

Invisible Guardrails

Enabling Developer Velocity with a Secure Platform



James Sigurðarson

Staff Infrastructure Engineer



Vignir Hafsteinsson

Senior Engineering Manager

Our mission

To help humanity thrive by enabling the world's teams to work together effortlessly.

The world's leading companies trust Asana

Health



McKesson

Johnson & Johnson

Genentech



Retail

T-Mobile

Kohl's



Media

HEARST
media solutions



Clear Channel

BUSINESS
INSIDER

Financial Services

affirm

OCBC Bank

nerdwallet



Tech

amazon



PHILIPS

OpenAI

The only work management platform that scales to **200k+** users in a single deployment

Agenda

Background

What trade-offs were we making with security?

Implementation

How did we reject those trade-offs?

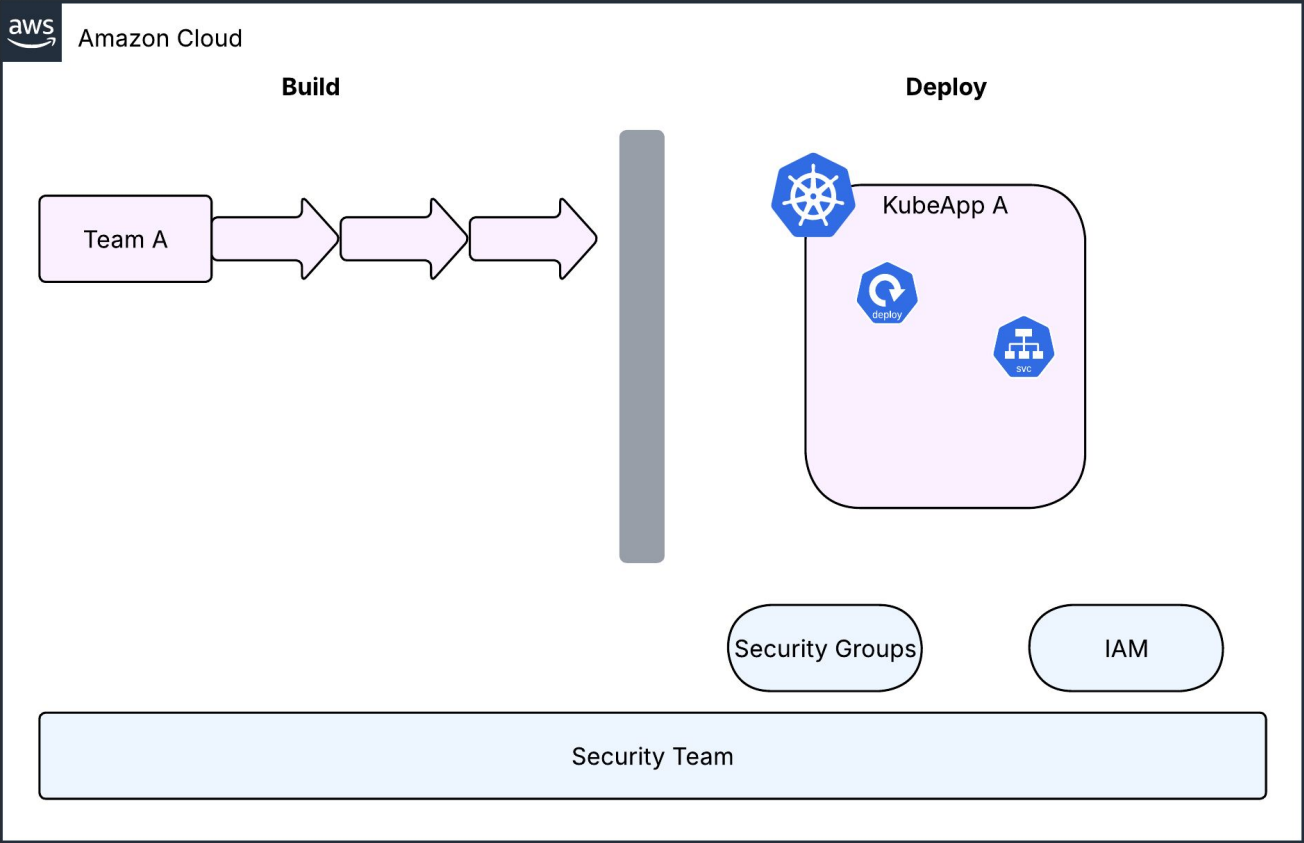
Guardrails

How did we prioritize security **and** velocity?

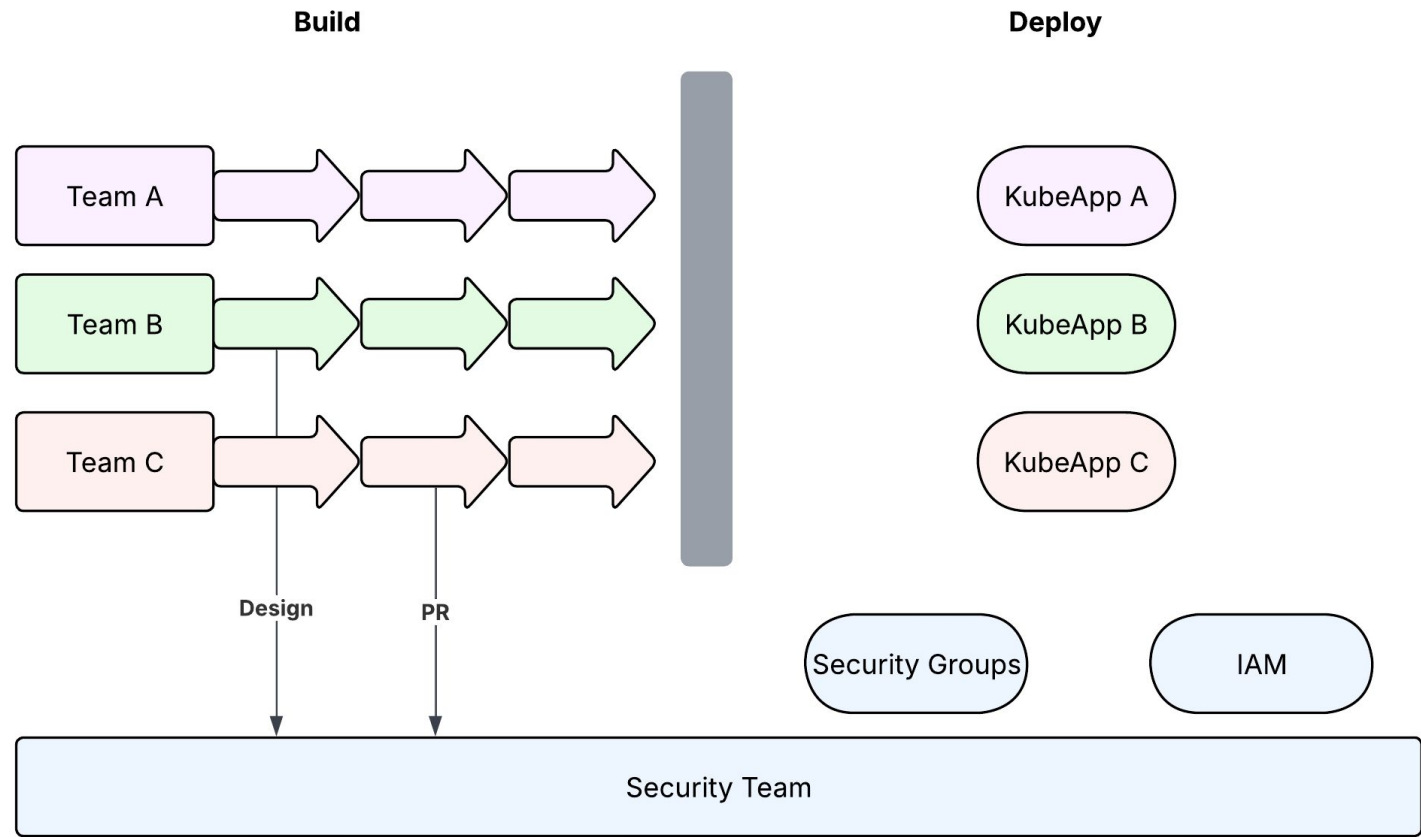
Lessons Learned

What did we learn? What mistakes were made?

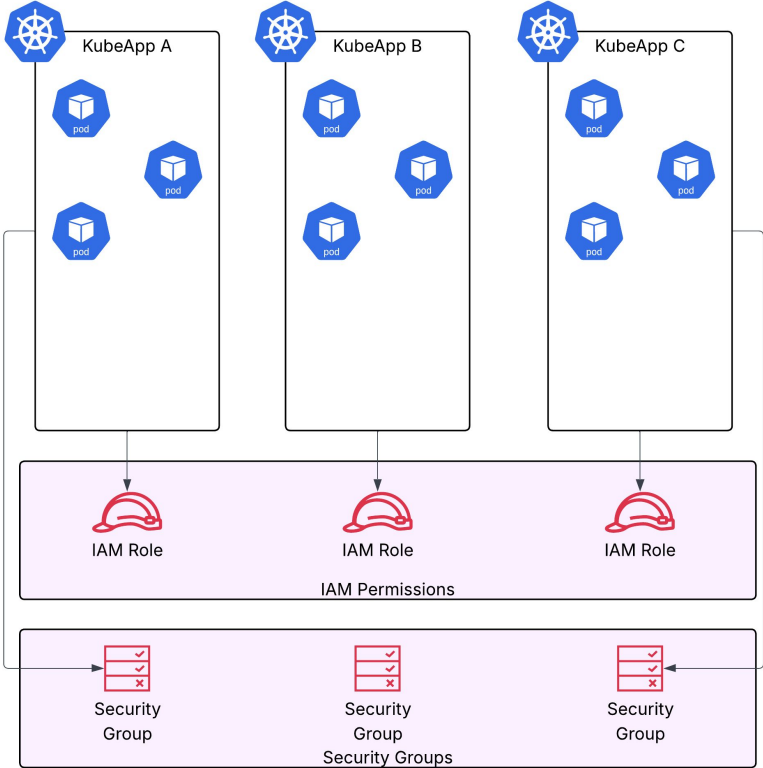
Background



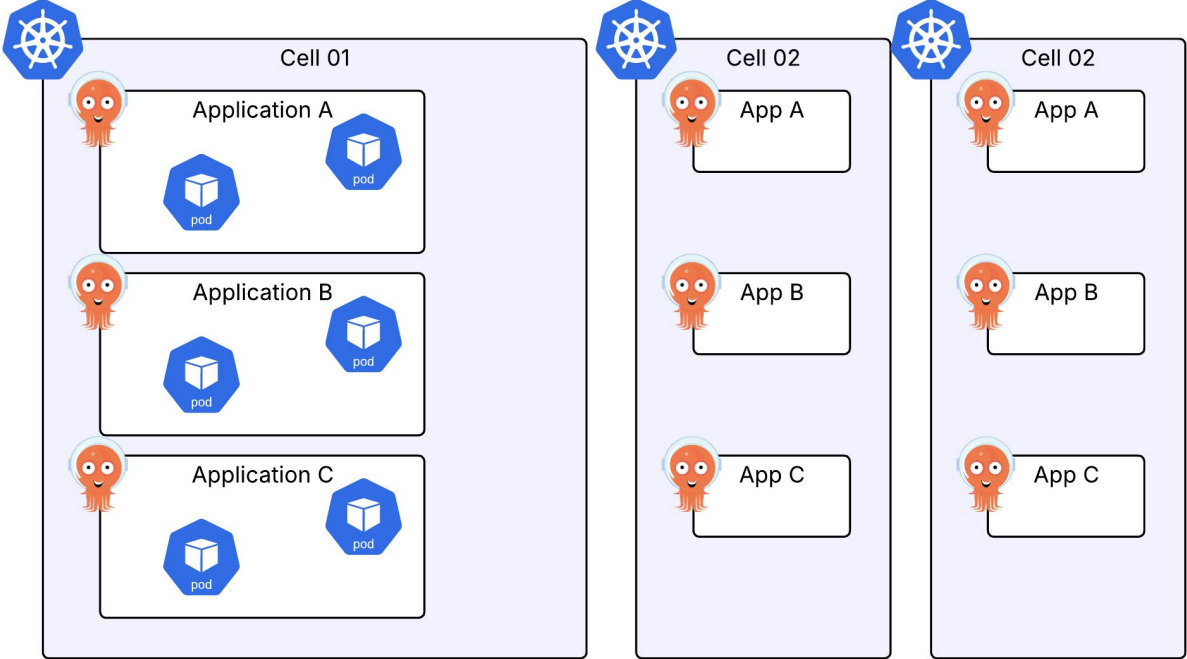
Background



Background



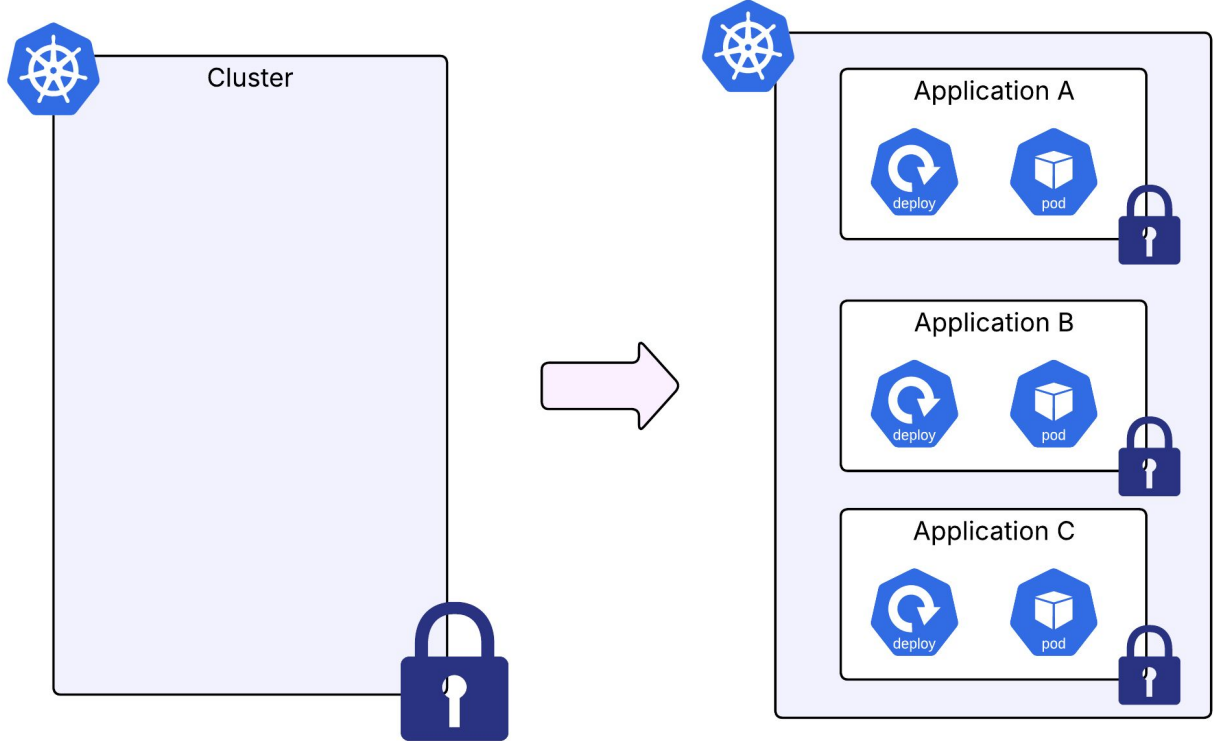
Background: Cells



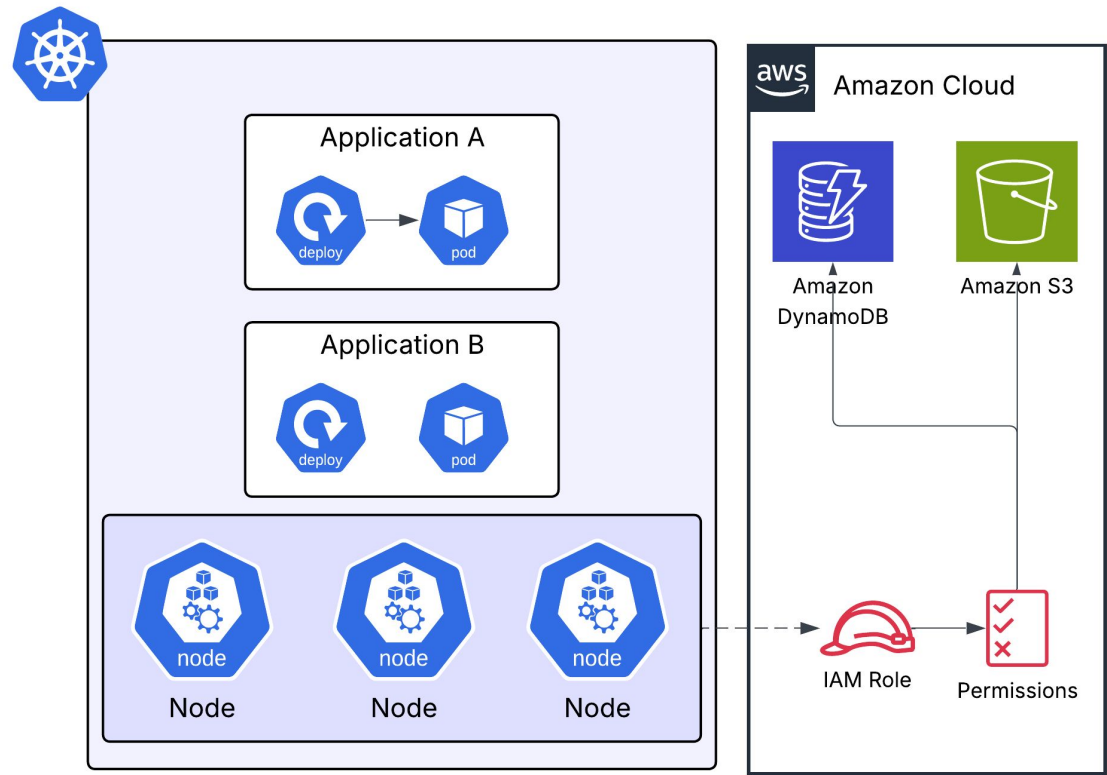
Problem

- **Velocity** was being sacrificed for **security**, causing friction
- Security Infrastructure had become a **Single-Point-of-Failure**
- Infrastructure teams were not taking **full responsibility** for security
- Our infrastructure **needed to scale**, causing **new** security concerns

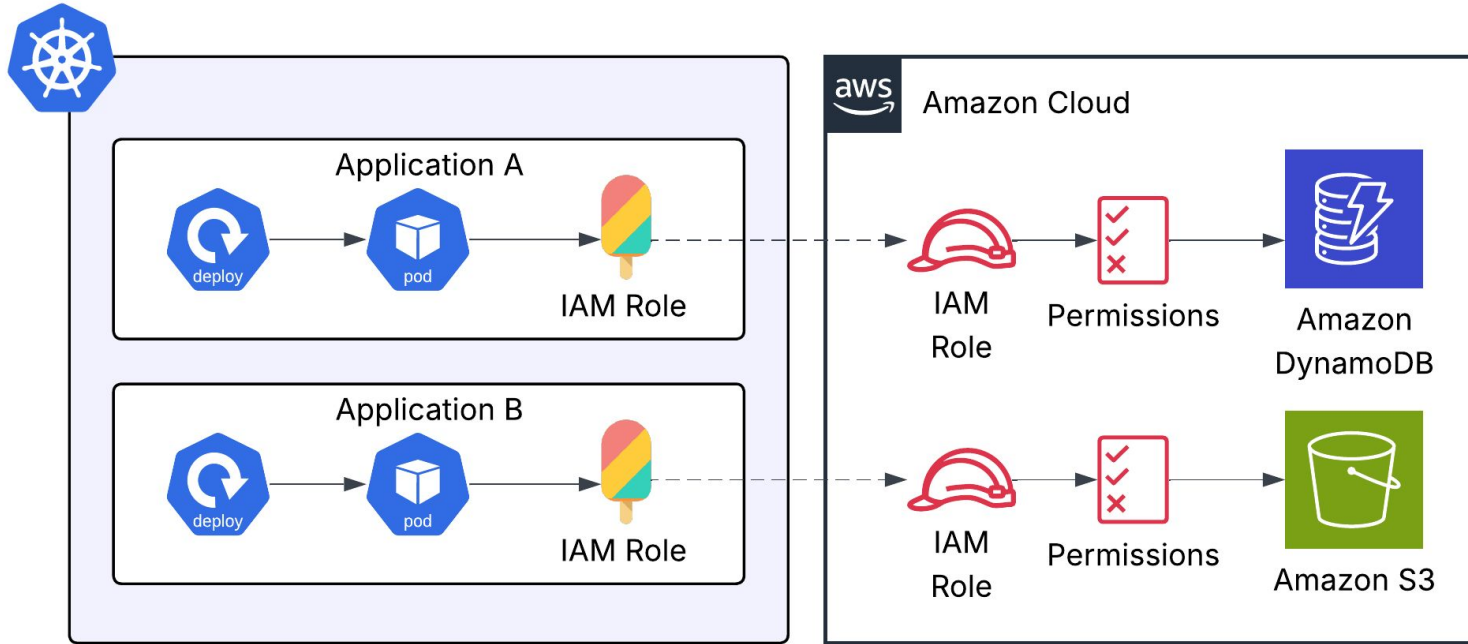
Shrinking the Permission Unit



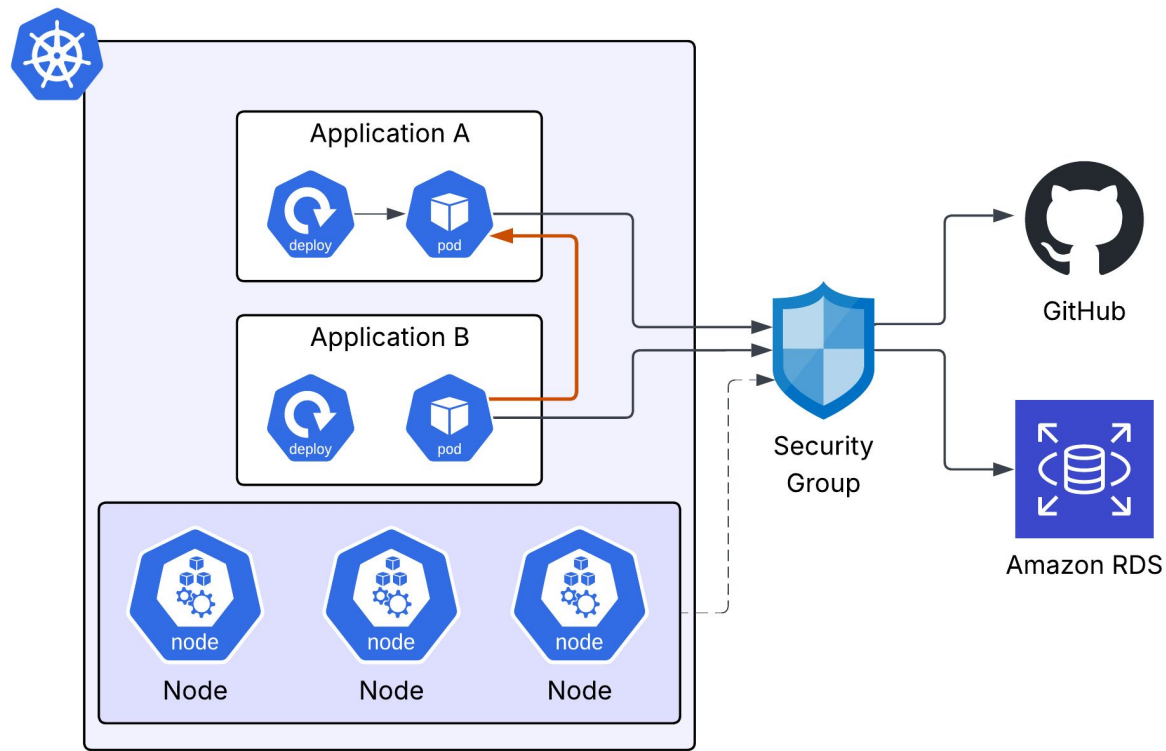
IAM access



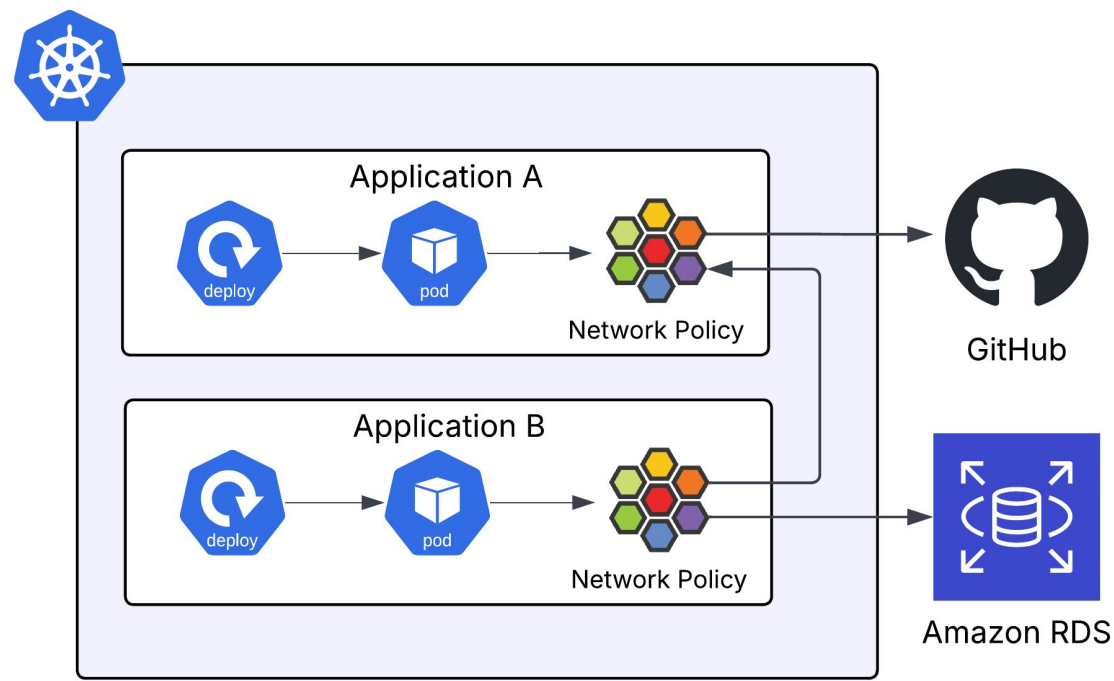
IAM access



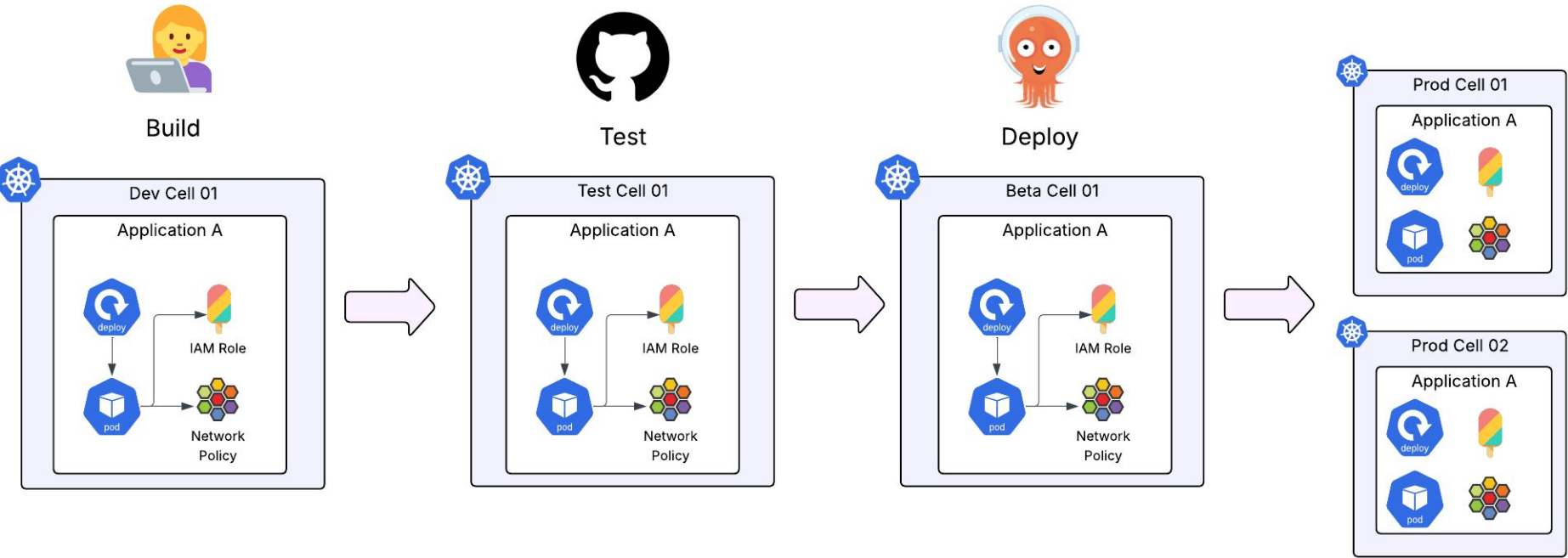
Network Access Control



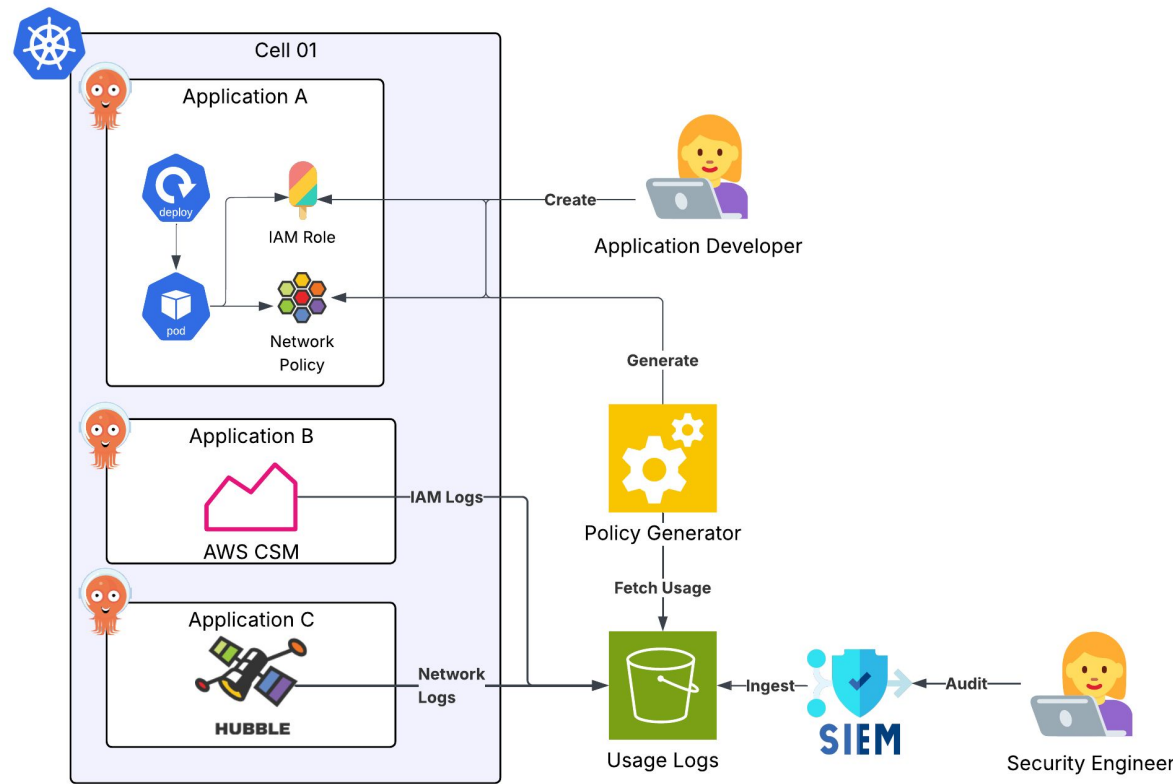
Network Access Control



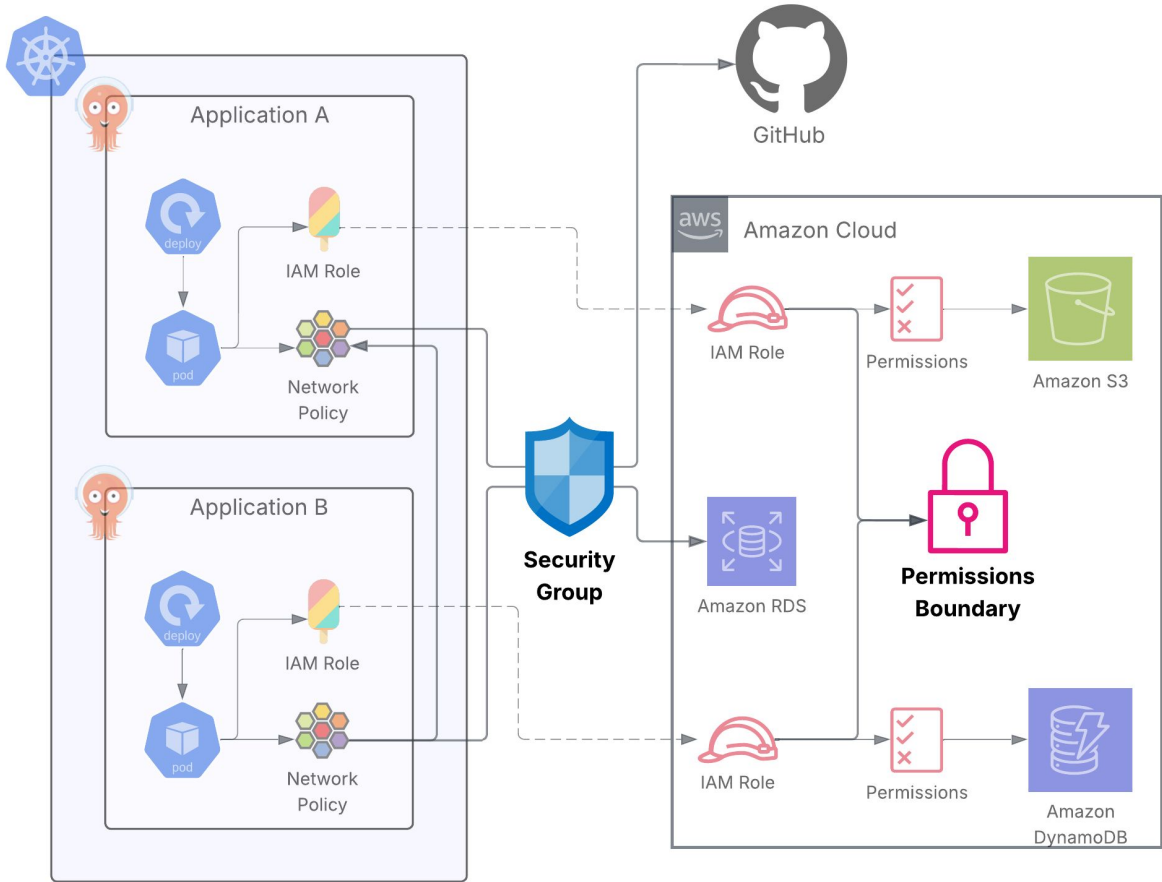
The Development Workflow



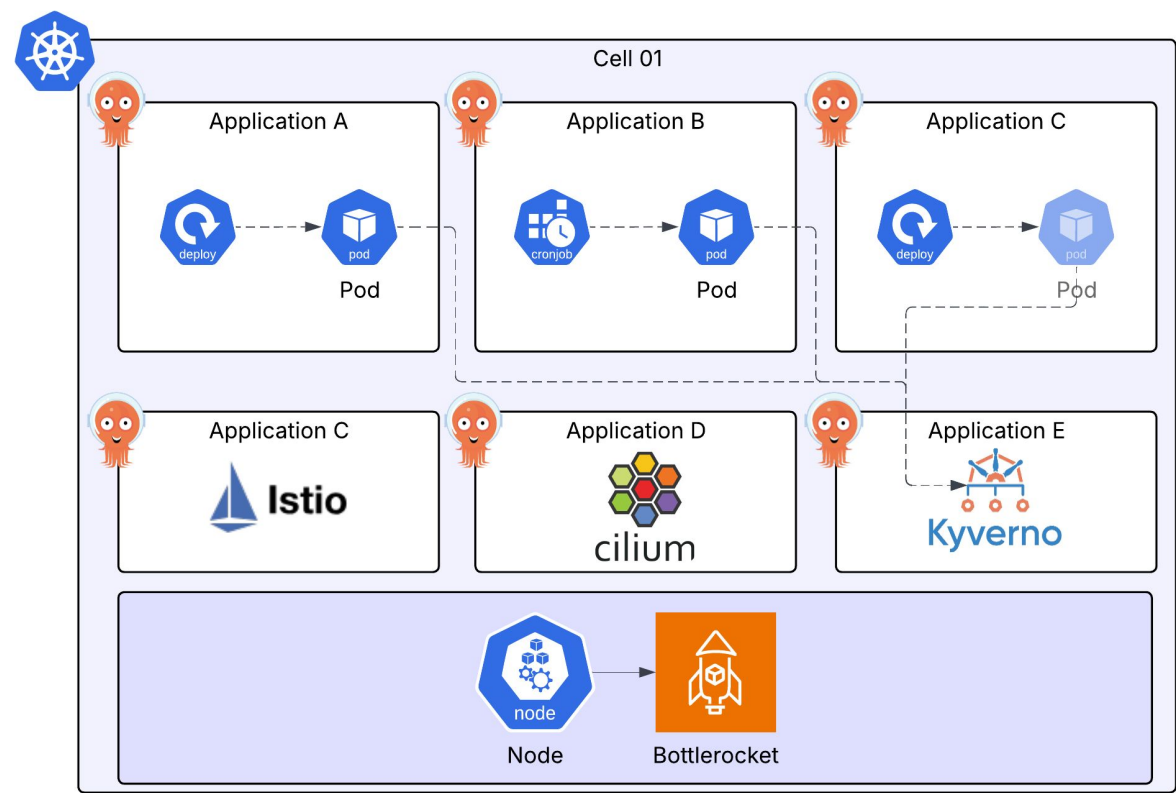
Automation



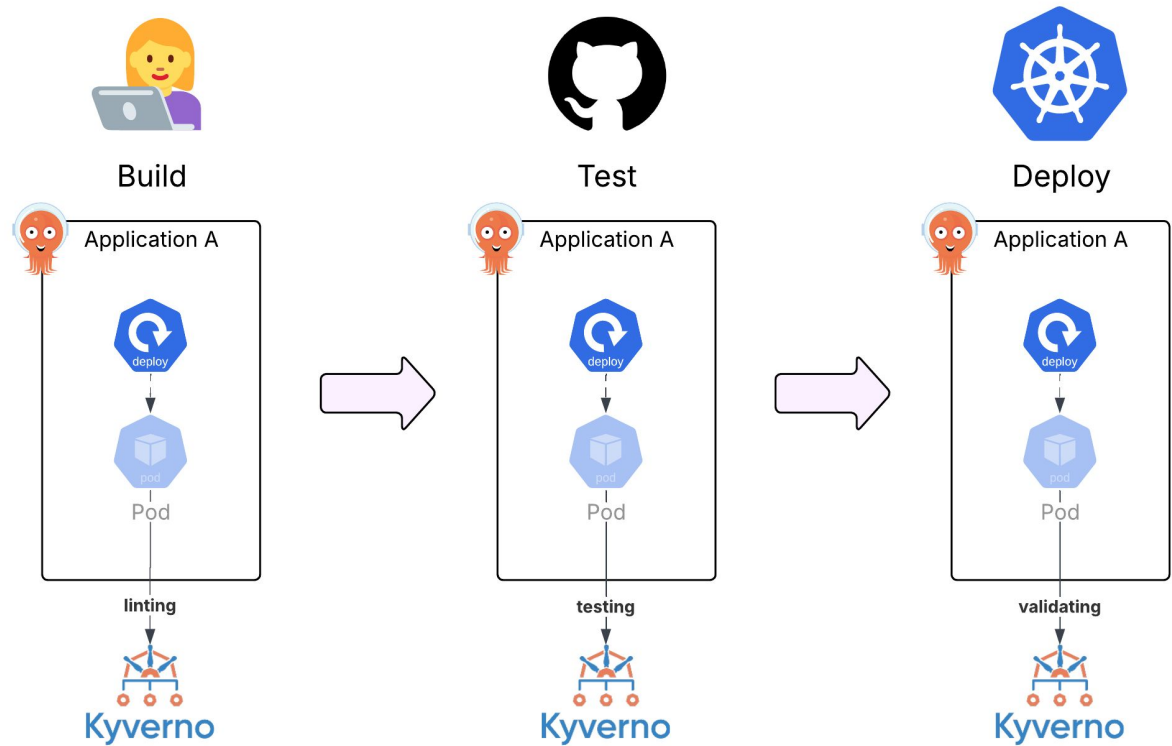
Guardrails



Secure Platform



Shifting Left



Lessons Learned

Silent Enforcement

Invisible = Confusing

If a developer doesn't know *why* a deploy failed, "invisible" just feels like "broken."

Observability is Mandatory

Knowledge Debt

Abstraction ≠ Simplicity

Removing manual friction also removes the "learning by doing" that builds infrastructure literacy.

Make the “Magic” Transparent

One Team to All Teams

Security is everyone's responsibility

While the *process* is faster, the *cognitive load* on developers has increased.

Scale through Education

80/20

Exceptions are the hardest part of the rule

90% of apps fit the guardrails perfectly. The final 10% required new guardrails.

Build for Extensibility

Guardrails should be invisible during success,
but **loud** and **clear** during failure

Summary

Security as Invisible Guardrails

Strive for making the easiest path the most secure path

High trust requires high automation

Build platforms that prevent mistakes

Shift Left on Security

Discover security issues as soon as possible

Q&A

Thank you!



<https://sched.co/2CW34>