
AWS
re:Invent

ARC210

Well-architected containers: 10x performance with DevOps

Mai Nishitani
Solutions Architect
AWS

Mitch Beaumont
Principal Solutions Architect
AWS

Peter Kiem
Senior DevOps Engineer
Domain Group

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Well-Architected containers game

Win by collecting all 5 keys in:

 Cost optimization

 Security

 Reliability

 Performance efficiency

 Operational excellence

Well-Architected containers game

Stage 1 - Applying Well-Architected to containers

Stage 2 - Demo: Container and DevOps tools

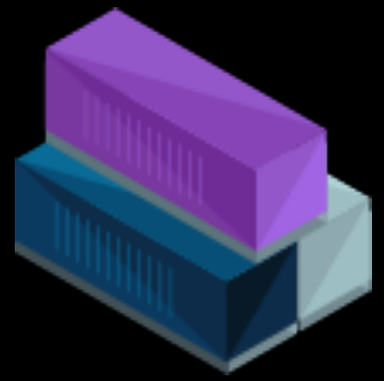
Stage 3 - Domain Group talks about their container journey

Final stage - Useful info to get started

Well-Architected containers







Collect all 5 keys to the pillars on containers!

Optimize Amazon Elastic Container Service (Amazon ECS) and AWS Fargate

DevOps





x5



Level: 200

Finance:

“Great work on your containers!
Can we also reduce our AWS costs?”





DevOps



Yes

No

Cost optimization 

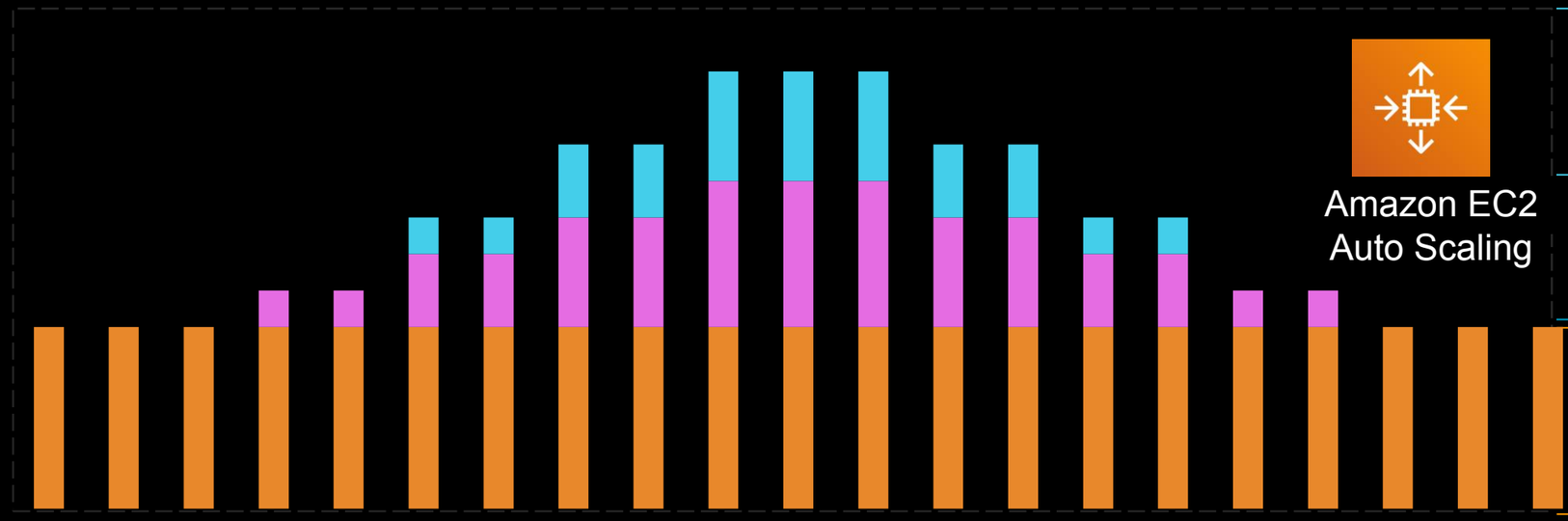
Amazon ECS on Amazon Elastic Compute Cloud (Amazon EC2)

- Save 90% with Spot Instances
- Mix and match On-Demand and Spot Instances

Amazon ECS on AWS Fargate

- Pay for what you use
- Flexible pricing options on AWS Fargate
- Compute savings plans to save up to 66%

Cost Optimization with Purchase Models



Scale using Spot for short-lived, fault-tolerant, stateless workloads

Scale using On-Demand for short-lived, stateful workloads

Use Savings Plans for known / steady-state workloads

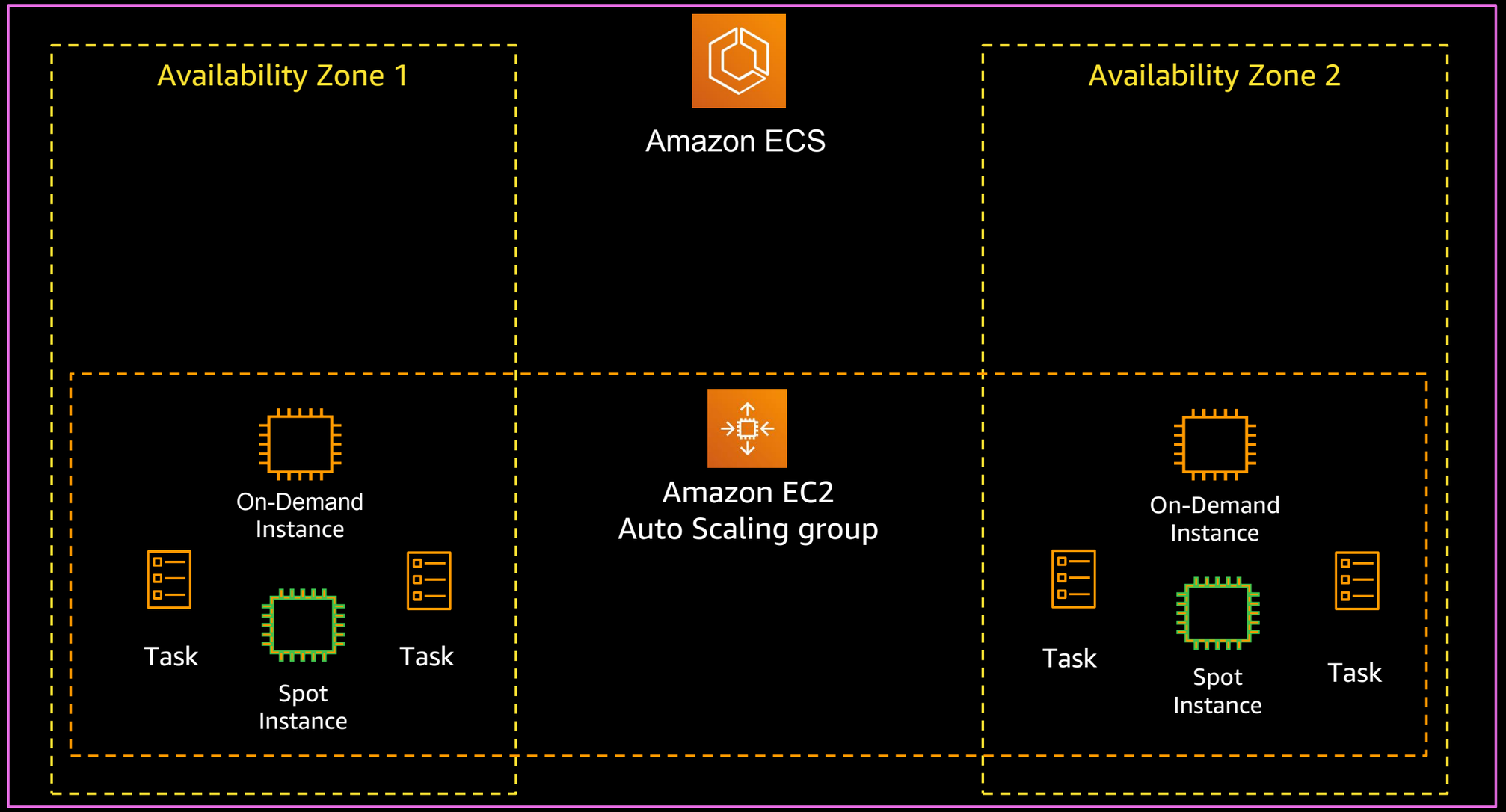
Save up to 66% with Compute Savings Plans

Amazon ECS

AWS Fargate

aws

Cost optimization: On-Demand and Spot Instances on Amazon EC2



Cyber security:

“There’s vulnerabilities out there.
Can we secure our containers?”



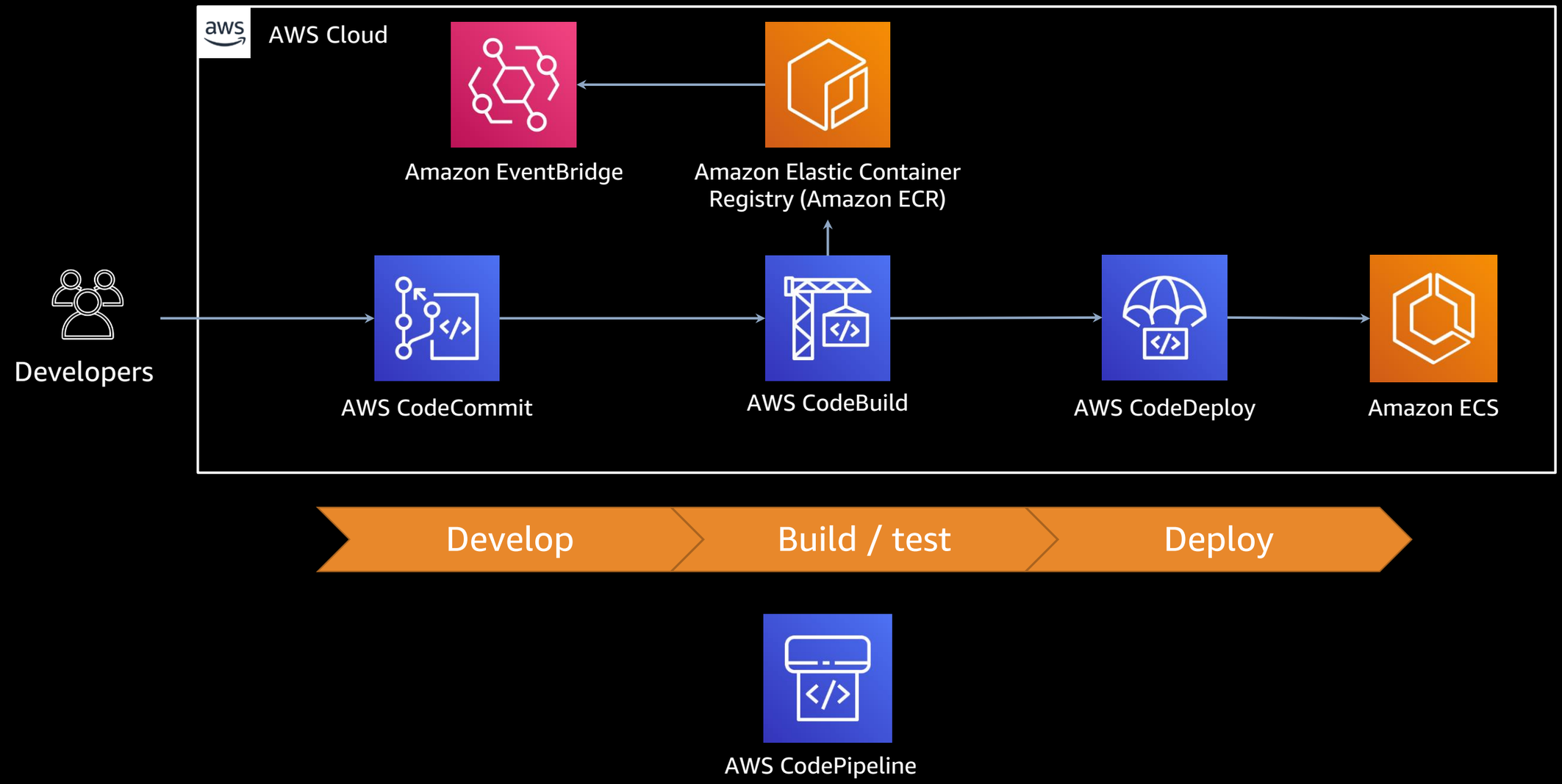
DevOps



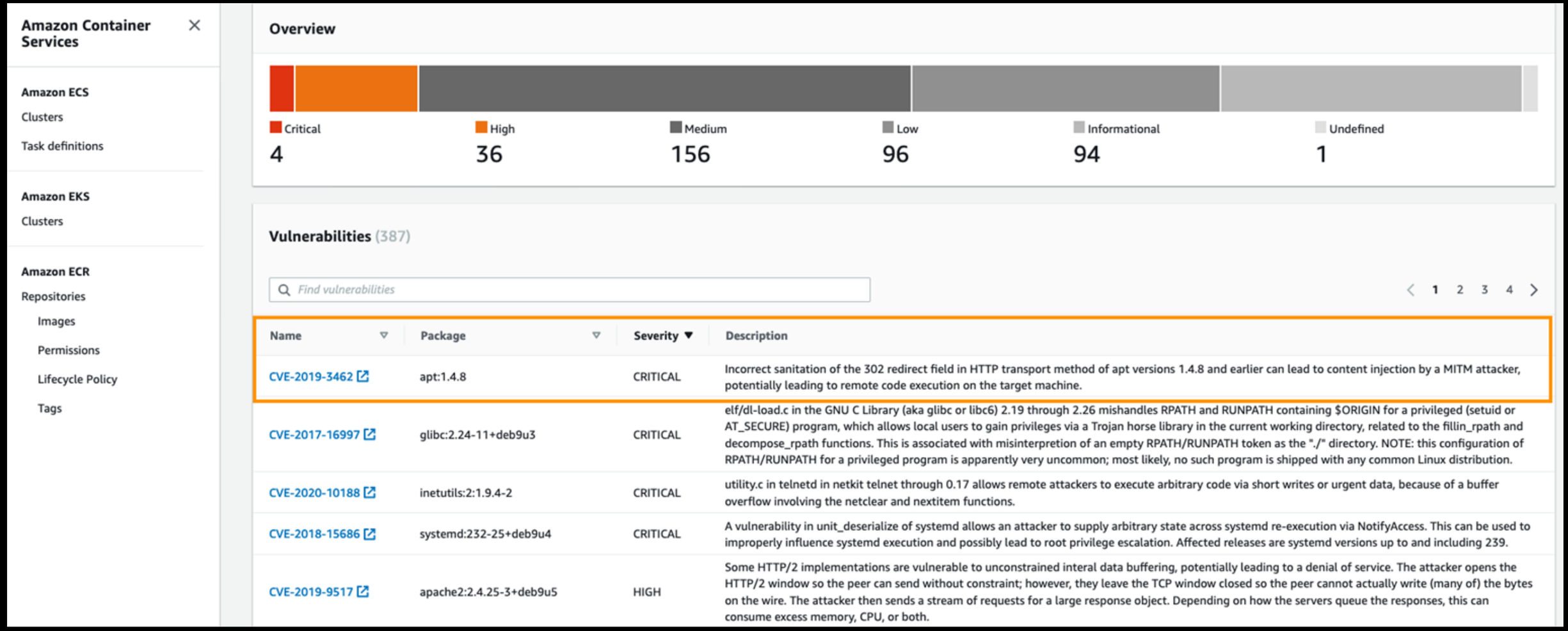
Yes

No

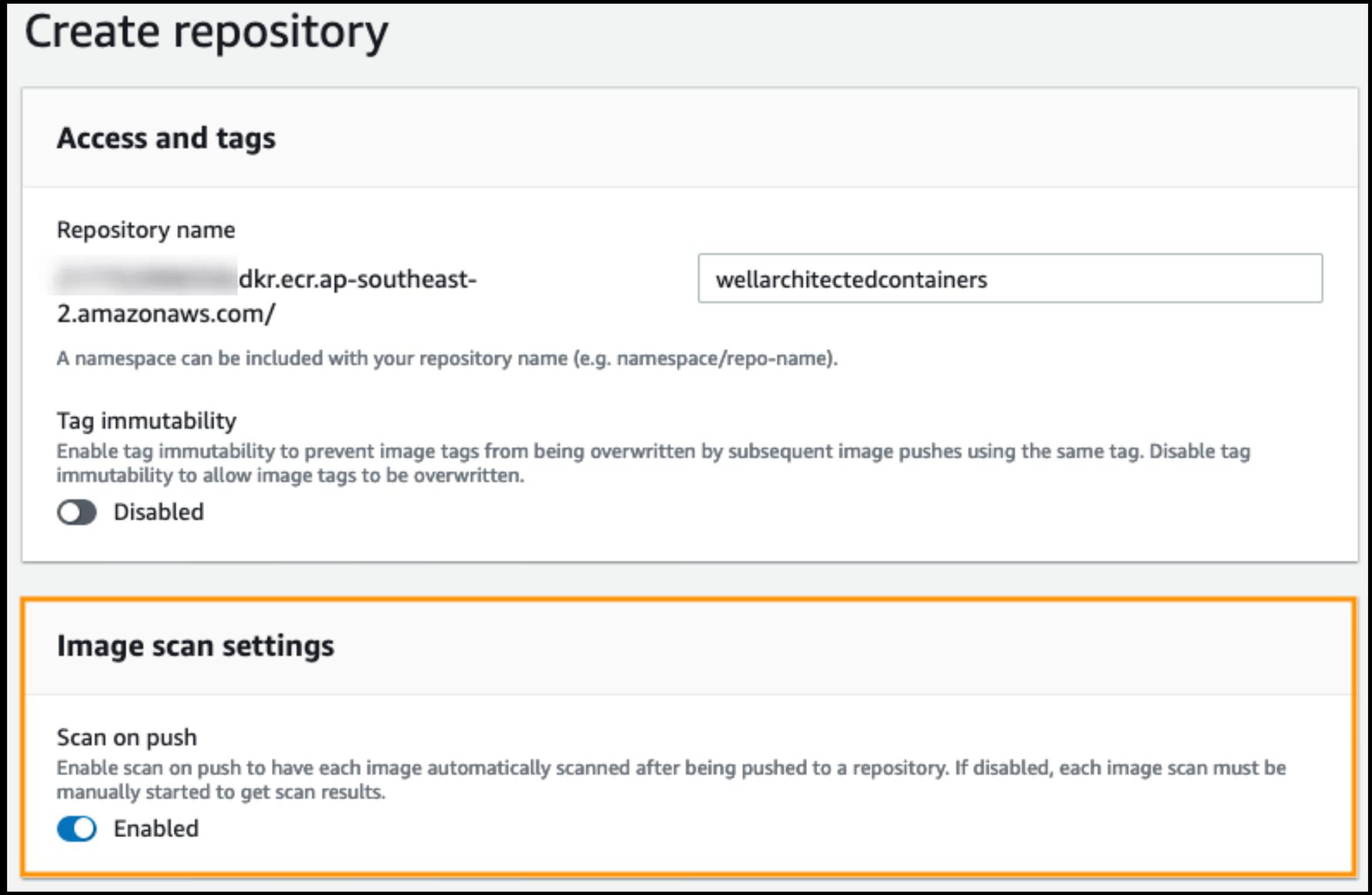
Container security: Amazon ECR image scanning



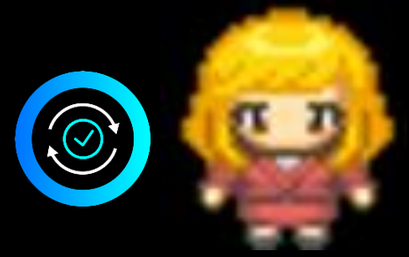
Amazon ECR image scanning: Dashboard



Amazon ECR image scanning: Configuration



Marketing:



“We have an upcoming event.
Can our containers handle the traffic?”

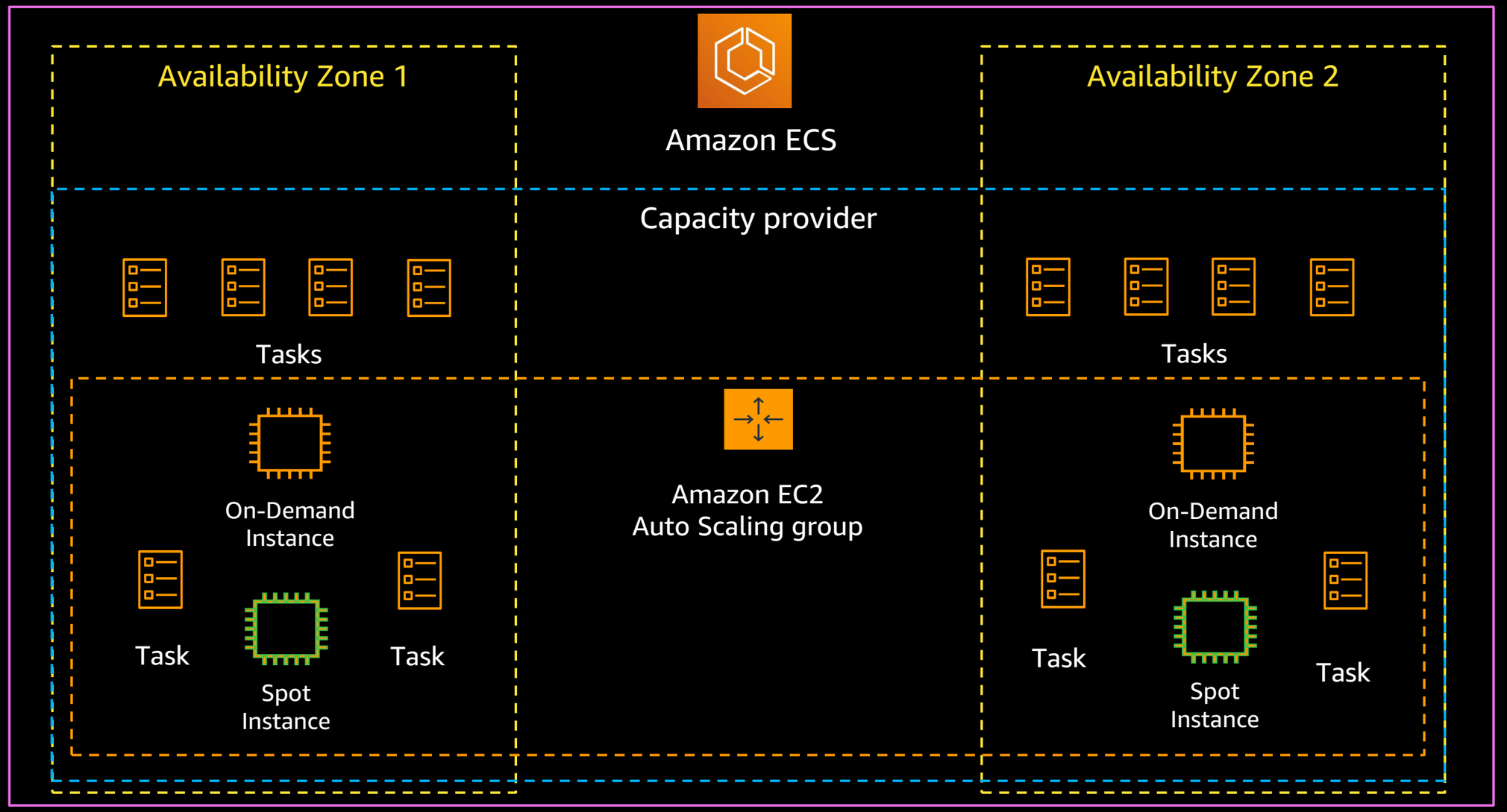
DevOps



Yes

No

Reliability: Amazon ECS capacity providers



IT manager:

“Can you check whether our containers are performing?”



DevOps



Yes

No

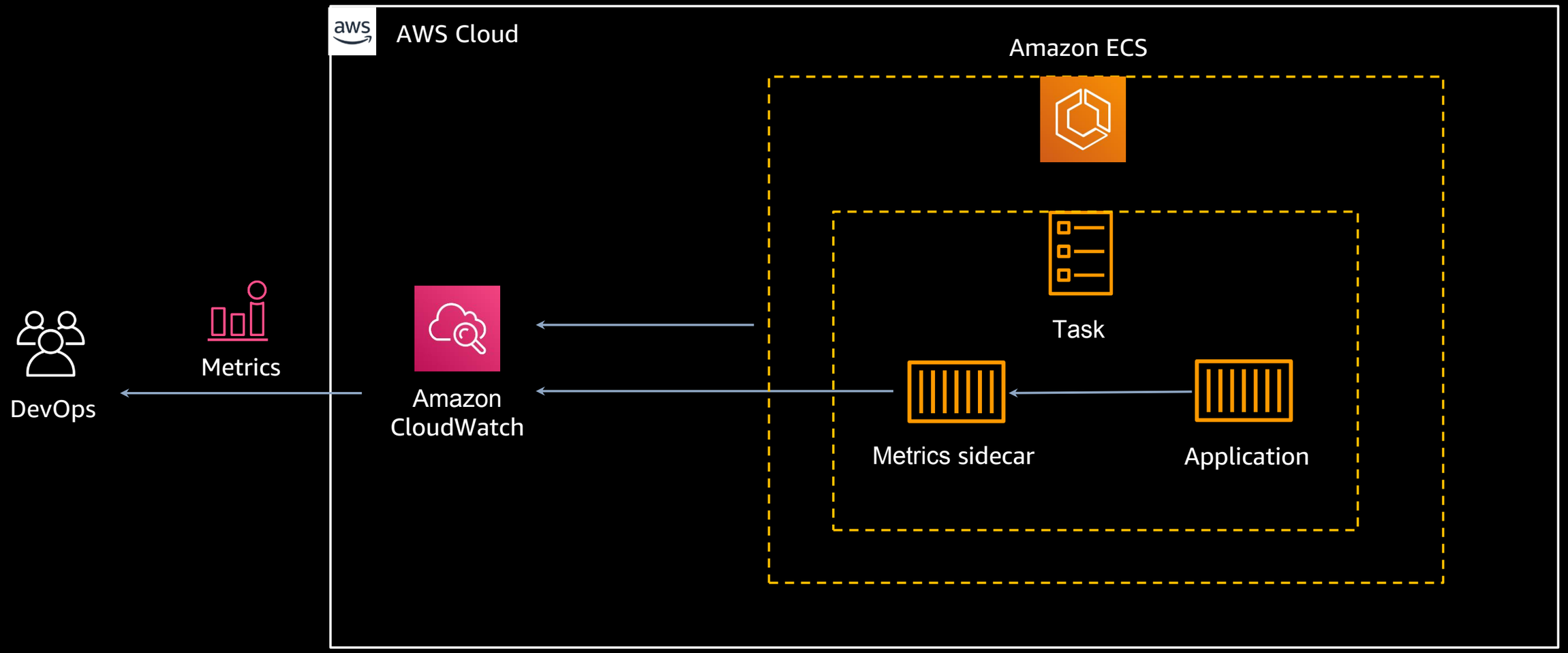
Performance efficiency 

Amazon CloudWatch Container Insights

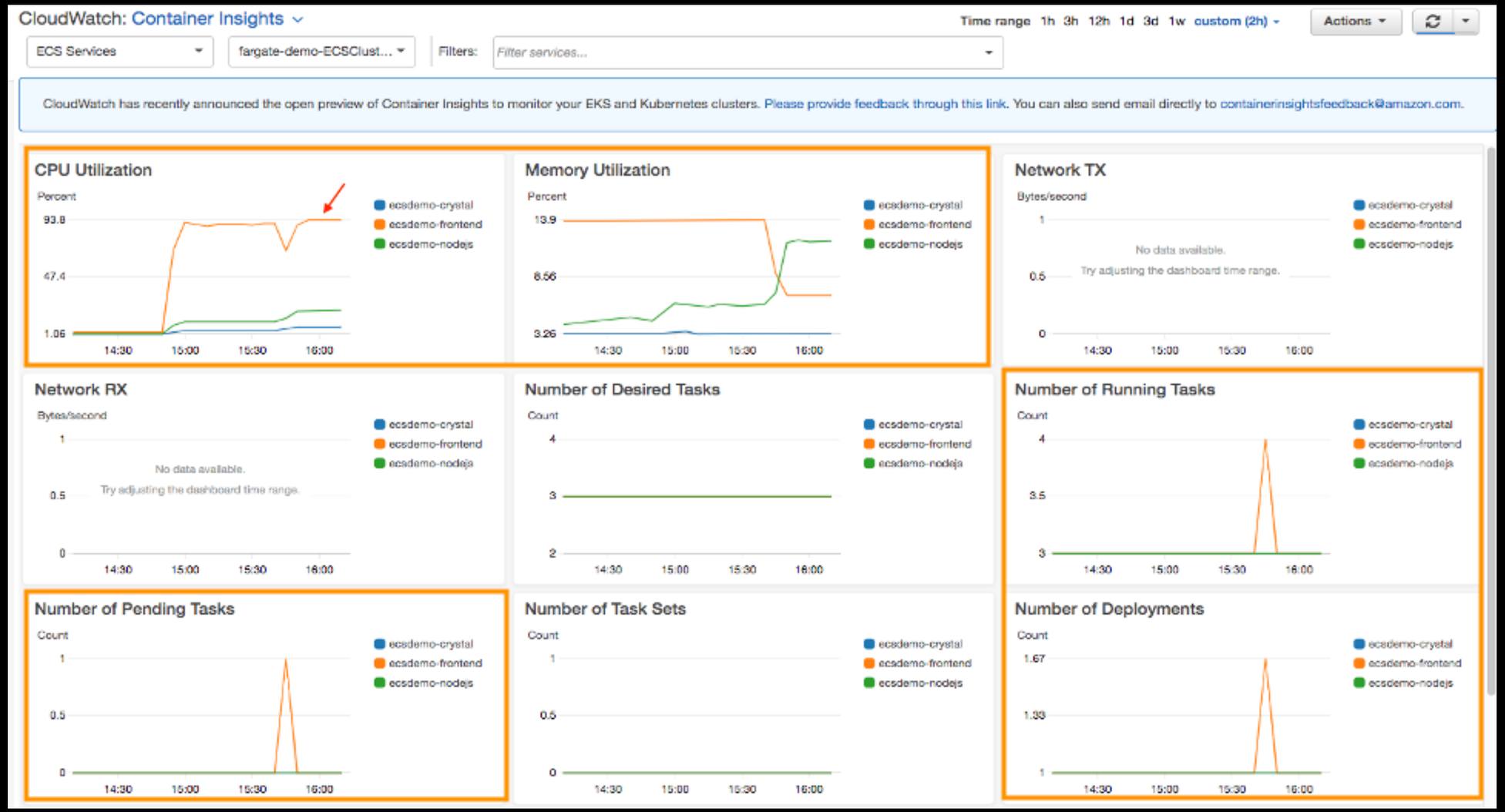
Analyze performance trends and automate workflows with:

- Detailed logs and metrics of your containers and apps
- Automated alarms and scaling
- Lower risks for unexpected changes

Performance: Amazon CloudWatch Container Insights



Amazon CloudWatch Container Insights: Dashboard



Stage 1 clear!



DevOps achieved 25% cost optimization with containers!



Container security went up 5 points!



Container reliability went up 8 points!



Container performance efficiency went up 10 points!

Collected 4 keys in Well-Architected containers!





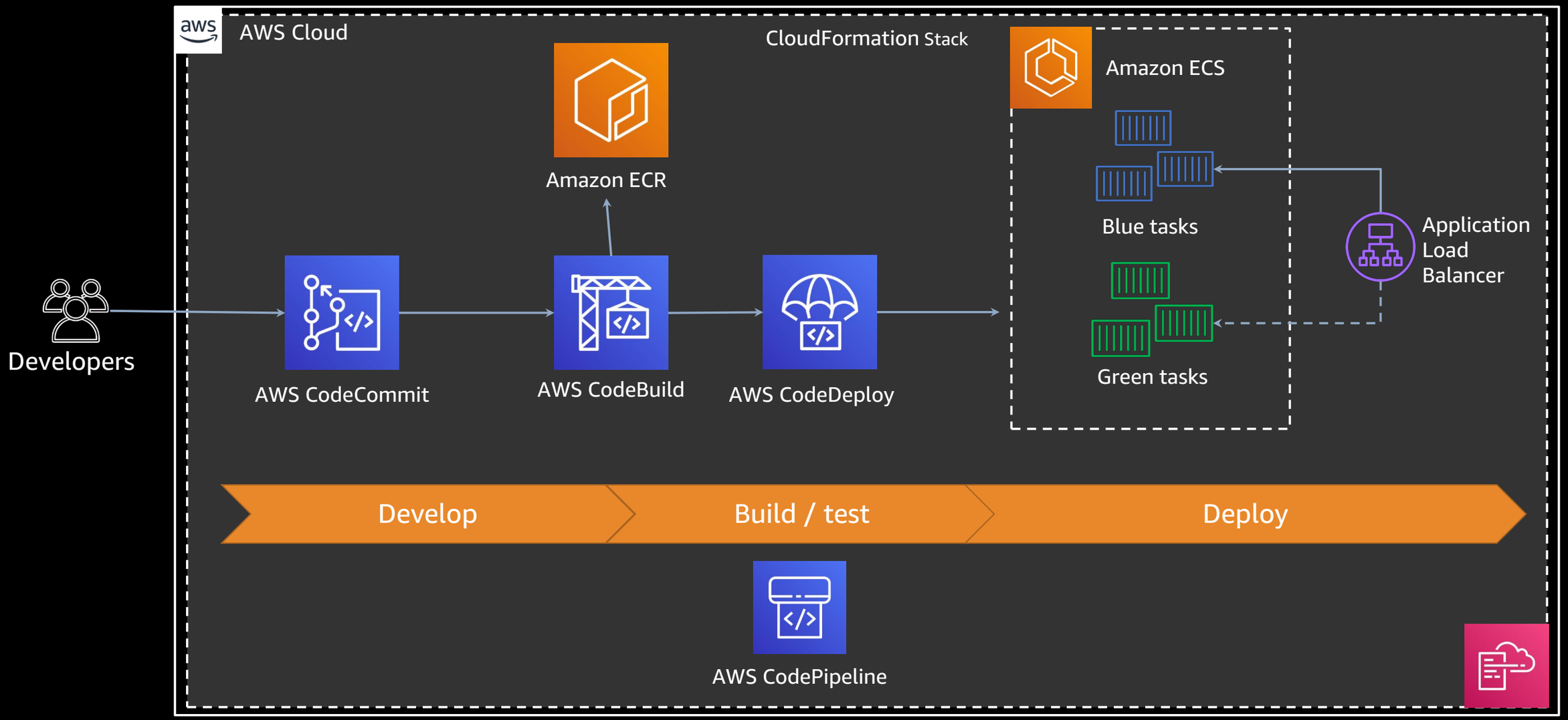
Stage 2: Demo - Container and DevOps Tools





Mitch Beaumont
Principal Solutions Architect
Containers Specialist

Operational excellence: Frequent, small, and reversible changes



Developer Tools
CodePipeline

- Source • CodeCommit
- Artifacts • CodeArtifact
- Build • CodeBuild
- Deploy • CodeDeploy
- Pipeline • CodePipeline
  - Getting started
  - Pipelines
    - Pipeline
    - History
    - Settings
- Settings

Go to resource
Feedback

Developer Tools > CodePipeline > Pipelines > ARC210-Pipeline

ARC210-Pipeline

Notify | Edit | Stop execution | Clone pipeline | Release change

download-source Succeeded

Pipeline execution ID: 444e3c72-8fe8-47fc-97c1-37eac206e9c1

SourceAction
GitHub (Version 1)
Succeeded - 24 minutes ago
daea8953

daea8953 SourceAction: Increasing service count to 6

Disable transition

deploy-infrastructure Succeeded

Pipeline execution ID: 444e3c72-8fe8-47fc-97c1-37eac206e9c1

Deploy
AWS CloudFormation
Succeeded - 20 minutes ago
Details

daea8953 SourceAction: Increasing service count to 6

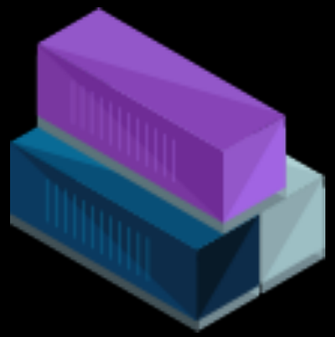
Disable transition

build-container Succeeded

Pipeline execution ID: 444e3c72-8fe8-47fc-97c1-37eac206e9c1

© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Stage 3: Domain - Containers in the real world



+





Peter Kiem

Senior DevOps Engineer

Domain Group

Domain

- Real estate media and technology services business
- Major part of the business is a real estate listings portal
- Quite literally on the side of a bus

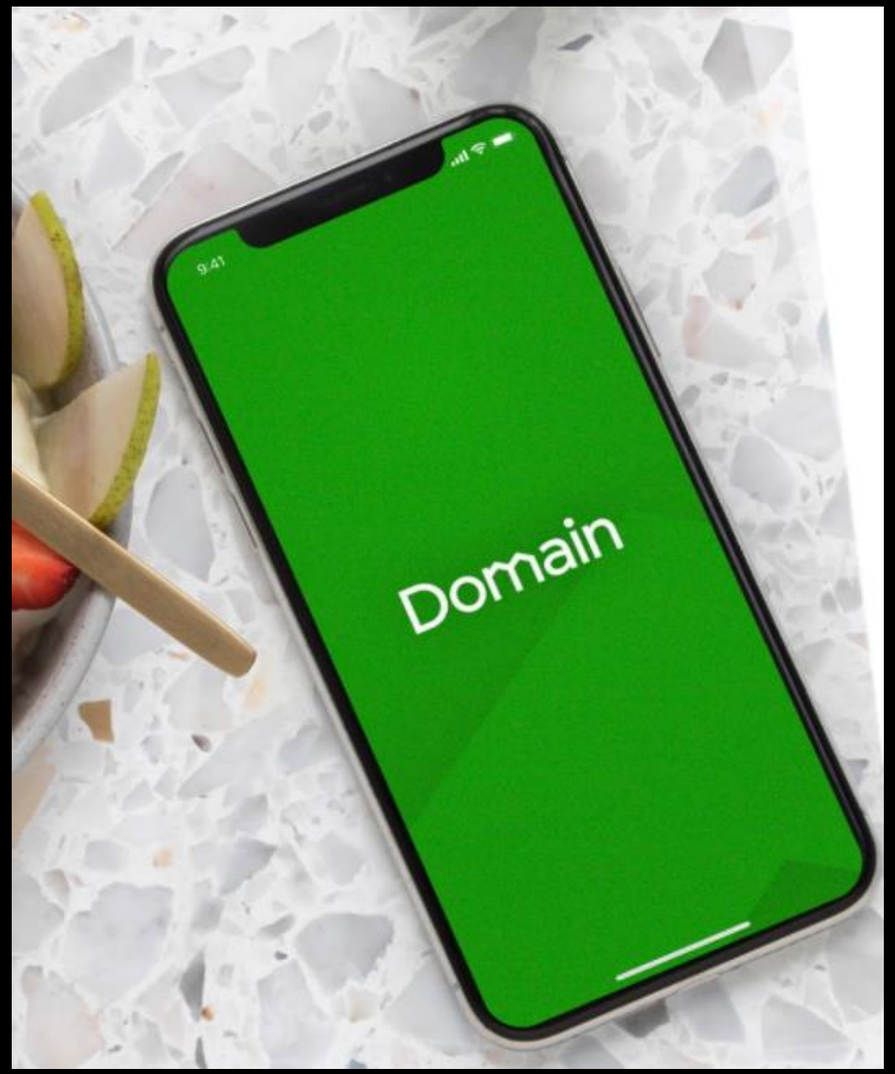


. . . and the bus stop



. . . and a tram



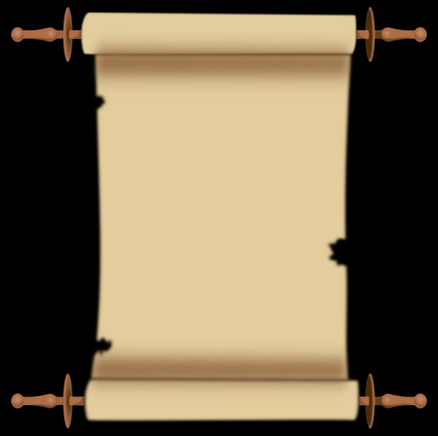


iOS and Android apps have more than 1 million users each month

Up 20% from the same time last year

Featured in the Australian TV series “The Block”

Problem statement



How can we improve our mobile API services?


Cost optimization


Security


Reliability


Performance efficiency


Operational excellence

Challenges





Use of older third-generation Amazon EC2 instances



Complex and inefficient AWS Auto Scaling



Old and dated infrastructure as code setup that has multiple dependencies on scaling



Poor cost optimization

Solution





Containers on AWS



Utilize build-and-deploy pipeline

Lift and ~~shift~~ ✖ transform! ✔

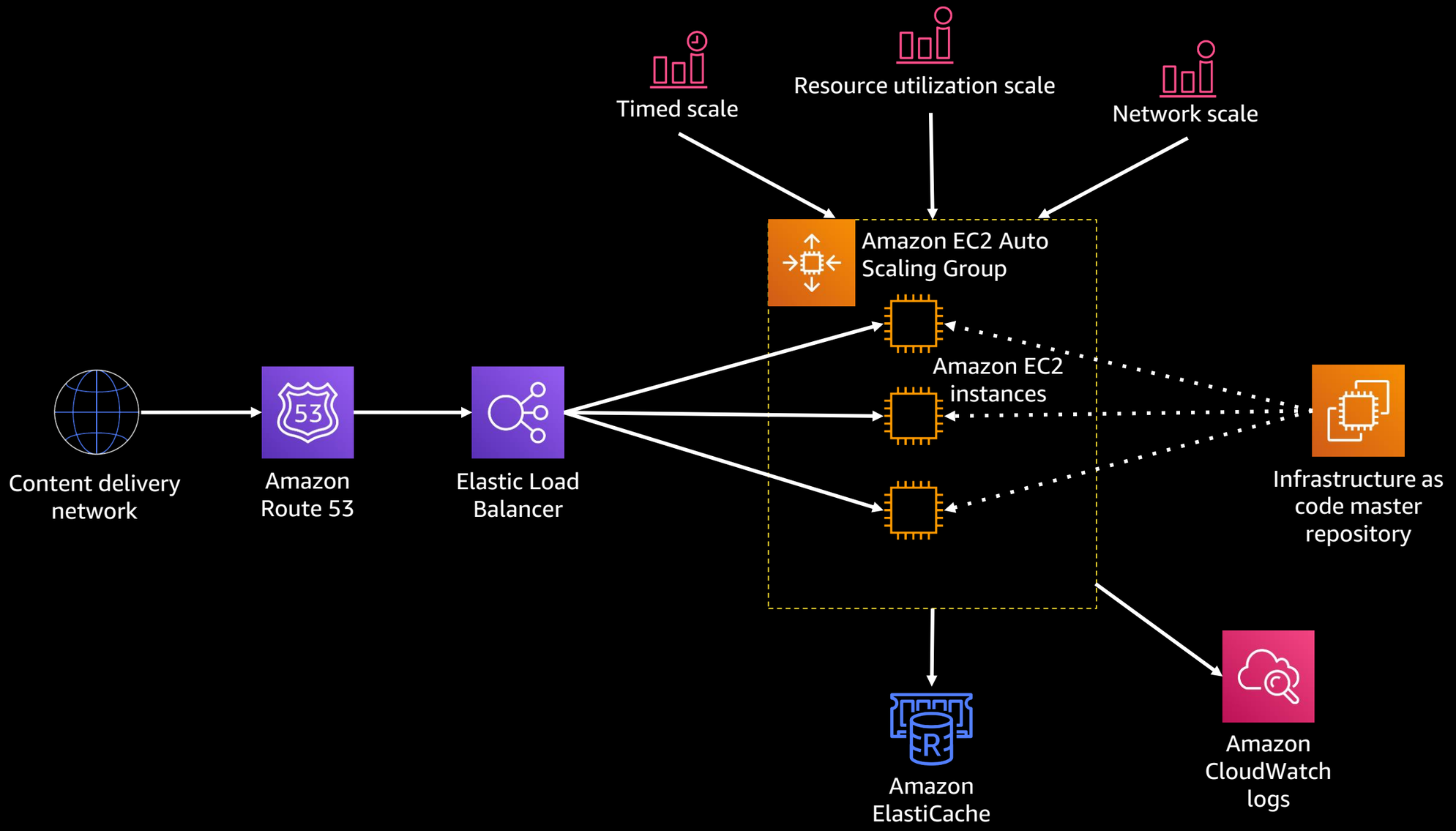


Managed Scaling through Amazon ECS Capacity Providers

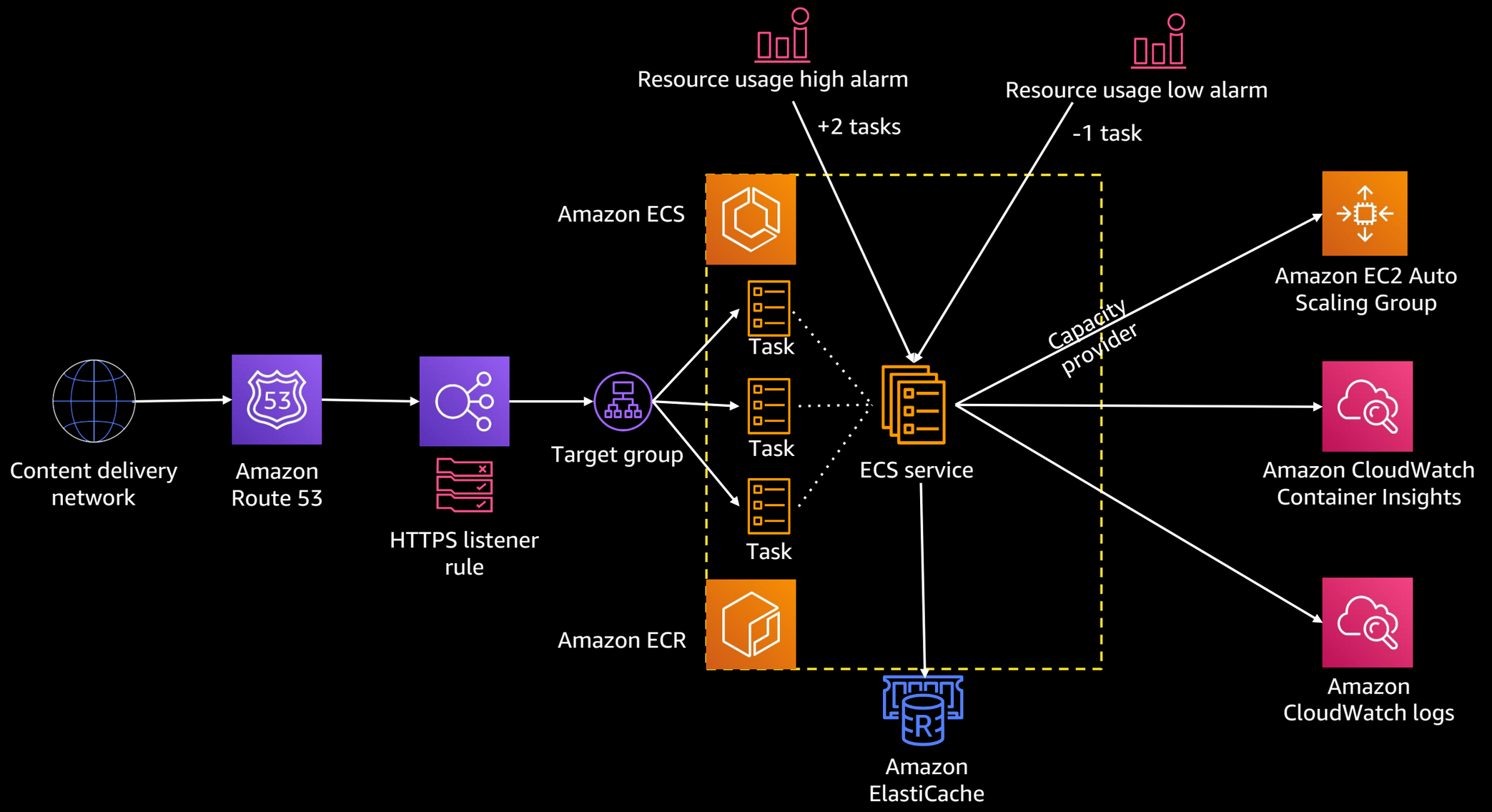
Architecture



Before


Timed scale
Resource utilization scale
Network scale
Amazon EC2 Auto Scaling Group
Amazon EC2 instances
Content delivery network
Amazon Route 53
Elastic Load Balancer
Infrastructure as code master repository
Amazon ElastiCache
Amazon CloudWatch logs

After


Resource usage high alarm
+2 tasks
Resource usage low alarm
-1 task
Amazon ECS
Task
Task
Task
Amazon ECR
Capacity provider
Amazon EC2 Auto Scaling Group
Amazon CloudWatch Container Insights
Amazon CloudWatch logs
Content delivery network
Amazon Route 53
HTTPS listener rule
Target group
ECS service
Amazon ElastiCache

Findings



Cost optimization



AWS Trusted Advisor to help with correctly sizing our infrastructure



Used Gzip compression to bring down Elastic Load Balancer costs



Validated our cost-savings using AWS Cost Explorer



Security



Automate rollout of Immutable AMIs with AWS Systems Manager Parameter Store





Security patches applied with latest ECS Optimized AMIs





Image scanning on push to Amazon ECR

Reliability



Placement strategy - needs to be appropriate to effectively use capacity provider

For us that was binpack on CPU



Amazon ElastiCache cluster - Higher connection count from containers caused strain on the ElastiCache cluster

Upgrading to the 5th Gen ElastiCache instances allowed us to support the increase in connection count

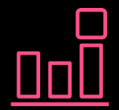
Performance efficiency



Used Amazon CloudWatch Container Insights to track usage and performance



Fine-tuned our ECS task definition to have appropriate resource reservation



Used CloudWatch to provision enough infrastructure when needed

Operational excellence



Need to scale the cluster x2 capacity quickly
ECS Capacity Provider wasn't adequate to support this



Initial workaround - chatbot command to scale before deploy





Solution - configured AWS CloudFormation > Amazon SNS notification > AWS Lambda

Outcome



Stage 3 clear!



25% Infrastructure cost optimization!



Automated AMI rollout - 5x faster patching - improved security!



0 Infrastructure incidents! 🎉 Reliability went up 20 points!



Scaling time reduced by 250% Performance efficiency went up 15 points!



Operational excellence - Frequent and reliable deployments allow us to support organizational priorities with 3x more efficiency

Collected 5 keys in Well-Architected containers!





Future



Operational excellence



Use AWS Cloud Development Kit (AWS CDK) to improve the build-and-deploy process

Performance efficiency



Use AWS Graviton2 6th-generation EC2 instances with Amazon ECS

Cost optimization



Further cost savings by having 20% of the cluster use Amazon EC2 Spot Instances

Security



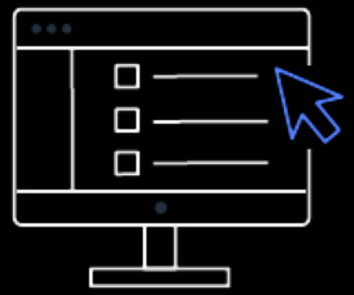
Bottlerocket AMI for the ECS cluster

Reliability



Apply this architecture to other workloads that are also on older EC2

Final stage: Get hands on!



Container resources

Sample code for this session and container resources

Get started with GitHub repository for the sample code in the demo and container tutorials and blogs!

github.com/aws-samples/aws-reinvent2020-arc210-well-architected-containers





Architecture resources

AWS Well-Architected Framework

The official best practices for architecting in the AWS Cloud

aws.amazon.com/architecture/well-architected

AWS Well-Architected Labs

Hands-on labs to help you learn, measure, and build using architectural best practices

wellarchitectedlabs.com

AWS Architecture Center

Official AWS repository for all Architecture resources

aws.amazon.com/architecture

AWS Solutions Library

Vetted reference implementations and well-architected patterns

aws.amazon.com/solutions



Thank you!



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Please complete the session survey

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.