Agentics Day: MCP + Agents
NORTH AMERICA

Introducing Agent Substrate

Scaling Thousands of Stateful AI Agents on Kubernetes Without the Waste

Alex Van Boxel
alexvanboxel.name

BEFORE WE START

About me, and this talk

Alex Van Boxel

Principal Systems Architect

Collibra

30 YEARS OF BUILDING

Web · 3D · Middleware · NFC · Mobile · Big Data · Observability · Infrastructure

Community

Apache · CNCF

AGENDA

1. Idle agents, and actors that are not pods
2. Kubernetes, frameworks, and where Substrate fits
3. Isolation with gVisor
4. Sleep, wake up, and the birth of an agent
5. Credentials on the network path
6. A2A: sleep until the callback

PLATFORM INFRASTRUCTURE RESEARCH

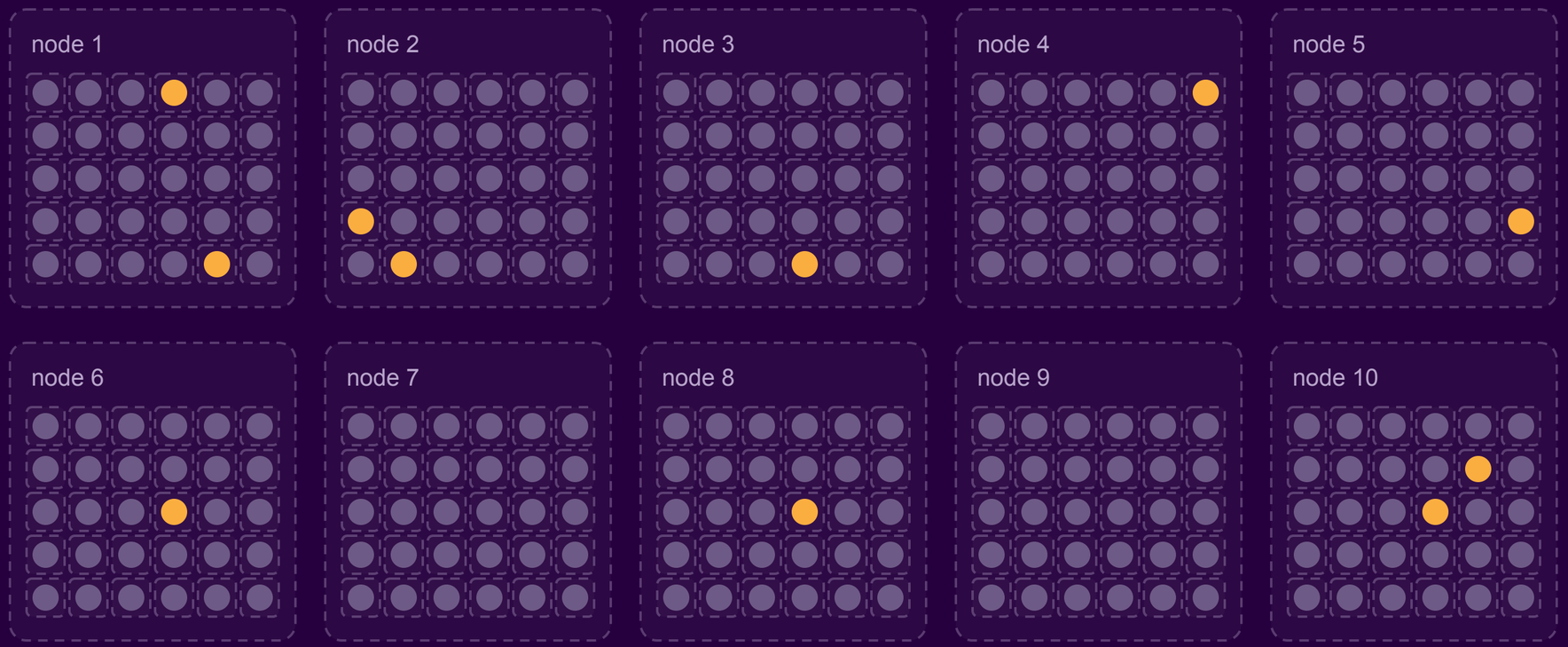
Collibra is a multi-tenant SaaS platform for data and AI governance, bringing isolation challenges.

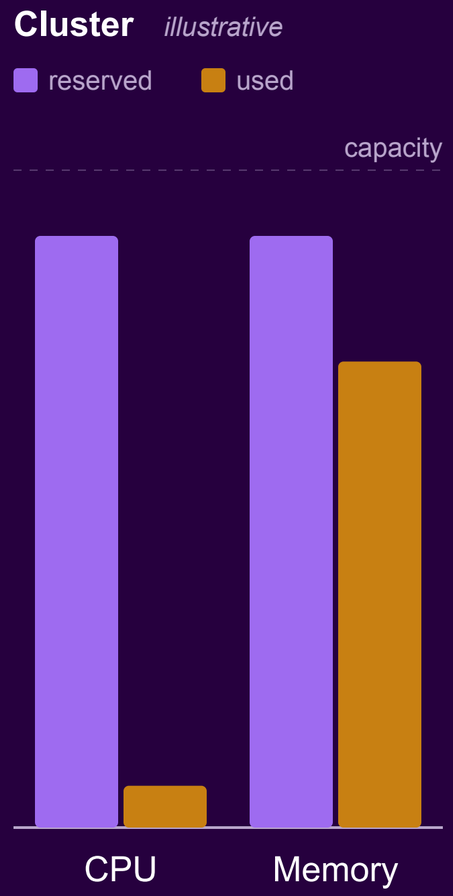
Specialised agent agents could bring untrusted code, and touch governed data, and sit idle most of the time.

Idle agent cost money, so we need to pack them densely, will keeping them isolated and not leaking credentials.

THE PROBLEM, BRIEFLY

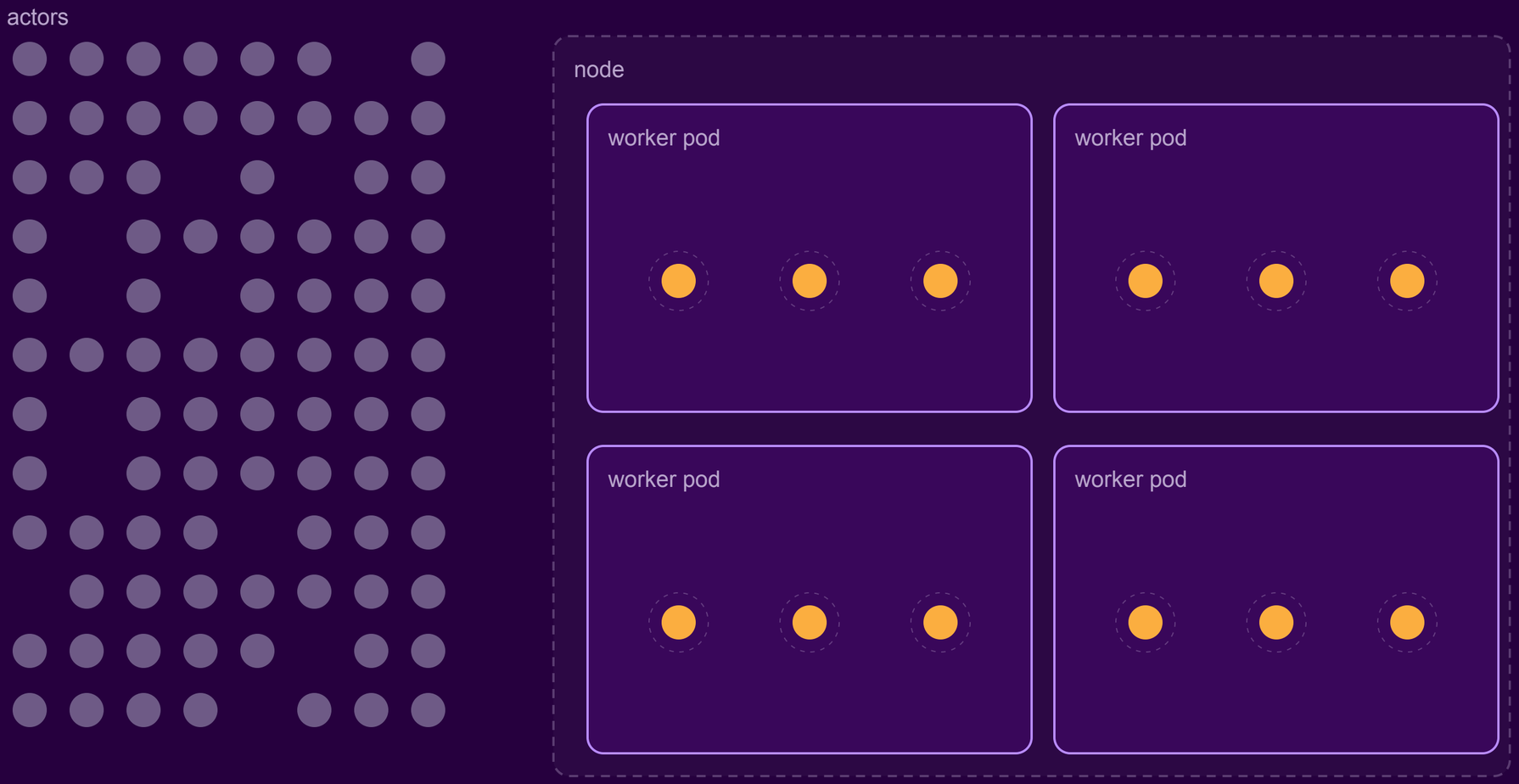
Your agent is mostly waiting





CORE IDEA

Actors are not pods



CONTROL PLANE

Kubernetes stays out of the hot path

one pod per agent

kube-apiserver → etcd → scheduler → kubelet → image pull → container start

Agent Substrate

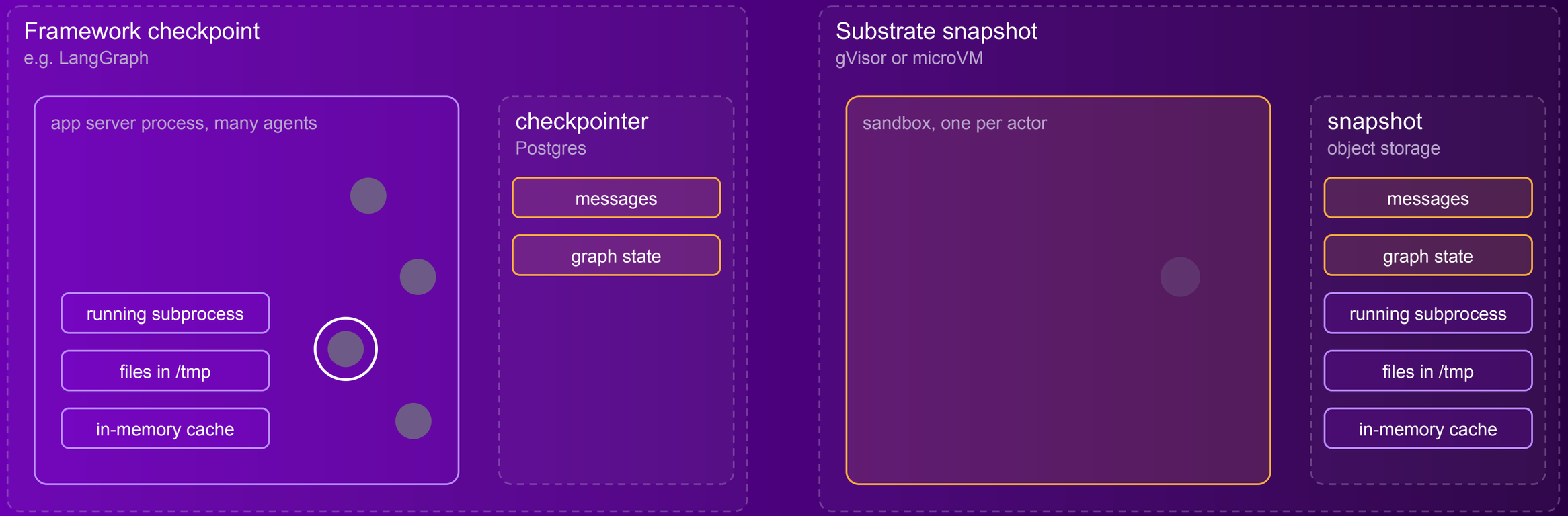
atenet-router → ate-api-server → Postgres (actor row + lease) → atelet → restore (from snapshot)

Kubernetes still runs the platform

WorkerPool and SandboxConfig are CRDs. Workers are ordinary pods, started ahead of time.

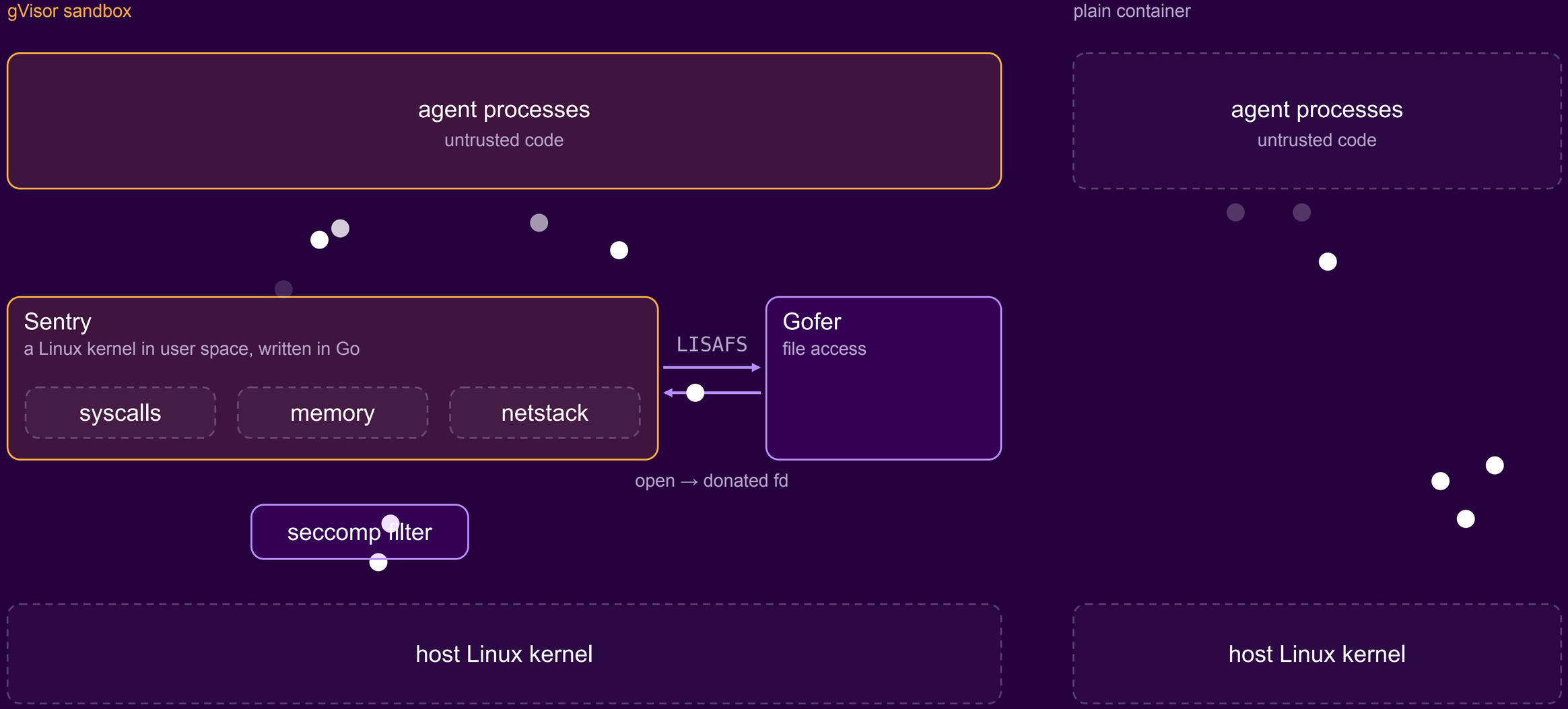
A QUICK PAUSE

Doesn't my framework do this already?



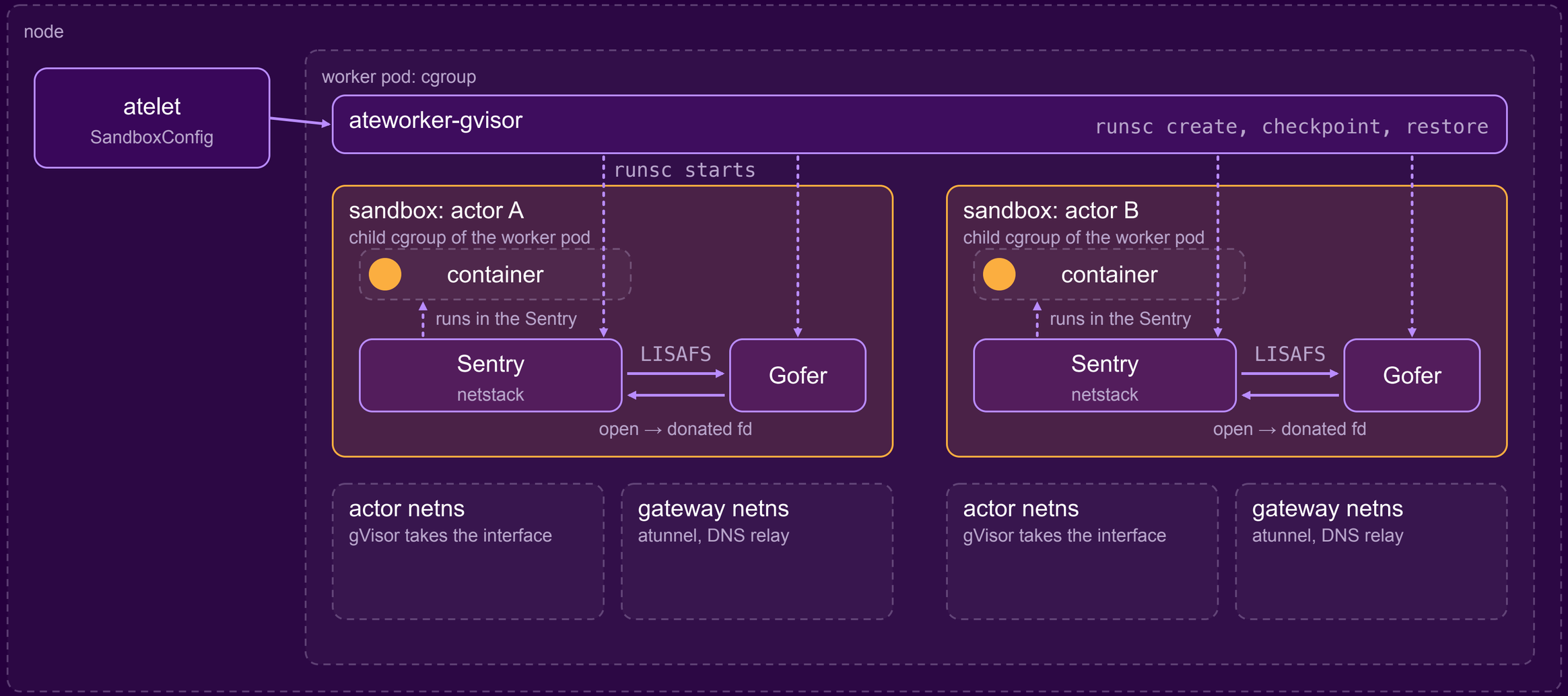
ISOLATION

How gVisor isolates an agent



ISOLATION

How Substrate runs gVisor



ANOTHER QUICK PAUSE

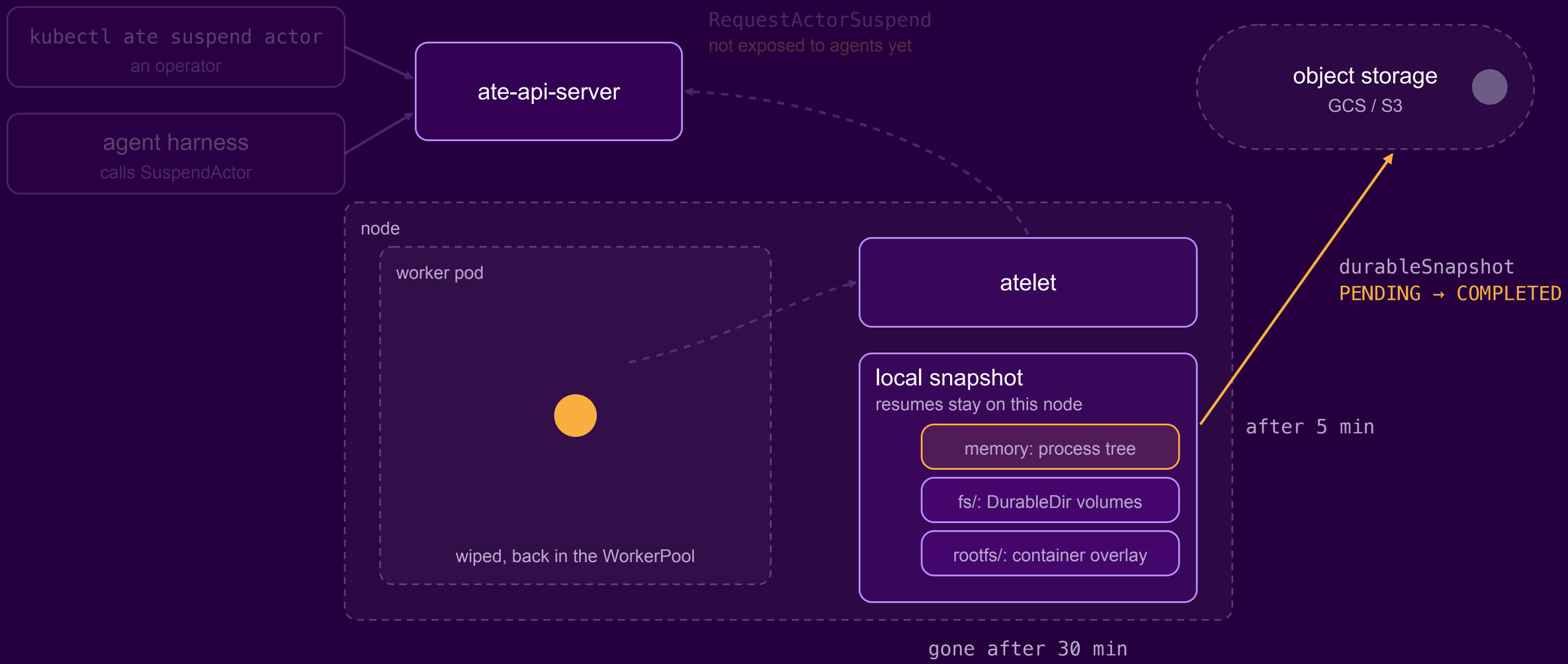
How isolated is each agent?

	framework agents in one process	gVisor the default sandbox	microVM Cloud Hypervisor, Kata guest agent
boundary	none, one shared process	a user-space kernel per actor	a guest kernel per actor
memory	one shared heap	separate per sandbox	separate per VM
host kernel sees	every syscall	a small, filtered set	the VMM and KVM
resume	re-run from checkpoint	runs while memory pages in	loads memory, then runs
workloads	code written for it	most Linux syscalls	a full Linux kernel

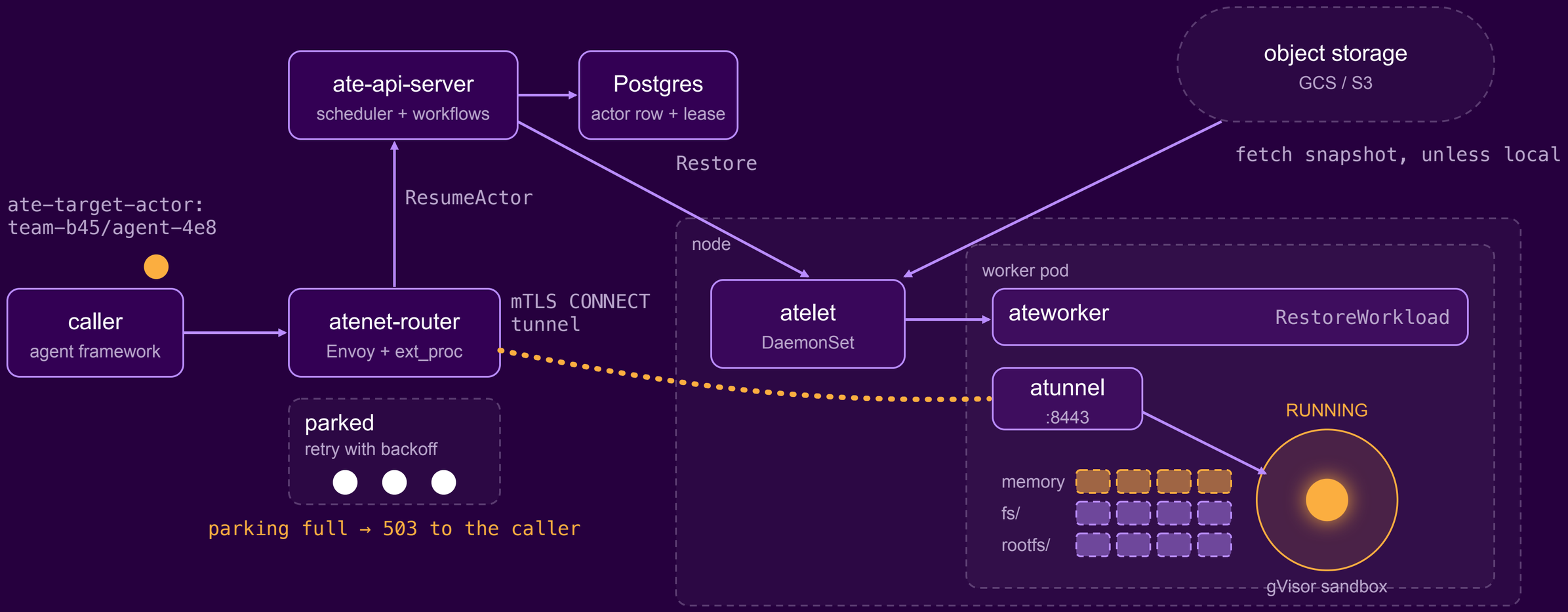
Same Substrate API for both sandboxes. Pick per WorkerPool.

GOING TO SLEEP

Suspend while the LLM thinks

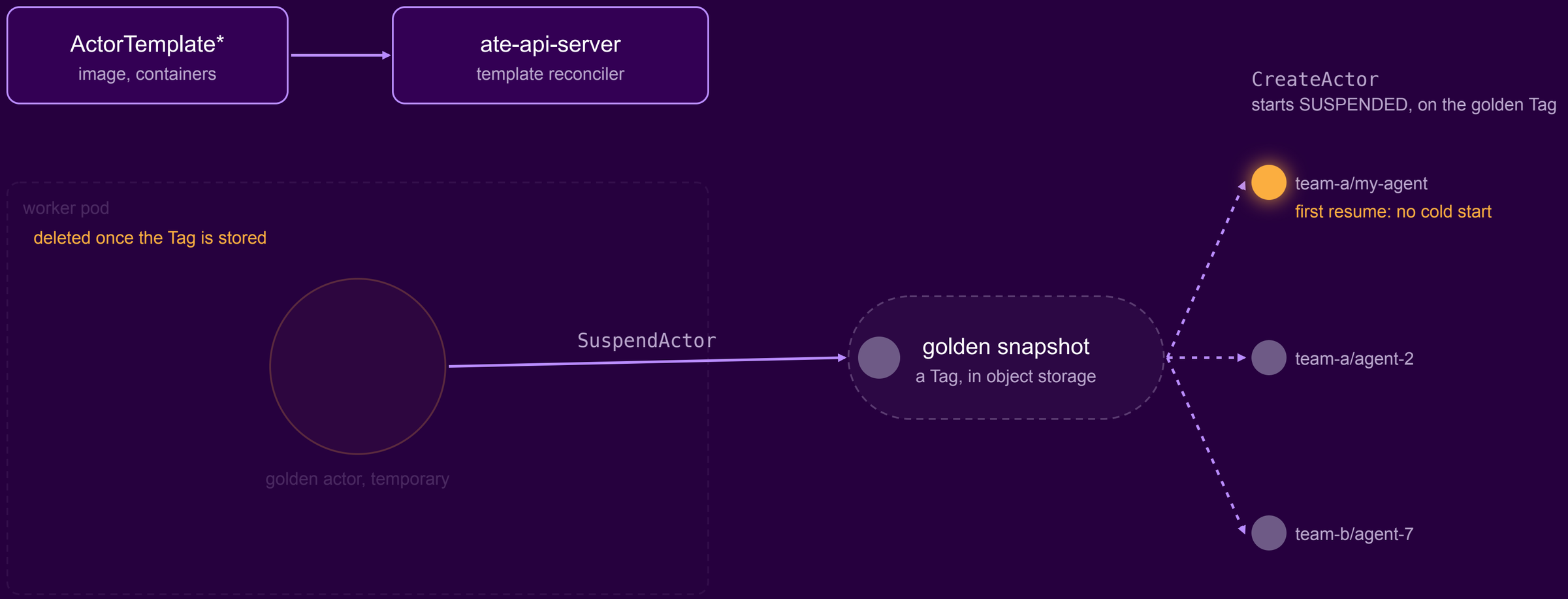


Waking up


ate-api-server
scheduler + workflows
Postgres
actor row + lease
object storage
GCS / S3
Restore
fetch snapshot, unless local
ResumeActor
ate-target-actor:
team-b45/agent-4e8
node
worker pod
caller
agent framework
atenet-router
Envoy + ext_proc
mTLS CONNECT
tunnel
atelet
DaemonSet
ateworker
RestoreWorkload
atunnel
:8443
RUNNING
parked
retry with backoff
memory
fs/
rootfs/
gVisor sandbox
parking full → 503 to the caller

BIRTH OF AN AGENT

Born from a golden snapshot



* Upgrading the container means a new ActorTemplate: an actor switches while suspended, keeps its volumes, and loses its memory.

ONE MORE QUICK PAUSE

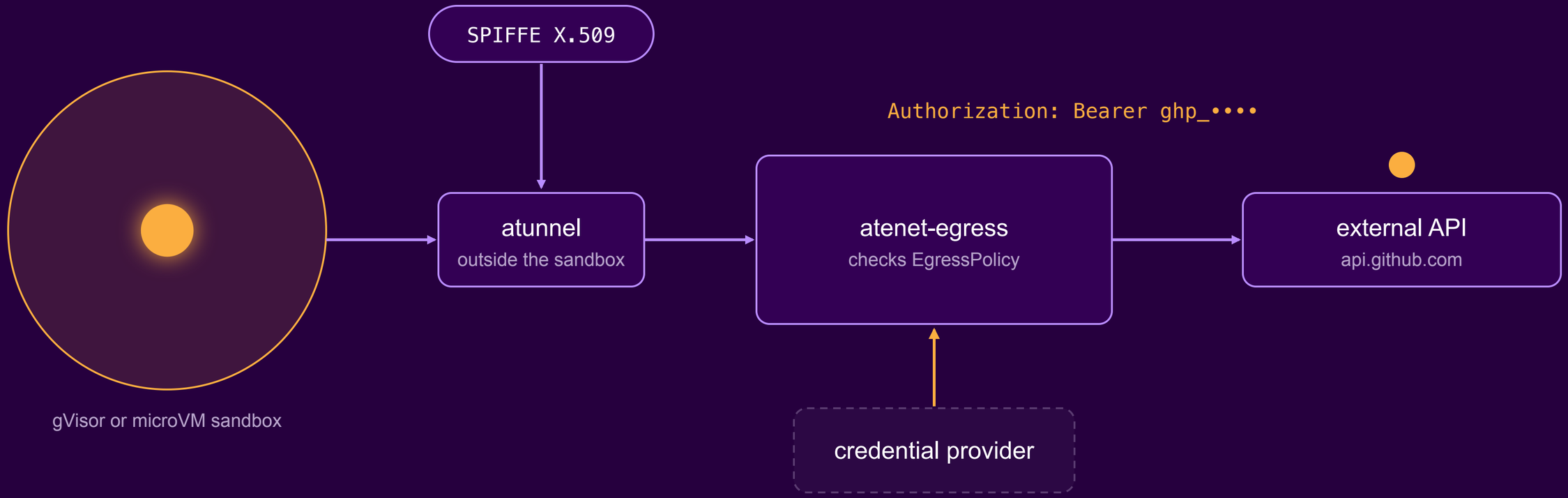
Framework, or Substrate?

	framework only e.g. LangChain in your app server	Substrate any agent, as an actor
agents	built on the framework's API	any language, any harness
isolation	tenants share one process	a sandbox per agent (tenant)
state kept	what your code checkpoints	the whole process: memory, files
idle agent	a row in a database	a snapshot in object storage
wake-up	load state, no restore	restore: under 400ms
tools	run inside the shared server	any Linux process, sandboxed
secrets	the app server holds them	injected at egress, outside
to operate	a service you already run	a new control plane

Not either/or: your LangChain agent runs inside an actor, next to any other.

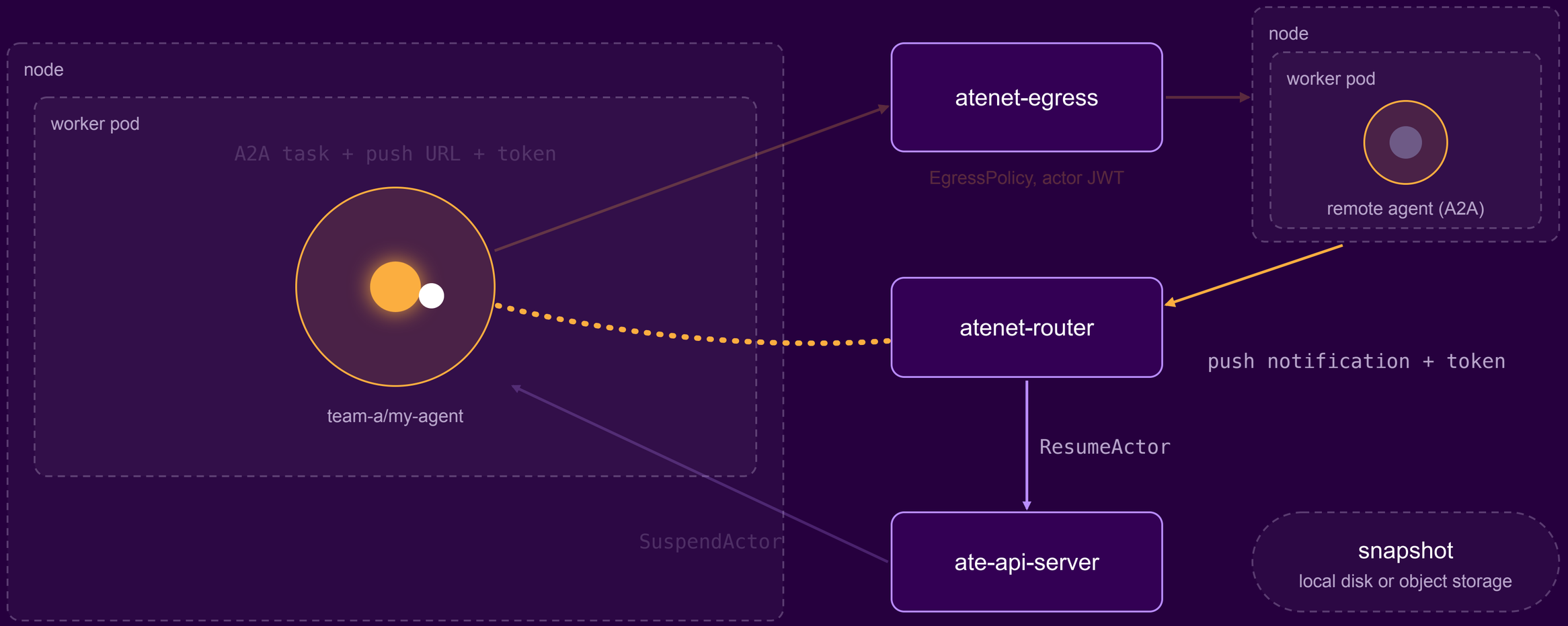
FOR MCP AND TOOL BUILDERS

Untrusted code, real credentials



PUTTING IT TOGETHER

A2A: sleep until the callback



Agentics Day: MCP + Agents
NORTH AMERICA

Thank you

THE PROJECT
github.com/agent-substrate/substrate

THESE SLIDES
alexvanboxel.name

Give feedback

Introducing Agent Substrate: Scaling Thousands of Stateful AI Agents on Kubernetes Without the Waste