Cosa serve oggi per costruire una piattaforma cloud indipendente?

Dawit Abate Woldeamanuel

Linux Day 2026

Dawit

Senior Software & Cloud Engineer

Sto costruendo Flui cloud

un PaaS open source per i provider europei

Parliamo dei componenti open source che servono

Per costruire una piattaforma cloud

Chi usa il cloud?

Foto Posta Documenti Biglietti del treno



Un concerto in piazza

L'associazione decide: i biglietti ce li vendiamo da soli

Meno commissioni

Su ogni biglietto, una parte va alla piattaforma che lo vende.

I dati restano nostri

Chi viene, cosa compra, dove abita.

Software libero

Il programma per vendere biglietti esiste già: si usa e si cambia.

Posti limitati. Sabato, ore 12:00

si aprono le prenotazioni.

Indipendente non vuol dire “in cantina”

Una definizione prima di costruire

Poter scegliere il fornitore.
Poterlo cambiare.

Le quattro libertà del software libero

eseguire **studiare** **modificare** **condividere** + una **andarsene**

Il test: se domani cambio provider, quanto ci metto?

Non è una fissa: è nelle leggi

Le due metà della definizione stanno entrando nelle norme europee

Poterlo cambiare

DORA

Regolamento UE · in vigore da gennaio 2025

Banche e assicurazioni devono avere un piano per uscire dal fornitore cloud.

Non chiede un fornitore europeo. Chiede di poterlo lasciare.

Poterlo scegliere

CADA

Proposta della Commissione · giugno 2026

Quattro livelli di sovranità per il cloud della pubblica amministrazione.

1 2 **3** **4**

dal 3: proprietà e controllo europei

Proposta, non ancora legge.

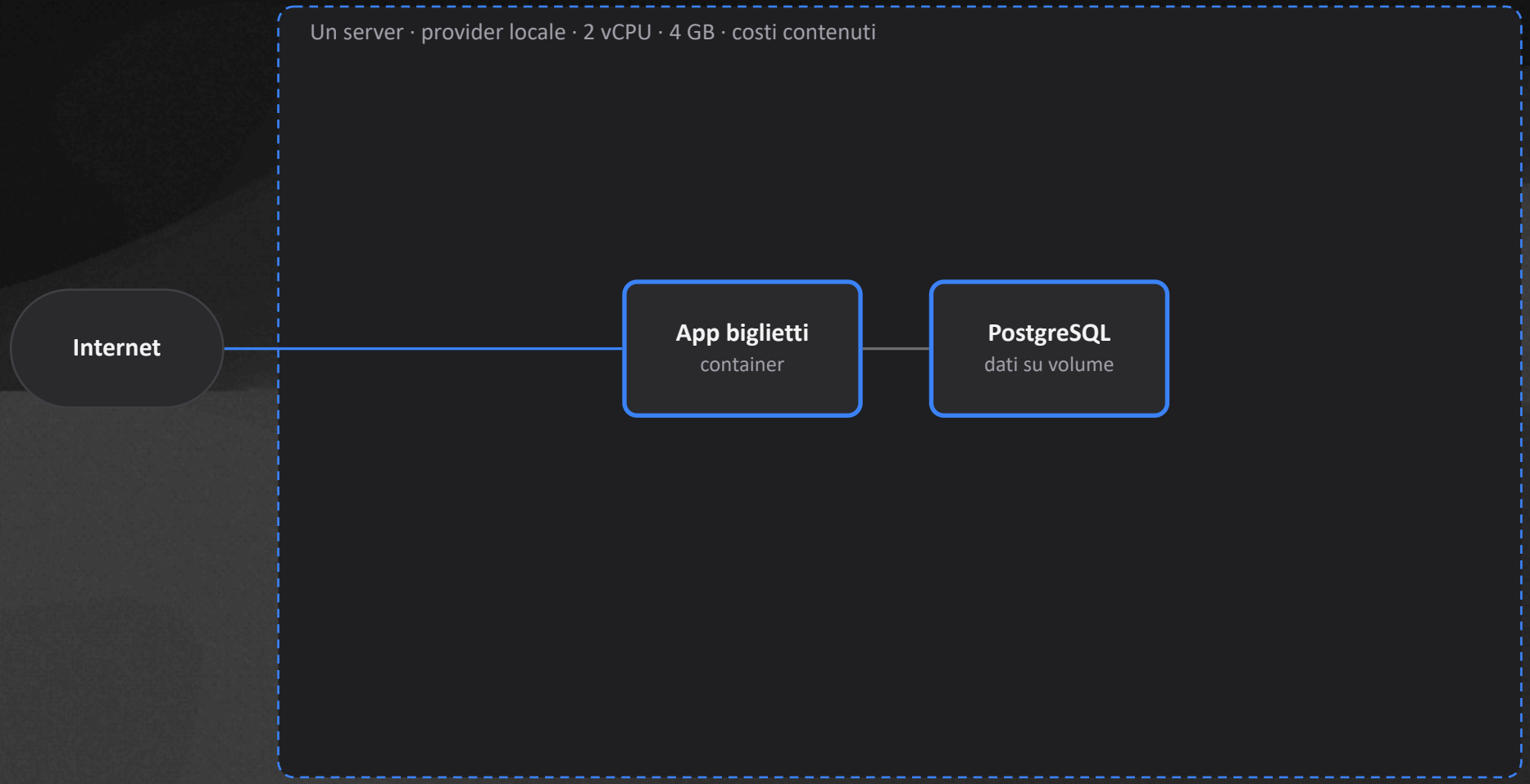
La quinta libertà, scritta nelle leggi.

Tre strade

Dove facciamo girare la piattaforma dei biglietti?

	Hyperscaler	Provider europeo	Server propri
Ti danno	Tutto: macchine, database, code, pure l'IA. Nel loro modello.	Macchine, rete, un'API, qualche servizio. Poi ti guardano.	Il controllo. Tutto il controllo.
Resta a te	Capire la fattura.	Tutto il resto: i pezzi li metti tu.	Anche l'hardware. Anche sabato alle 12:00.

Prima versione: un server, l'app, il database

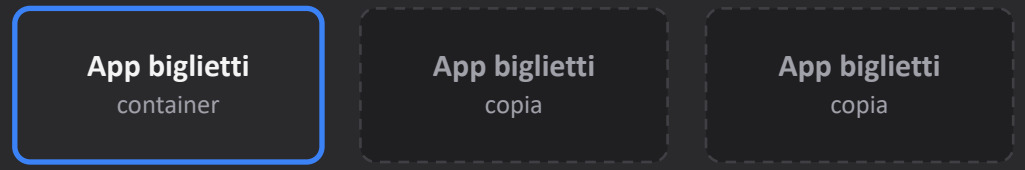


Il server si cambia, i volumi no

Il mattone su cui poggia tutto il resto

Il server

l'app, in un container



- si reinstalla da un'immagine, identico ogni volta
- se si rompe, ne prendi un altro
- se serve, ne accendi tre

I volumi

i dati, in un volume



- stanno su disco, e restano quando il container muore
- si copiano e si spostano con cura
- sono l'unica cosa che non si può rifare

Tutto quello che viene dopo funziona perché questi due sono separati.

Il DNS

Senza un nome, il concerto c'è e non ci arriva nessuno

~~https://203.0.113.42~~ → **biglietti.concerto.it**

L'elenco

Il DNS traduce il nome in un indirizzo. È l'elenco telefonico di Internet, e la vostra riga la scrivete voi.

La scadenza

Ogni risposta ha un TTL. La rispettano tutti quelli in mezzo, fino al browser. A 60 secondi, un cambio arriva in un minuto.

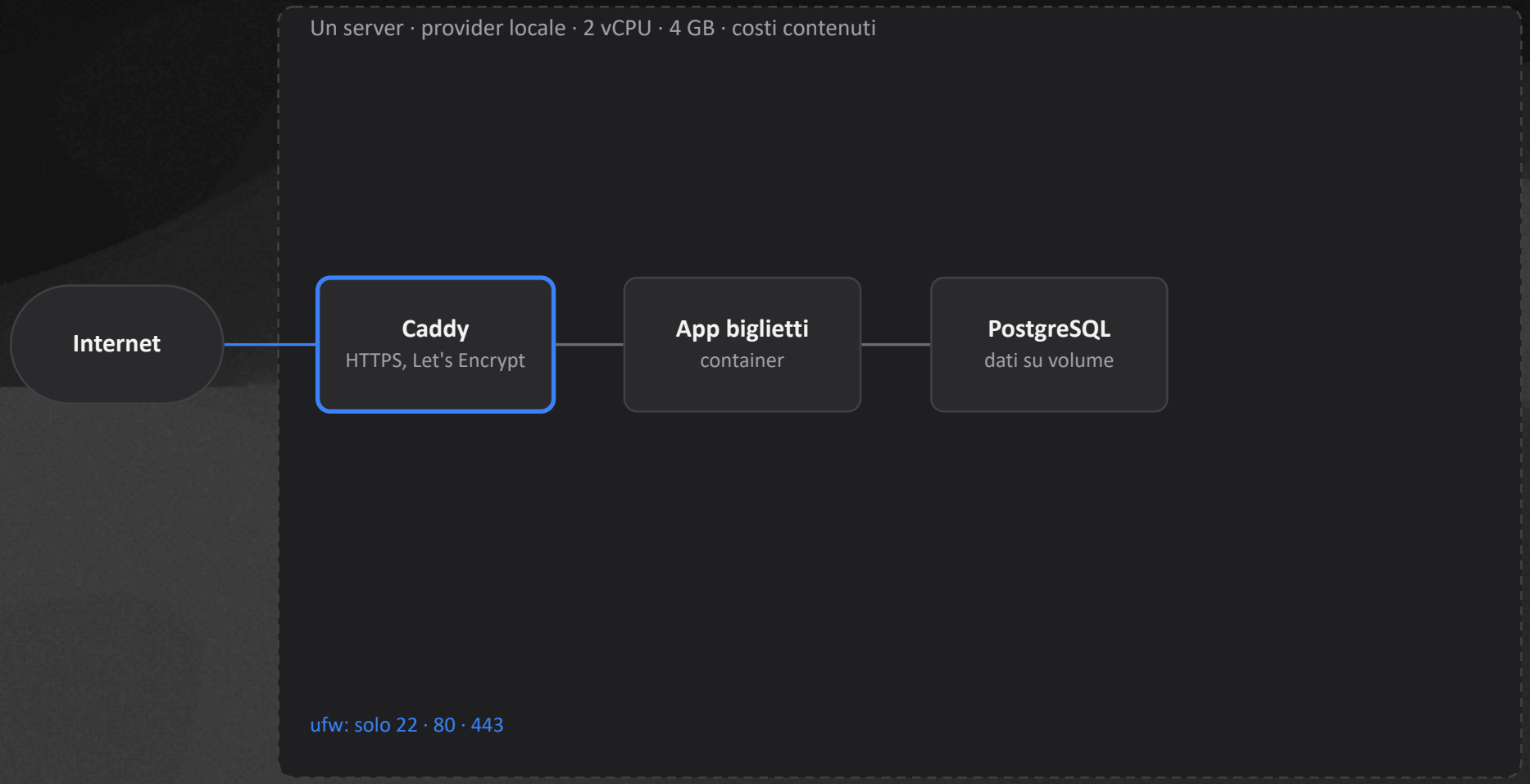
Il nome è vostro

L'indirizzo è del provider. Quando cambiate provider, cambia l'indirizzo. Il nome no.

Il nome è la cosa che resta quando cambiate tutto il resto.

Cosa mettiamo alla porta

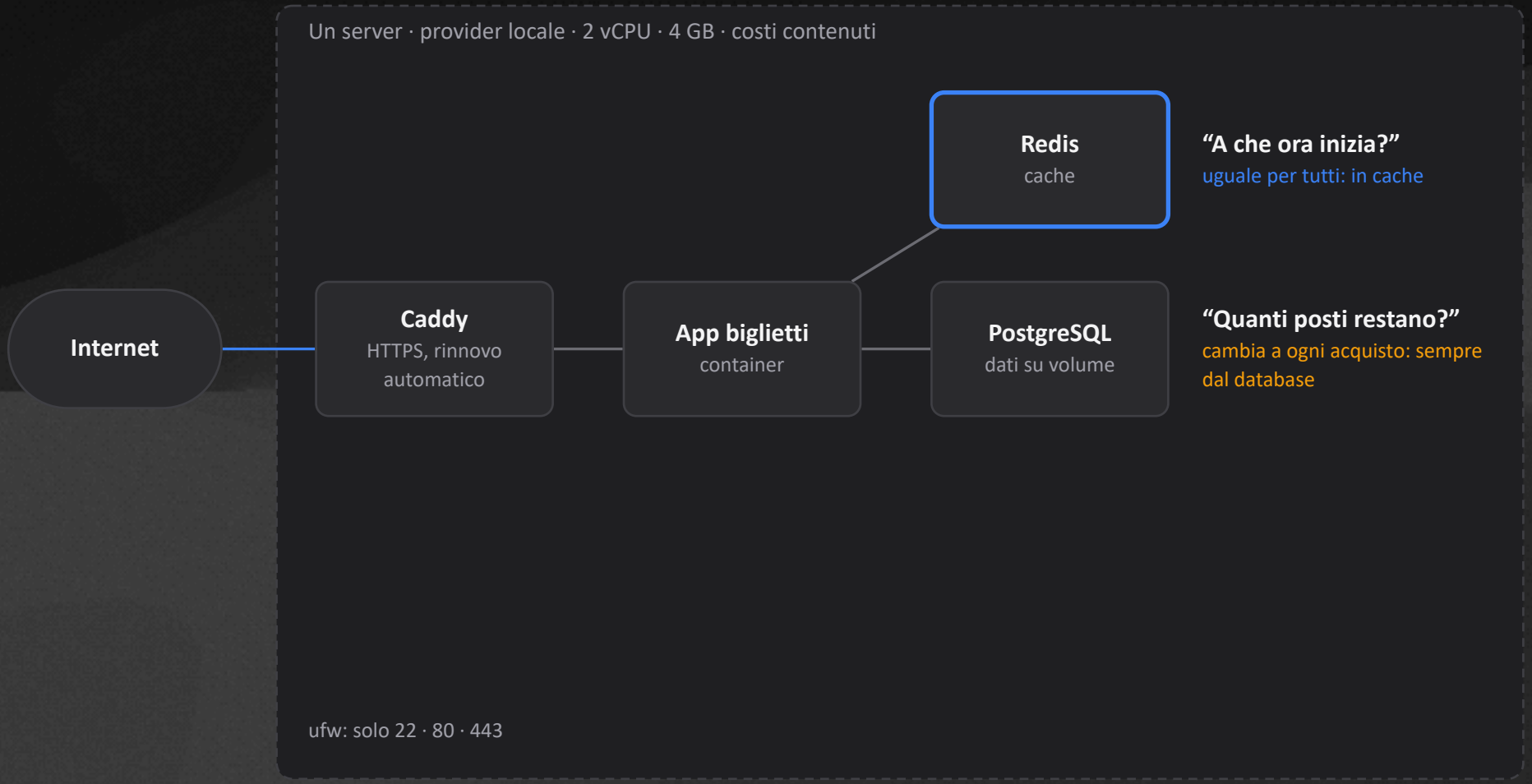
Su un server solo bastano tre programmi



HTTPS che si rinnova da solo, porte chiuse tranne tre, e chi insiste resta fuori.

Annuncio dell'ospite: mille persone chiedono la stessa cosa

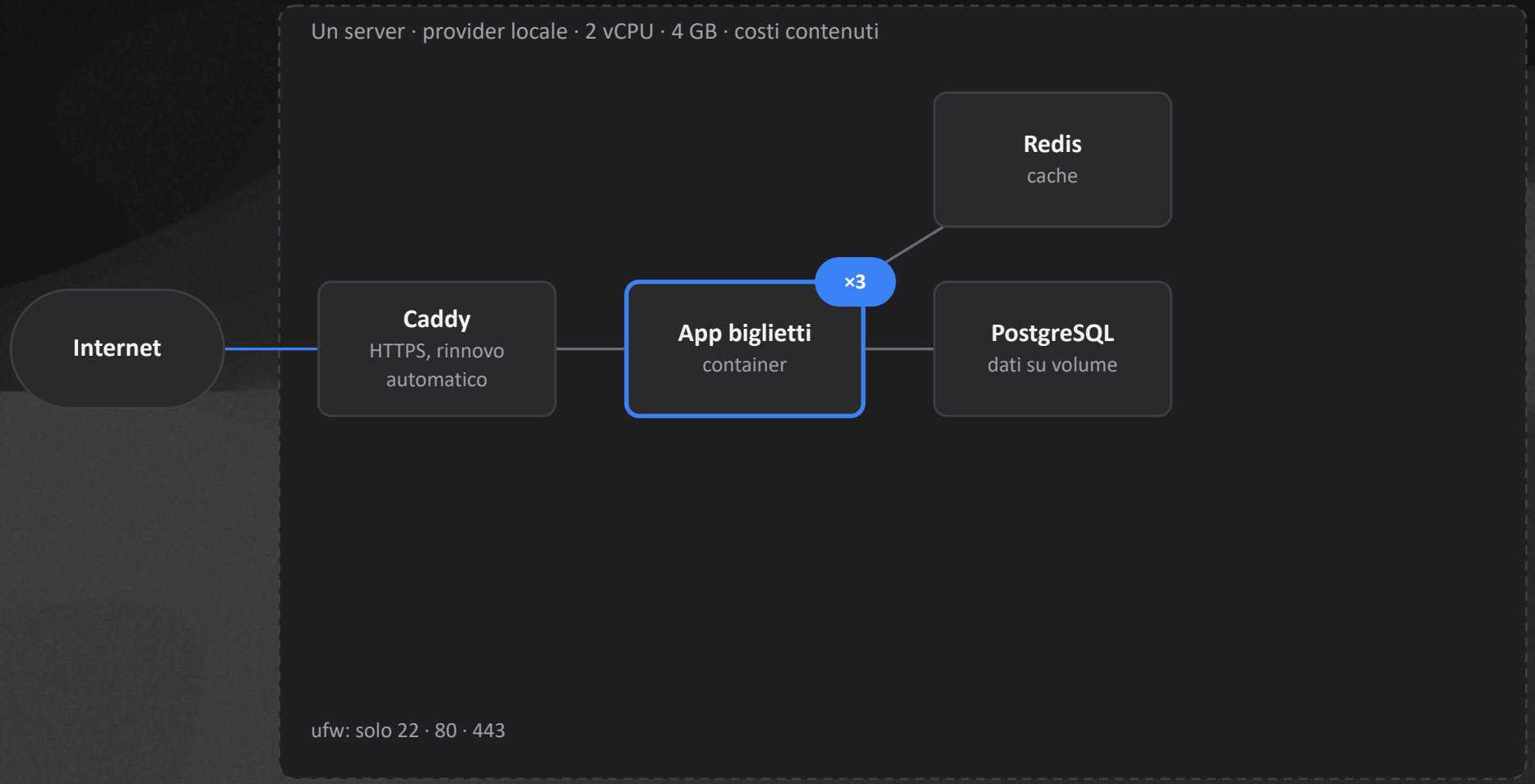
“A che ora inizia?” Il database dà mille volte la stessa risposta.



In cache ci va quello che può essere un po' vecchio senza danni.

Sabato, 11:59

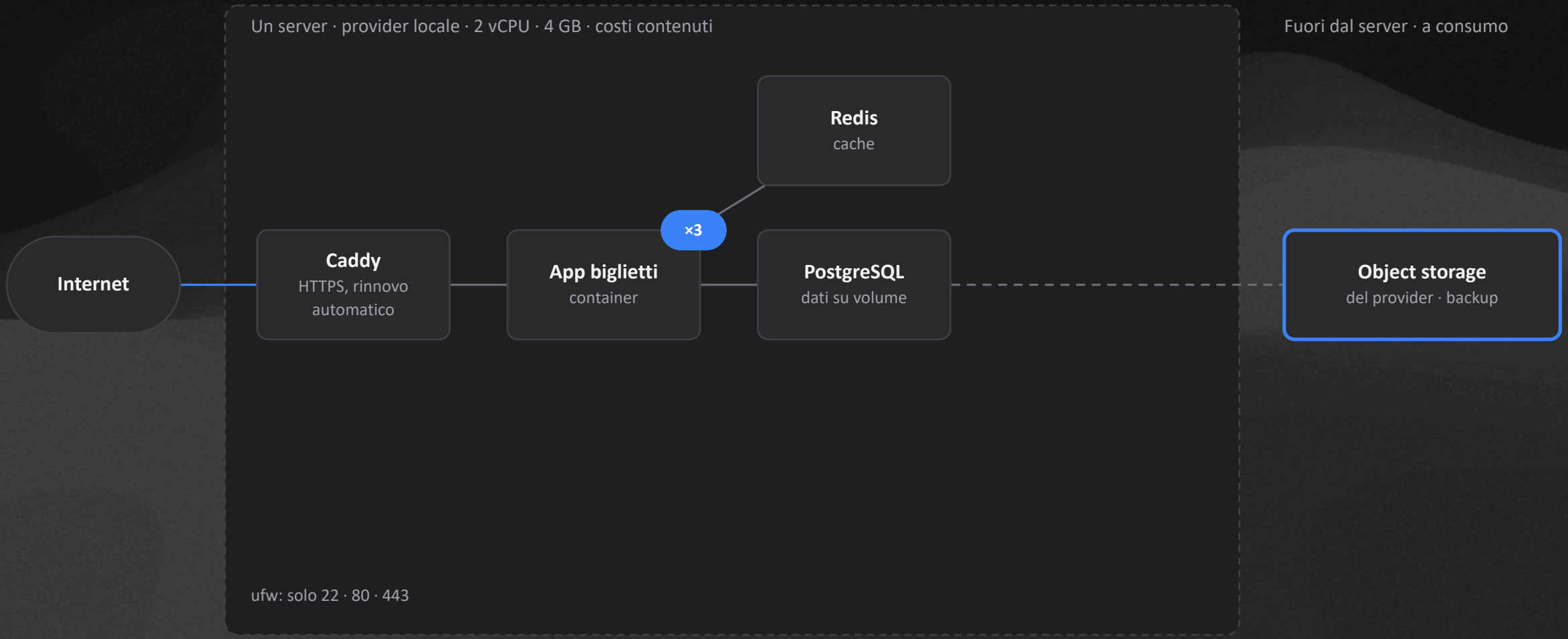
Alle 12:00:03 la zia preme F5 quaranta volte. Non è un attacco. È la zia.



Tre copie dalla stessa immagine, in dieci secondi. Ma devono comportarsi come una.

Se il server muore stanotte, cosa resta?

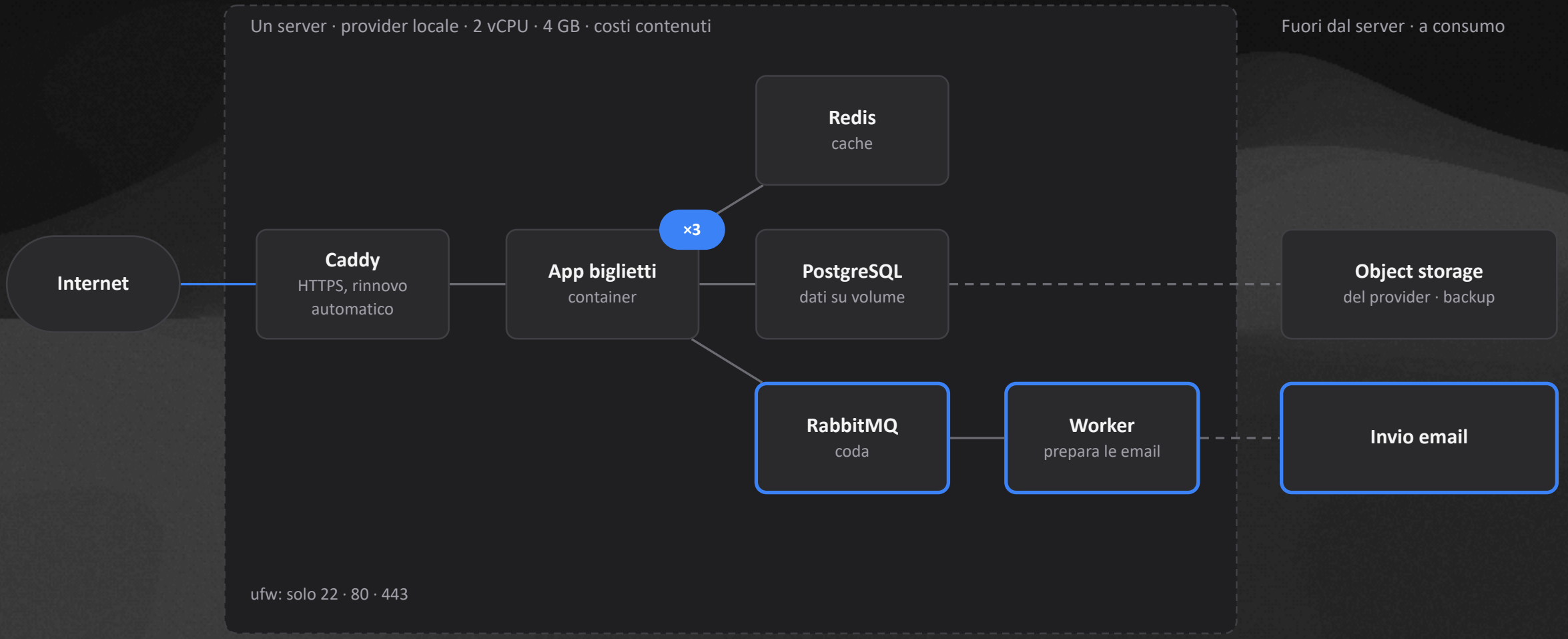
Il database sono le foto: il backup va fuori dal server



Un backup che non avete mai ripristinato non è un backup. È una speranza.

Se l'email si ferma, blocchiamo le vendite?

Il biglietto è venduto quando è nel database. L'email arriva quando arriva.



La coda la teniamo noi. La consegna la fa chi lo fa di mestiere.

Cosa non si fa in casa

Si compra a consumo, e si può cambiare

DNS

Deve rispondere anche quando il vostro server è giù.

DNS del provider

Storage dei backup

Se sta sul server, muore con il server.

Object storage

Email

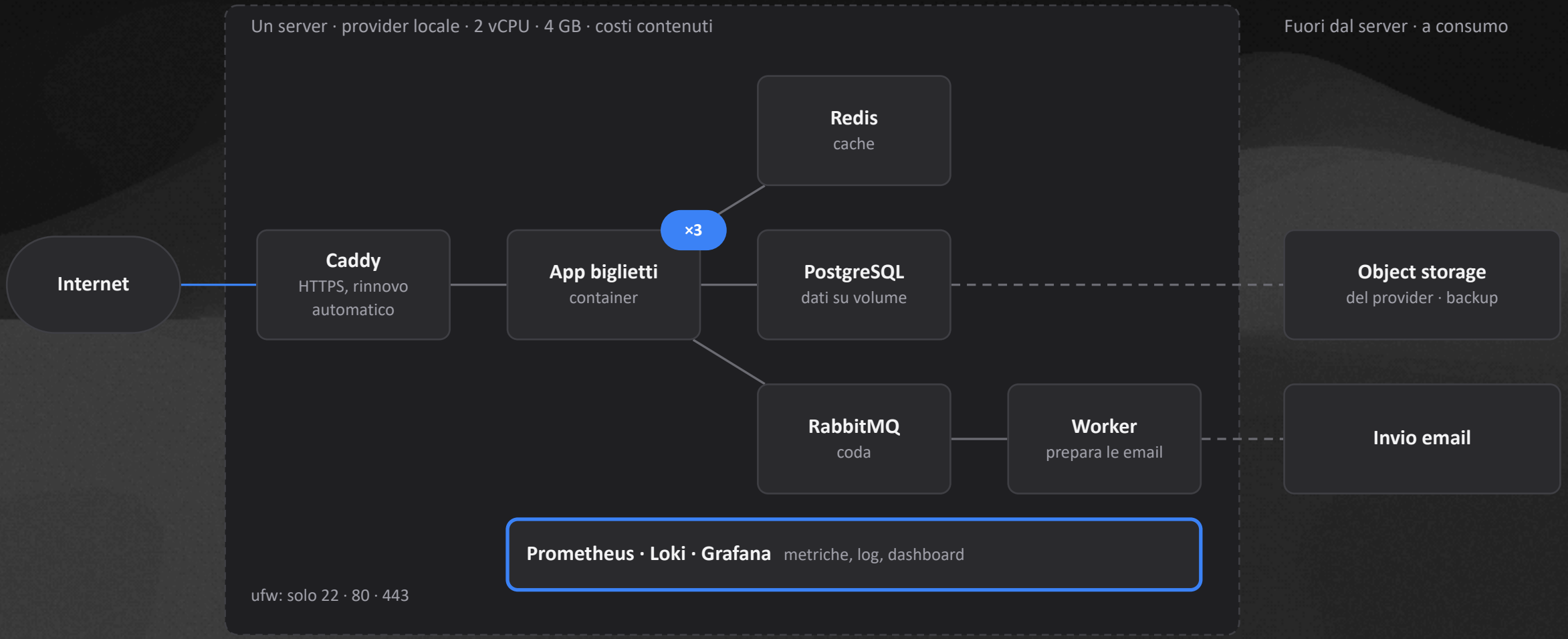
Consegnare posta senza finire nello spam è un mestiere.

Server di posta del provider

Indipendente non vuol dire fare tutto in casa. Vuol dire poter cambiare chi lo fa.

Come capiamo cosa sta succedendo?

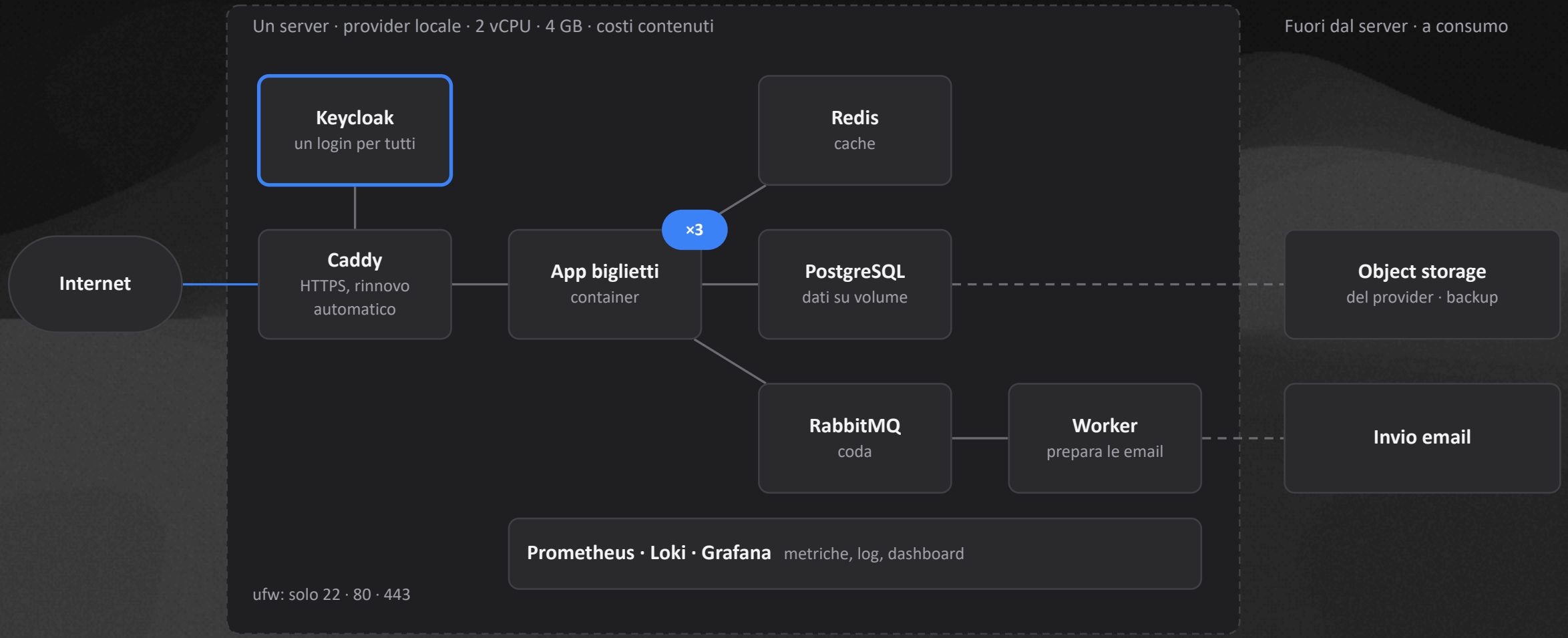
Senza strumenti, il monitoraggio siamo bendati



Prima richieste, tempi ed errori. Il resto dopo.

Adesso le porte sono due

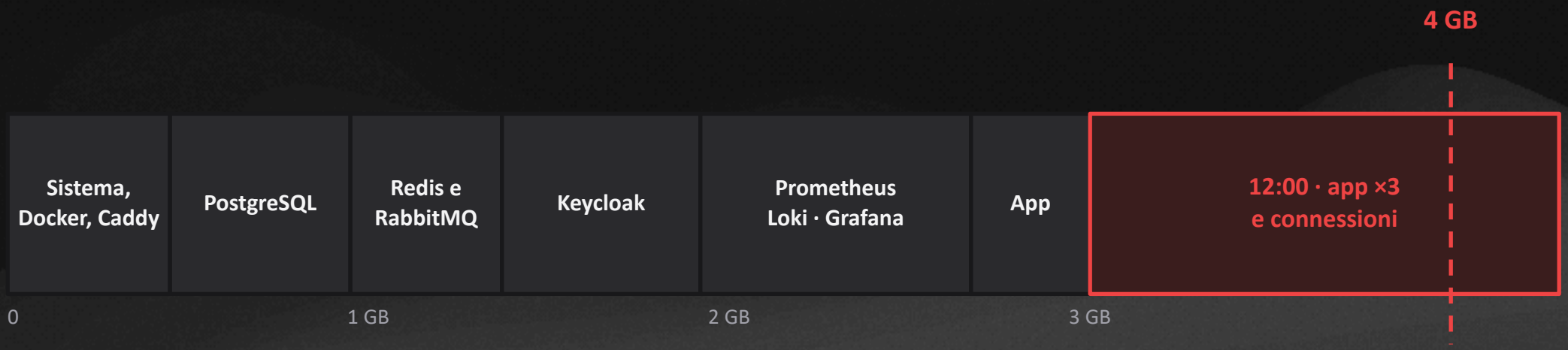
L'area organizzatori dell'app e Grafana: un login solo per tutte e due



Chi sei lo chiedi una volta. Poi tutti si fidano di quella risposta.

Il server è pieno

Quattro giga, e tutto quello che ci abbiamo messo dentro



Tutto quello che abbiamo aggiunto ha un costo in memoria.

E alle 12:00 arriva il picco.

Proporzioni a spanne: misurate le vostre con docker stats.

Più grande, o più server?

Verticale o orizzontale: allo stesso prezzo, due mondi diversi

Un server più grande

- semplice: cambi taglia e riavvii
- ma ha un tetto
- e resta uno solo: se muore, muore tutto

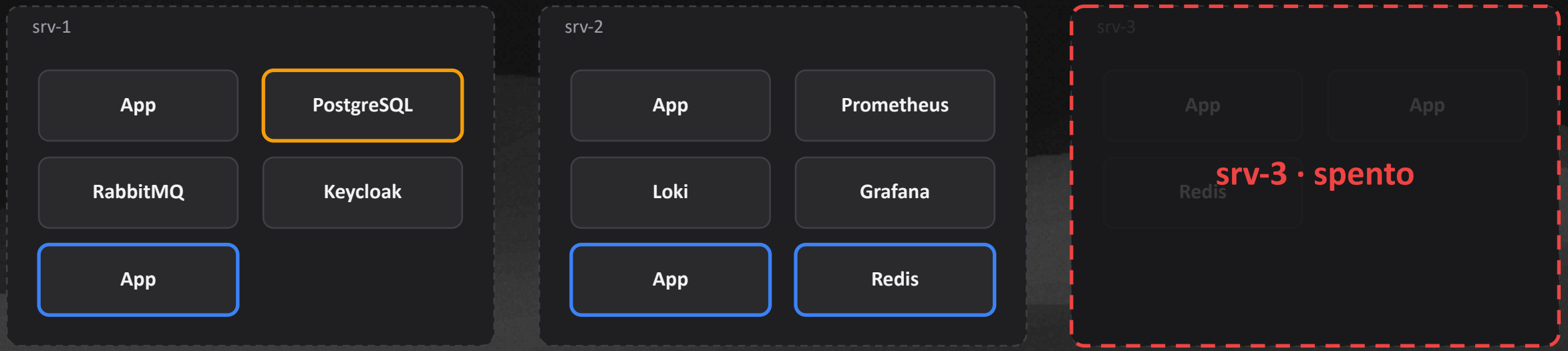
Più server

- nessun tetto: ne aggiungi un altro
- se uno muore, gli altri restano
- **ma qualcuno deve decidere cosa gira dove**

Quel “qualcuno” è il motivo per cui esiste Kubernetes.

Chi decide cosa gira dove?

Tre server. Alle 12:05 uno si spegne.



Serve qualcuno che decida cosa gira dove,
e che se ne accorga quando un server muore.

Si chiama orchestratore. Il mio è K3s: Kubernetes in un binario, circa mezzo giga di RAM.

I container ripartono altrove. Il database no: le foto, ricordate?

Gli stessi pezzi, da un server a un cluster

La lista che vi portate a casa

	Un server	Un cluster
Ingresso e HTTPS	Caddy	**Traefik + cert-manager**
Firewall	ufw	**nftables su ogni nodo + firewall del provider**
Container	Docker	**K3s: Kubernetes, con containerd**
Rete	tutto sulla stessa macchina	**rete privata del provider · WireGuard tra provider**
Database	PostgreSQL	**PostgreSQL con operatore: CloudNativePG**
Cache e coda	Redis · RabbitMQ	**gli stessi, come servizi del cluster**
Identità	Keycloak	**Keycloak**
Metriche e log	Prometheus · Loki · Grafana	**gli stessi, raccolti da ogni nodo**
Backup, email, DNS	fuori, a consumo	**fuori, a consumo**

Dareste la vostra carta di credito a un programma?

Alle 12:10 tre server non bastano. Autoscaling vuol dire: li compra lui.

Le regole che gli date

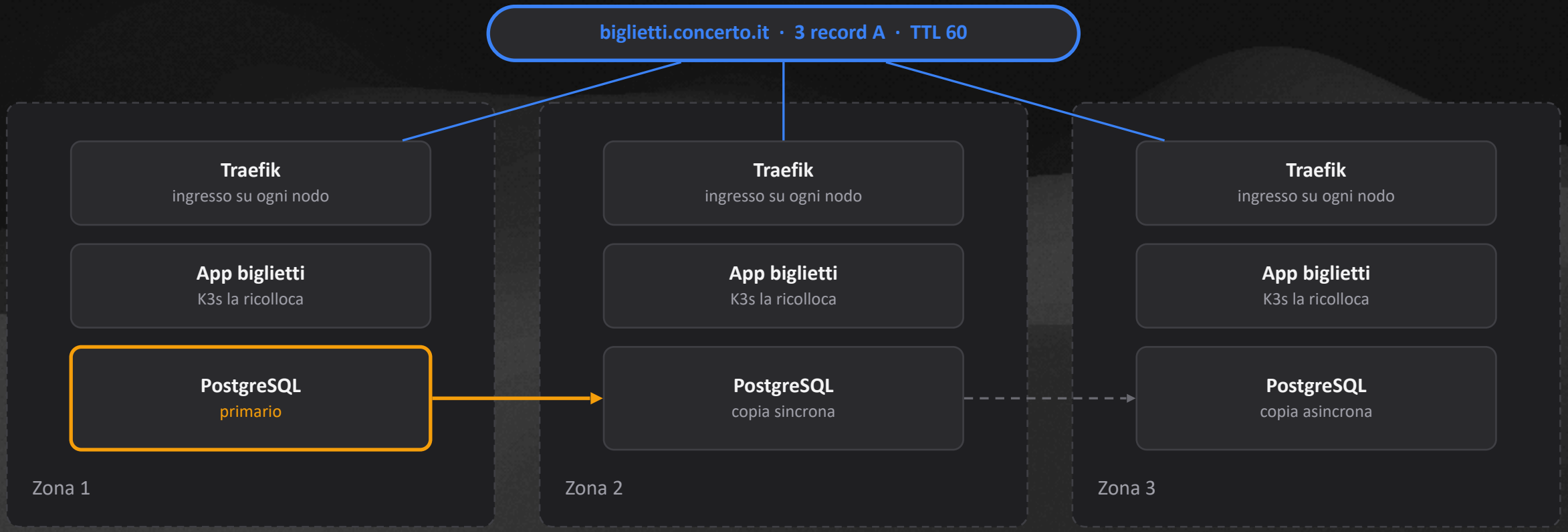
Quando	**CPU sopra il 70% per 5 minuti**
Minimo	**1 server**
Massimo	**4 server**
Tetto	**60 € al mese**
Oppure	**avvisami, e aspetta il mio sì**

Alle 14:00 il picco è finito

- restituisce i server in più
- aspetta che finiscano i lavori in corso
- non spegne quello con il database
- non oscilla su e giù ogni cinque minuti

Senza un tetto, l'autoscaling è una carta di credito in mano a un ciclo for.

Tre datacenter, un cluster

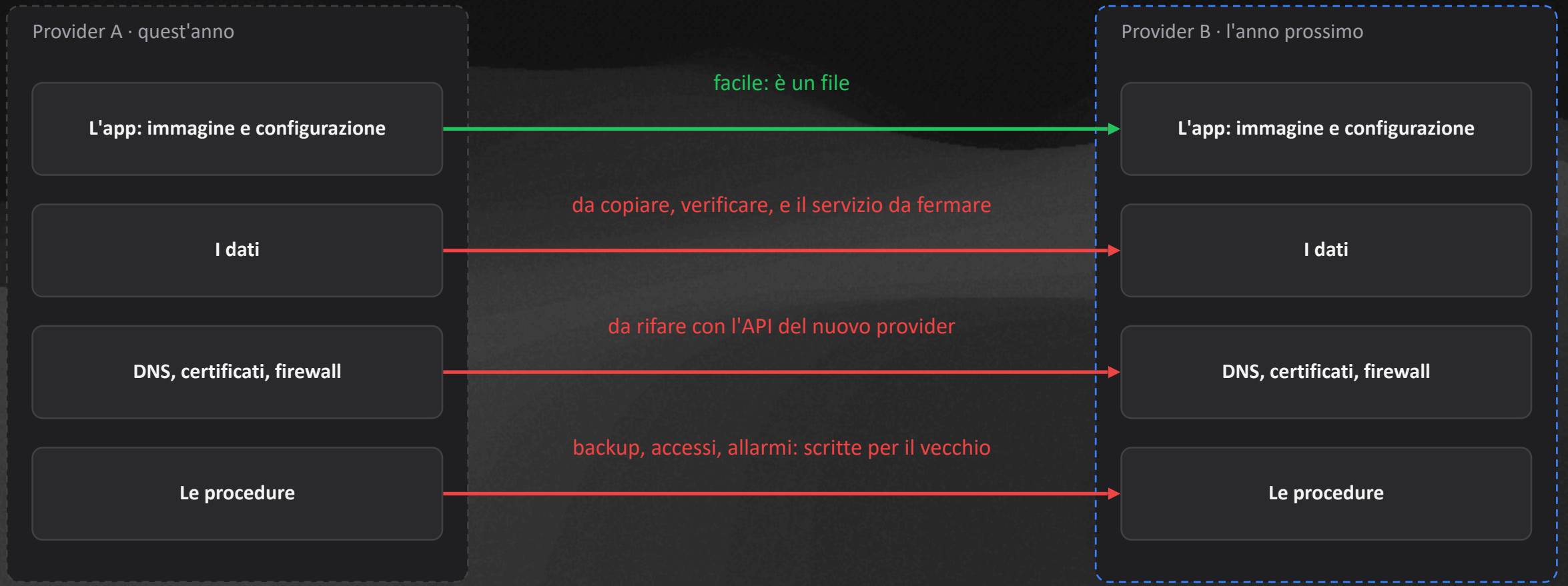


CloudNativePG: uno scrive, gli altri copiano · il commit aspetta la copia più veloce · promozione automatica

È il multi-AZ. Fatto da voi, con software libero, su un provider di Norimberga.

L'anno prossimo cambiamo provider?

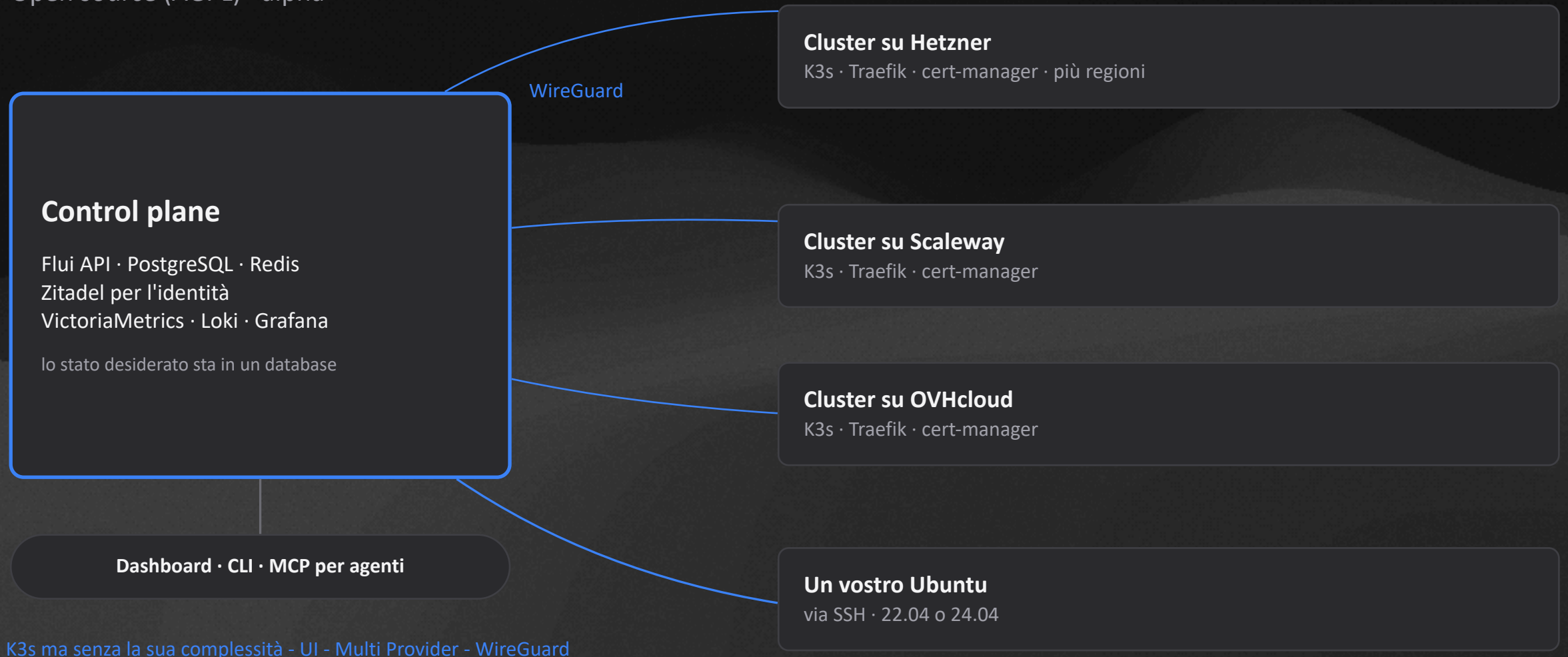
Portare via il container è come traslocare portando via solo il frigorifero



Il test dell'inizio: se domani cambio provider, quanto ci metto? Questa è la risposta onesta.

Flui: mettere insieme questi pezzi

Open source (AGPL) · alpha



K3s ma senza la sua complessità - UI - Multi Provider - WireGuard

Se vi basta un server, Coolify o Dokploy.

Requisito per un provider: un'API e il pagamento orario. In Europa ce ne sono più di uno.

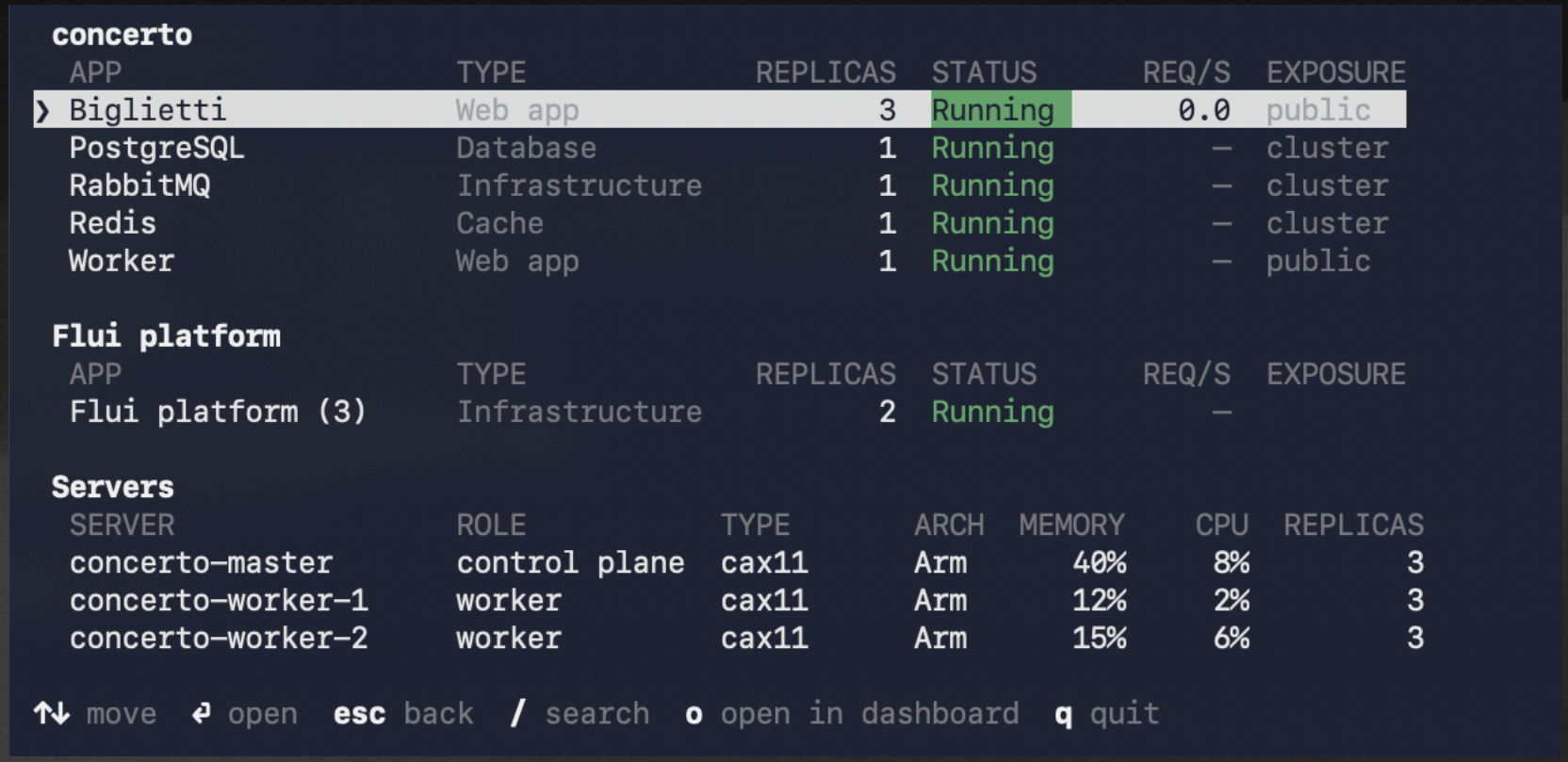
I problemi di oggi, in Flui

Dove sono finiti, uno per uno

Alla porta	HTTPS e firewall impostati quando nasce il nodo; limiti e login unico per ogni rotta
Il nome	record DNS scritti da Flui, se la zona sta su un provider che sa pilotare
Le foto	PostgreSQL dal catalogo, backup con point-in-time
Capire cosa succede	metriche, log e allarmi dal primo minuto
Più server	un minimo, un massimo e un tetto di spesa mensile
Più provider	un control plane solo, cluster collegati via WireGuard
Cambiare	sposta un'app e il suo PostgreSQL su un altro cluster, e passa quando sei pronto
Cosa non fa	un'app gira su un cluster solo: niente failover automatico tra provider

La demo è online, in sola lettura: il link è nell'ultima slide.

flui topology



flui topology app Biglietti

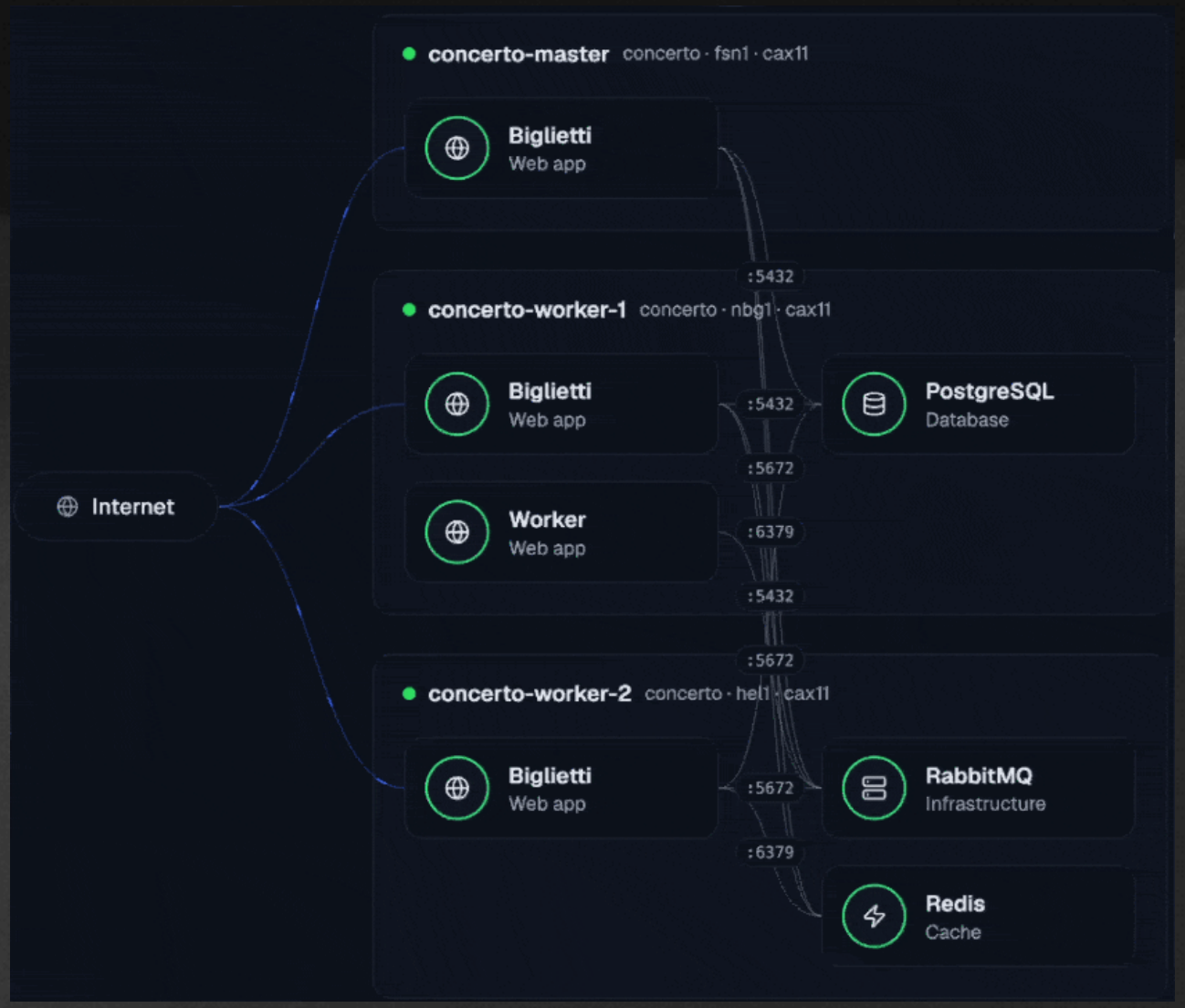
```
Fleet › concerto › Biglietti
Biglietti  Running  concerto · public · Web app
Replicas      3 · manual              Traffic in    0.0 req/s
Memory        16 MB, limit 64 MB      CPU           10m, limit 100m
Stack         —                       Project       concerto

Reachable from the Internet
  https://biglietti.fluicloud.eu            → :80       0.0 req/s  — errors

Talks to
  Redis                    concerto            :6379
  RabbitMQ                 concerto            :5672
  PostgreSQL               concerto            :5432

Runs on
  concerto-worker-1        ×1  worker          nbg1    memory 12% · cpu 2%
  concerto-worker-2        ×1  worker          hel1    memory 15% · cpu 7%
  concerto-master          ×1  control plane   fsn1    memory 42% · cpu 10%
dawit@192 ~ %
```

flui topology cluster concerto --open



There is no cloud.
It's just someone else's computer.

Grazie

Dawit Abate Woldeamanuel

Demo in sola lettura: demo-app.flui.cloud/try



github.com/flui-cloud/flui-core