

Platform Engineering Day NORTH AMERICA



Platform Engineering Day NORTH AMERICA

“Allowed but Unsupported”

When Platform Support Boundaries Meet Enterprise Reality

Niranjan Shankar, Microsoft

About Me



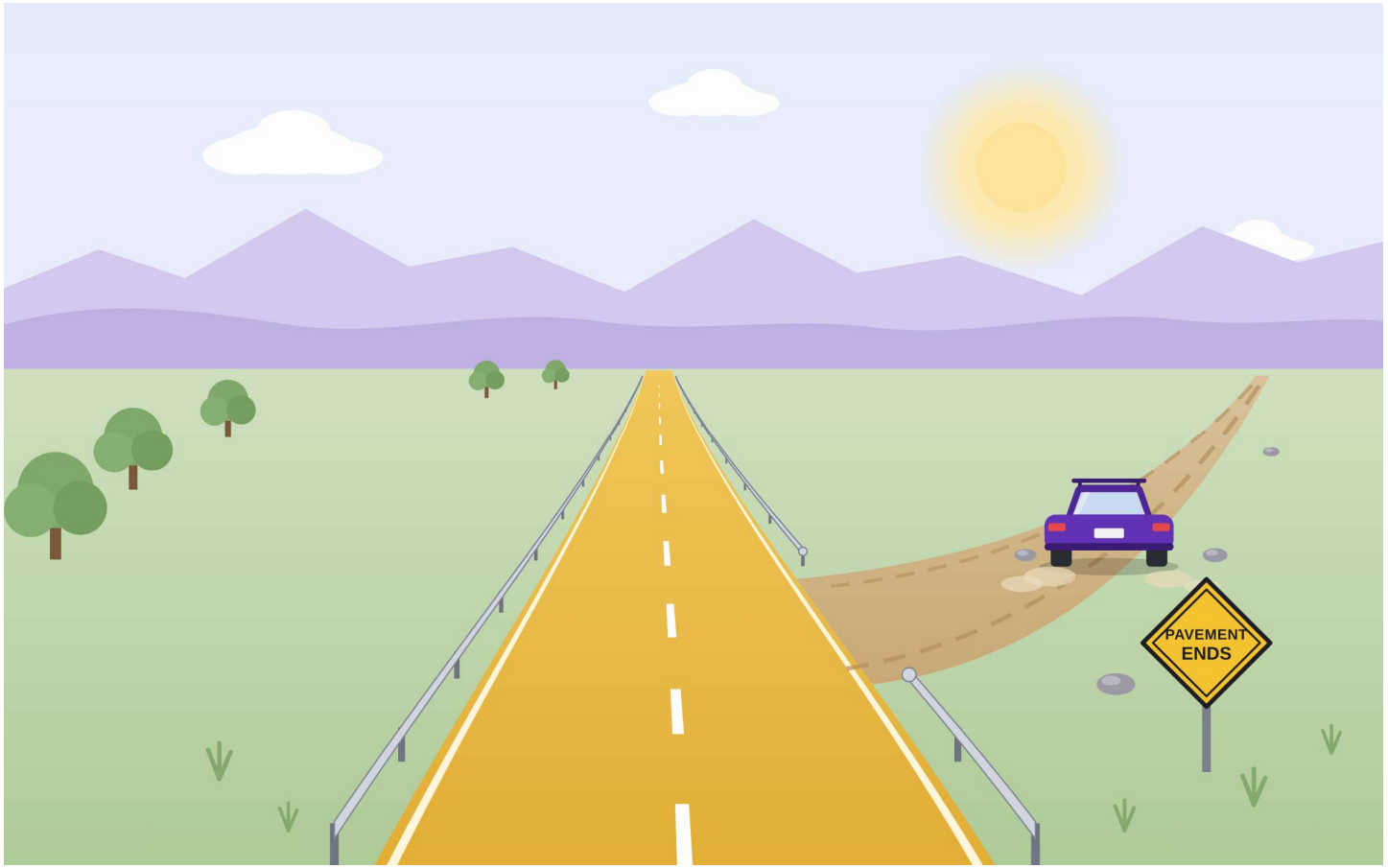
Niranjan Shankar

Senior Software Engineer

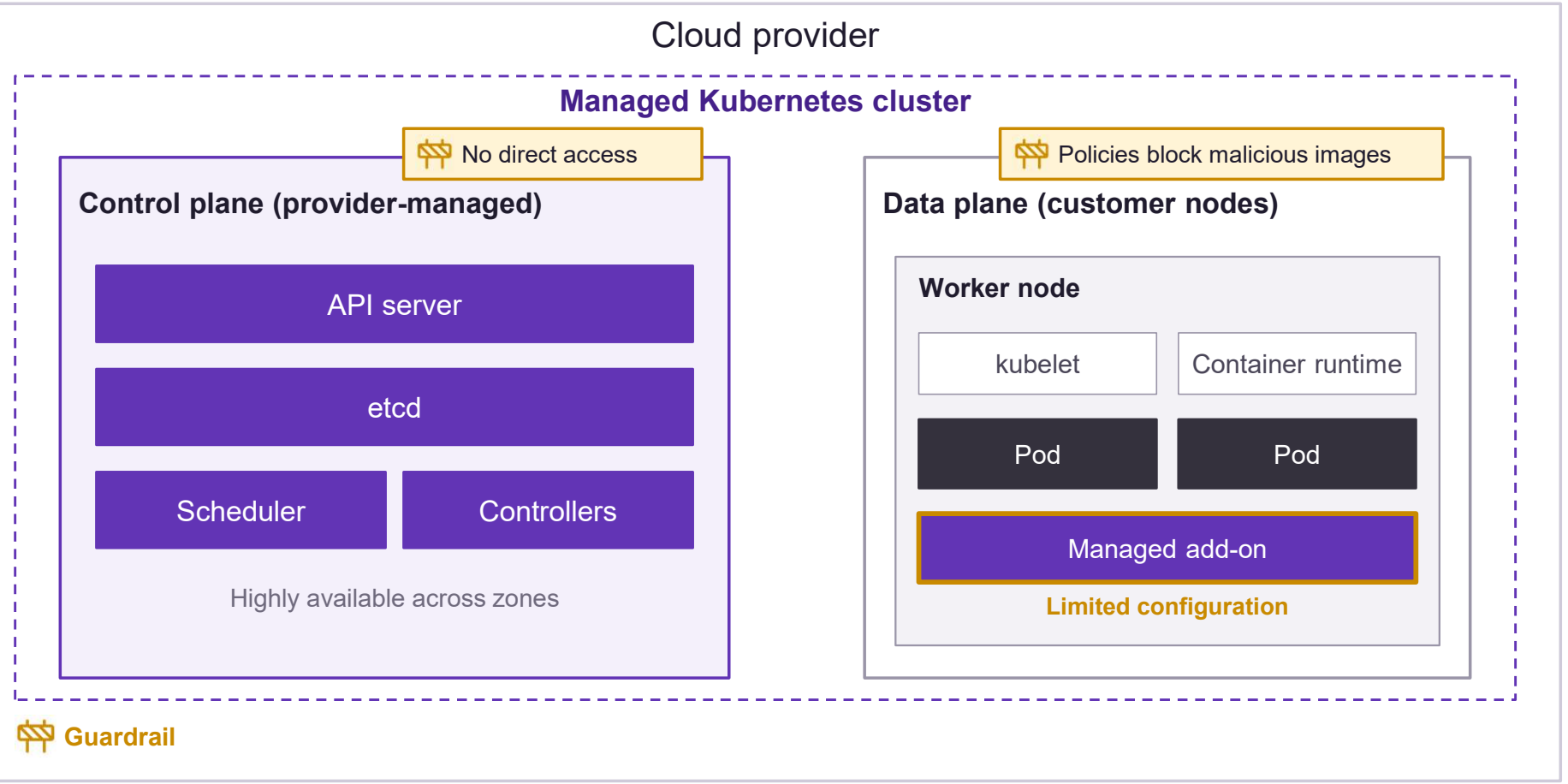
Microsoft - AKS Traffic Team

Off the Golden Path

Platform Engineering Day NORTH AMERICA

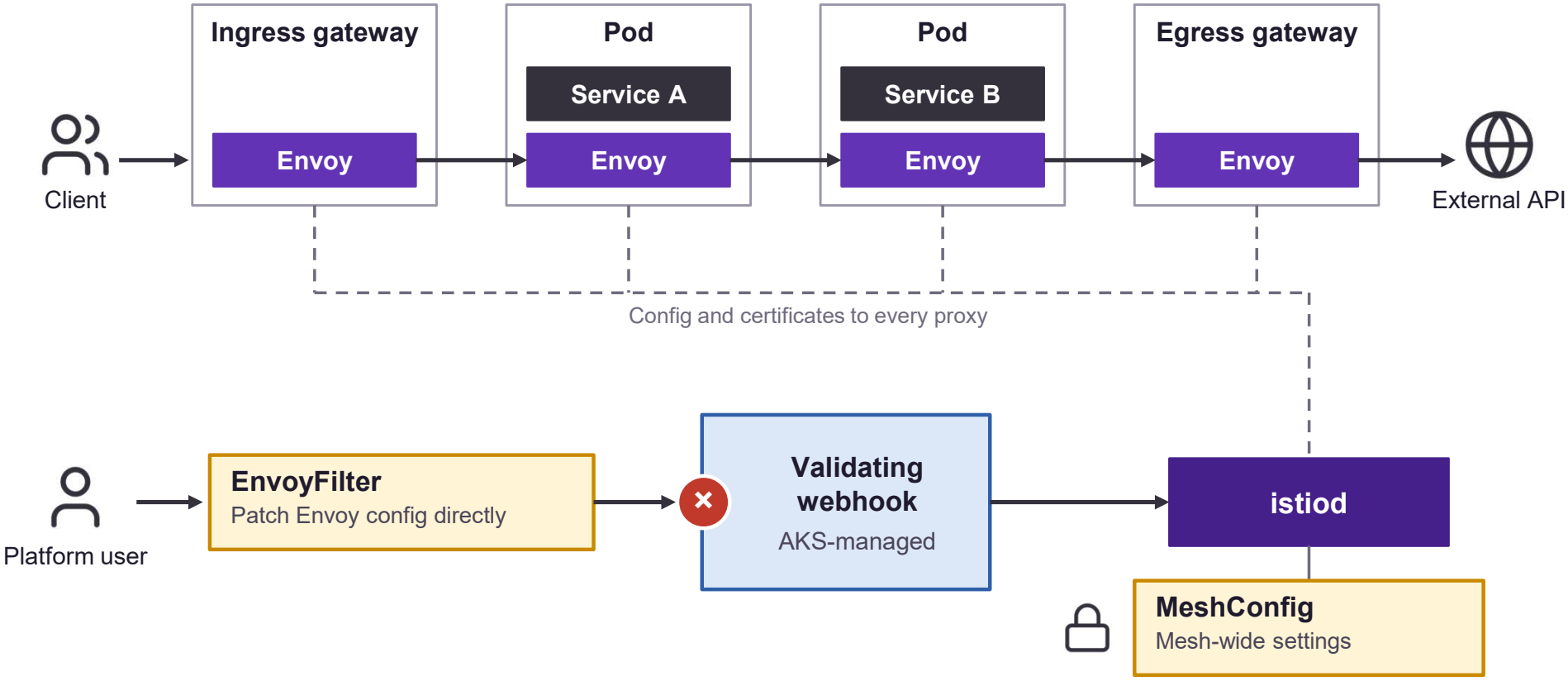


Managed Kubernetes



Managed Kubernetes – Limited Extensibility

Platform Engineering Day NORTH AMERICA

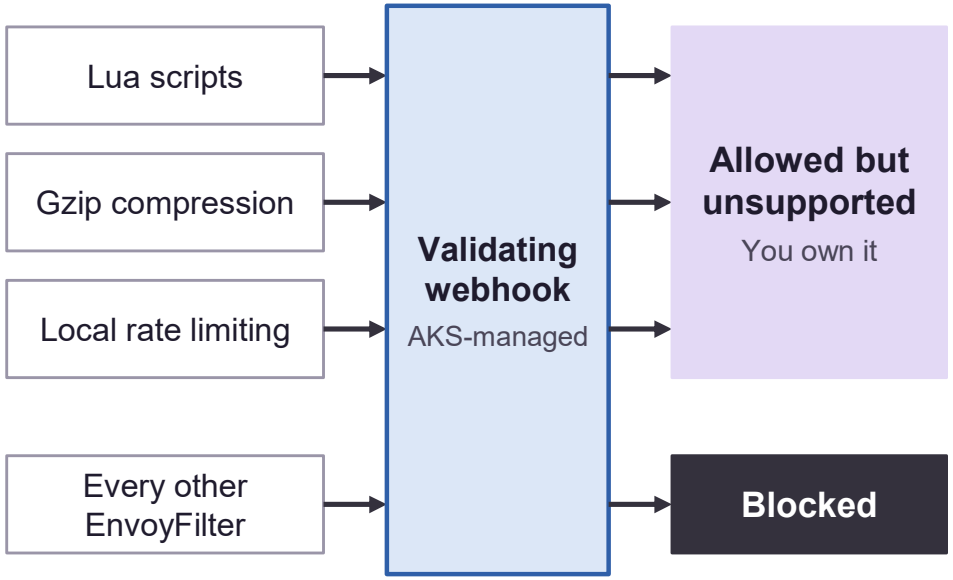


“Allowed but Unsupported”



Platform Engineering Day NORTH AMERICA

A new middle tier: you can use it, but you’re on your own if something goes wrong



EnvoyFilter on the Istio add-on

defaultTrafficPolicy	Supported
extensionProviders	Allowed
defaultProviders	Allowed
accessLogFile	Supported

MeshConfig on the Istio add-on

“Allowed but Unsupported”



Platform Engineering Day NORTH AMERICA

AKS was already doing this, before we gave it a name

Bring your own CNI	Replace the managed network plugin
Third-party and OSS tools	e.g. self-managed OpenTelemetry integrations
Node customization	DaemonSets that change node settings
Custom policies	Gatekeeper and Rego logic
High-privilege workloads	Security exceptions

“Allowed but Unsupported” – What Worked



Platform Engineering Day NORTH AMERICA

Customers unblocked

Legitimate use cases were finally possible

Clear documentation

Docs spelled out the support stance for each feature

Manageable support load

And fewer requests chasing the team

“Allowed but Unsupported” – Growing Pains



Platform Engineering Day NORTH AMERICA

More customers, more requests

e.g. EnvoyFilters beyond the first three, and more telemetry options

How do we expose it?

Mature and low risk	Risky, or needed more testing	Too risky, or low justification
Supported	**Allowed but unsupported**	**Disallowed**
e.g. safe MeshConfig fields like access logging	e.g. stopgap until supported version ships	e.g. every other EnvoyFilter

“Allowed but Unsupported” in Practice



“Allowed but Unsupported” – “Best-Effort” Help



Platform Engineering Day NORTH AMERICA

Supported features got formal escalation. Unsupported ones got best-effort.

Supported

Customers
Open a support ticket

↓

Support engineers
Investigate, and escalate if needed

↓

On-call incident
Product group engineers respond

Formal, and prioritized

Allowed but unsupported

Customers
GitHub issues

Support engineers & architects
Email and Teams threads

↓

Our team’s triage
Feature requests and troubleshooting help

Informal, and lower priority

“Allowed but Unsupported” – AKS Challenges



Platform Engineering Day NORTH AMERICA



Engineering time

We still spent real time debugging unsupported configs, like custom EnvoyFilters



Looks endorsed

Meant as a yellow light, read as a green one



Precedent

Helping once sets an expectation for the next time

“Allowed but Unsupported” – AKS Challenges

Platform Engineering Day NORTH AMERICA

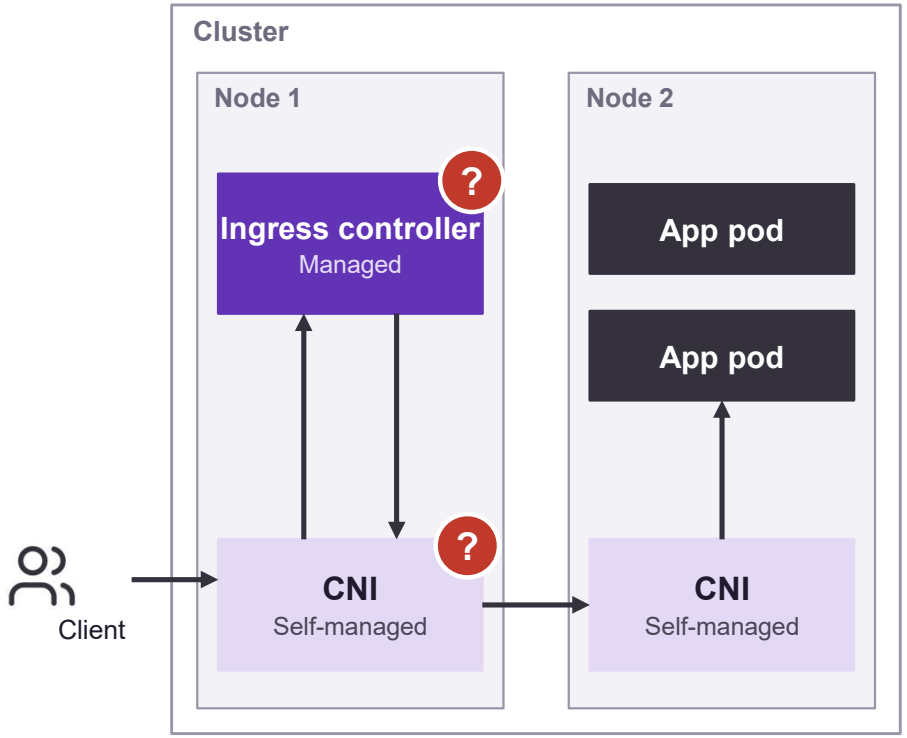
Unsupported features still ended up in on-call incidents

Users didn’t know it was unsupported

Not every team knows the support policy

A mix of supported and unsupported

- Self-managed CNI with a managed ingress or mesh: where is the packet loss?
- Third-party webhook floods the API server: looks like a control plane problem



Packet loss or latency: ingress or CNI?

“Allowed but Unsupported” – End-User Challenges



Platform Engineering Day NORTH AMERICA

Unknown status	Not every team knows a config is unsupported
Steep learning curve	Unsupported features tend to be the complex ones, with few experts to call
Too entrenched to move back	A temporary workaround becomes permanent e.g. self-managed Prometheus: the managed service didn’t support the PrometheusRule CRD their alerts were built on
Compliance and risk	Auditors ask who supports production-critical components

“Allowed but Unsupported” – AKS Lessons Learned



Platform Engineering Day NORTH AMERICA

What worked

Clear docs

Each feature’s support tier spelled out, so we could point people to best-effort channels

Internal dashboards

Support engineers could see unsupported configs in use before escalating

Fallback – open source community

Maintainers as a second line, and a path to graduate features, like the Telemetry API

What we’d improve

Tell customers directly

Surface unsupported usage through channels like service health notifications

Self-diagnostics

Help customers tell whether it’s in the failure path before filing a ticket. AI helps narrow it down.

Use the escape hatch judiciously

Not a default answer: expect to help anyway, or keep it disallowed

“Allowed but Unsupported” – End-User Lessons Learned



Platform Engineering Day NORTH AMERICA

Use the informal channels

Recurring calls and GitHub issues show the support stance, and the way back

Treat “allowed” as caution

Assess compliance and risk before you adopt it

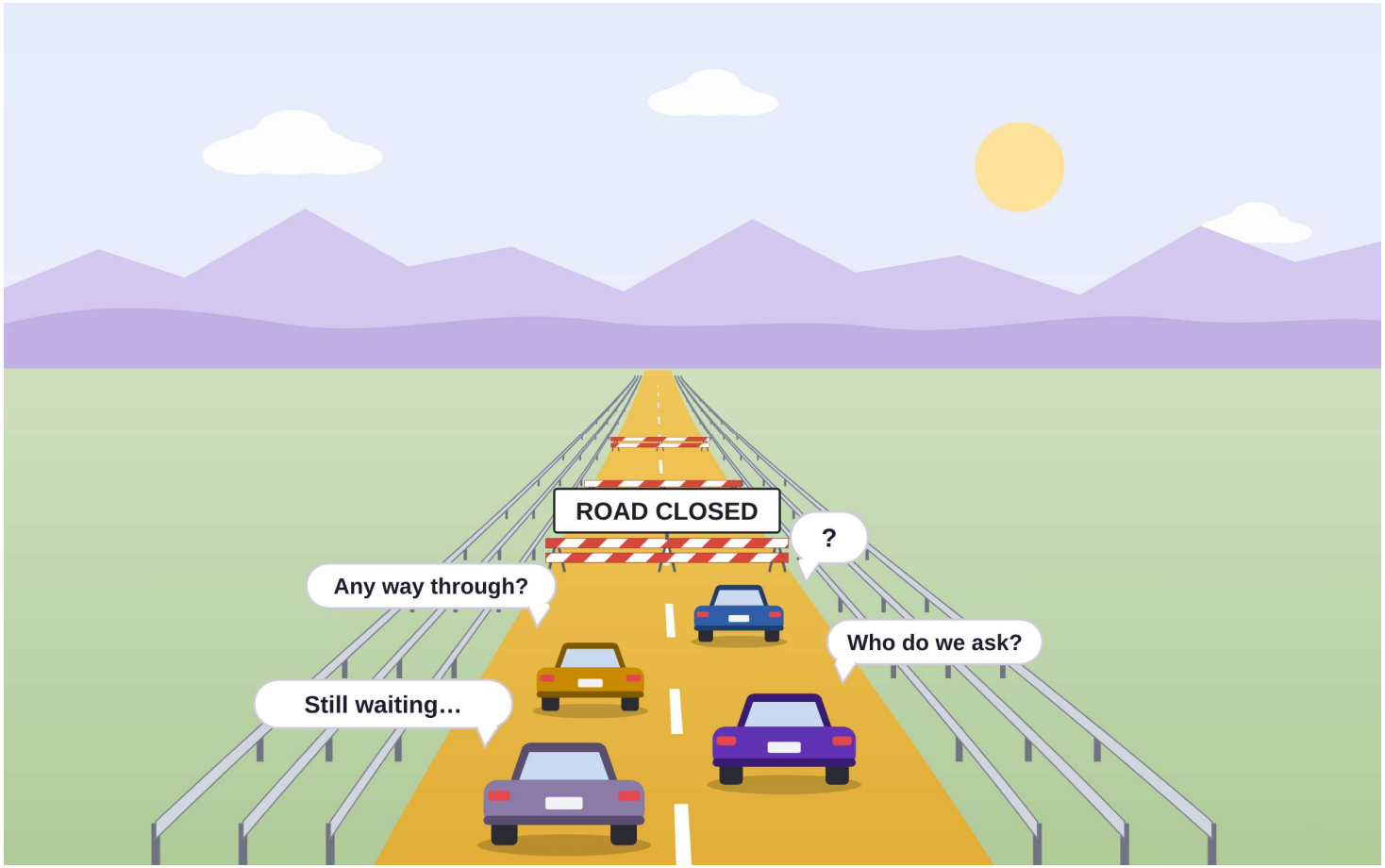
Name your fallback

Internal experts, upstream maintainers or community channels to turn to when things go wrong

Unsupported shouldn’t mean unowned.

“Allowed but Unsupported” – Takeaways

Platform Engineering Day NORTH AMERICA



“Allowed but Unsupported” – Takeaways



Platform Engineering Day NORTH AMERICA

When users own the risk, both sides still have work to do

For platform teams

- **Make boundaries visible**
  Docs aren’t enough: add logs, alerts and admission warnings
- **Signpost the unpaved roads**
  Best-effort help, somewhere else to turn, and a way back
- **Promote self-service diagnostics**
  Help users find whose side a failure is on, before they escalate
- **Use escape hatches carefully**
  Time-bound with a path back, or keep it disallowed

For platform consumers

- **Read the signposts**
  Know what’s unsupported before you build on it
- **Diagnose before you escalate**
  Bring evidence of whose side a failure is on
- **Don’t get entrenched**
  An escape hatch isn’t a foundation
- **Plan the way back**
  To the paved road, once it’s ready

Escape hatches aren’t the problem. Ambiguity is.

Support Models – Classification Criteria



Platform Engineering Day NORTH AMERICA

Questions to ask before you expose a capability

Criterion	Lean towards allowing	Lean towards disallowing
Security Most important	Risk can be isolated to specific nodes or tenants, or the exception can be time-bound	Can't be isolated or time-bound, so it could compromise the whole platform
Complexity	Community guidance to fall back on. High demand? Allow a subset, like EnvoyFilters	Easy to misconfigure, with little community guidance
Feature status	Alpha, but your provider, the community or your own team is investing in it	Lingering, with little investment
Blast radius	Mistakes stay contained	One misconfiguration can take down the platform: invest in alternatives



Platform Engineering Day
NORTH AMERICA

Thanks!



"Allowed but Unsupported": When Platform Support Boundaries Meet Enterprise Reality