Integrating Fargate with the CDK

David Tucker
TECHNICAL ARCHITECT & CTO CONSULTANT
@_davidtucker_   davidtucker.net

Globomantics

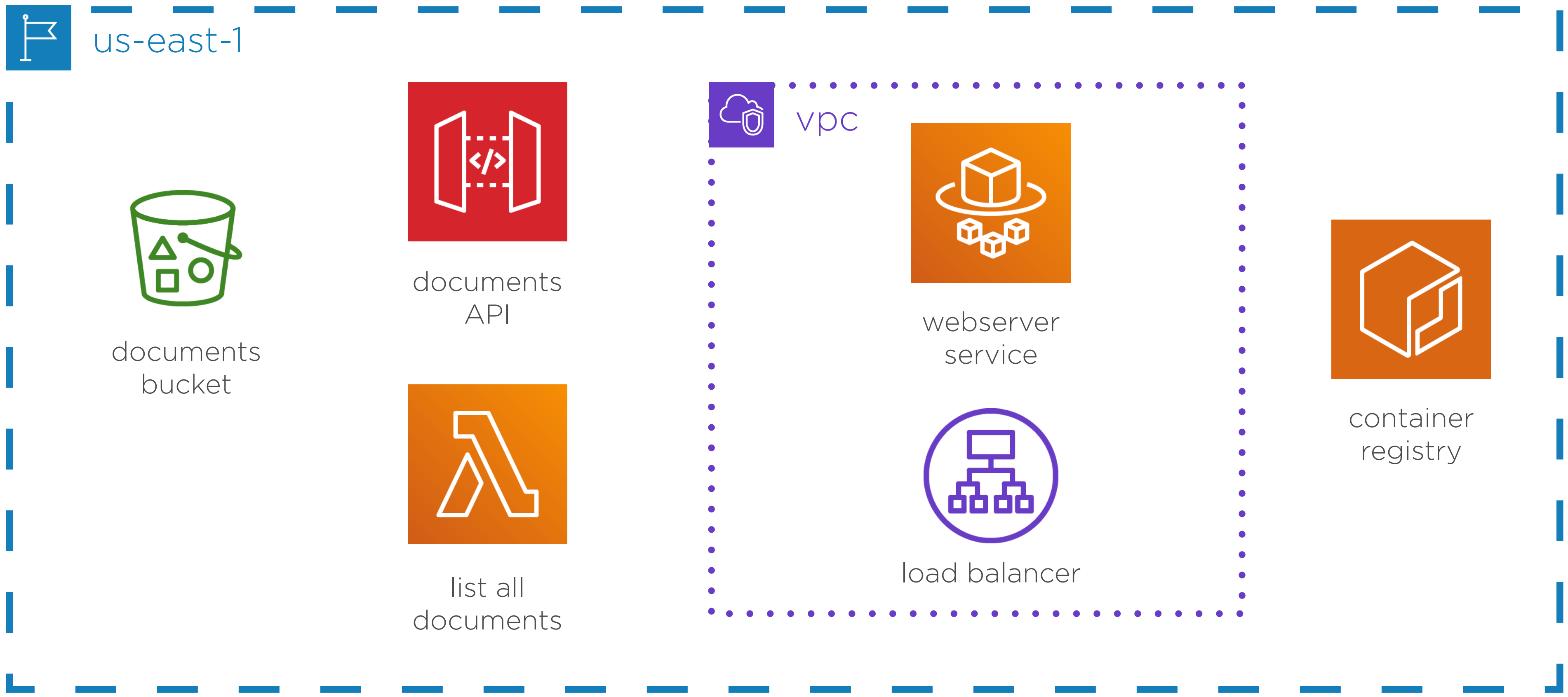
Josh
Cloud Architect

Successfully built the API tier for the proof of concept

Needs to be able to:

- Create a custom container for a Node.js express web server
- Deploy the container on AWS
- Implement a load balancer for the service

Sample Architecture



Overview

Reviewing container services on AWS

Creating a TypeScript and Express Docker container

Deploying our Docker container using AWS Fargate

Verifying our Fargate deployment

Discussing next steps with the CDK

Container Services on AWS

Container Management Services for AWS

Amazon ECS

Provides a container orchestration service on AWS

AWS Fargate

Enables containerized applications without managing servers

Amazon ECS for Kubernetes (EKS)

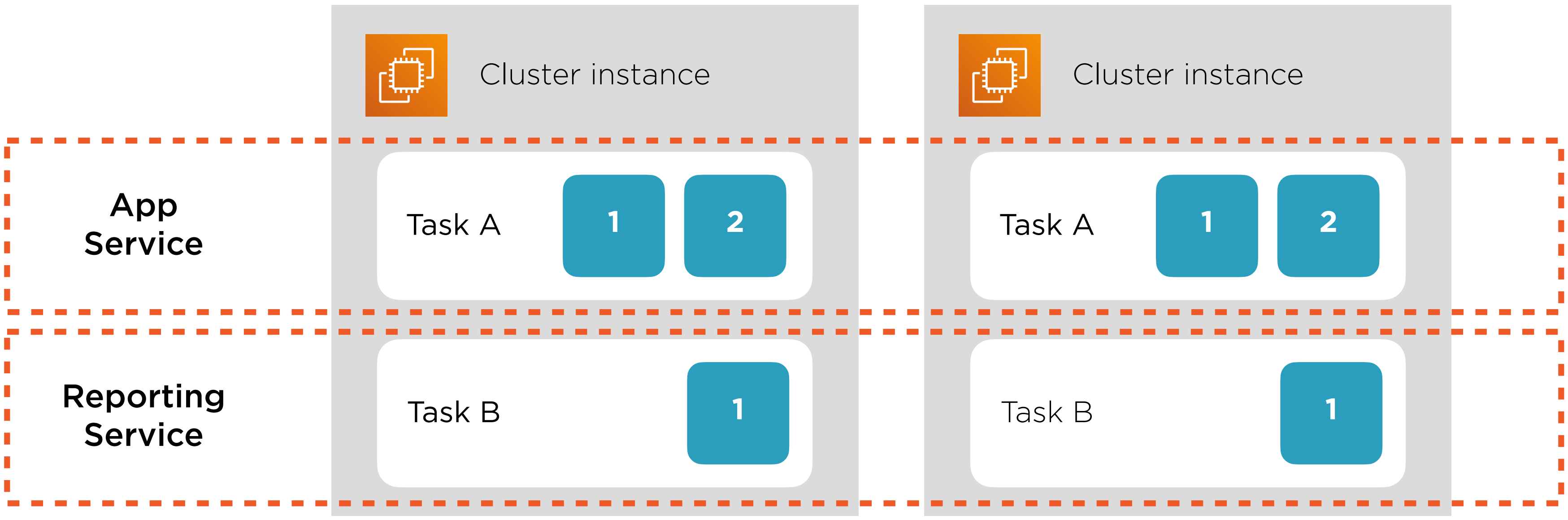
Manages Kubernetes applications in AWS

Container Registry for AWS

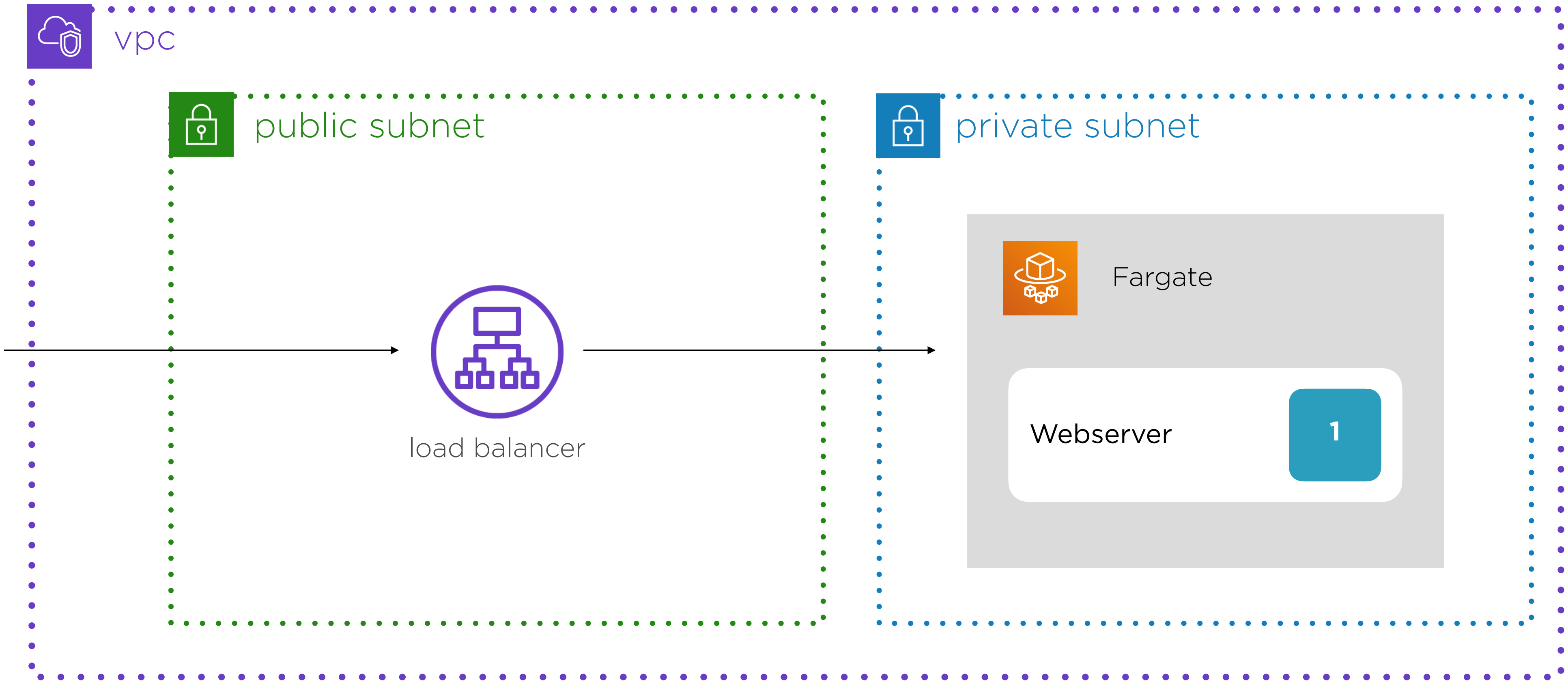
AWS ECR

Container registry for use on AWS

Amazon ECS Terms



Fargate Deployment



Creating a Docker Container for the CDK

Demo

Creating a directory structure for Docker containers within our project

Implementing a TypeScript express webserver

Configuring a Dockerfile

Testing our container locally

Deploying a Container with the CDK

Demo

Creating a new construct for our web server

Configuring a Docker image asset to store in the ECR

Deploying our container as a service on Fargate

Verifying our deployment

Next Steps with the CDK

Summary

Created a CDK project using the CLI

Reviewed the entire CDK lifecycle

Examined best practices for managing deployed CDK stacks

Deployed an API using Lambda, API Gateway, and S3

Deployed a load-balanced Fargate service

Next Steps with the CDK

- **Multiple Environments**
- **Multiple Stacks**
- **Continuous Delivery**
- **Reusable Constructs**
- **Enforcing Security**