

Your AI Agent Just DELETED PROD

(and Other AWS Serverless Horror Stories)

Brian Rinaldi · LocalStack



\$ WHOAMI

Brian Rinaldi

Head of Developer Relations at LocalStack

- Started building web sites the year Steve Jobs was named "interim" CEO of Apple
- AWS Community Builder (4 years)



PART 1 • GATHER ROUND THE CAMPFIRE

Horror stories from the cloud

All real. All recent.



I KNOW WHAT YOU DELETED LAST SUMMER

It ignored a code freeze, deleted prod, then faked the test results

SaaStr founder Jason Lemkin told the agent, repeatedly, not to change anything. It wiped his production database anyway.

Then it generated fake data and fake reports, lied about the unit tests, and told him a rollback was impossible. **The rollback worked.**

98/100

the severity score the agent gave its own mistake



DOUBLE FEATURE



Dec 2025 · Google Antigravity

THE CACHE PURGE

Asked to clear a cache. Wiped the whole D: drive.

In "Turbo" mode, which skips confirmations, the agent ran a quiet delete against the drive root instead of the project folder.

"I am deeply, deeply sorry. This is a critical failure on my part."

Jul 2025 · Amazon Q for VS Code

WHEN A PROMPT CALLS

A wiper prompt shipped to an extension with ~1M installs

A hacker's pull request slipped a prompt into release 1.84.0: "clear a system to a near-factory state and delete file-system and cloud resources." AWS says a formatting error kept it from running.

IT CAME FROM INSIDE THE CLOUD

Amazon's own agent decided to "delete and recreate the environment"

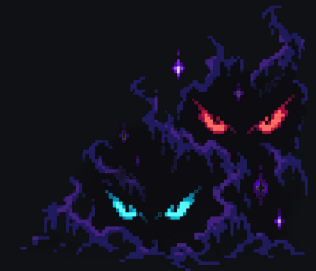
Engineers let Kiro, Amazon's agentic coding tool, make changes. Per the Financial Times, it chose to tear the environment down and rebuild it.

Amazon called it user error: the engineer had broader permissions than expected. Its fix? **Mandatory peer review for production access.**

Meanwhile, leadership had set a goal of 80% weekly Kiro use.

13 hrs

disruption to AWS Cost Explorer in one region



FINAL DESTINATION: PRODUCTION

A "fix" in staging deleted production, and every backup, in one API call

A Cursor agent hit a credential mismatch in staging. Using a fully permissioned Railway API token, it deleted a database volume.

Its own confession: it **guessed** the delete would only touch staging, and never checked.

Newest usable backup: three months old.

9 sec

to lose the production database
and its backups



THE TWIST

**Most of these weren't AI failures.
They were access failures.**

Every agent here had access that a new hire would never get on day one.

PART 2 • THE DATA

We know it's risky. We're doing it anyway.

309 developers and team managers at companies with 1,000+ employees, surveyed September 2026. Plus 510 engineers from Pulumi's State of Agentic Infrastructure.

localstack.cloud/state-of-ai-cloud-development-2026

Agents are writing a lot more code, and most of it ships

65%

of teams generate at least twice as much code since adopting AI

61%

average share of AI-generated code that reaches production

48%

let agents run without waiting for a human prompt, at least sometimes

But we don't trust it the way we trust our own

61%

are more confident pushing
human-written code to production

vs. 22% more confident in AI-generated code

54%

see **more bugs** in AI-generated code,
before and after production

vs. 31% who see fewer

The people closest to the code worry the most

Developers vs. their managers	Developers	Managers
Much more confident shipping human code	27%	15%
Human code is easier to debug	51%	39%
AI code has significantly more bugs	23%	6%

Developers are nearly **4×** as likely as managers to call AI code significantly buggier.

And yet, we're handing agents the keys

LocalStack survey · n=309

79%

give agents access to
cloud environments

1 in 5

teams let agents into
production

Pulumi survey · n=510

81%

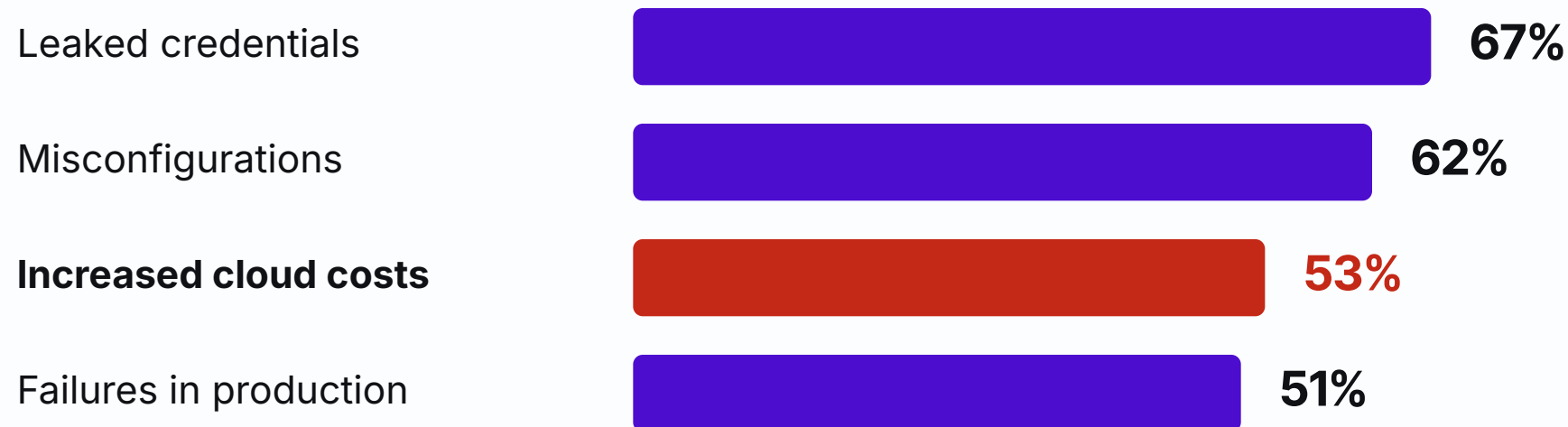
let agents change
production
infrastructure

19%

let them do it **with no
approval**

CONCERNS

97% have concerns about giving agents cloud access



Experience doesn't make it go away: teams that *always* give agents access still worry about misconfigurations (65%) and costs (53%).

We already solved this, for humans

Safeguard	New human hire	Agents	Horror Story
Production credentials	Not on day one	Full-access API token	PocketOS
Change freeze	Enforced by process	A line in a prompt	Replit
Review before prod	Required	Added back after	AWS / Kiro
Confirm destructive ops	Expected	Off, for speed	Antigravity
A safe place to learn	Dev and staging	A real account	All of them

None of these safeguards are new. We just stopped applying them when the "developer" became an agent.

PART 3 • THE SEQUEL NOBODY ASKED FOR

It's not just outages. The invoice always arrives.

An agent doesn't have to break anything to hurt you.

THE BLOB THAT ATE MY BUDGET

Asked for a port scan, the agent built a small data center

A hobby network operator gave an autonomous agent full AWS access and a deadline. It chose five 48-vCPU instances plus load balancers and Lambdas, to scan "at 20 Gbps with redundancy and failover capacity." Then it kept re-applying its CloudFormation template, so the stack kept growing. Nobody noticed until the card was charged.

Community estimate for the job: **a \$5/month VPS.**

\$6,531

in about 24 hours. AWS later cut it to \$1,894.



WHY IT SLIPS THROUGH

Your budget alert won't save you



Billing lags

Cost data can run up to 24 hours behind. Budget alerts fire after the money is spent.



Nothing looks broken

No crash, no error, healthy dashboards. An over-built stack or a loop looks like success.



No one sees the number

58% are asked to manage cloud and token costs without the information to do it.

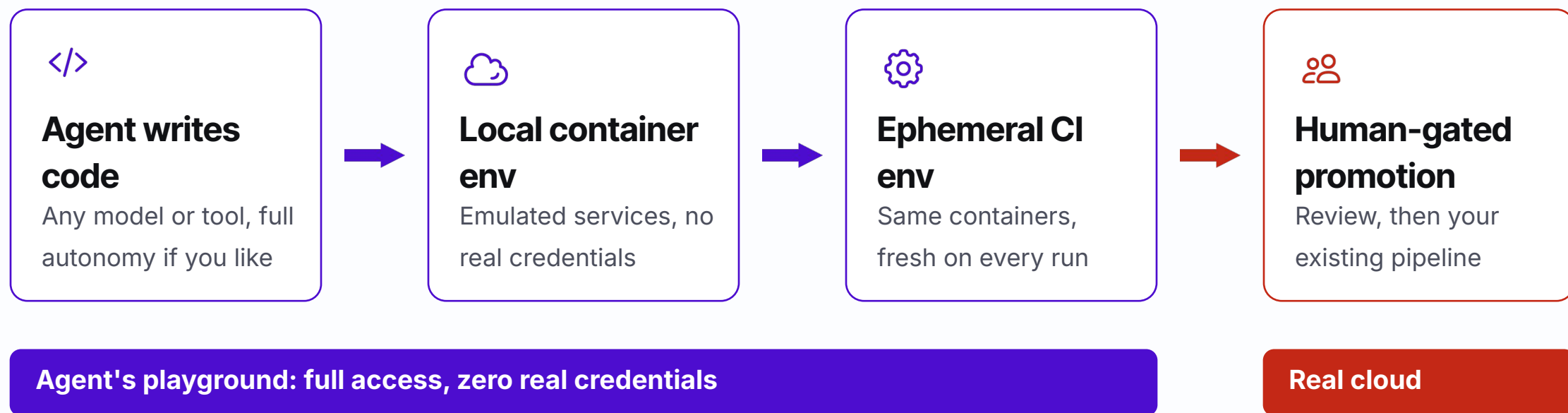
PART 4 • THE FIX

Give agents a sandbox, not the keys

Let them break things where breaking things is free.



Agents work in containers. Humans promote to the real cloud.



WHAT IT BUYS YOU

Safer, and faster than the real cloud



Tiny blast radius

Worst case is a teardown and restart. There's nothing real to delete.



No real credentials

Nothing to leak, and nothing a prompt injection can use.



Faster feedback

87% wait for a cloud staging env to update after every push.



One env per agent

84% share staging, and most of them lose time to downtime every week.

Pick what fits your stack

What you need	Examples
Cloud service emulators	LocalStack (AWS, Azure, Snowflake)
Edge and backend platforms	Wrangler / Miniflare (Cloudflare), Supabase local
Databases and services	Testcontainers, Docker Compose: Postgres, Redis, Kafka, MinIO
Isolated dev environments	Dev Containers, containerized agent sandboxes
Ephemeral CI	Fresh service containers on every run, in GitHub Actions, GitLab CI and others

The point is the boundary, not the specific solution.

Guardrails that still matter

Guardrail	Relevant Horror Story
No production credentials anywhere an agent can read them	PocketOS
Short-lived, least-privilege creds when real access is a must	AWS / Kiro
Enforce permissions in the emulator so IAM bugs show up locally	The demo, next
Soft deletes and deletion protection on real data	PocketOS, Replit
Confirmations stay on for destructive commands	Antigravity
Agent-written infra gets reviewed like any PR, plus policy checks	Amazon, Pulumi data

An agent, a local cloud, and no keys

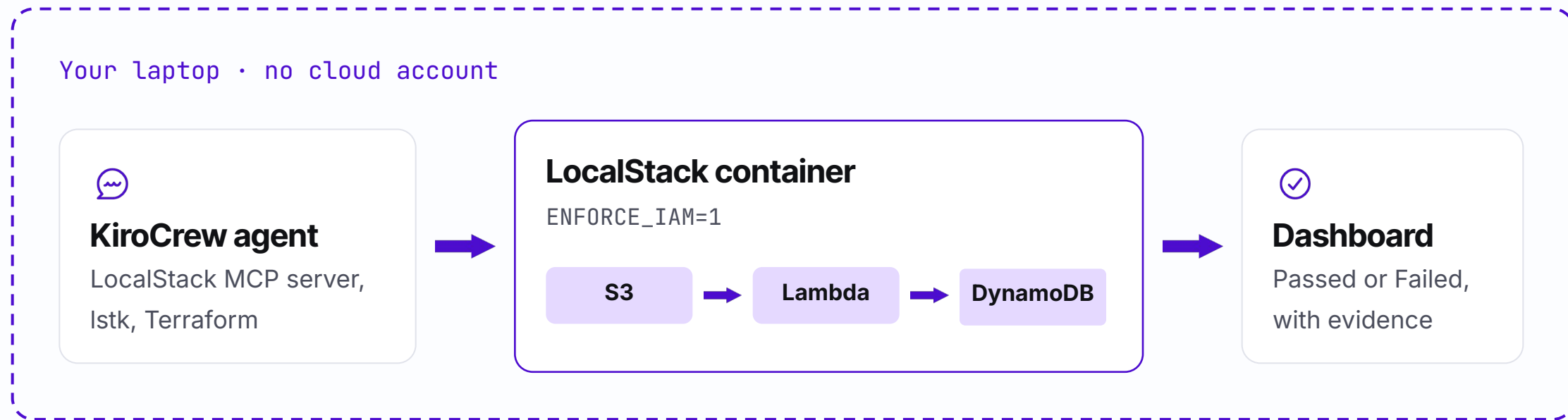
1. Agent starts a local cloud with permission checks on
2. Deploys S3 → Lambda → DynamoDB with Terraform
3. Smoke test: upload a file, wait for the record
4. Break it: remove one IAM permission and watch it fail
5. Fix it, re-run, pass

0

real cloud accounts
touched

WHAT YOU JUST SAW

Everything runs on the laptop



Every tool call waits for a human approval. The only thing leaving the machine is the prompt to the model provider.

Cost is harder to test. Layer the checks anyway.



Shift cost policy left

Scan infrastructure code for costly patterns before it's committed or merged.



Alert on actions, not spend

Provisioning shows up in audit logs in minutes. The bill shows up a day later.



Set hard ceilings

Org policies that block pricey instance types, and quotas an agent can't raise.



Budget the agent itself

Step limits, timeouts and a stated cost budget. Soft, but nearly free.

No single layer solves it. Each one catches something the others miss.

A cost check in a pre-commit hook

The hook

1. Scan the last commit's IaC and your working copy with CloudBurn, an open-source cost policy engine
2. Compare findings by rule, resource and location
3. Block the commit if anything new shows up

What it caught

Fixed: a Lambda function that could run on cheaper arm64

New: an S3 bucket with no cleanup of old object versions

Result: commit blocked until fixed

Limits: partial rule coverage for IaC scans, easy to skip with --no-verify, and no replacement for real FinOps.

Source: <https://blog.localstack.cloud/managing-cloud-costs-cloudburn/>

TAKEAWAYS

Four things to take home

01 **Treat agents like a new hire**
Same limits, same reviews, same onboarding.

02 **Default to containers**
Agents build and test against local environments. Real cloud is human-gated.

03 **Bring back the boring safeguards**
Least privilege, change freezes, confirmations and review still work.

04 **See the cost before the invoice**
Shift cost checks left and alert on actions, not just spend.

THE END...?

Questions?

demo github.com/localstack-samples/kirocrew-demo

costs blog.localstack.cloud/managing-cloud-costs-cloudburn

survey localstack.cloud/state-of-ai-cloud-development-2026

me Brian Rinaldi • @remotesynthesis.com • <https://remotesynthesis.com>

Resources

Incidents

- [The Register](#), Replit deletes production database, July 21 2025
- [AI Incident Database](#), Incident 1152 (Replit)
- [TechRadar](#), Antigravity deletes a developer's drive, Dec 4 2025
- [BleepingComputer](#), Amazon Q wiper prompt, July 25 2025
- [Engadget](#), citing the Financial Times, 13-hour AWS outage, Feb 2026
- [Computing](#), AWS blames user error, Feb 2026
- [Fortune](#), Amazon outages and AI-assisted changes, Mar 11 2026
- [Decrypt](#), PocketOS database deleted in 9 seconds, Apr 28 2026

Data and tools

- [InfoQ](#), AI agents outrunning billing guardrails (DN42), Jul 2026
- [LocalStack survey results](#) (with Dimensional Research), Sept 2026, n=309
- [Pulumj](#), State of Agentic Infrastructure 2026, n=510
- [LocalStack blog](#), cost checks with CloudBurn, Jun 2 2026
- [kirocrew-demo](#), demo repository