

Virtualization and containers

Marina Generalova | Integration Delivery Manager, Zabbix



Solutions

Six templates — VMware, Proxmox, Nutanix, Hyper-V, Docker and Podman. Three of them changed in 8.0.
More to come.

What matters?

A real-life story from our own experience.



Zabbix 8.0: What's new

Proxmox, Hyper-V, Podman

Nested LLD for even easier discovery and scaling along with the cluster

New dashboards with the key metrics at a glance

Expanded monitoring for nodes, VMs, LXC containers, and even per-interface traffic, to spot spikes on a specific interface

Customizable and flexible alert thresholds

Demo

Proxmox setup



Proxmox: what you get

Cluster overview

Memory, CPU, and storage across the whole cluster — what is left to allocate

Nodes and LXC

Performance per node and per container, discovered automatically

Quiet problems

Certificates, user activity, SMART status and pending updates — alerted, not found later

Placement

A VM or container moving between nodes raises a problem, planned or not

How the data is collected

User data

Users and login activity in the cluster

Node data

Nodes, their metrics, and the VMs and containers running on them

Cluster resources

Shared storage and per-VM detail.

Dependent item:

JSONPath: \$.data.cluster.memory_total

Discard unchanged with heartbeat: 1h

Dashboards

Monitoring

Problems

Hosts

Latest data

Maps

Discovery

Services

Inventory

Reports

Data collection

Alerts

Users

Administration

Subscriptions

Integrations

Help

6.1.2

linux/amd64

PODMAN API

version

6.1.2

PODMAN IMAGES TOTAL SIZE

166.36

MB

my-nginx

Exited (2)

zbx-created

Created (6)

zbx-exited

Created (6)

zbx-normal-01

Exited (2)

zbx-normal-02

Restarting (4)

zbx-normal-03

Running (1)

zbx-normal-04

Exited (2)

zbx-normal-05

Paused (3)

zbx-paused

Dead (7)

zbx-removing

Created (6)

zbx-restarting

Dead (7)

zbx-running

Created (6)

zbx-stopped

Unknown (0)

zbx-stopping

Stopped (9)

Host: CPU utilization

	min	avg	max
Host: System CPU utilization	2.0523 %	7.1632 %	11.9879 %
Host: User CPU utilization	15.5079 %	30.7928 %	44.7752 %
Host: Idle CPU utilization	48.5581 %	63.9395 %	78.952 %

Host: Swap memory usage

	min	avg	max
Host: Used swap memory	1.05 GB	1.9 GB	2.98 GB
Host: Free swap memory	3.1 GB	5.02 GB	6.92 GB
Host: Total swap memory	10 GB	10 GB	10 GB

Host: Memory usage

	min	avg	max
Host: Used memory	14.05 GB	14.81 GB	15.28 GB
Host: Free memory	480.79 MB	962.21 MB	1.7 GB
Host: Total memory	15.75 GB	15.75 GB	15.75 GB

Severity

Disaster

High

Average

Problem history list

Time	Recovery time	Status	Info	Host	Problem • Severity	Operational data	Duration	Update	Actions
02:26:47 PM		PROBLEM		PODMAN	Podman: Container [zbx-running]: Not in normal state	Container status: Created (6), Normal state: 1	1m 23s	Update	
02:26:27 PM		PROBLEM		PODMAN	Podman: Container [zbx-normal-05]: Not in normal state	Container status: Paused (3), Normal state: 1	1m 43s	Update	

Proxmox VE: Overview

Proxmox VE: VMs

Proxmox VE: LXCs

Start slideshow

QEMU: Navigator

▼ docker

Node [pve]: QEMU [docker]: CPU utilization

Node [pve]: QEMU [docker]: Memory utilization

Node [pve]: QEMU [docker]: Network input speed

Node [pve]: QEMU [docker]: Network output speed

▼ vm-01

Node [pve]: QEMU [vm-01]: CPU utilization

Node [pve]: QEMU [vm-01]: Memory utilization

Node [pve]: QEMU [vm-01]: Network input speed

Node [pve]: QEMU [vm-01]: Network output speed

▼ vm-02

Node [pve]: QEMU [vm-02]: CPU utilization

Node [pve]: QEMU [vm-02]: Memory utilization

Node [pve]: QEMU [vm-02]: Network input speed

Node [pve]: QEMU [vm-02]: Network output speed

QEMU: Graph

	min	avg	max
Node [pve]: QEMU [docker]: CPU utilization	0.8643 %	1.2515 %	3.7064 %

VM: CPU utilization

Items	PROXMOX
Node [pve]: QEMU [docker]: CPU utilization	1.30 %
Node [pve]: QEMU [vm-01]: CPU utilization	1.02 %
Node [pve]: QEMU [vm-02]: CPU utilization	0.95 %

VM: Status

docker

Running (2)

vm-01

Running (2)

vm-02

Running (2)

VM: Memory utilization

Items	PROXMOX
Node [pve]: QEMU [vm-02]: Memory utilization	73.37 %
Node [pve]: QEMU [vm-01]: Memory utilization	71.40 %
Node [pve]: QEMU [docker]: Memory utilization	69.66 %

Hyper-V

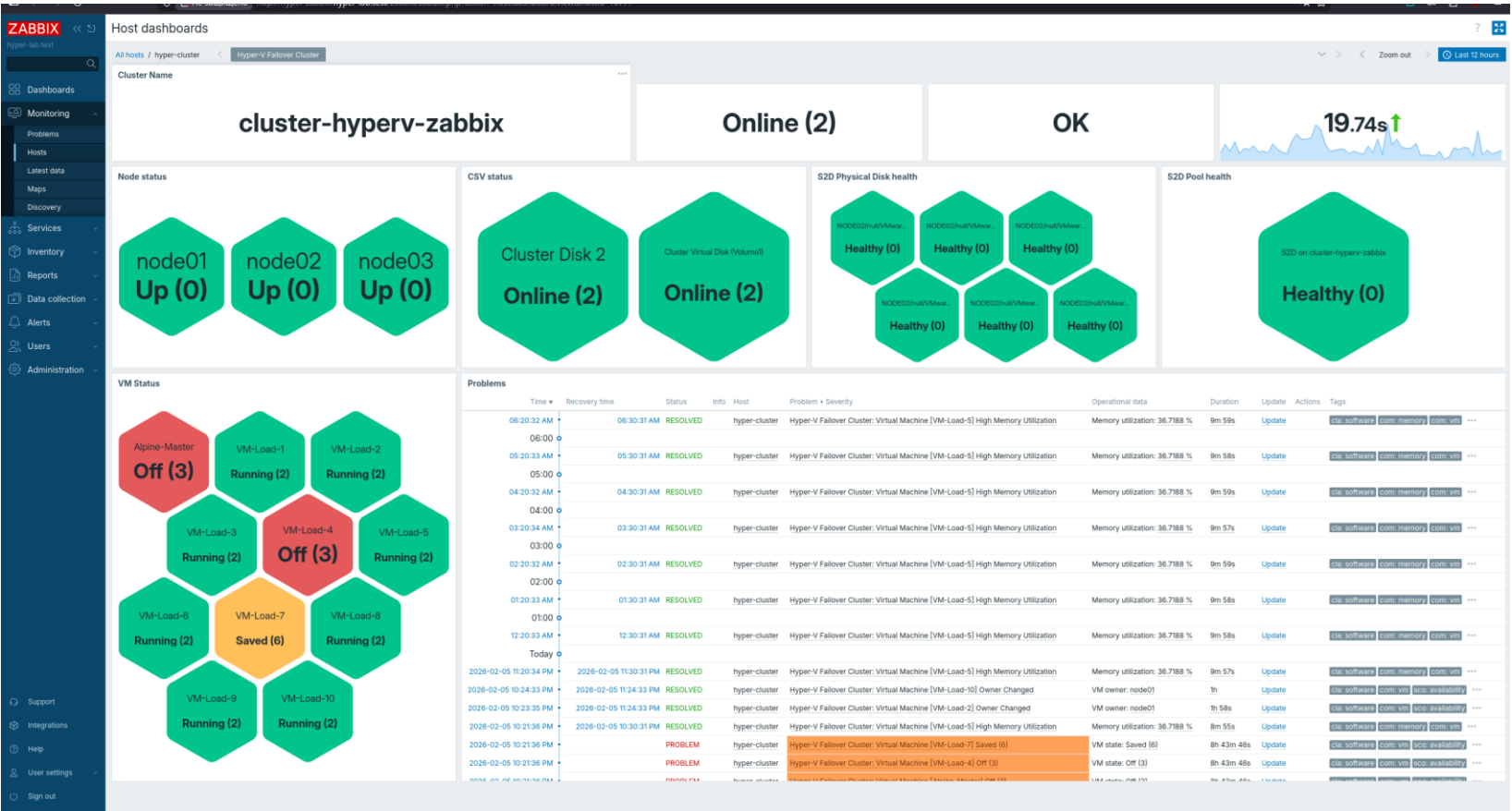
Different protocol & structure, same goal

Works over SSH

Automated node, VM, and CSV discovery

Templates for Standalone and Cluster

Available since Zabbix 8.0



VMware

Sophisticated triggers: Maintenance mode suppresses every alert from VMs on that hypervisor

Monitors snapshot count, age and consolidation — the classic way a datastore fills up

vCenter's own alarms and hardware sensors, discovered into Zabbix

3 nested templates: VMware (generic data on datastores, alarms, clusters, hypervisor and VM discovery) > VMware Hypervisor > VMware guest

Higher user permission granularity, but more hosts

Nutanix

Same structure as VMware: Nutanix Prism Element > Cluster > Host

Hosts and clusters are discovered automatically

Will be refactored to include more VM metrics via nested LLD

Same questions, different platforms

Is there room left?

Proxmox: cluster memory and storage. VMware: datastore free space. Hyper-V: CSV utilization.

Survive a failure?

Hyper-V and Proxmox: quorum.
Nutanix: redundancy factor.
VMware: cluster overcommitted.

Failing quietly?

Proxmox: disk SMART. VMware: hardware sensors. Nutanix: host degraded.

Is this planned?

VMware: maintenance mode suppresses the VMs on that host.
Nutanix: pending reboots.

Key things

Virtual machines

Early warning signs:

Memory at the cluster level

Storage, with enough warning to act

VMs switching nodes

Unexpected restarts

Detailed diagnostics: CPU, disk I/O,
network

Containers

Key warning signs:

Container state

Memory-against-limit

No data coming back

Detailed diagnostics: CPU, disk I/O,
network, mounts

The pattern

With VMs you can prevent issues by acting on early cluster-level warnings; with containers it's about knowing immediately and acting before users notice.

Containers: Docker and Podman

Docker

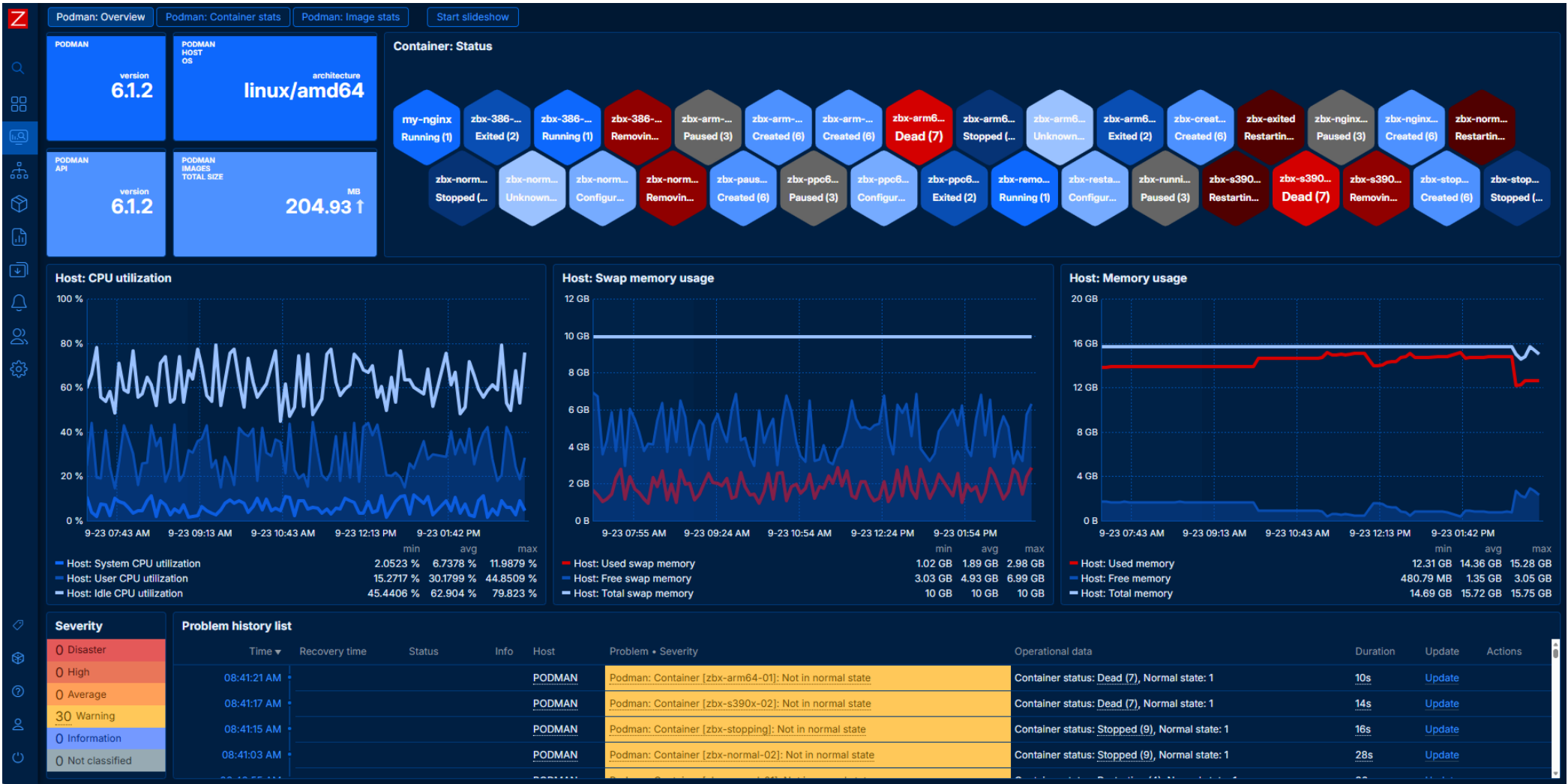
Dedicated agent 2 plugin
Exit codes, OOMKilled and restart counts. HEALTHCHECK failure alert

Podman

Agent or HTTP (use firewall or VPN)
Per-container thresholds: each container gets its own normal state, CPU and memory limits

Both

Every container discovered automatically, with state, CPU, memory, network and block I/O



Containers: Same questions

Is it healthy?

Docker: your HEALTHCHECK and its failing streak. Podman: per-container normal state

Hitting its limits?

Podman: memory against the limit. Docker: CPU throttling periods and time

Why did it stop?

Docker: exit code, OOMKilled and the error field

Is it still there?

Docker: running, dead, restart count. Podman: per-container collection failure



Demo

Podman



Reducing false alerts

Maintenance mode

A hypervisor in maintenance suppresses every alert from the VMs on it

Normal is per object

Thresholds, alerts and expected states are easily customizable via template macros

One cause, one alert

Trigger dependencies collapse a cascade into a single problem

Spikes and flapping

Thresholds averaged over minutes; health checks need a repeat failure

Setup and collection by platform

	Proxmox	VMware	Hyper-V	Docker	Podman
Data collection	REST	SOAP	script	socket	agent or API
Access	API token	read-only user	SSH account	agent 2 on host	agent + API service
Object discovery	automatic	automatic	automatic	automatic	automatic
Result	Actionable insights				

What you get

Control

You know what is happening, so you can act on it

Peace of mind

You know nothing is happening, so you can work on something else

Same setup

Link a template, set macros, discovery does the rest

Next

Nutanix with nested LLD, then more virtualization platforms

Questions

Missing a platform? Tell us. Community track is tomorrow.

