Checkmk Conference #11

Running Checkmk at scale

Max Linke
Team Lead
Development
Checkmk GmbH

Cristian Blanaru
Team Lead
Site Reliability Engineering
Checkmk GmbH

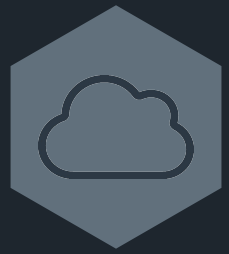
Tales from the trenches

Preparing Checkmk

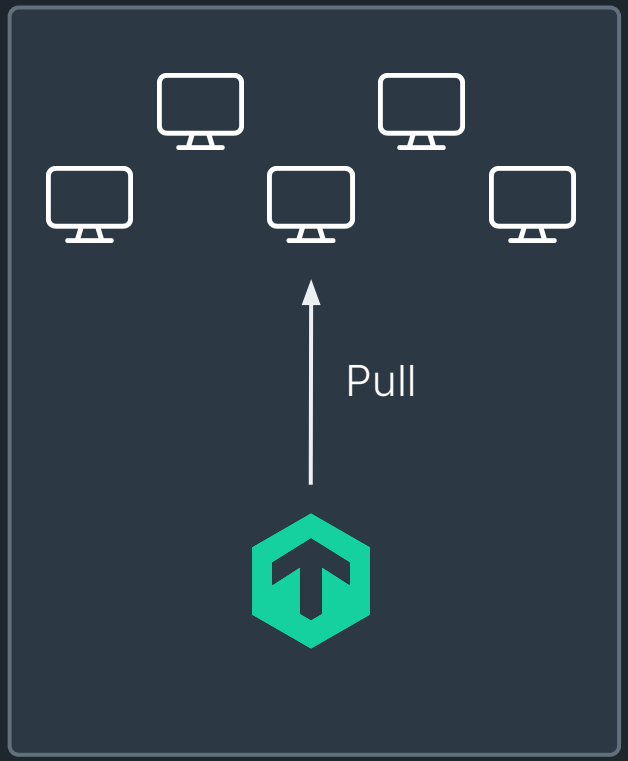
- Security made easy for you
- Security made easy for us

Building a SaaS platform

- Authentication
- Updating Checkmk



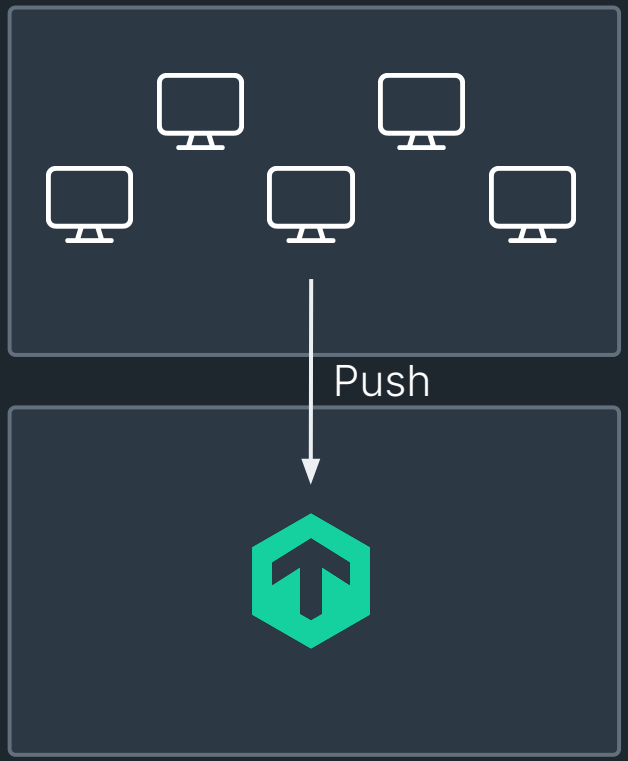
Security made easy for you



Secure by design

- default option is secure
- default is easy to use

Security made easy for you



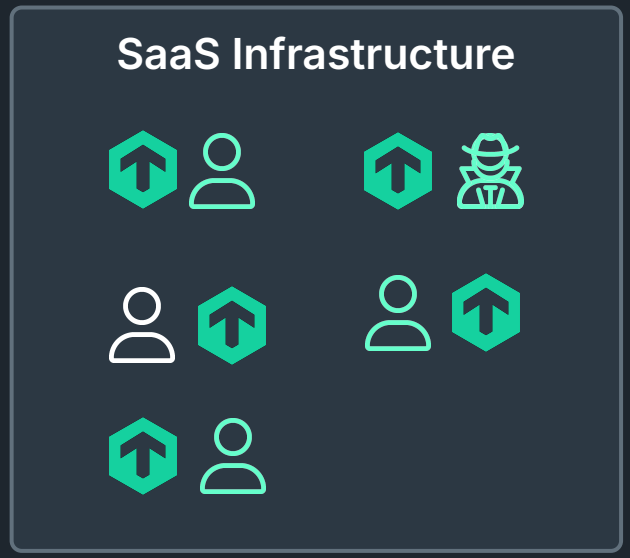
Push agent available since 2.2

- Not the default choice in the CCE or CME
- Requires extra configuration

Push mode is the only and easy choice

- Pre-Configure push agent
- Two step onboarding guide to install agent
- Disable pull mode

Security made easy for us



Remote Code Execution as a Feature (for admins)

- Checkmk extension packages (MKP)
- “Individual program call instead of agent access”
  - Run arbitrary shell commands on your checkmk server

Prevention in SaaS

- Explicitly disable features in SaaS
- Explicitly allow rules in SaaS
  - Check at start with rules and features are enabled
  - Regularly update enabled rules

Tales from the trenches

Preparing Checkmk

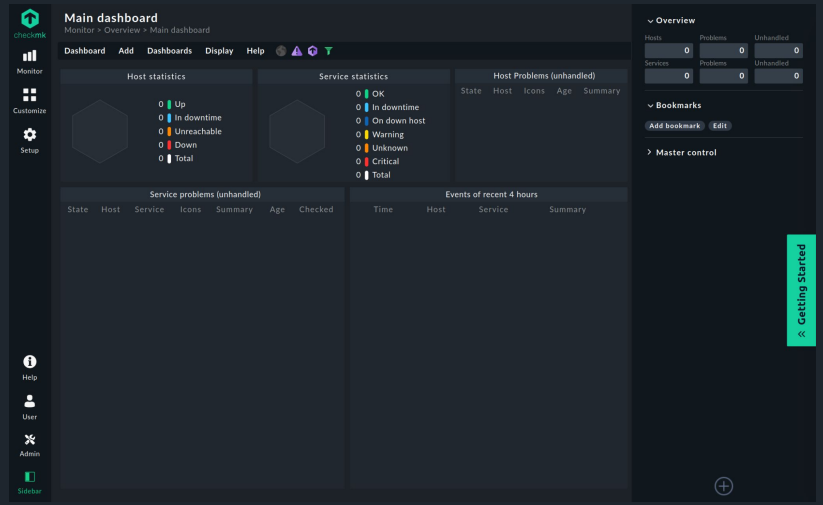
- Security made easy for you
- Security made easy for us

Building a SaaS platform

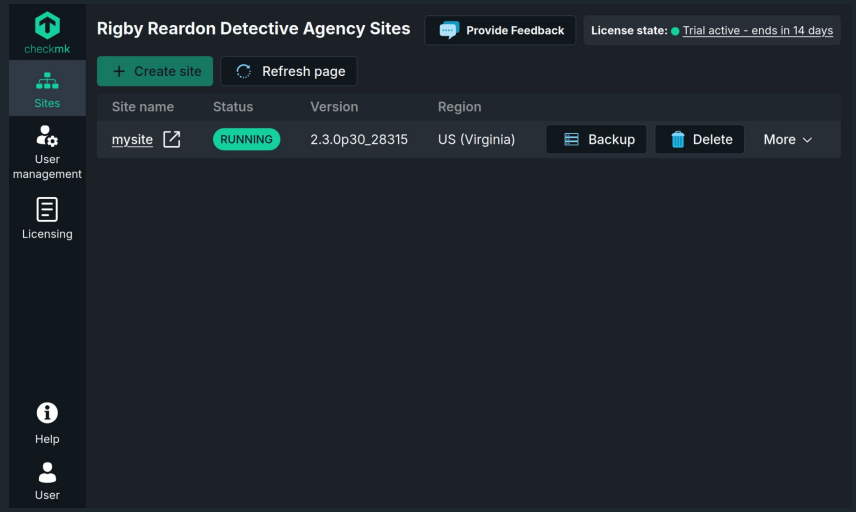
- Authentication
- Updating Checkmk

What is Checkmk Cloud?

Checkmk



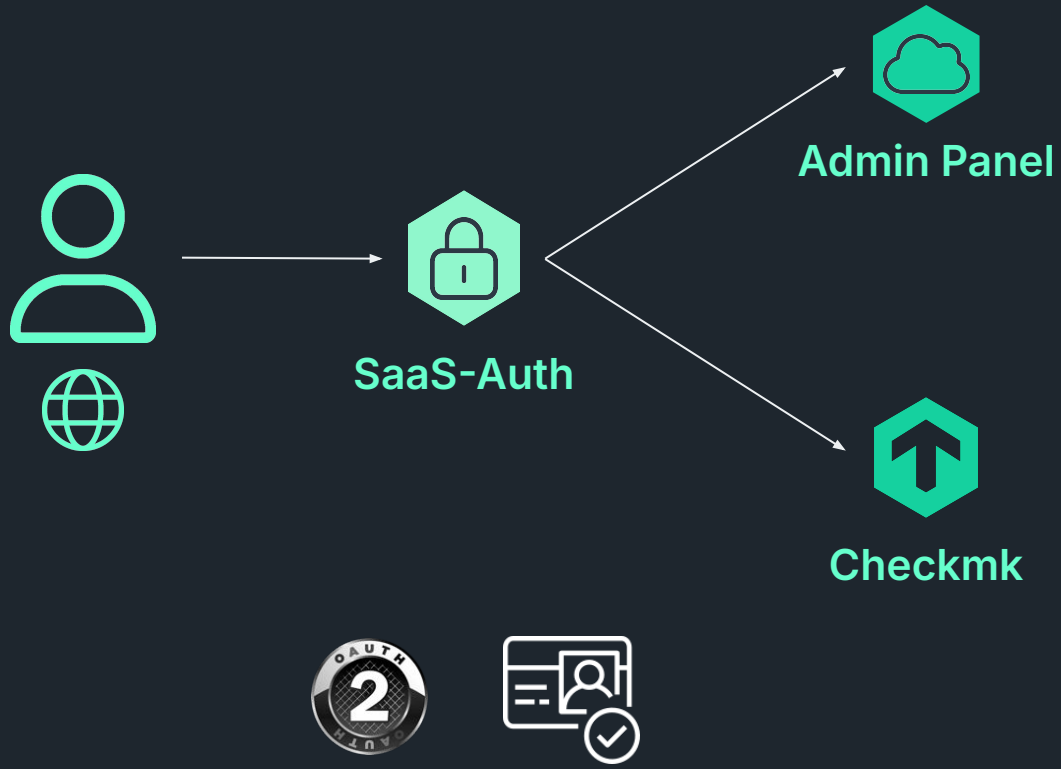
Admin Panel



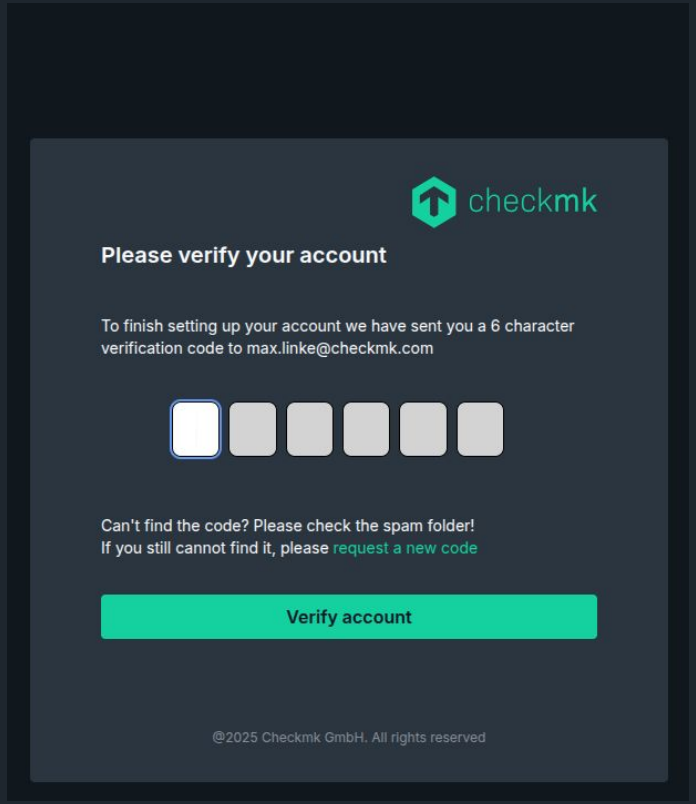
Solving Authentication

Desired features

- Seamless transition
- Self service
- Secure



Good UX in Authentication is hard



The next day

What should we do?

Updating Checkmk sites

Minimal Impact → Nightly Maintenance

Work for thousands of sites → Automated

Reliable → How can we do that?

What happens before the update?

Test `omd update` extensively on all editions daily

Test the current development of checkmk in SaaS daily

Test backup and restore functionality in SaaS daily.

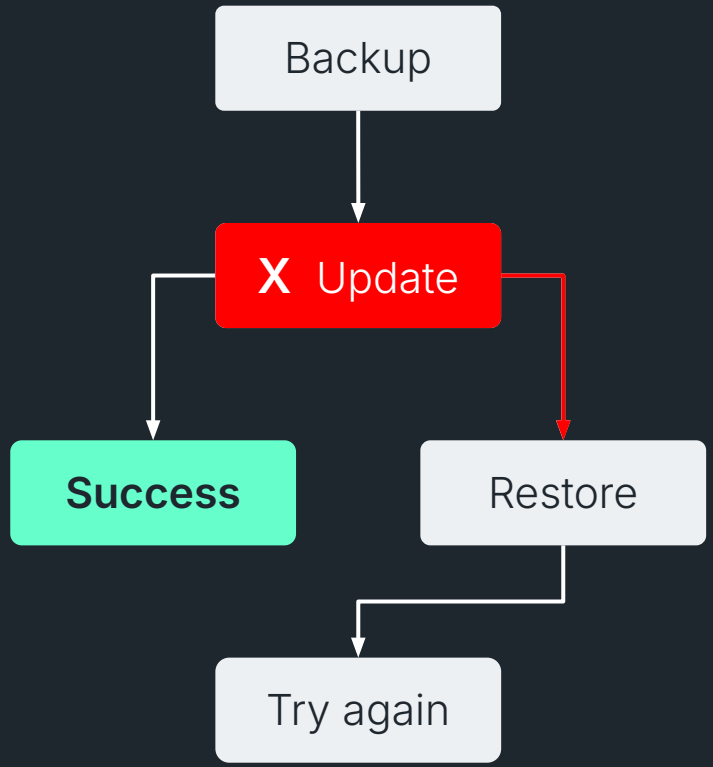
Updating a site

Handle errors in omd update

- Backup before the update
- Restore after failure

Handle errors with the update job

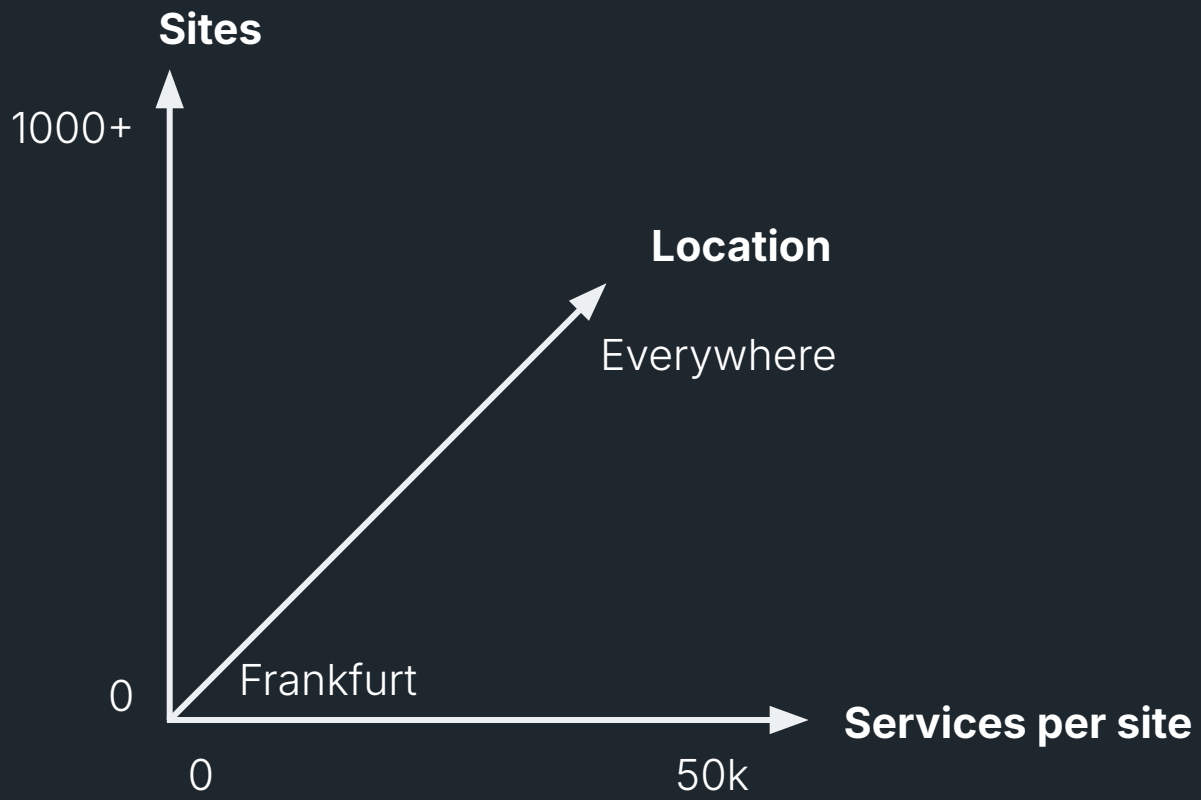
- Run update job as a kubernetes Job
- Handle restarts gracefully



How we built our SaaS platform

First things first

What does scalable mean?



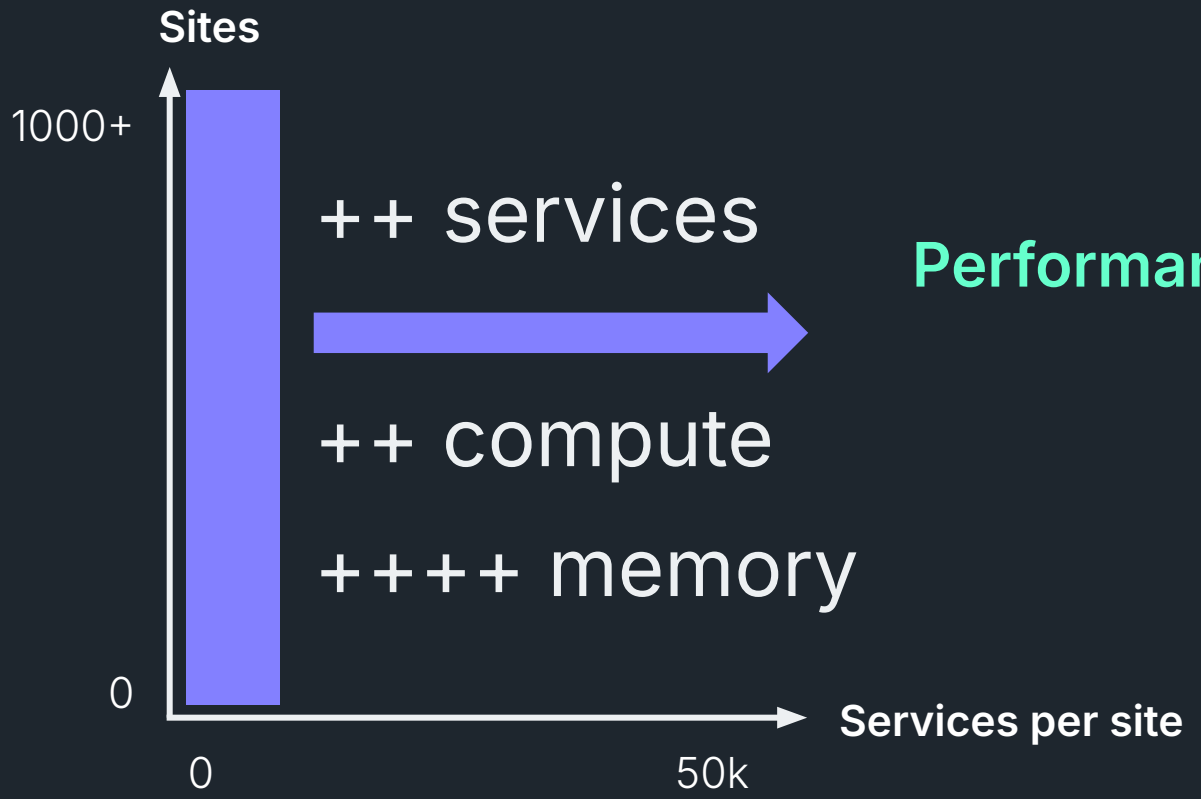
Scaling challenge #1: A lot of short-lived sites!

100+ trials

And they should be:

- On-demand
- Short lived
- Cheap

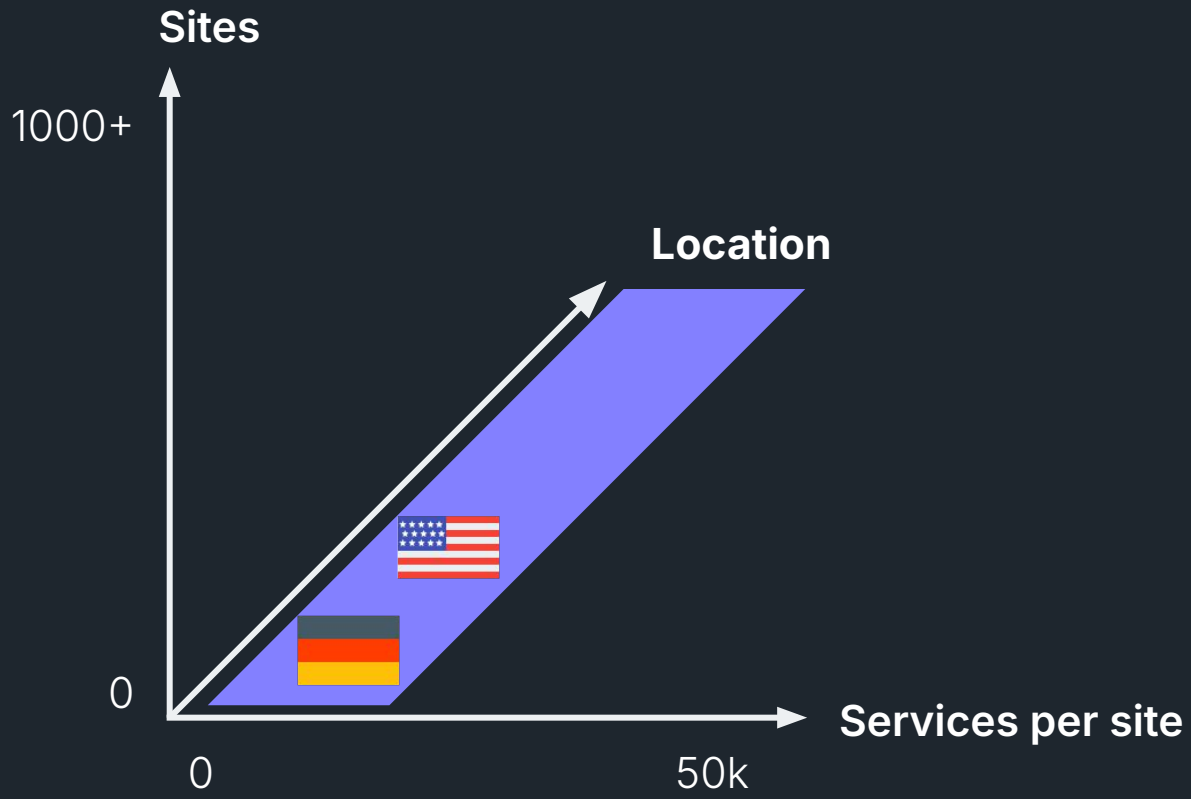
Scaling challenge #2: Unpredictable growing sites



Performance

Cost

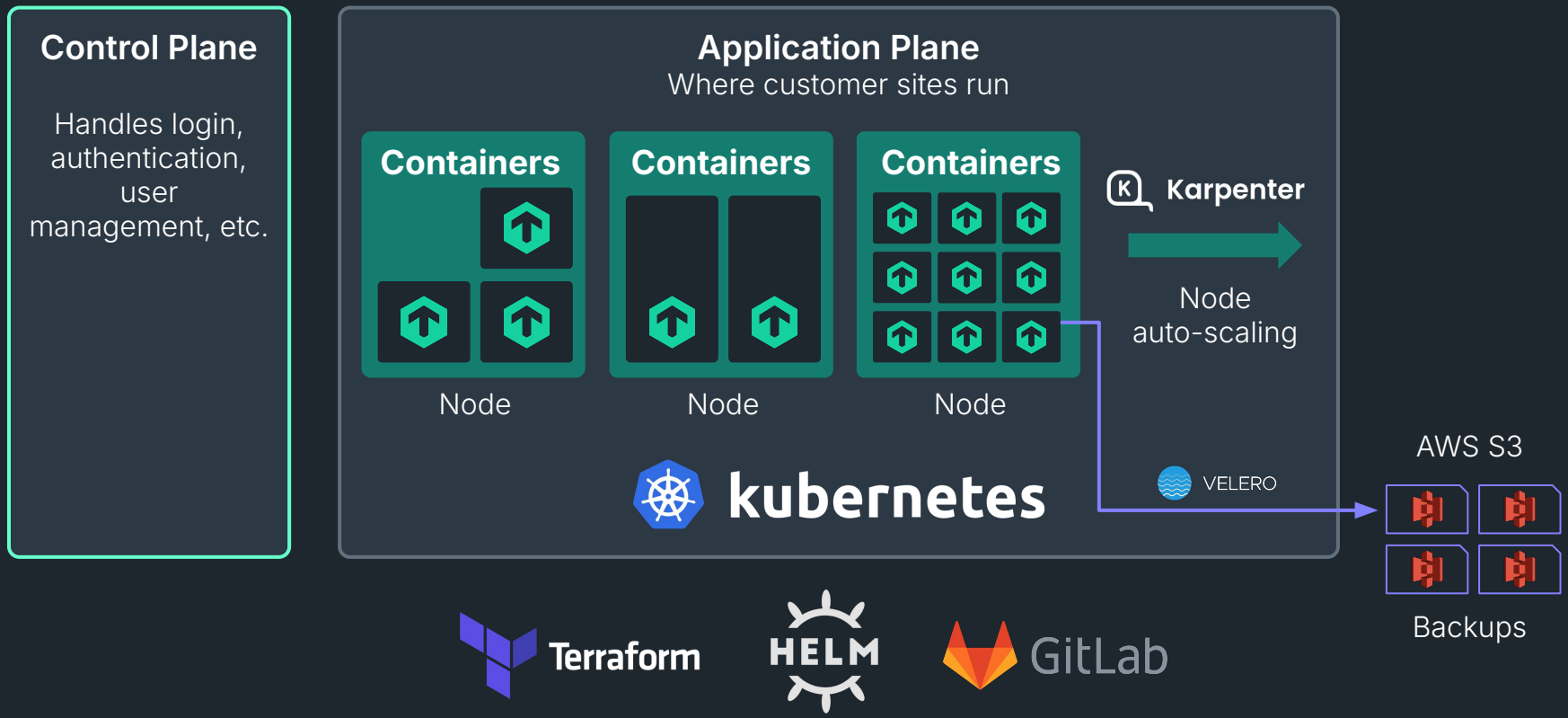
Scaling challenge #3: Across the world



How to tackle the scalability problem?

No click-ops	Infrastructure-as-code
On-demand resources	Auto-scaling infrastructure
Resource pooling	Containers
Orchestration	Kubernetes
Globally replicable	Hyper-scaler

Our SaaS platform ... super simplified illustration!



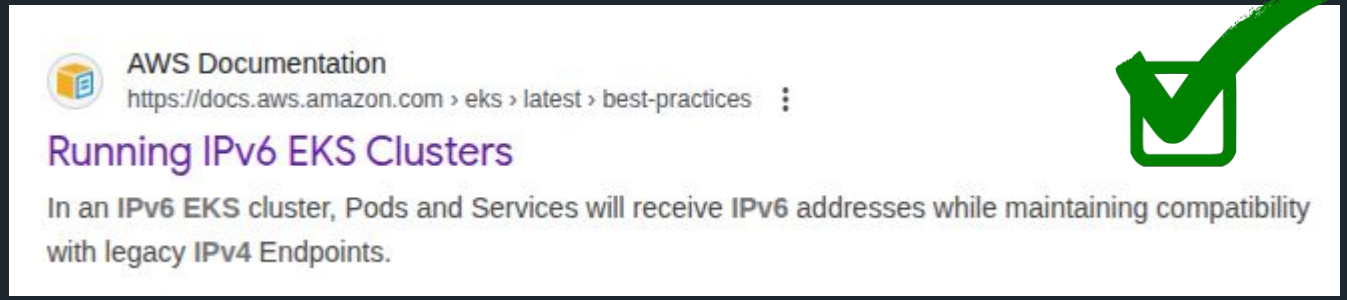
How we built our SaaS platform

The Dual-Stack Challenge (in AWS)

IPv4 CIDR
172.16.0.0/16

OR

IPv6 CIDR
2001:ODB8::/64



Who stirred the pot?

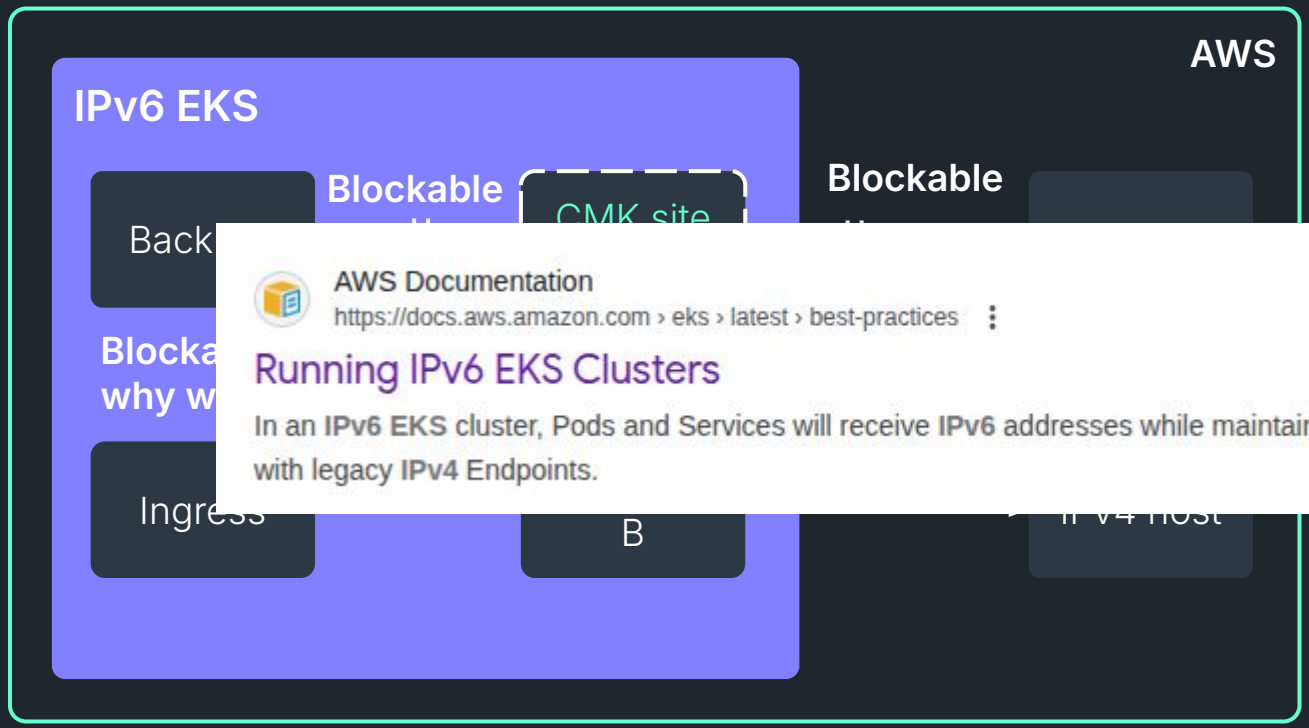
Martin Mar 22nd, 2024 at 2:58 PM

Ah, the IPv6 fallout. I should have said no to it and then we would all be in a happy world 😉

Cristian B. Mar 22nd, 2024 at 3:00 PM

:))) the IPv6 storm is here :) hey we fixed all our deployments in a day, I'm sure it'll be ez (last famous words)

Always test, never assume!



also have IPv4
s!

at if an external
omplains and
we need to block IPv4
traffic?

The Dual-Stack Challenge (in AWS)

Problem:

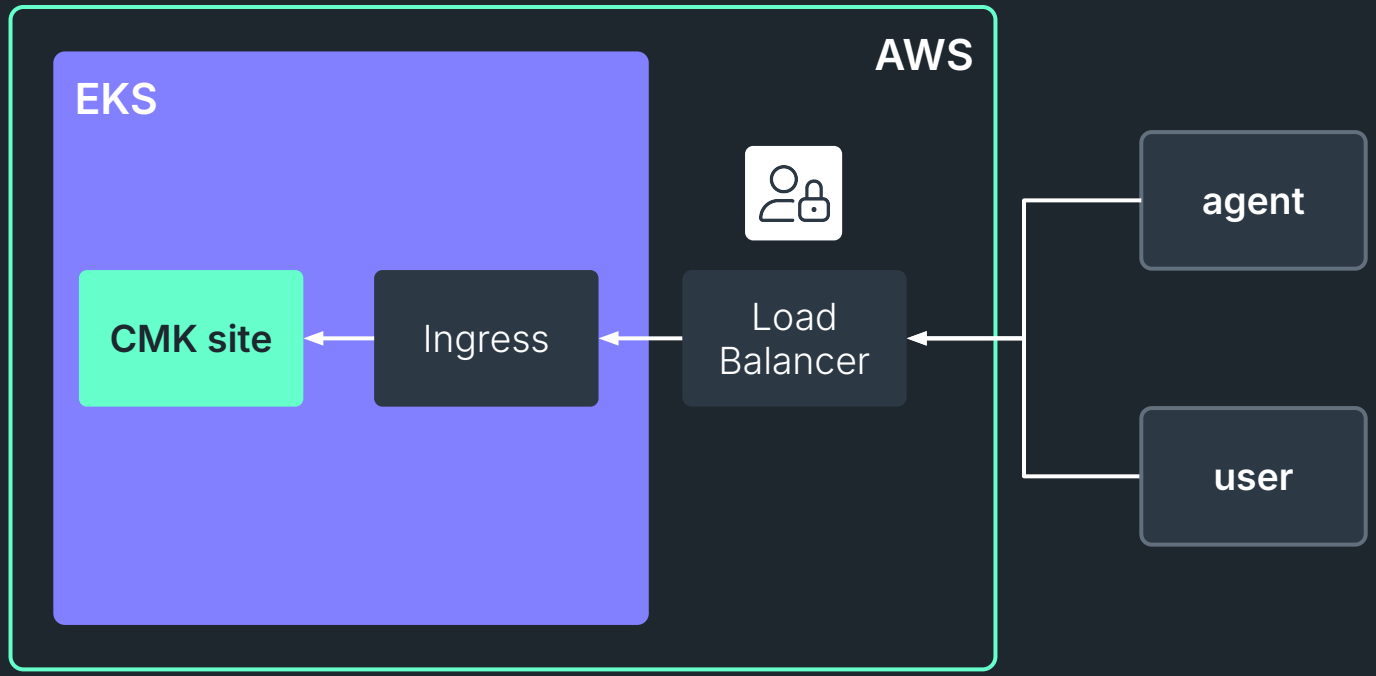
- In IPv6 only cluster, IPv4 support is implemented via a virtual interface
- Kubernetes Network Policies are applied only on PRIMARY interfaces
- A virtual interface is NOT a primary interface
- You cannot block IPv4 targets

Solution:

- Understand how network policies work at the node level
- Implement your own filtering for IPv4 until enough people cry out so that AWS can support dual-stack configurations the correct way

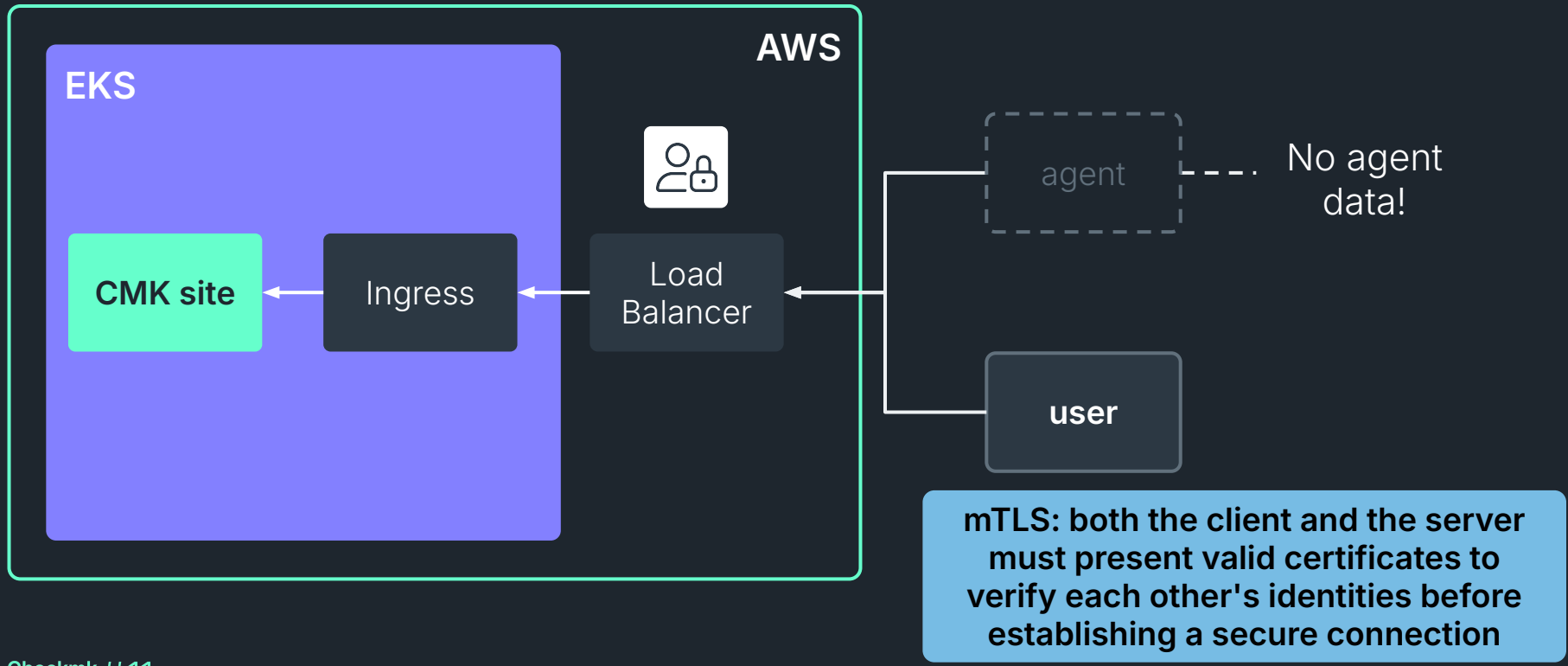
Remember the Ingress

Generic pattern for traffic from user to application



Remember the Ingress

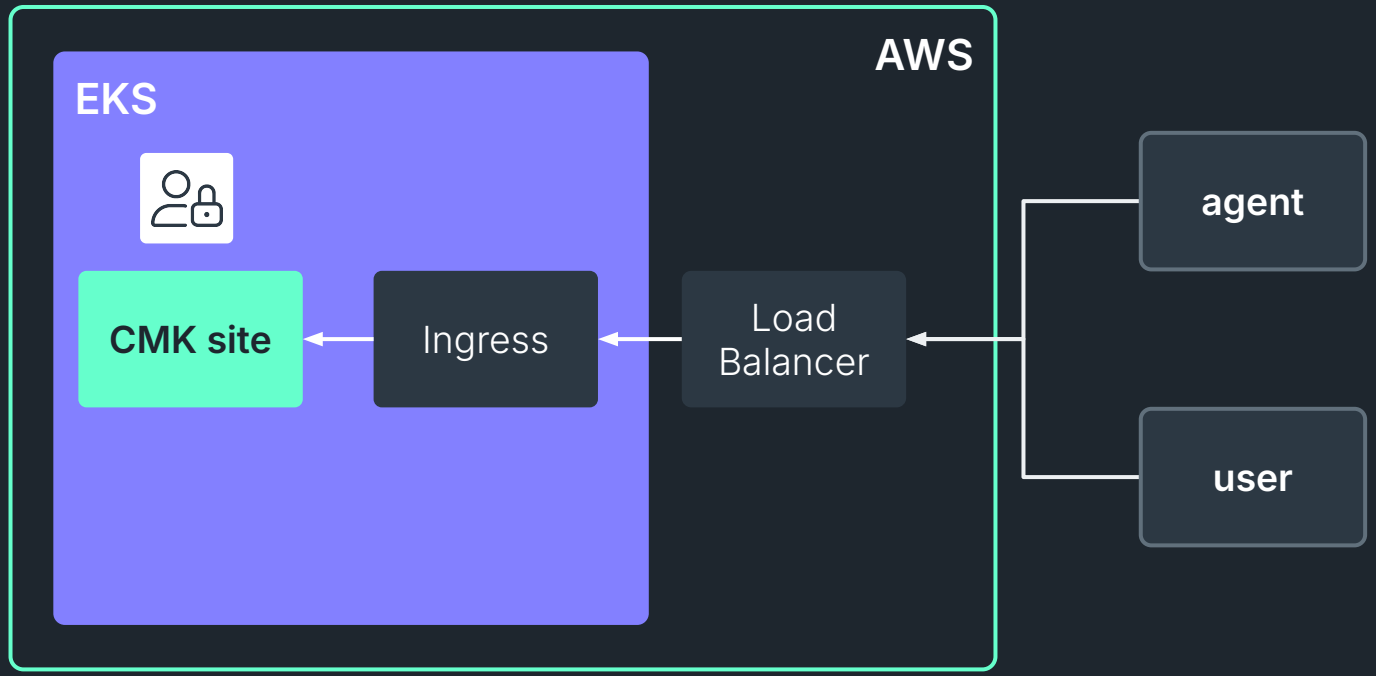
Where's the agent data?



mTLS: both the client and the server must present valid certificates to verify each other's identities before establishing a secure connection

Remember the Ingress

Approach 1 - passthrough all traffic



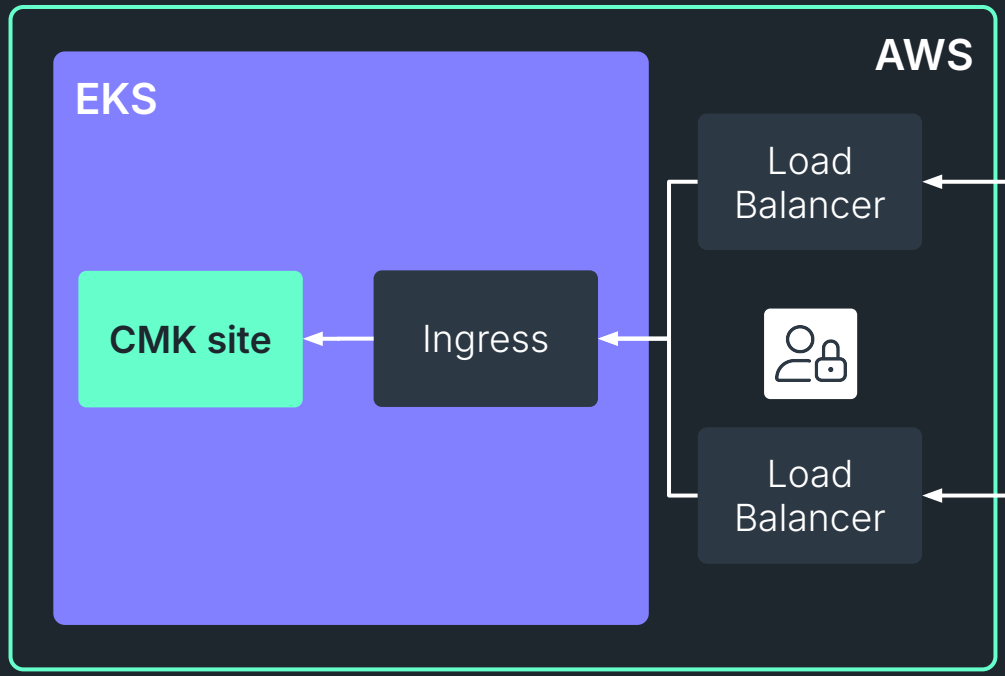
Nope!

→ Third party rate limiting when you request a new certificate

→ We do not want a wildcard certificate to be distributed in client sites

Remember the Ingress

Approach 2 - separate LB for the agent



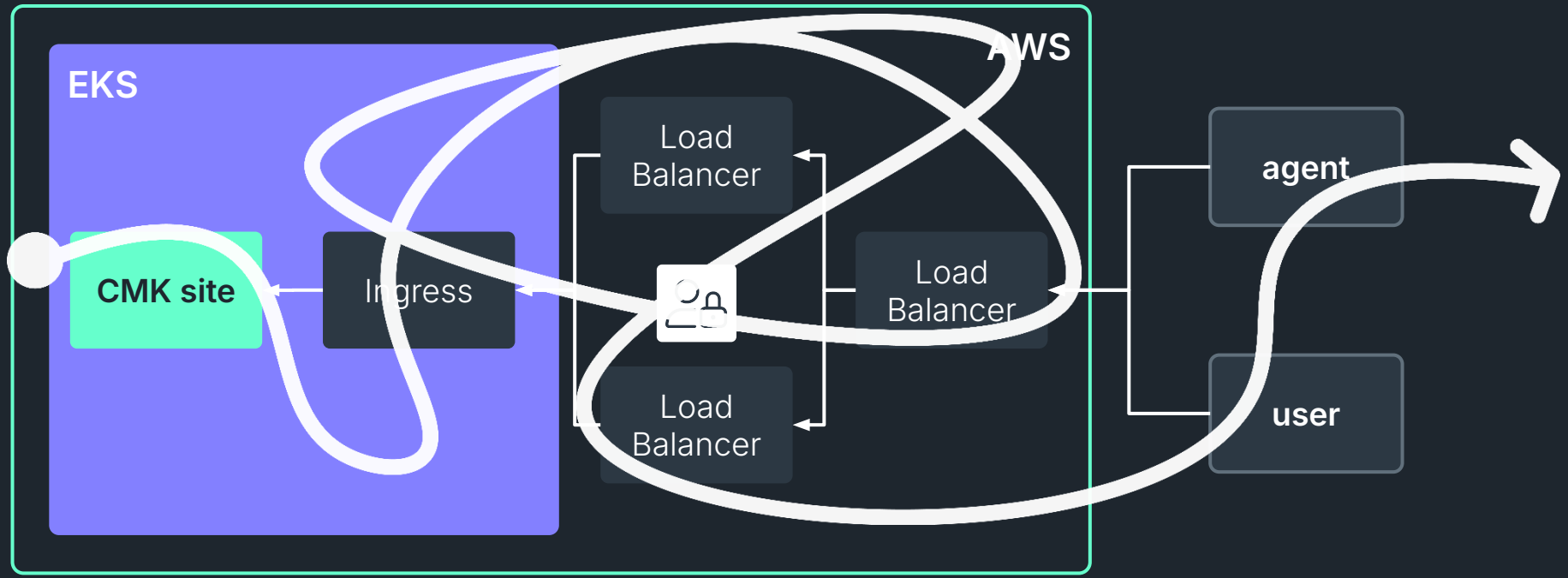
No go!

→ Trickier to bake agents

→ Users might get confused by the two different names if they want to bake their own agent

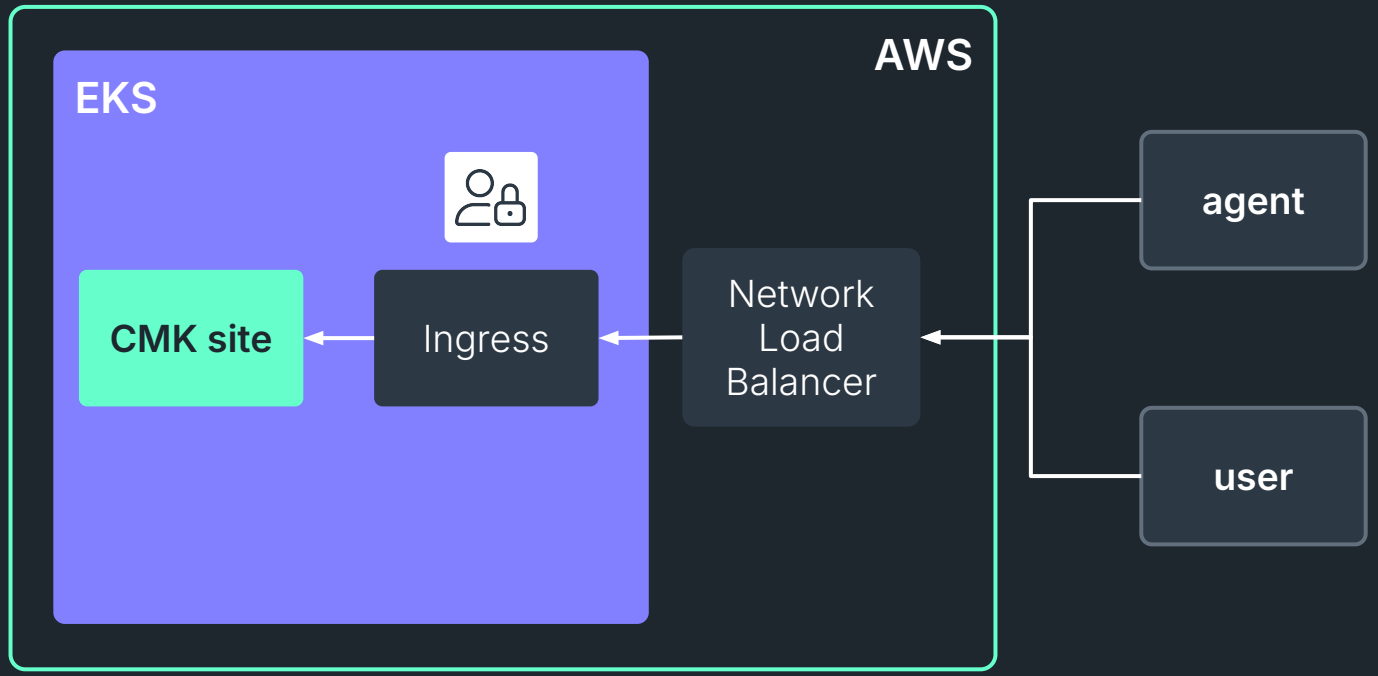
Remember the Ingress

Approach 3 - an LB in front of the other two LBs



Agent communication solution

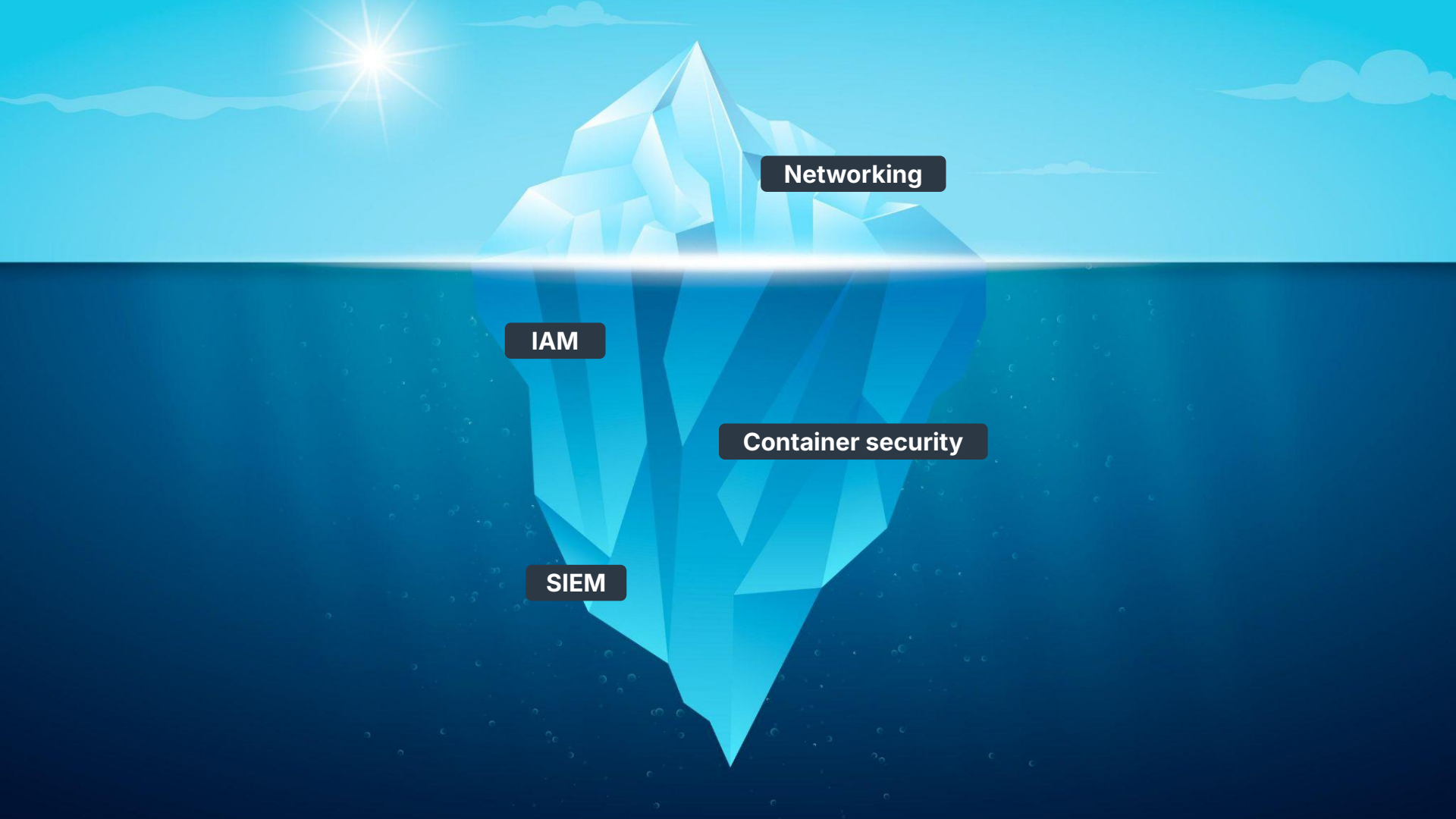
Simple, elegant, best of both worlds



Solution
Discriminate traffic at the **Ingress** level

→ Agent traffic destined for port 8000 is not touched

→ Everything else flows as before


Networking
IAM
Container security
SIEM

How we built our SaaS platform

Easy if you have the right abstractions

Containers

Kubernetes

Hyper-scaler

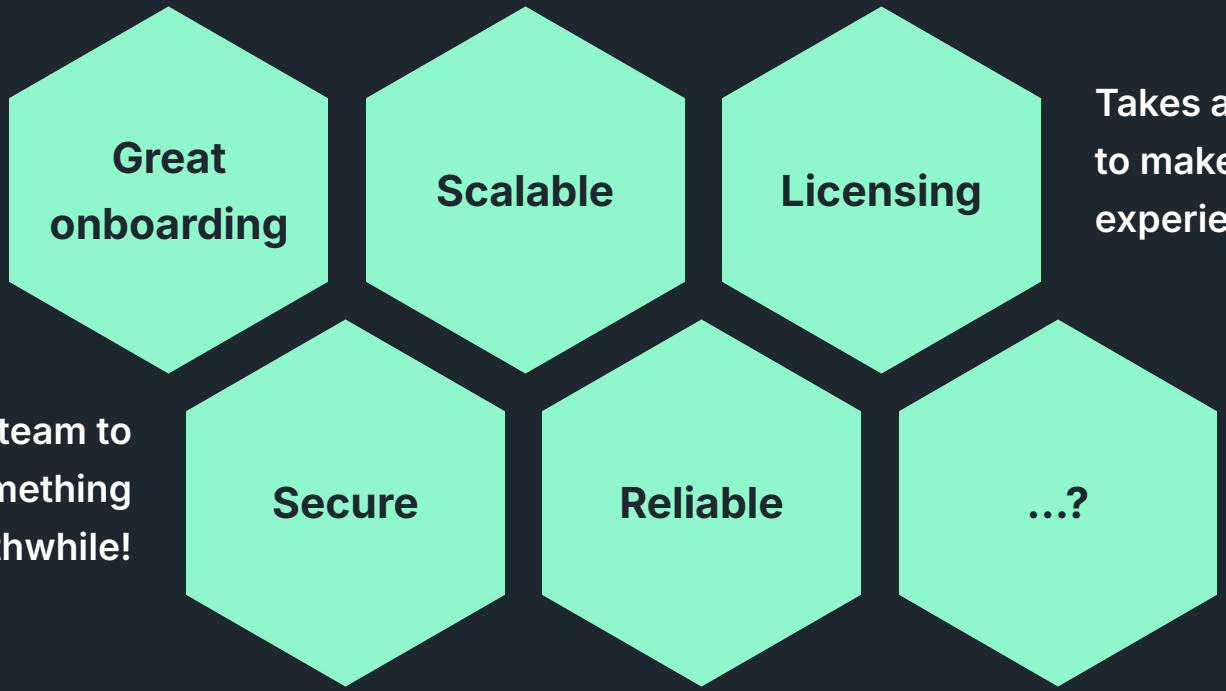
Imagine if ...

... a Control Plane service fails → Redundancy.
Multiple copies of each service.

... an individual node fails → Kubernetes auto-recovery.
~5 min restore time.

... a data center (Availability Zone) fails → Automation. Global backups.
~20 min restore time.

Exciting stuff



Takes a lot of pieces to make a great SaaS experience.

It takes a great team to build something worthwhile!


11