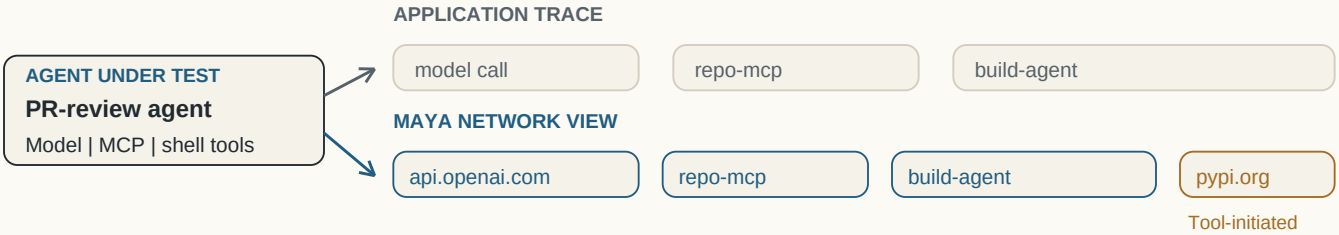
Know what your agents reach. Control what they can reach.

Agents need room to choose tools and adapt as they work. Your team remains responsible for the access you give them. Models, tools, and dependencies can change what an agent connects to on each run.

Maya attributes network activity to the agent responsible, compares it with approved boundaries, and enforces connection policy independently of the agent. It complements your logs, traces, tests, and evaluations without changing agent code. Maya Zoom* extends this with prompt and response context for deeper investigation and richer policies within approved connections.

Find unexpected dependencies when debugging failures or assessing exposure.



Build: map dependencies

Attribute observed connections to each agent, including those initiated by tools, SDKs, and third-party components.

Compare actual reach with the intended workflow, including agent-to-agent connections across processes and hosts.

Test: catch changed behavior

Spot new connections after a prompt change, model swap, or dependency update. Keep test agents away from live systems.

Check observed activity against the agent's contract in CI, and fail the build when it violates approved boundaries.

Deploy: keep environments separate

Use agent tags and policy to keep development agents away from production agents and production resources.

Keep agent rules consistent as pods restart and workloads scale. Boundaries follow the agent rather than its IP address.

Operate: investigate and contain

See which agents depend on a model, API, tool server, or peer. If a service is compromised, identify which agents reached it and when.

Limit the blast radius by restricting an affected agent or connection while permitted activity continues.

RUNS WHERE AGENTS RUN

Private cloud | Mac endpoints | Kubernetes (EKS, GKE) | Containers | AWS Bedrock | Vertex | Cloud Run

* Roadmap

Copyright 2026 Mayagentic, Inc. All rights reserved.

Use Maya in your development workflow.

Ask your developer assistant about the agent you are building or testing.

YOU	Did the SDK update change how the refunds agent reaches Stripe?
YOUR ASSISTANT	Yes. At **14:07**, it attempted to contact api.stripe.com directly for the first time. Maya recorded **12 connection attempts**.
YOU	Did those connections get through?
YOUR ASSISTANT	No. Its contract permits payment access only through payments-service. Maya blocked all **12 direct attempts** while keeping the approved path available.

FOUR WAYS TO USE MAYA

Command line

Plain answers, JSON for scripts and agents, and exit codes for CI.

Coding-agent skill

Guides your coding assistant on when to ask Maya and how to use the answer.

MCP server

The same questions for IDE chat and hosted assistants that have no shell.

CI check

A GitHub Action that checks behavior against the approved contract.

CLI QUERY EXAMPLES

COMMAND	ANSWER
`maya reach`	Connections made
`maya new`	New connections
`maya run`	Run activity
`maya check`	Contract conformance
`maya blocked`	Blocked connections
`maya who-calls`	Resource dependencies
`maya tokens --trend`	Estimated token trend

One contract, from development to production.

Draft	Review, commit	Check in CI	Enforce in prod	Evidence
Maya drafts it from observed activity	Kept in your repo next to the code	Build fails if the agent strays	Same file, enforced by Maya	What was allowed, what happened

Read-only by design.

Developer questions use revocable read-only keys. Only administrators can change policy.

Evidence for security review.

The contract and activity history connect what was allowed, what happened, and the controls applied.

Get early access for your team. Email developers@mayagentic.com.

Copyright 2026 Mayagentic, Inc. All rights reserved.