers, livez, cluster CPU and memory, utilization graphs, object counts, problems

Namespaces

Top namespaces by CPU and memory, usage graphs, honeycombs of pending and failed pods

Workloads

Top deployments by CPU and memory, usage graphs

PV & PVC

PVC requested storage, PV capacity, honeycombs of PV and PVC status



Kubernetes Node

1 page per node

- Gauges: CPU, memory, swap
- Pressure top list
- Network and filesystem graphs
- Root filesystem pie charts
- Load, uptime, problems

## Widgets used

Top items

Honeycomb

SVG graph

Gauge

Pie chart

Item value

Problems by severity

Scatter plot

# Dashboards out of the box

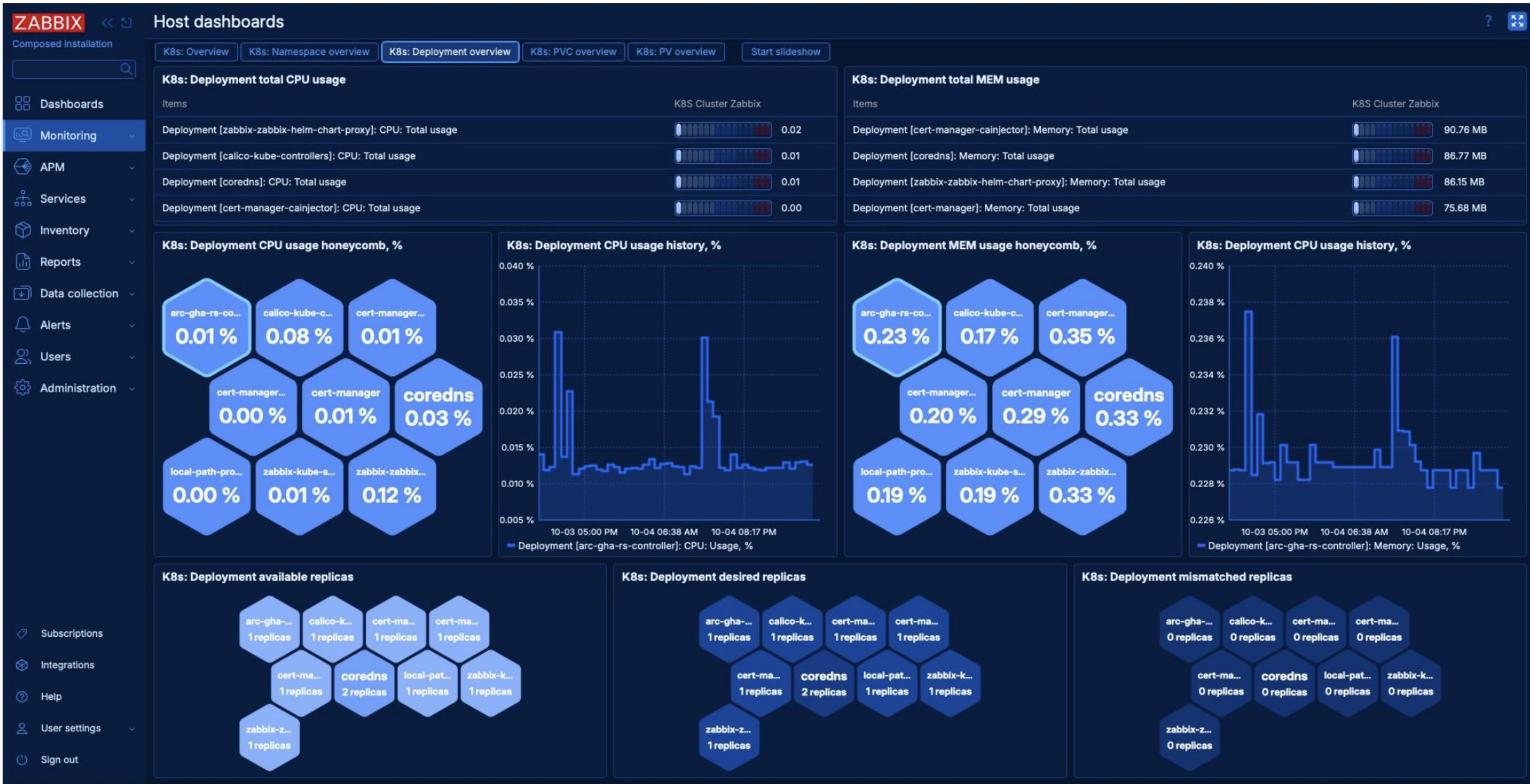


**ZABBIX**  
**SUMMIT<sup>26</sup>**  
OCTOBER 7 - 9, 2026  
RIGA - LATVIA

## Widgets used

- Top items
- Honeycomb
- SVG graph
- Gauge
- Scatter plot
- Problems by severity
- Pie chart
- Item value

# Dashboards out of the box



## Widgets used

- Top items
- Honeycomb
- SVG graph
- Gauge
- Scatter plot
- Problems by severity
- Pie chart
- Item value



OCTOBER 7 - 9, 2026  
RIGA - LATVIA

# Classic vs new, side by side

|                                       | Classic (6.0 – 7.4)                           | New (8.0)                      |
|---------------------------------------|-----------------------------------------------|--------------------------------|
| Templates                             | 6                                             | 2                              |
| Hosts you create                      | 2                                             | 1                              |
| Hosts for 3 control-plane + 6 workers | 29                                            | 10                             |
| Agent on every node                   | Zabbix agent 2 DaemonSet                      | Not needed (node-exporter)     |
| Items per pod                         | 29, plus 6 per container                      | 2                              |
| One crash-looping pod                 | 2 duplicate problems                          | 1 problem                      |
| API server health                     | componentstatuses (deprecated) + readyz/livez | readyz + livez                 |
| Namespace view                        | —                                             | 12 metrics per namespace       |
| Deployment CPU & memory               | —                                             | Usage, requests, limits, share |
| Node pressure (PSI)                   | —                                             | CPU, memory, I/O               |
| Dashboards                            | Classic graphs on component hosts             | Cluster (4 pages) + node       |
| API token macro                       | Plain text                                    | Secret                         |

## Built for **big** clusters



### Trim at the source

Limit kube-state-metrics collectors and metrics in the chart values.



### Filter discovery

Include or exclude namespaces, nodes and PVs with the `{ $KUBE.LLD.FILTER.* }` macros.



### Tune per object

Context macros: thresholds per namespace, deployment (regex), filesystem, interface.

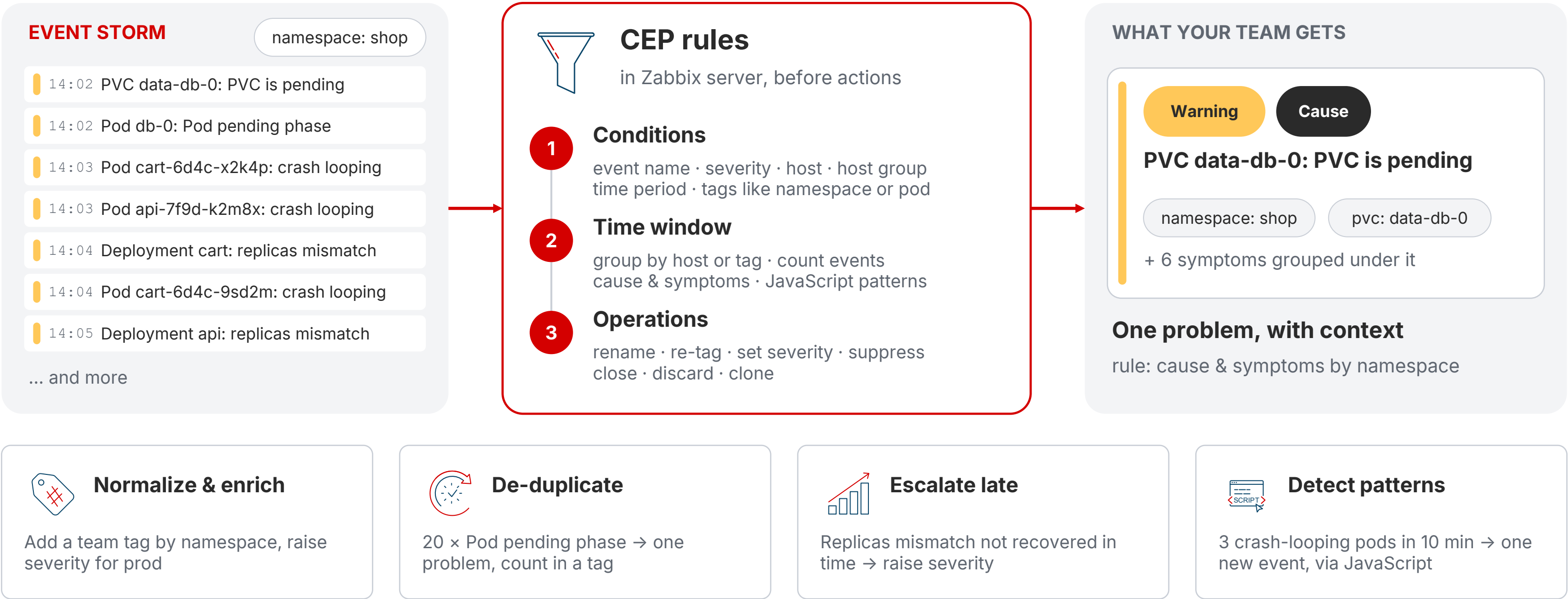


### Lean by design

2 items per pod, lost pods removed at once, slow values stored on change.

# 8.0: Complex event processing

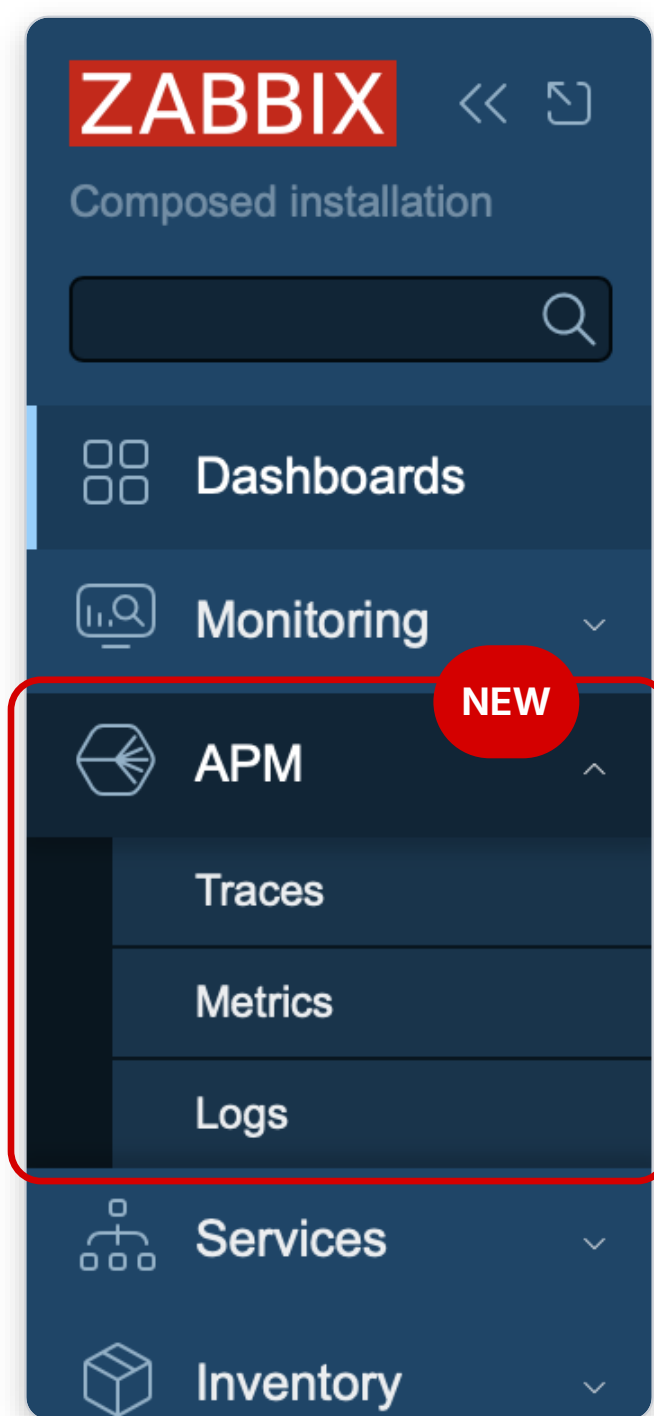
Shape the event stream before it becomes problems and alerts



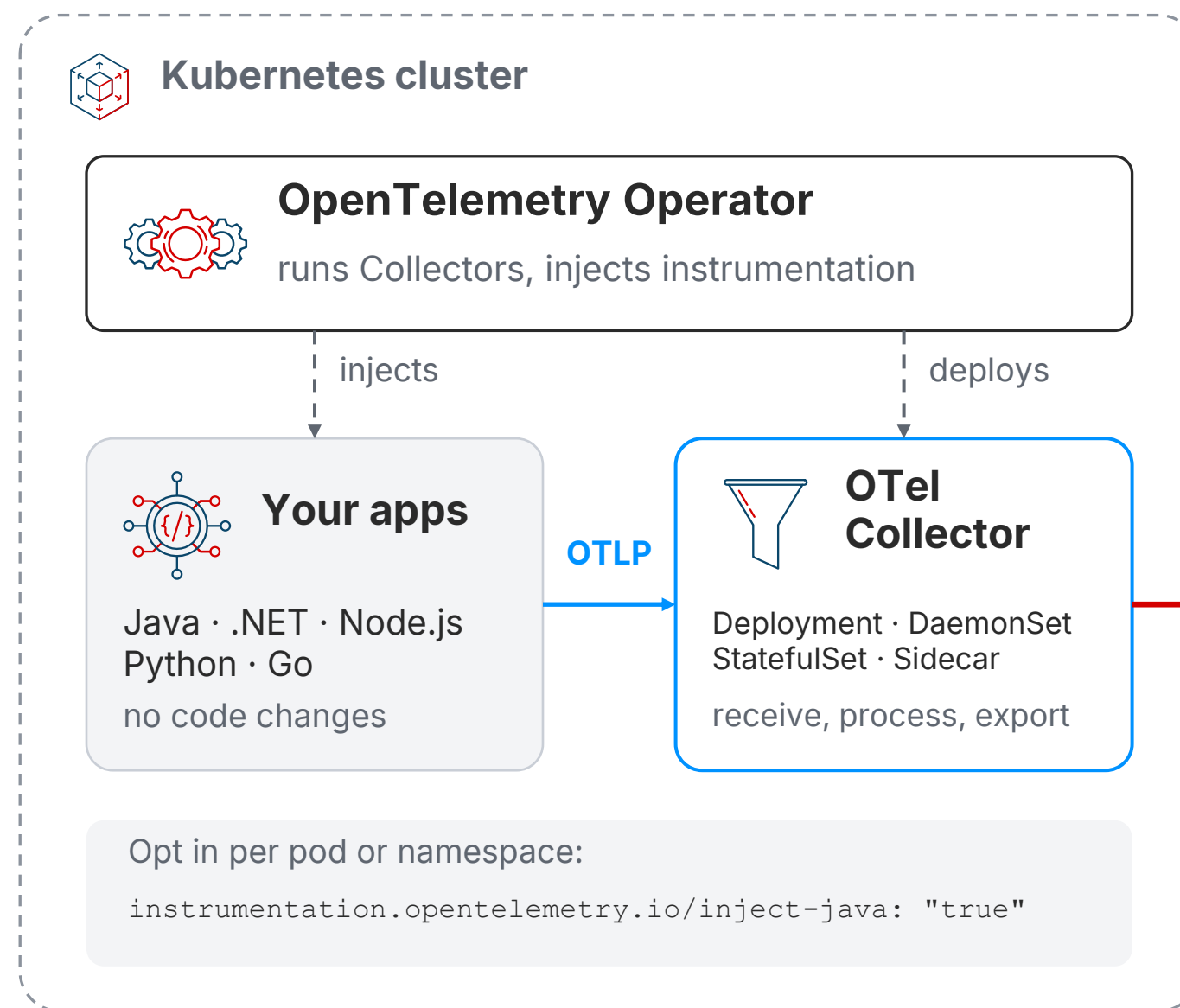
**Where:** Data collection → Event processing

Changes made by rules show up in the event history

## 8.0: APM with OpenTelemetry



8.0 UI



### Zabbix 8.0

+ ClickHouse

- APM: Traces · Metrics · Logs**  
a new menu, next to Monitoring
- Telemetry query items**  
spans, logs, metrics → items and triggers
- Ready OT templates (in-progress)**  
Generic · Envoy Proxy  
Proxmox VE · OpenRouter
- Same alerting and dashboards**  
as for your Kubernetes infrastructure



**We've got you covered.**

**Infrastructure** → new Kubernetes templates  
**Applications** → OpenTelemetry APM

## 8.0: ClickHouse/ElasticSearch: performance internals

### THE RESULT

# 1 M+

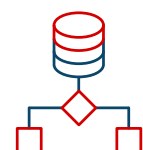
values per second, processed and written to the database

# 1 M+

triggers

# 0

preprocessing queue



### Async history providers

Pick ClickHouse, Elasticsearch or SQL per value type. Providers are async, over HTTP(S). Was: synchronous, Elasticsearch only.

```
HistoryProvider=clickhouse;value_types=...
```



### Batched cache warm-up

Reads only what's about to be evaluated. One query returns recent rows for N items.

**1M values: 267–320 s → 0.6 s**



### Optimized history sync

Batch per cycle 1000 → 2000. No hashing under locks. One failing provider no longer fails the cycle.

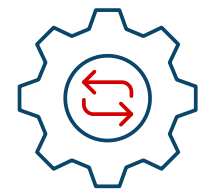
**Warm-up SQL requests: 4M → 2K**



### Offload to OLAP

Stop doing work someone else can do for you: the OLAP engine takes the load off Zabbix.

**No trends · No housekeeping**



## Moving from classic

Requires Zabbix 7.4 & 8.0

Tested with:

- Kubernetes 1.32.6+
- kube-state-metrics chart 7.2+
- node-exporter chart 4.51+

### Both sets ship with 8.0

The classic templates stay — still the place for control-plane internals: API server, scheduler, controller manager, kubelet.

### Run side by side, then switch

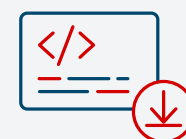
It's a new host, so history doesn't carry over. Keep the classic hosts until the new ones have enough history.

### Check the chart defaults

Chart 2.0.0 turns the Zabbix agent DaemonSet off. Still using classic node hosts? Set `zabbixAgent.enabled: true`.

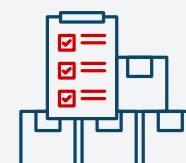
# Try it today

Zabbix 8.0 · templates and Helm chart on  
[git.zabbix.com](https://git.zabbix.com)



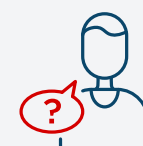
## Templates

```
zabbix repo >  
templates/app/kubernetes_cluster_http
```



## Helm chart

```
kubernetes-helm repo > master  
chart v2.0.0
```



## Feedback

[support.zabbix.com](https://support.zabbix.com)  
or  
find me after this talk



# Thank you!

Questions?

**ZABBIX**  
**SUMMIT**<sup>26</sup>  
OCTOBER 7 ■ 9, 2026  
RIGA ■ LATVIA