DevFest

DEVFEST 2026

Beyond CRUD

```
$ who runs prod?
> you
```

Build your own internal cloud, one trade-off at a time.

A REAL ENGINEERING STORY

Tachera Sasi · Backend & Infrastructure Engineer

the part of software that runs after the database

Google Developer Groups

You run a small agency.

Twelve clients. Fifteen servers. Three regions. One engineer: you.

EU · 5

vps-fra-01	Laravel
vps-fra-02	Go API
vps-fra-03	Next.js
vps-fra-04	Postgres
vps-fra-05	Redis

US · 5

vps-nyc-01	Laravel
vps-nyc-02	Go API
vps-nyc-03	Next.js
vps-nyc-04	Postgres
vps-nyc-05	Redis

AF · 5

vps-jnb-01	Laravel
vps-jnb-02	Go API
vps-jnb-03	Next.js
vps-jnb-04	Postgres
vps-jnb-05	Redis

The architecture diagram is a person. **That person is you.**

Also the developer, the DevOps team, and the on-call rotation.



Google Developer Groups

Every server comes with a second job.

Same checklist. Different stack. Fifteen times over.

- [ ] Create users + keys
- [ ] Firewall + SSH
- [ ] Install runtime
- [ ] Nginx vhost
- [ ] TLS certificates
- [ ] Env + secrets
- [ ] Deploy the app
- [ ] Restart process
- [ ] Read the logs
- [ ] Watch metrics
- [ ] Backups + cron
- [ ] Patch + update

×15

servers

And every client
has a different stack.

Laravel | Go | Next.js

The work isn't hard. **It's just endless.**



Google
Developer
Groups

The math is not on your side.

180

manual steps

`15 servers × 12 jobs = 180`

And every deploy, cert renewal,
and client change resets the board.



you, probably

meme: KC Green / Imgflip

You are the cron job.

Google Developer Groups

So you write the wish list.

One internal dashboard. No terminal sessions. No 'just SSH in and run this'.

- ☐ **Provision** 01
- ☐ **Lifecycle** 02
- ☐ **Deploy** 03
- ☐ **Nginx** 04
- ☐ **SSL** 05
- ☐ **Logs** 06
- ☐ **Metrics** 07
- ☐ **Commands** 08
- ☐ **Env & secrets** 09
- ☐ **Domains** 10
- ☐ **Backups** 11
- ☐ **SSH keys** 12

This is an internal tool — **for your team, not a SaaS.**

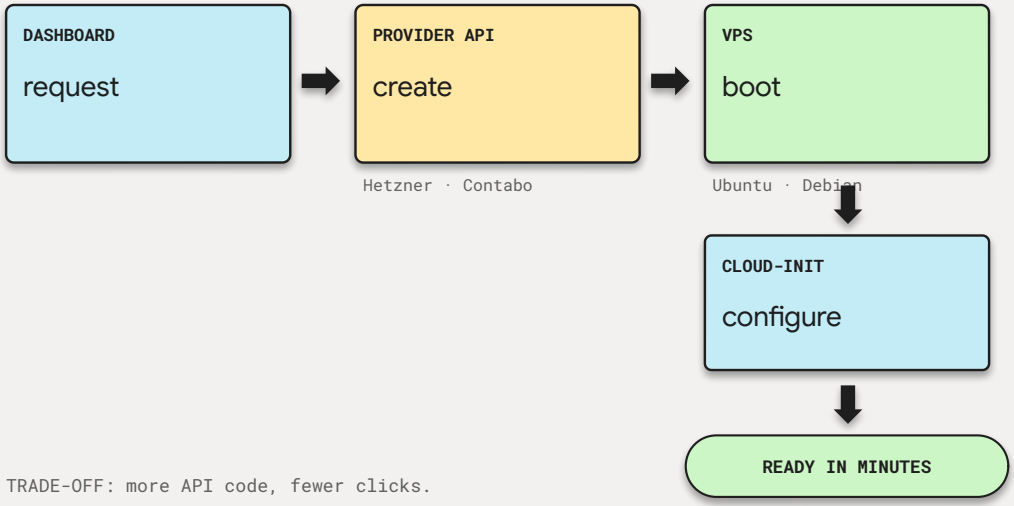
No Kubernetes. No Terraform. No AWS. Just Linux, Go, and cheap VPS APIs.



Google Developer Groups

Start with the box.

Hetzner and Contabo already have APIs. No data center required.



cloud-init.yaml

```
#cloud-config
users:
  - name: deploy
    groups: [sudo]
    ssh_authorized_keys:
      - ${DEPLOY_KEY}
packages:
  - nginx
runcmd:
  - ufw allow 80,443/tcp
  - systemctl enable --now nginx
```

a server's onboarding doc

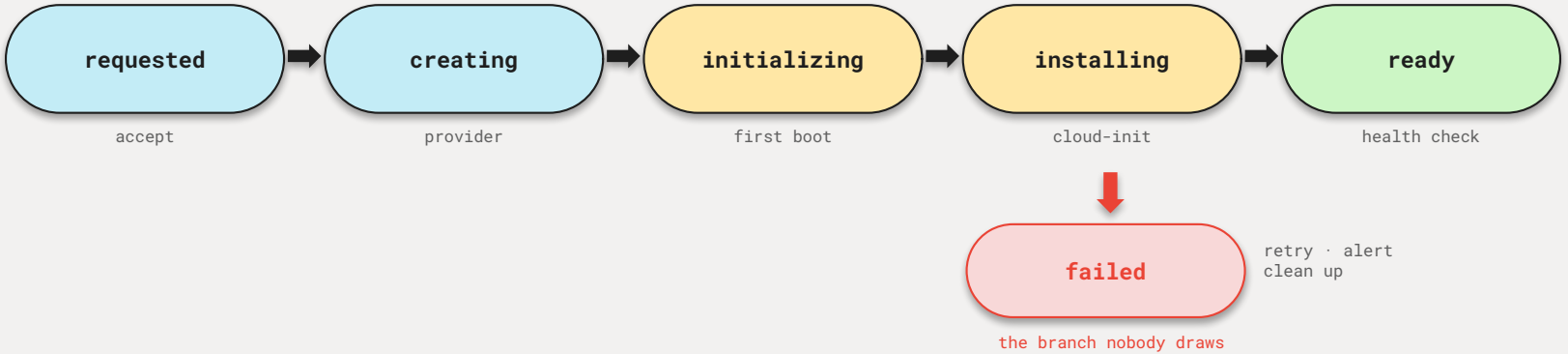
TRADE-OFF: more API code, fewer clicks.

One click replaces a 40-step checklist. **It's just code now.**

Google Developer Groups

The API said yes. The server wasn't ready.

Provisioning is asynchronous. Treat it like it.



YOU CAN ALSO

START STOP REBOOT DESTROY

But only the box.

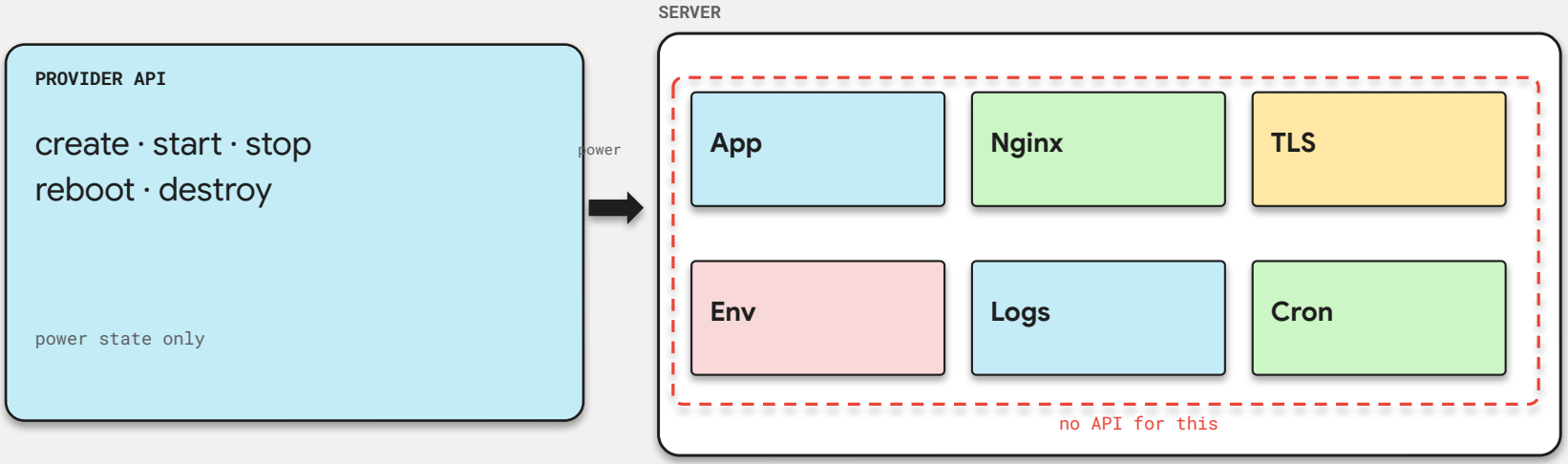
Track reality, not HTTP responses.

A 200 means “accepted”, not “ready to deploy”.

Google Developer Groups

The box is not the app.

Provider APIs manage power state. They don't know what's inside.

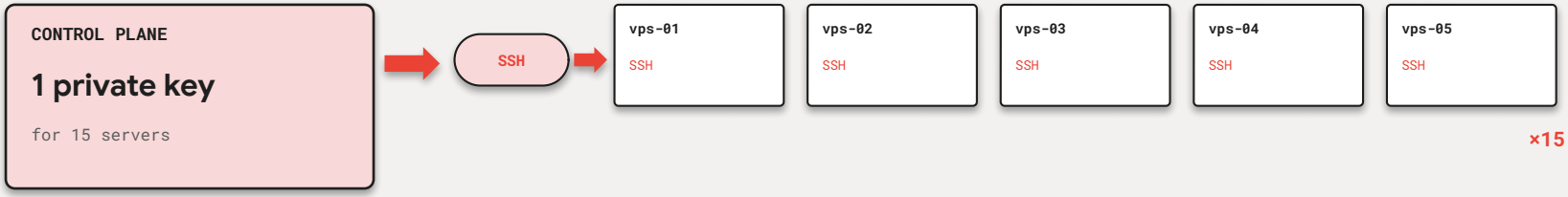


The provider gives you a machine. **It does not give you a deployment.**

Google Developer Groups

Option 1: let the backend SSH in.

The obvious move. Tools like Laravel Forge do exactly this.




```
ssh — deploy.sh

$ ssh deploy@203.0.113.10 "bash /opt/deploy.sh"

> Connection timed out.   $ retry...   > Permission denied (publickey).
```

One key. Fifteen servers. **What could go wrong?**

Google Developer Groups

You try it. Then the trade-offs show up.

SSH is a great human tool. It is a poor control plane.

01

Key custody

one key opens 15 doors

02

Inbound surface

port 22 on every VPS

03

Ephemeral

drop = unknown state

04

Concurrency

15 sessions, partial failures

05

Observability

exit code ≠ live logs

For 3 servers, fine. **For 15, you are building an SSH orchestrator.**

Credentials · connections · concurrency · observability · blast radius.



Google Developer Groups

You set out to remove the human from SSH.

You built a computer that SSHs into every server.

Yes. That is the downgrade.

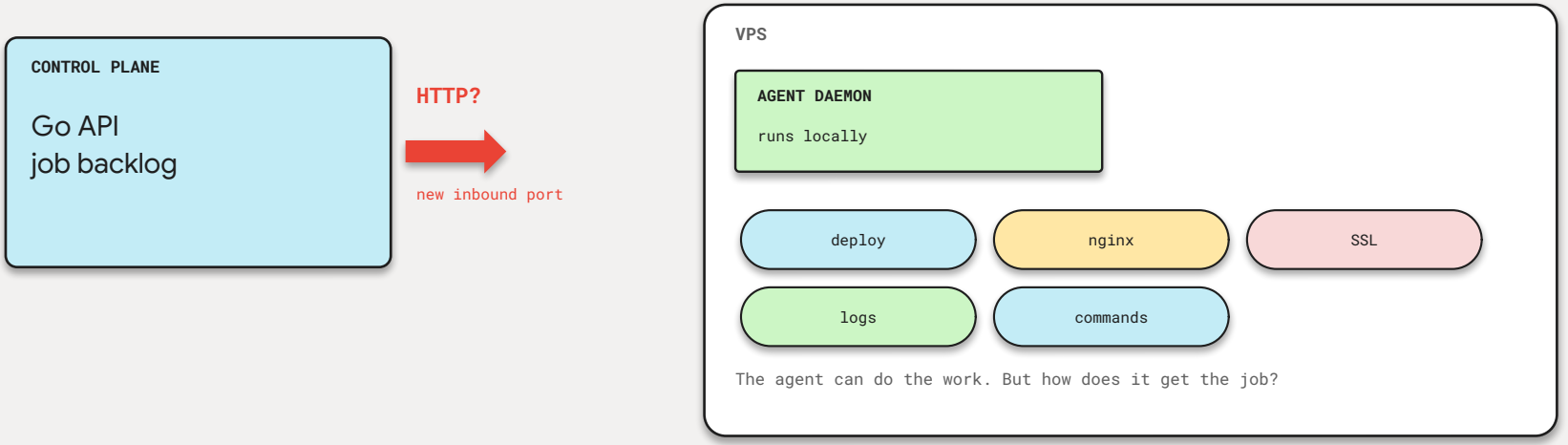


you, probably

meme: Lucasfilm / Imgflip

Option 2: give every server an agent.

Install it with cloud-init, or run one command on a VPS you already have.



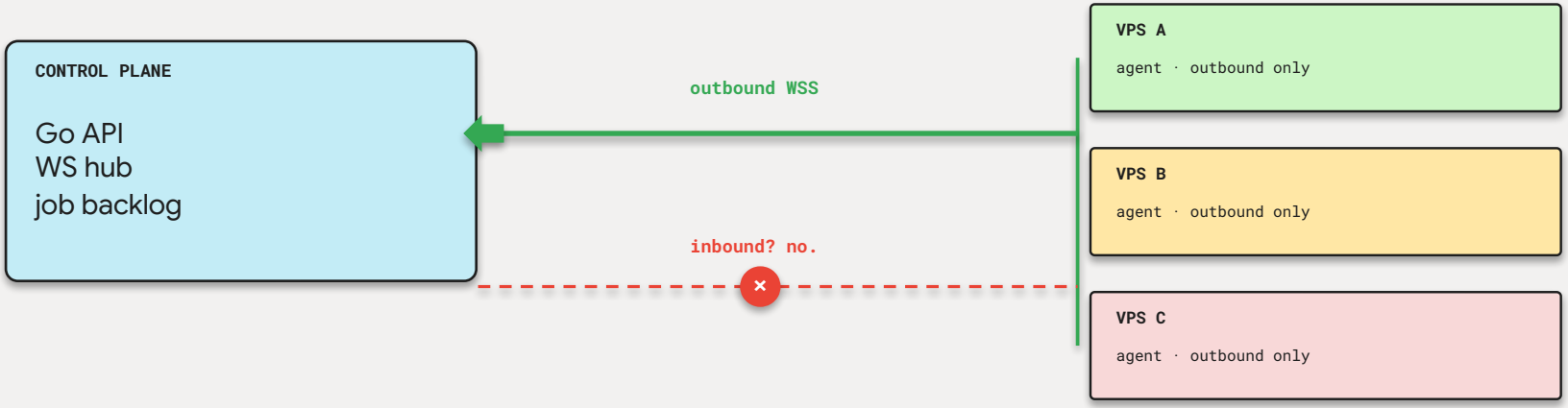
You just replaced SSH with **another inbound port.**

TRADE-OFF: local execution is powerful; inbound HTTP is the same attack surface in a new coat.

Google Developer Groups

So you flip the connection.

No inbound SSH. No inbound agent port. The agent calls home.



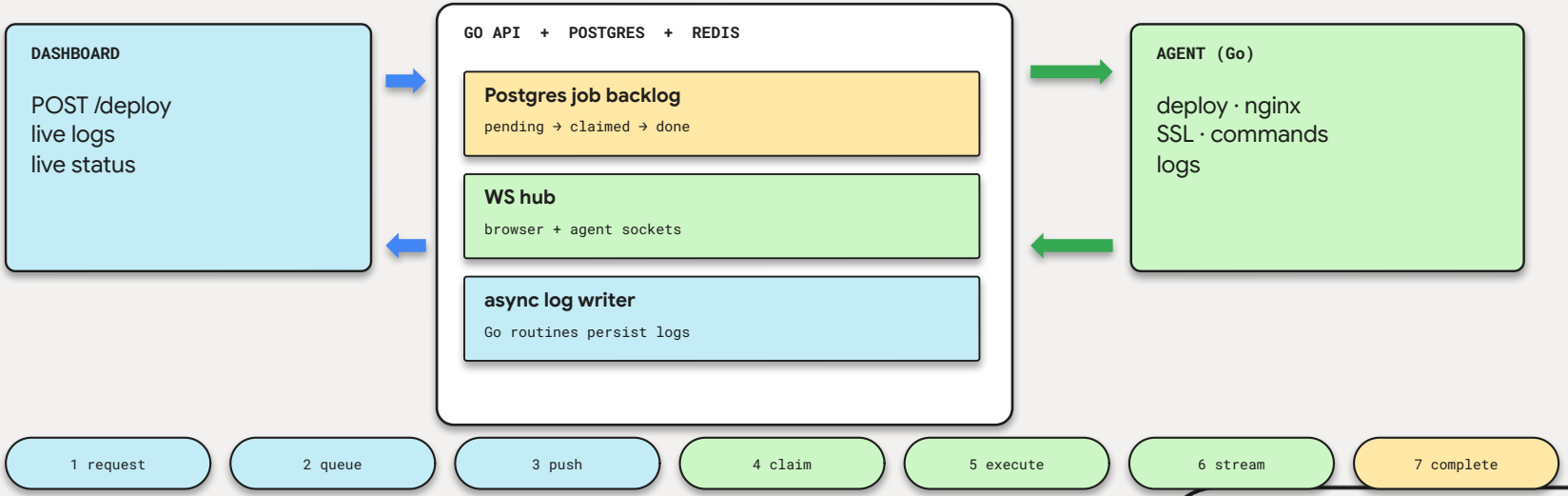
This is the decision that makes the agent safe: **outbound only.**

TRADE-OFF: you need reconnect logic and a job backlog, but you remove the inbound surface.

Google Developer Groups

The communication loop.

One click. No terminal. No polling.



The agent never opens an inbound port. Every step is authenticated. On completion: notify email / SMS / webhook.

Google Developer Groups

You secure every hop.

Three credentials. Three relationships. They never mix.

BROWSER → API

JWT for REST
60s one-time ticket
for the WebSocket

AGENT → API

per-VPS token
stored on the VPS
revocable + rotated

EVERY JOB

atomic claim
logs scoped to owner
audit trail

No inbound port. No shared SSH key. No anonymous job.

Your own root SSH access stays yours. The platform stops using it.



Google Developer Groups

Now the wish list is possible.

Same execution layer. Every capability is a job the agent can run.

- Provision ✓
- Lifecycle ✓
- Deploy ✓
- Nginx ✓
- SSL ✓
- Logs ✓
- Metrics ✓
- Commands ✓
- Env & secrets ✓
- Domains ✓
- Backups ✓
- SSH keys ✓

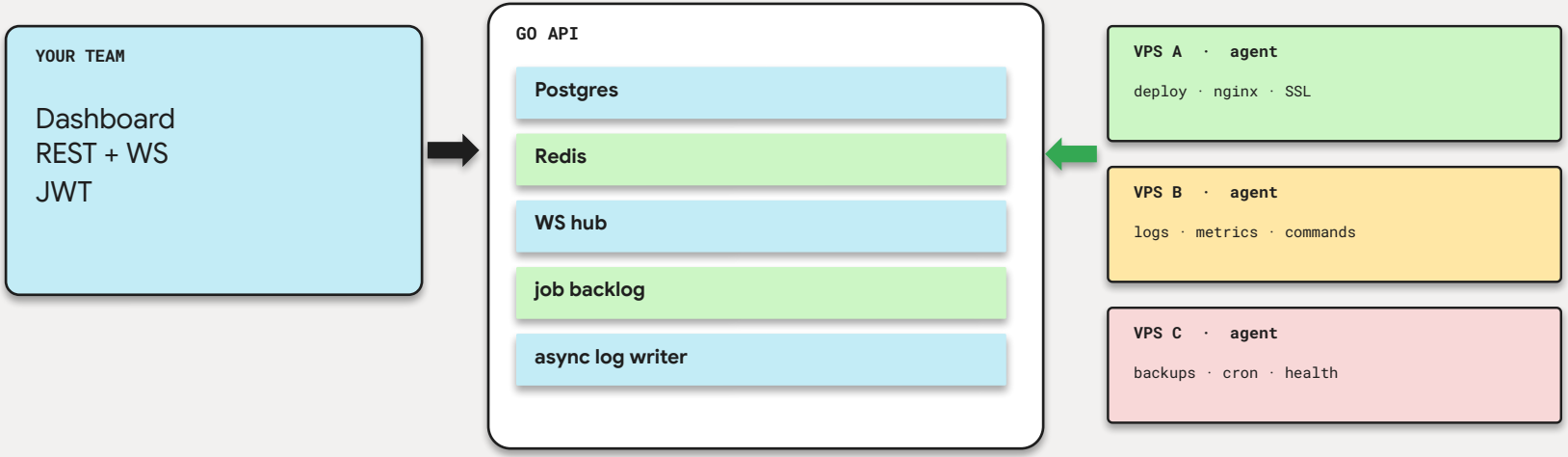
Not magic. Each capability still has its own auth, failure modes, and recovery.



Google Developer Groups

The internal tool, end to end.

Your team on the left. Your clients' servers on the right.



One dashboard. One control plane. Many servers.

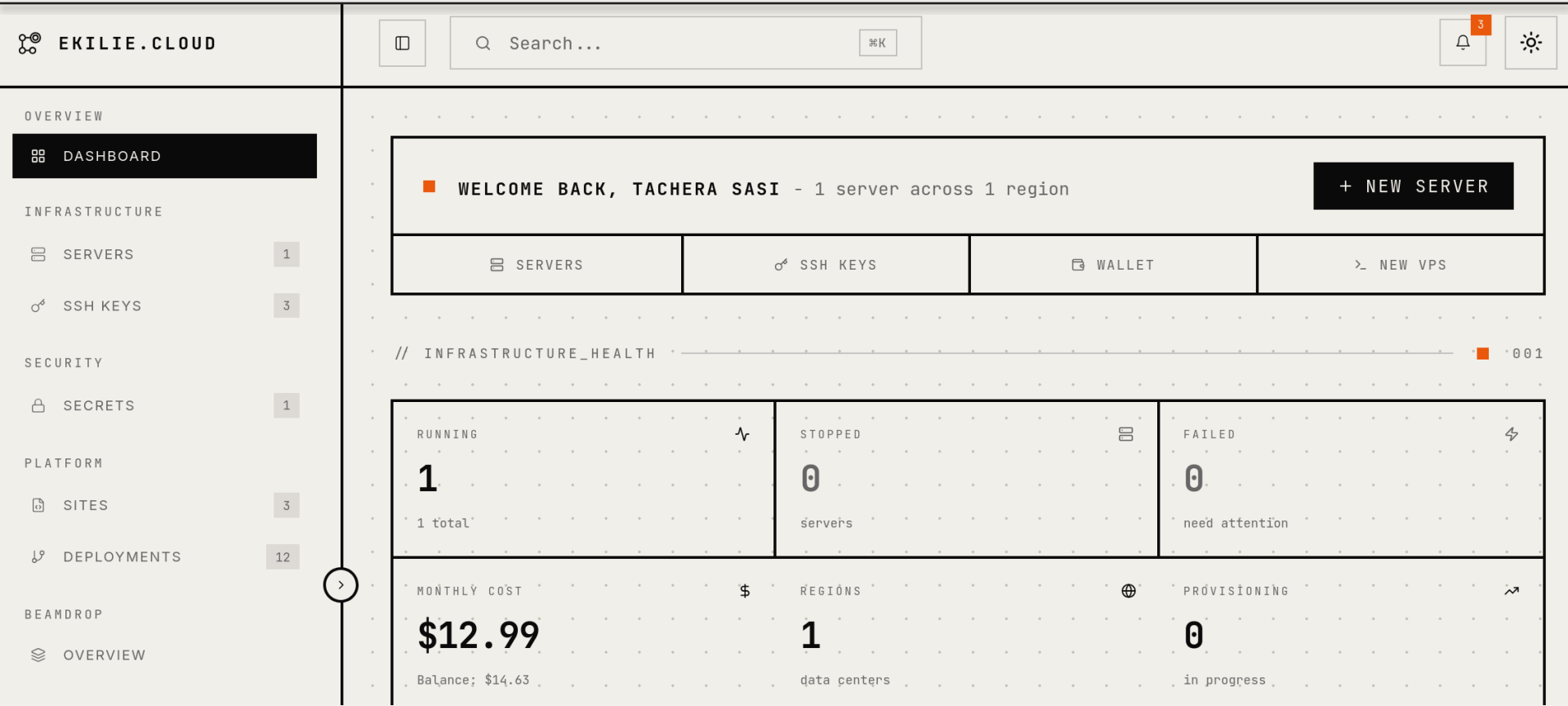
This is an internal tool — not a SaaS, not a Kubernetes cluster, not a hyperscaler bill.

Google Developer Groups

THE REVEAL · EKILIE CLOUD

This is exactly how Ekilie Cloud works.

The same architecture, in production. The agent is ekilied — open source Go, one per server.



CRUD is the interface.
Infrastructure is the product.

You didn't just build CRUD. You built the thing that runs it.

01

Start with the box, not the app.

02

Earn every shortcut. Every one has a trade-off.

03

Flip the connection: outbound, authenticated, observable.

Thank you.

Questions, arguments, and memes.

ekilie.cloud · docs/how-ekilied-works



Google Developer Groups