K8s and TF ‘in the field’

How CircleCI’s Field Engineering team accelerates enterprise POCs with IaC

‘Moo’ Olaniyan is a Sr. Enterprise Field Engineer at CircleCI with a backgrounds in air-gapped device automation and deployment.

linkedin.com/in/moo-olaniyan/

‘Eddie’ Webbinaro is Global Head of Field Engineering at CircleCI with a background in SDLC platforms and enterprise transformation.

linkedin.com/in/eddiewebbinaro/

>Agenda

Brief Introduction

What are we talking about? (preview)

What was the problem?

Why IaC and K8s/TF were the solution (for us)

What solution looks like (C.E.R.A.)

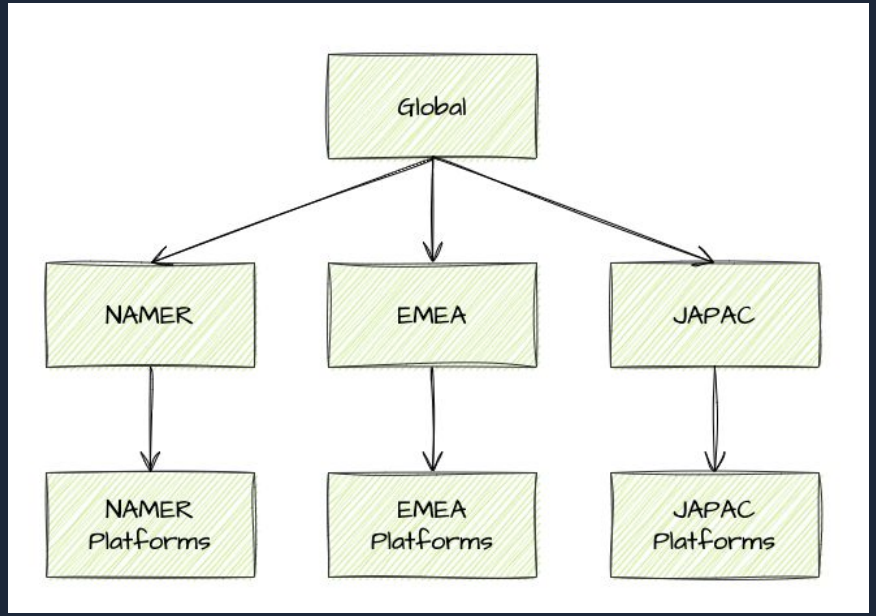
Impact and Summary

Solution Preview

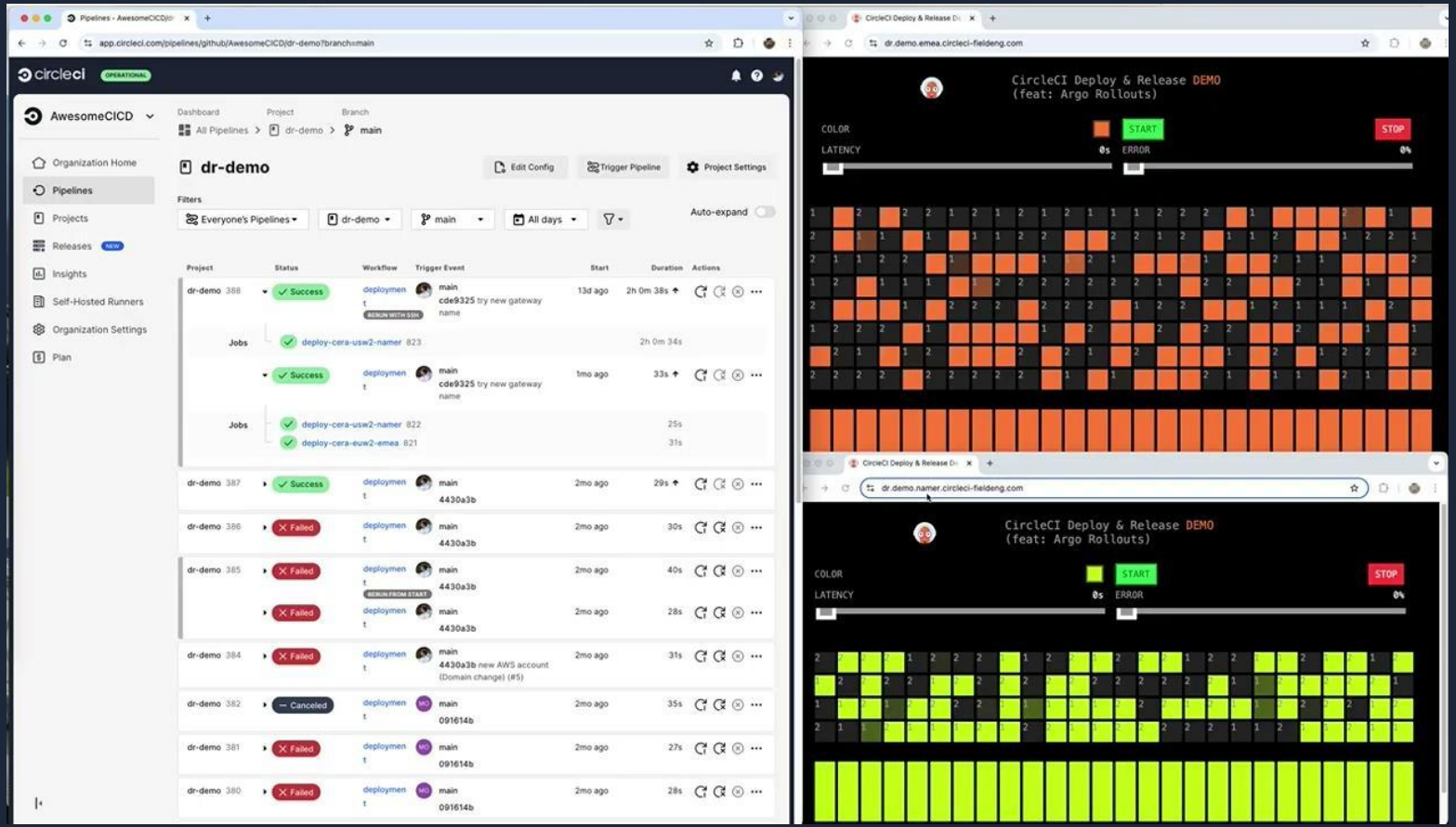
What are we talking about anyway?

CERA Preview Demo

- **Multi-Region EKS Deployment:** Trigger a pipeline to roll out a new release across multiple regions.
- **CircleCI Releases Dashboard:** Walk through steps of a progressive rollout.
- **Auto-Rollback:** Demonstrate automatic rollback in case of deployment errors.
- **No Credentials Needed:** Deploy without manual intervention or credentials.
- **Enterprise-Like Environment:** Mimics the setup an average developer would use in a real-world enterprise setting.



Demo Time

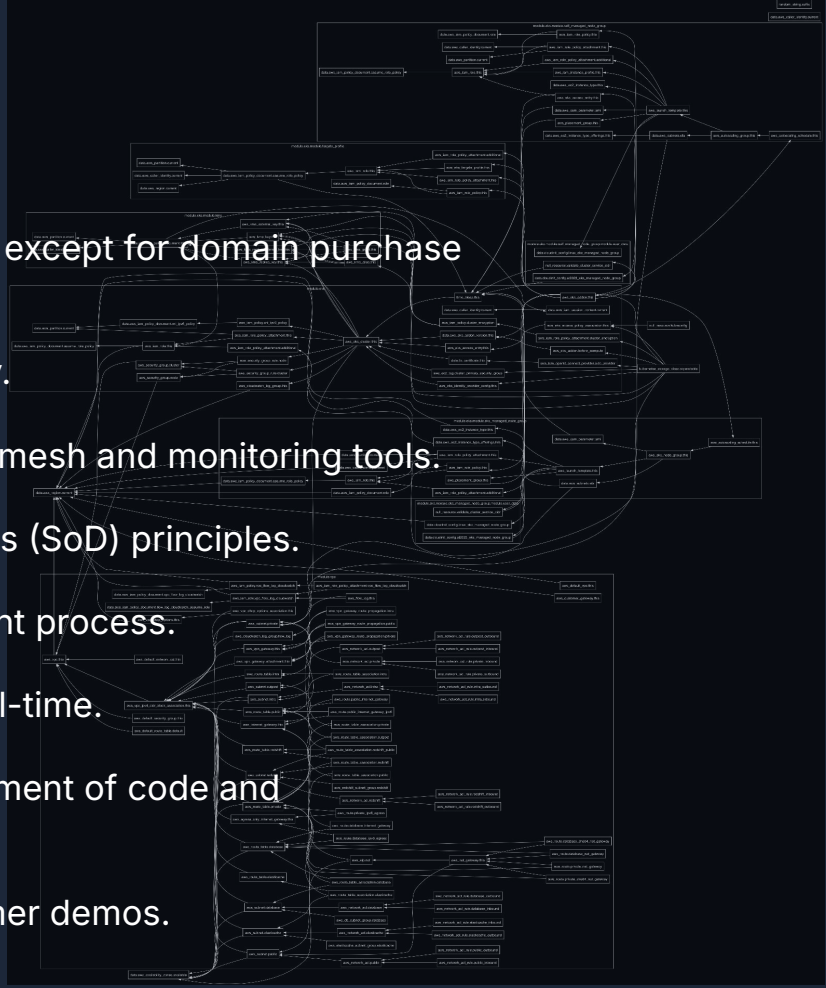


https://drive.google.com/file/d/1OHaTIAJCH3Kap8xqK8bz3iBZVNY4GZb7/view?usp=drive_link

Enterprise Reference Environment

Our IaC Solution..

- **100% Infrastructure as Code (IaC):** Fully automated, except for domain purchase and IAM setup.
- **Runs on EKS:** Kubernetes for flexibility and scalability.
- **Istio, Kiali, Grafana, Prometheus:** Integrated service mesh and monitoring tools.
- **Layered Access Control:** Follows Separation of Duties (SoD) principles.
- **Passwordless Deploys:** Simplified, secure deployment process.
- **Visual Release Tracking:** Monitor deployments in real-time.
- **Artifact Repository & Secrets Vault:** Secure management of code and credentials.
- **Turn-key Demo Apps:** Ready-to-use apps for customer demos.



Problem Statement

After the solution?

The problem..

Field Engineering Demo Debt

Inconsistent Impact

Varied Technology: Demos used different tech stacks, leading to inconsistencies.

Varied Stories: Demo narratives changed based on setup, losing focus.

Lack of Enterprise Reflection: Demos didn't accurately mirror enterprise environments.

No Separation of Duties (SoD): Lacked proper access control layers.

Lack of Controls: Inconsistent security and governance across demos.

High Effort

Many Re-Wheeled Wheels: Repeated effort creating custom demos for each customer.

Outdated Assets: Demos often used outdated or irrelevant components.

Prep and Cleanup Time: Significant time required to prepare and clean up demos.

Customer opportunity

Lacked Customer Relevance: Customers couldn't "see themselves" in the demos.

Need for Hands-On Access: Customers needed their own environment to test and explore.

Extended Sales Cycle: Longer sales process due to lack of tailored, interactive demos.

Two Goals

Goal #1

Reduce Demo Effort: Create a highly available, reusable demo environment to minimize manual setup and prep time.

Goal #2

Elevate Customer Conversations: Build a dynamic, scalable platform that mirrors real enterprise infrastructure, providing value to all teams—development, security, and operations.

The Work

Step 1: Technology Chats

Tech Selection First: Focused on choosing the right tools and technologies for enterprise scalability

Global Collaboration: Senior teams across NA, EMEA, and JAPAC, coordinated through late nights and early mornings to align across time zones

Step 2: Vision Chats

Defining the Impact: Management and Field Engineers focused on storytelling and delivering real customer value.

Reference Implementation: Initially, the goal was to create a trusted reference architecture that customers could see as a proxy for their own environments.

Ongoing Work

Meanwhile, the team continued generating pipeline opportunities with marketing and business development, winning deals using existing methods.

Solution Proposal

Now we're cooking

Design Choices

Build vs Buy

Considered External Tools: Explored demo automation tools available in the market.

Flexibility Limitation: Most tools didn’t provide the flexibility or customization we needed to mirror complex enterprise environments.

Decision to Build: Ultimately, we chose to build our own modular, Infrastructure as Code solution to ensure full control over customer demos.

Multi-tenant vs Each-FE

Multi-Tenant Consideration: Explored the idea of a shared, multi-tenant environment for cost efficiency.

Individual Environments: Considered providing each Field Engineer with an isolated environment to avoid conflicts between demos.

Final Decision: Opted for a multi-tenant setup to align with enterprise teams' shared infrastructure models while keeping deployment consistent and scalable.

IaC Tooling and Design

Terraform for Infrastructure: Chose Terraform as the foundation for managing infrastructure across multiple regions.

Kubernetes on EKS: Used Kubernetes on EKS to ensure scalability and flexibility in deploying containerized applications.

Additional Tools: Integrated Istio for service mesh, and Grafana, Kiali, and Prometheus for monitoring and visibility.

Modular Approach: Built the environment to be modular and reusable, ensuring components can be easily swapped or scaled based on demo needs.

Modular and Layered

The agreed design would be multi-tiered to solve technical challenges around initialization and secrets management, and would also give us a DRY'er code base as we scaled to multi-region HA model.

Global Layer

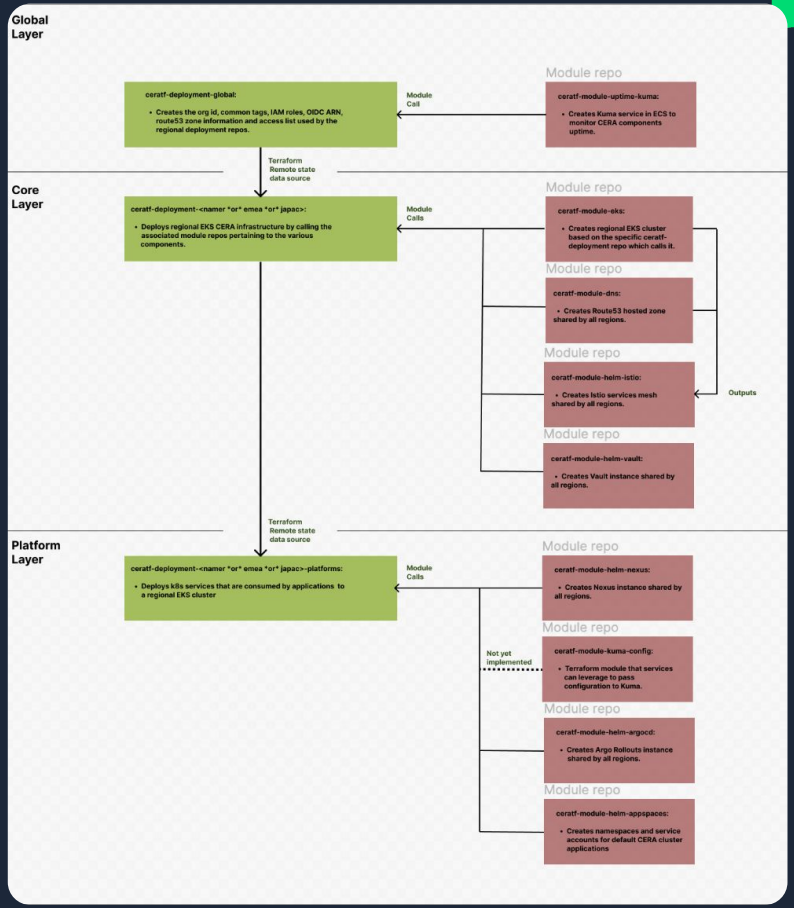
- Global DNS and IAM Policies

Region Core

- EKS cluster w/Networking routing and certs
- Cluster monitoring tools

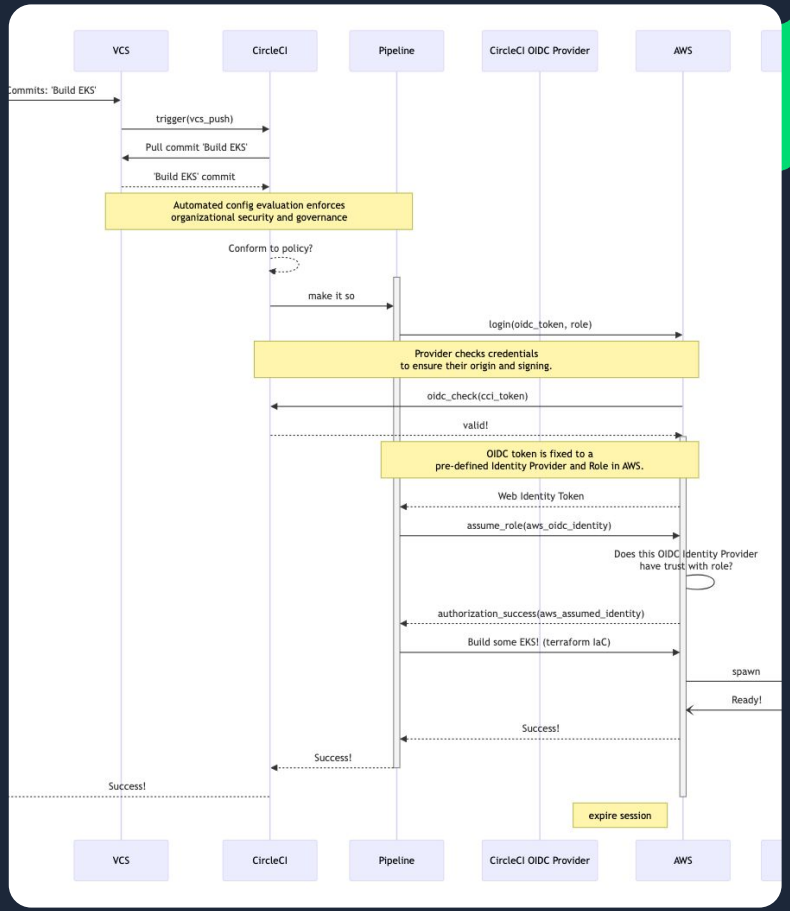
Region Platform (PaaS)

- App namespaces
- Vault, Nexus, and Rollouts



Keeping Secrets

- SSO Allows initial AWS Operator access to setup initial IAM profile with OIDC assumption.
- Pipeline runs with OIDC assumption (web identity) and provisions the entire stack.
- Vault Root key is placed in AWS KMS for auto-unseal.
- Operators have unique limited role outside pipeline access.
- Dev pipelines used OIDC connection with predefined IaC Vault policies to load any additional credentials/tokens.



Application and Evolution

It’s alive!

Adoption & Rollout

Slow Initial Adoption: Early adoption was slow due to complexity and lack of polish.

Customer Feedback Loop: Iterated based on customer feedback, improving the environment's usability and relevance.

Widespread Use: Over time, the environment became widely adopted not only by Field Engineering but also by other teams for demos and trials.

Consistent Platform: The environment became the go-to for delivering consistent, high-quality demos that mirrored customer environments.

Lightbulb moment

“How do we get access to your sandbox environment?”

Ah-ha!

Implementable reference architecture!

Removed Hardcoded Elements: Adapted the environment by eliminating hardcoded IDs and paths, making it flexible for external use

Fully Documented: Created detailed documentation for provisioning to ensure ease of use for customers and internal teams

Scalable & Portable: Refined the architecture to be portable and scalable, deployable across multiple environments

Customer-Ready: Built a robust, adaptable solution that could be handed over to customers for sandbox trials or POCs.

Making Deployment Portable

- **Parameterized Setup:** No more hardcoded elements, fully customizable for different environments.
- **Scalable Across Regions:** Easily adapted for multi-region deployments without complex changes.
- **Customer-Specific Configurations:** Tailored for unique customer needs while maintaining core architecture.
- **Faster Setup:** Streamlined provisioning process saves time across all deployments

Recap

- **Consistent Demos:** Reliable, repeatable environments
- **Customer-Reflective:** Mirrors enterprise infrastructure
- **Improved Security:** Passwordless, IAM, Vault integration
- **Portability:** Easily deployed across regions
- **Streamlined Operations:** Faster, more efficient demos

Thank you.