

ZABBIX SUMMIT 2026

Multi-Tenant Zabbix for MSPs

Scaling customers without scaling the ops team

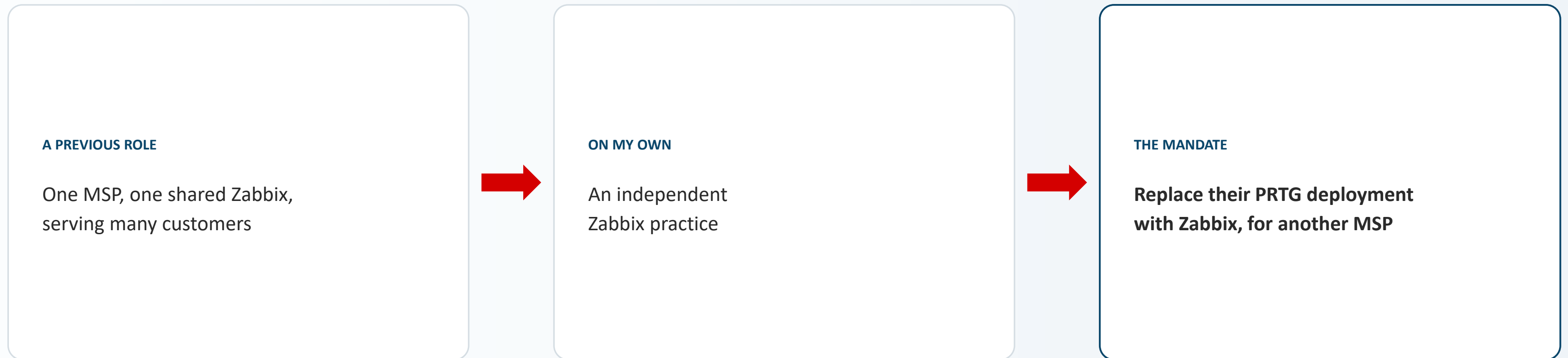


Alexandre Georges
mageo services sàrl



OCTOBER 7 ▪ 9, 2026
RIGA ▪ LATVIA

Where this comes from



**18 months later: in production, for them and for their customers,
proven in real conditions.**

Four walls of the shared instance

One instance, many customers: the limits are in the model, not in Zabbix.



Naming conflicts

no namespace per customer → every new tenant renegotiates the taxonomy



Template forking

one custom threshold → dozens of near-identical templates nobody dares touch



No upgrade window

one slot for all → the version freezes at the most conservative customer



Permissions

isolation lives in permission rules → hard to prove who sees what

The great idea

Let's have a dedicated instance per customer,
and use automation to solve all our problems.

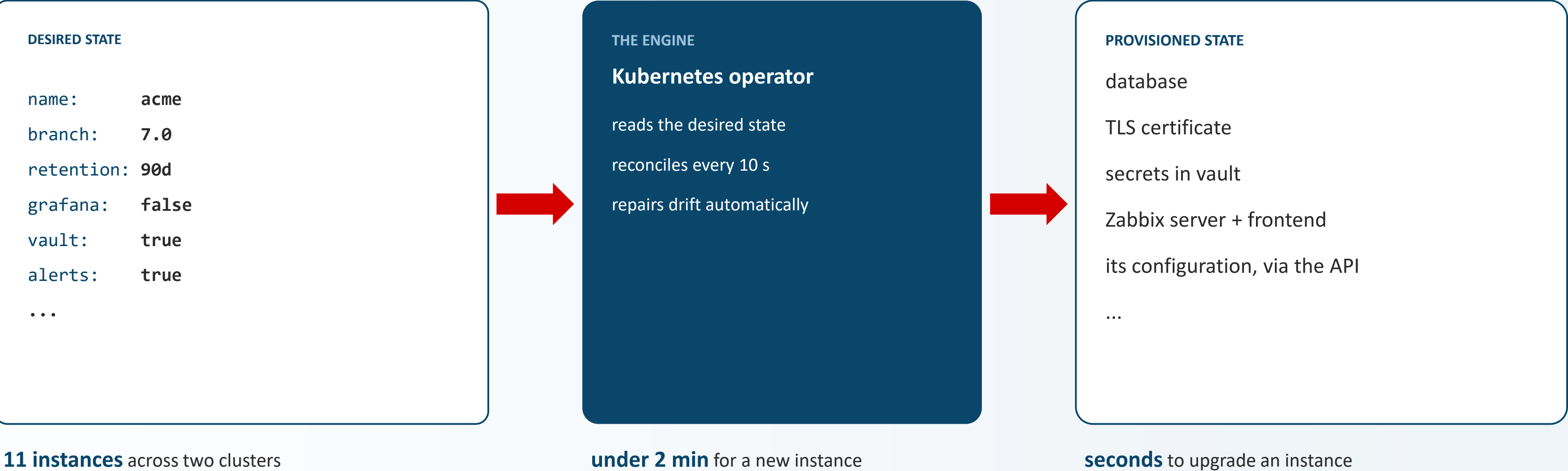
That will be easy!TM

Two models, and the third that takes the best of both

	Shared instance	Instance per customer	Shared infrastructure, dedicated instance
Isolation	Permission rules	Absolute	Absolute
Upgrades	One window for everyone	Per customer	Per customer
Templates	Clone and re-clone forever	Free	Free, from one catalogue
Each new customer costs	Little, at first	A full stack, and hands to run it	A form

The third column collects the winning cell of every row. So how do you build it?

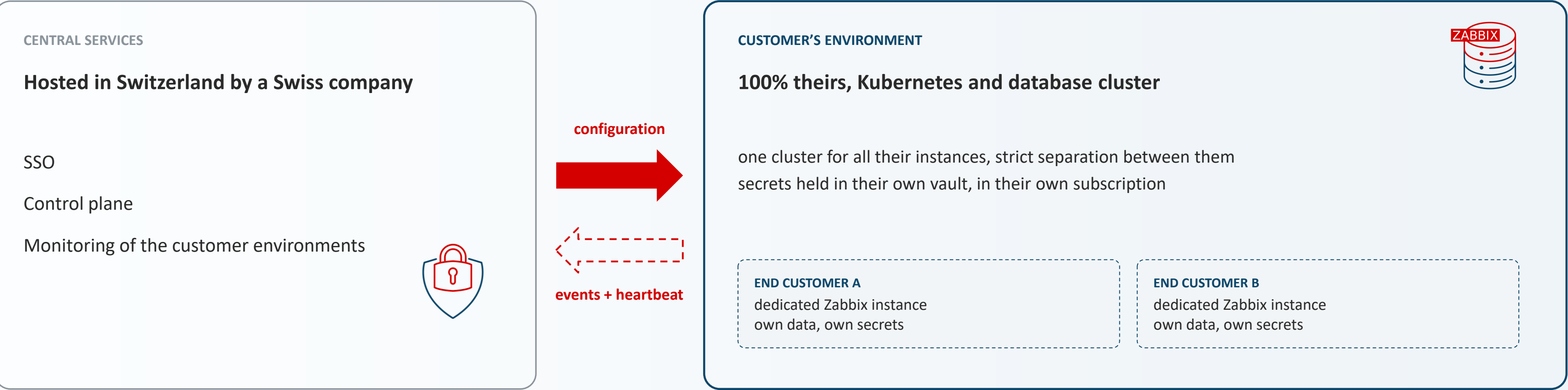
Provisioning: describe the tenant, don't build it



Onboarding and repair are the same code path. You do not need a platform team, you need a control plane.

And when the control plane is down, nothing downstream stops: every instance keeps monitoring on its own.

Where the data lives



Monitoring data never leaves the customer's cluster, and each end customer sits behind its own boundary.

Unmodified upstream Zabbix

No fork. No patches. No private build.

What ships upstream is what runs.

The platform drives Zabbix through its standard API, never by changing the product.



Nothing to unlearn: standard product, standard docs.

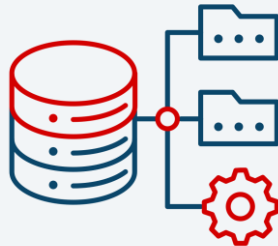
No derivative to maintain, no patch debt.

Fleet today: 7.0 LTS and 7.4. 8.0 LTS next.

Templates, revisited: one catalogue, many instances

PUSH

- one catalogue, outside Zabbix
- pushed to every instance through the API
- identity survives renaming and re-import
- nothing is ever deleted without asking
- drift detection → which instances are **outdated**



ONE REVISION PER BRANCH

7.0 LTS

7.4

8.0 LTS

Each template is published once per Zabbix branch.
An instance receives the revision for its branch.
Forward-only.

Move an instance to the next branch and it picks up that branch's revision. **No re-authoring.**

ONE CATALOGUE, TWO ORIGINS

Zabbix official templates: taken as shipped, never edited in place

The customer's own templates: theirs to author, same versioning, same push

No more clone-and-re-clone, and no more upgrade window that suits everybody: two of the four walls fall here.

What we build once, for every tenant

ONE IDENTITY

Same login everywhere

- one SSO for every instance and every app
- your own identity provider, whichever it is
- users created just in time, groups mapped at login

ONE VIEW

Every instance on one pane

- problems and health, aggregated live
- fleet-wide dashboards, same login, same groups
- fed by events and heartbeat, never metrics

ONE ALERT PATH

Sent, and arrived

- messaging, SMS and email held centrally
- delivery tracked end to end
- one contract beats one per customer

Mutualise the plumbing, never the data.

Each piece is a plug-in, not a project: switch any of it off, even SSO, and a plain Zabbix keeps running.

What runs today, and where we are going

IN PRODUCTION

- ✓ Data plane entirely in the customer's own cluster
- ✓ Instance provisioning and lifecycle
- ✓ SSO and JIT provisioning, no manual accounts
- ✓ Per-tenant secrets, no shared credentials
- ✓ Template push and drift detection across instances
- ✓ Branch-based version policy, one revision per branch

IN BETA

- ➔ Managed proxies, on VMs, hardware and in containers
- ➔ AI-assisted cross-instance incident correlation

IN DEVELOPMENT

- Seamless secret rotation
- Improved telemetry
- Retention driven by the tenant description
- Per-instance high availability

If you are at three customers today

1

Nest templates. Never clone them

a clone is a fork you will maintain by hand forever

2

Put the authorisation decision in the group name

then there is no second system to sync

3

Turn on JIT provisioning early

far cheaper at three customers than at thirty

4

Write your conventions down before you generate them

groups, tags and hostnames drift the moment a human types them

Three things to take away

The product, unmodified

standard Zabbix, standard docs, no fork to maintain

The data, with the customer

their cluster, their database, their boundary

The tenant, described

you fill in a form, the platform builds it and keeps it that way

Questions?

Ask now, or find me later: I am around for the whole summit. Alexandre Georges, mageo services sàrl.