How Harry’s Shaved off their Operational Overhead by Moving to AWS Fargate

Anuneet Kumar
Senior Product Manager, AWS

Bryce Lohr
Technical Lead on Core Services, Harry’s

Agenda

Motivation

Introduction to AWS Fargate

AWS Fargate at Harry’s

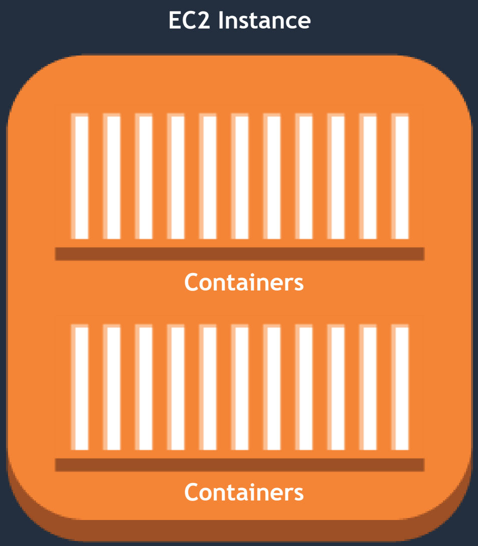
Motivation

At first there was

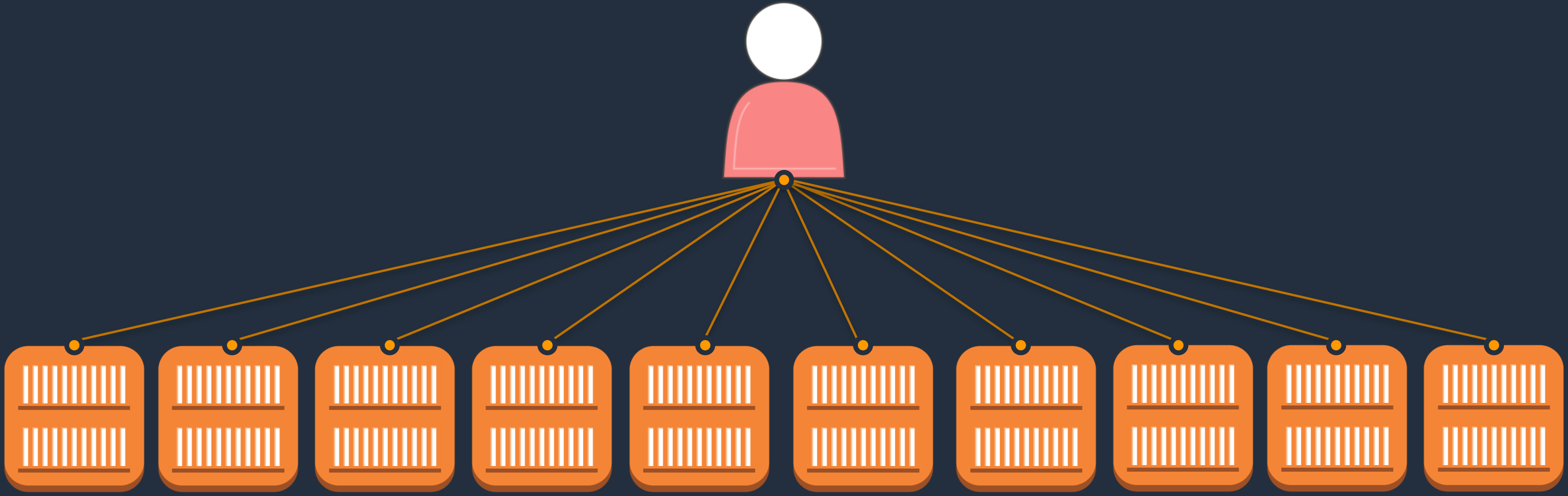
Amazon EC2

Then Docker

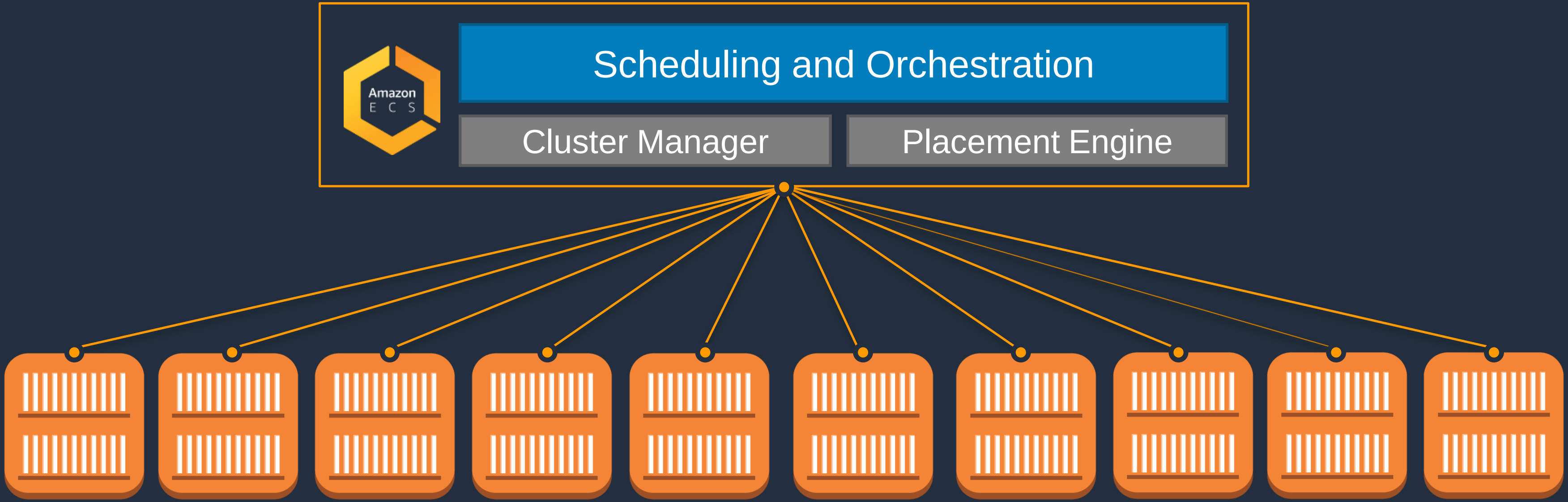
Customers started containerizing applications within EC2 instances



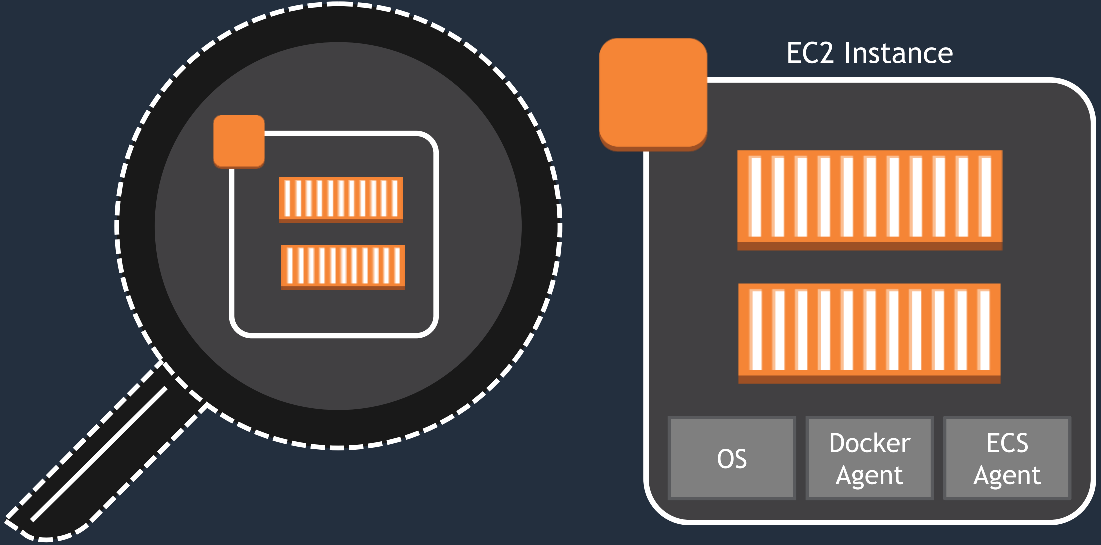
Containers made it easy to build and scale cloud native applications



Customers needed an easier way to manage large clusters of instances, place containers and run services

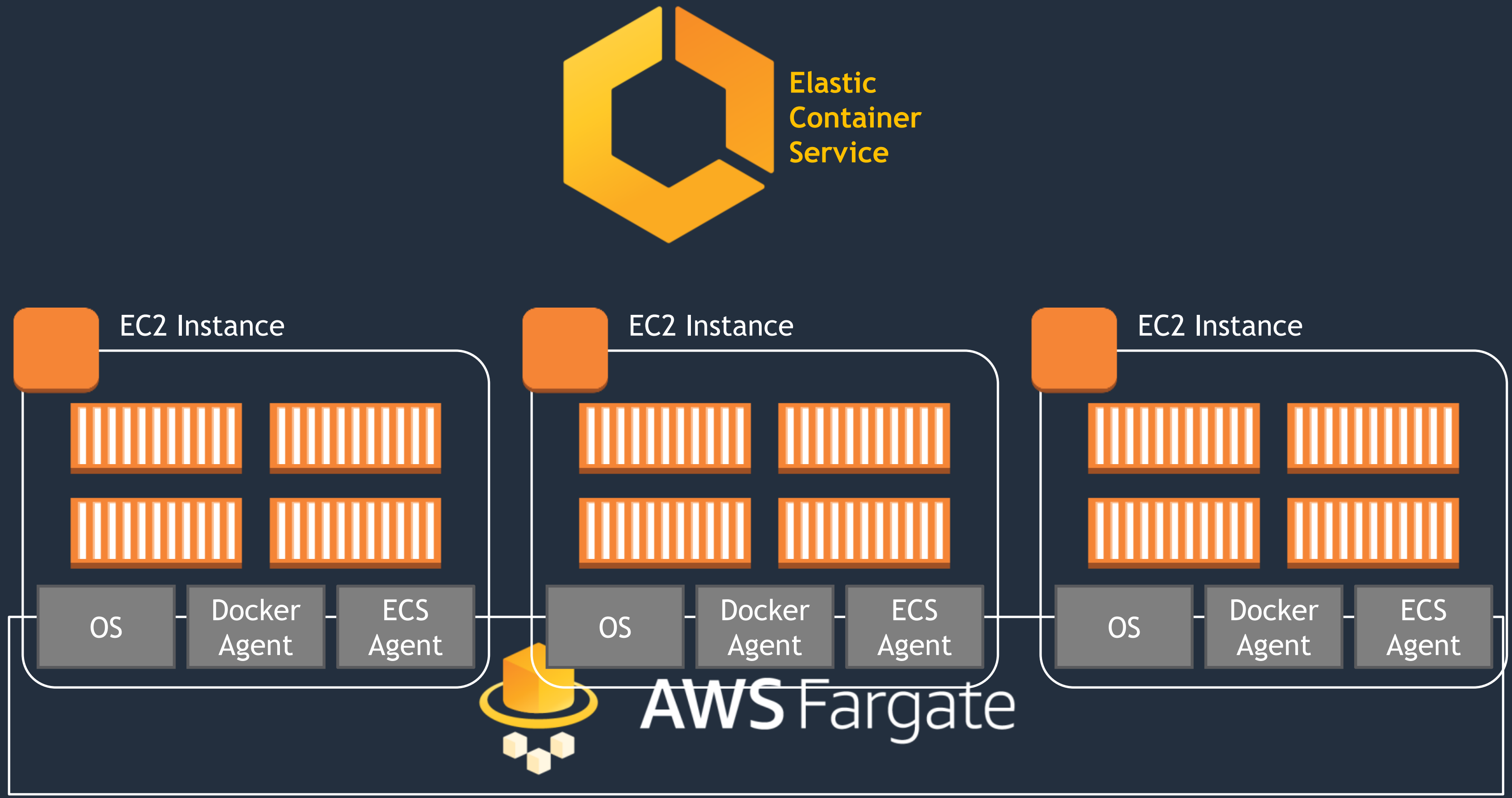

Amazon ECS
Scheduling and Orchestration
Cluster Manager
Placement Engine

But you still end up managing more than just containers



Patching and upgrading OS, Agents, etc.

Scaling the instance fleet for optimal utilization


Elastic
Container
Service
EC2 Instance
EC2 Instance
EC2 Instance
OS
Docker Agent
ECS Agent
OS
Docker Agent
ECS Agent
OS
Docker Agent
ECS Agent
AWS Fargate

Introduction to AWS Fargate

AWS Fargate

Managed by AWS

No EC2 Instances to provision, scale, or manage

Elastic

Scale up and down seamlessly, pay only for what you use

Integrated

With the AWS ecosystem: VPC Networking Elastic Load Balancing, IAM Permissions, Cloudwatch, and more

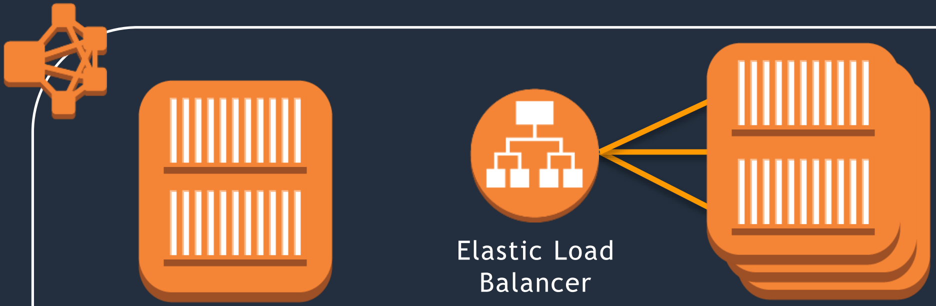
Your containerized applications

AWS Container Services Landscape

Management Deployment, scheduling, scaling, and management of containerized applications	Amazon Elastic Container Service	Amazon Elastic Container Service for Kubernetes
Hosting Where the containers run	Amazon EC2	AWS Fargate
Image Registry Container image repository	Amazon Elastic Container Registry	

Constructs when using Fargate with Amazon ECS

Constructs



Register
Task Definition
Define application containers: Image URL, CPU and Memory requirements, etc.

Run
Task
A running instantiation of a task definition

Use **Fargate launch type**

Create
Service
Maintain running copies

Integrated with ELB

Unhealthy tasks automatically replaced

Create
Cluster

Infrastructure isolation boundary
IAM Permissions boundary

Who is using AWS Fargate?

Smartronix®

CORPINFO

youi

ancestry®

accenture

Product Hunt

ironSource

TEAM BEACHBODY® Decide. Commit. Succeed.®

SGK® A Matthews International company

aws

AWS Fargate at Harry’s

Harry’s

Business

Harry's is here to give guys a high-quality, no-nonsense shaving and grooming experience at a fair price. We sell more than just razors to make sure men can get everything they need to look and feel their best, conveniently.

Products

Razors and shaving accessories

Skincare for the whole body

Availability

Online at harrys.com

Retail at: Target, Walmart, and select specialty retailers

AWS at Harry's

E–commerce

Powers the main harrys.com site customers use (main app running outside of AWS

Data Engineering

Provides data warehousing and analytics to the organization

Redshift, RDS, S3, Data Pipeline, EMR, Lambda, SNS, ECS (EC2, Elastisearch

Core Services

Central shared services: Shopping cart, order processing and fulfillment, email

ECS (Fargate), RDS, SQS, SNS, Elastisearch, Cloudwatch

Core Service: Transactional Emails

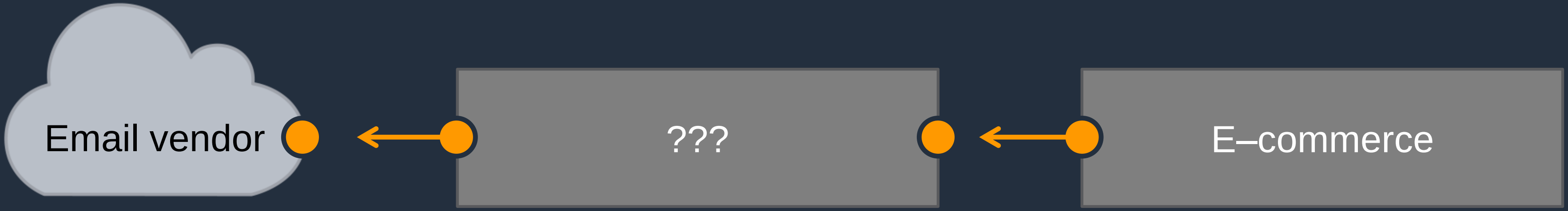
The problem

Allow Email Team to design, build, and test emails within 3rd party tool; Braze

Let e-commerce system "fire-and-forget" emails

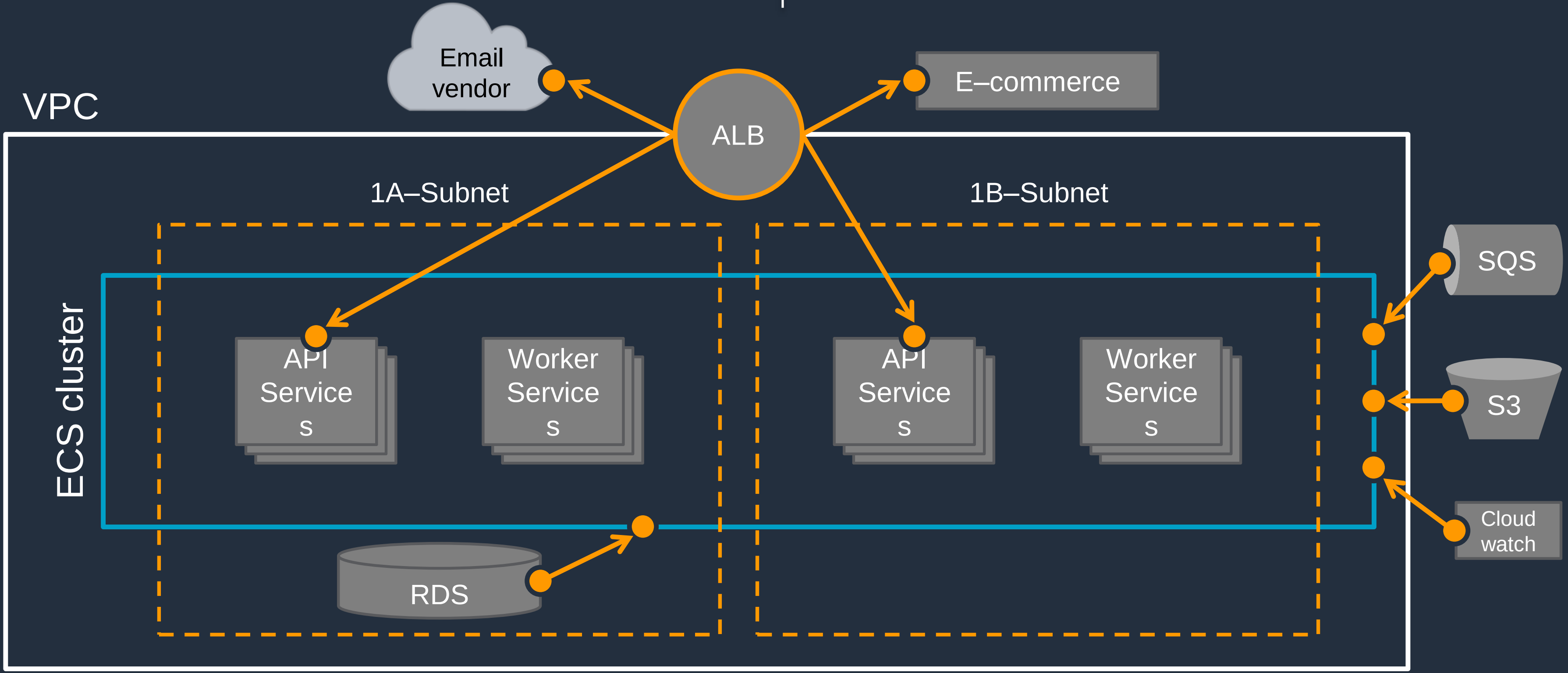
Handle long-lived, asynchronous workflows due to vendor API

Keep records and make sent emails, with content, available to customer service reps



Transactional Email System

Emails sent from E–commerce and relayed to vendor
Web hooks sent from vendor and queued for processing
User profiles created in vendor before sending emails



Pain points with Amazon ECS on Amazon EC2

Using containers was great, but operating them had some undifferentiated heavy lifting. We originally used 8 t2.small's, then switched to 4 m5.large's

Small team, less to manage is better

2 levels of auto–scaling, and they do not work well together

Patching EC2 Instances and ECS Agent

Cluster capacity planning (memory/CPU reservation vs. utilization, supporting rolling deploys

AWS Fargate benefits

We wanted to run containers, not EC2 instances

Eliminated EC2 instances, sizing concerns, instance profiles and policies

Replaced coarse-grained instance profiles with finer-grained IAM Task Roles

Directly leverage service auto scaling and target tracking policies

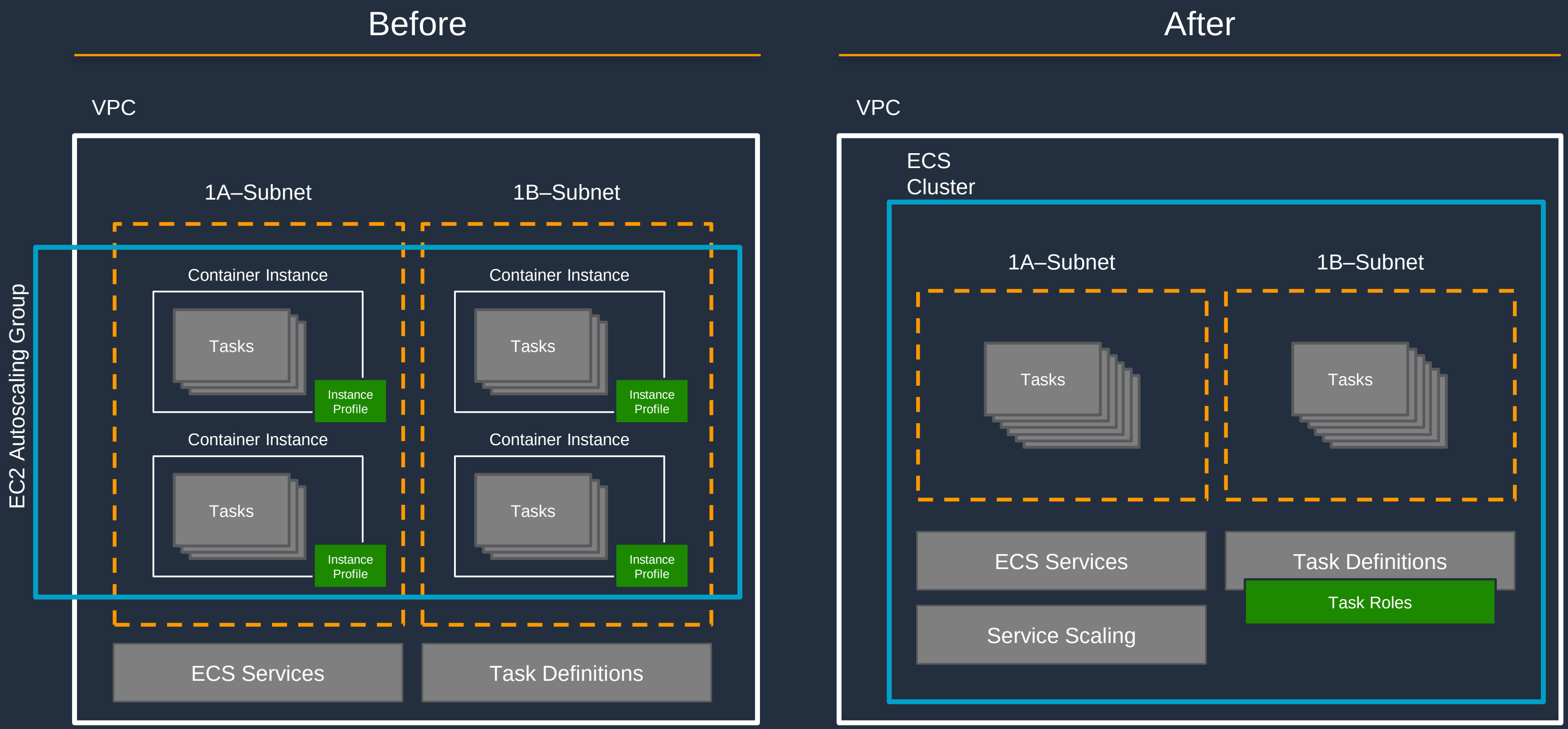
Shaved 2 hours off queue processing time for workers
Lowered average response time for API services by over half a second during peaks
Reduced 502/503 responses to client during peak loads

Migrated without any downtime

Simplified overall system

Less moving parts to operate and monitor in production
Removed a lot of CloudFormation code related to running instances

AWS Fargate benefits



Caveats and recommendations

- No–downtime migration requires 3–step deployment
  - Create new ALB target groups, services, and new identical, but lower-priority listener rules
  - Swap the priorities on the old and new rules; traffic will then go to new services
  - Drop the old rules and services after traffic dies off
- Low default limit on running containers hampers scaling for spiked workloads
  - Go ahead and submit for a limit increase
- Ensure services with different scaling requirements are different ECS services
- Watch the "Deep Dive on Fargate" talk from re:Invent 2017 (CON333)

Important Resources

- Deep-dive on Fargate (CON33): https://www.youtube.com/watch?v=CdxdjDpF8Eo&t=818s
- Build a modern web application: https://github.com/aws-samples/aws-modern-application-workshop
- AWS Compute Blog: https://aws.amazon.com/blogs/compute/
- Migrate ECS containers to Fargate: https://aws.amazon.com/blogs/compute/migrating-your-amazon-ecs-containers-tc aws-fargate

Questions