Running Amazon EC2 Workloads at Scale

Boyd McGeachie (@BoydMcgeachie)
Sr. Product Manager, EC2

Chad Schmutzer (@schmutze)
Principal Developer Advocate, EC2 Spot

March, 2019

Agenda

Intro

EC2 Launch Templates / Demo

EC2 Fleet / Demo

EC2 Auto Scaling / Demo

At first, there was Amazon EC2

m1.small

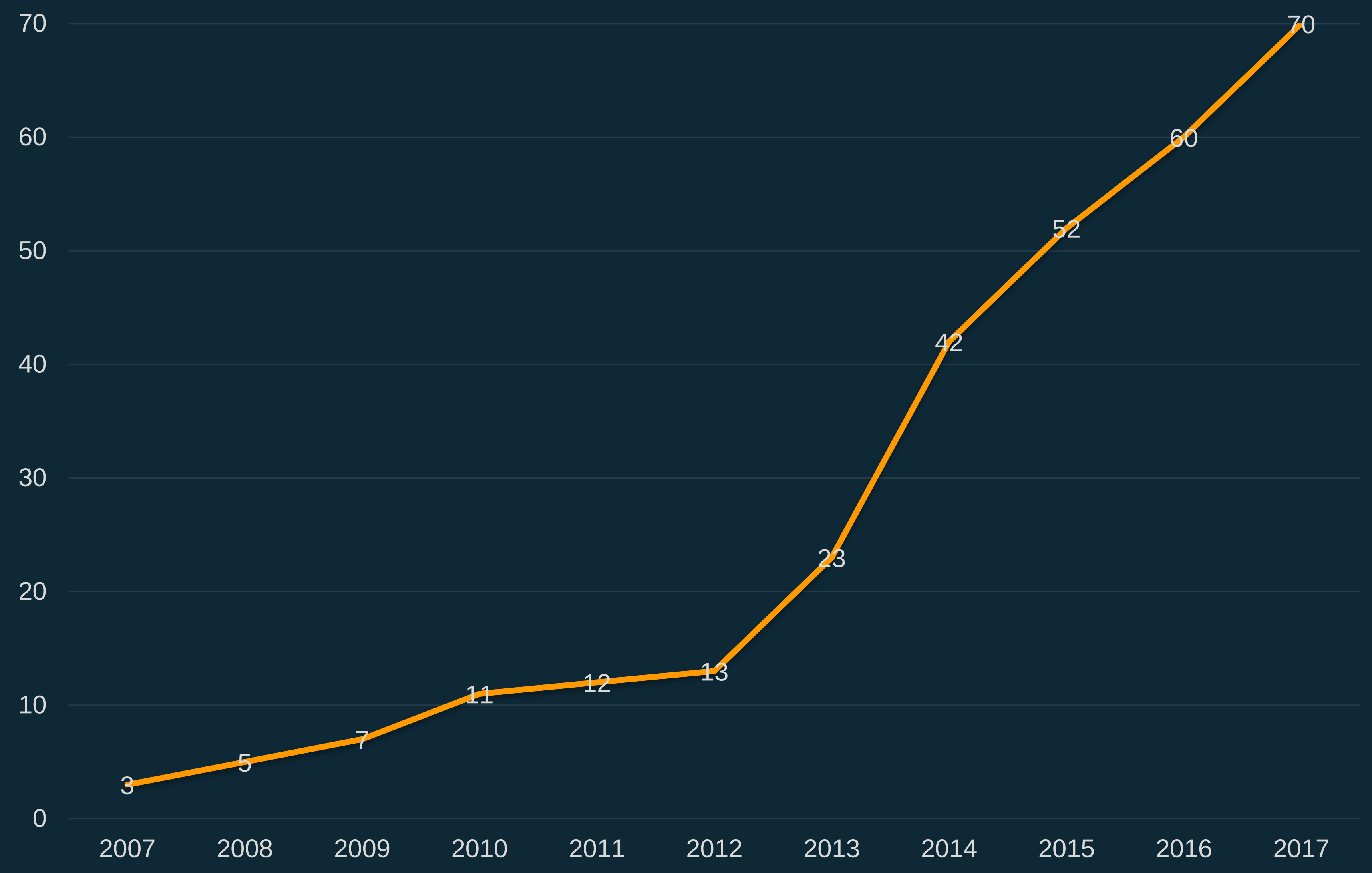
Then we added some new instance types

m1.small

m1.medium

m1.large

Then we added a lot more instance types



Recent launches:

Compute optimized: c5{d,n}

General purpose: m5{d,a}, t3, a1

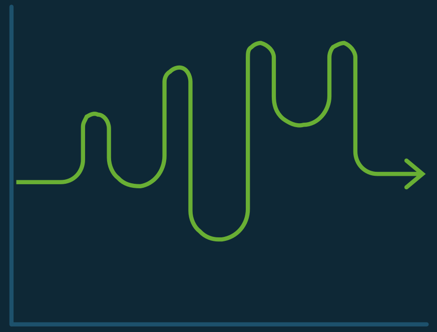
Accelerated computing: g3s

Memory optimized: z1d, r5{d,a}

Amazon EC2 purchasing options

On-Demand

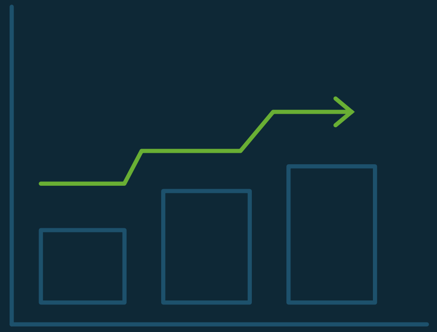
Pay for compute capacity by **the second** with no long-term commitments



Spiky workloads, to define needs

Reserved Instances

Make a 1- or 3-year commitment and receive a **significant discount** off On-Demand prices



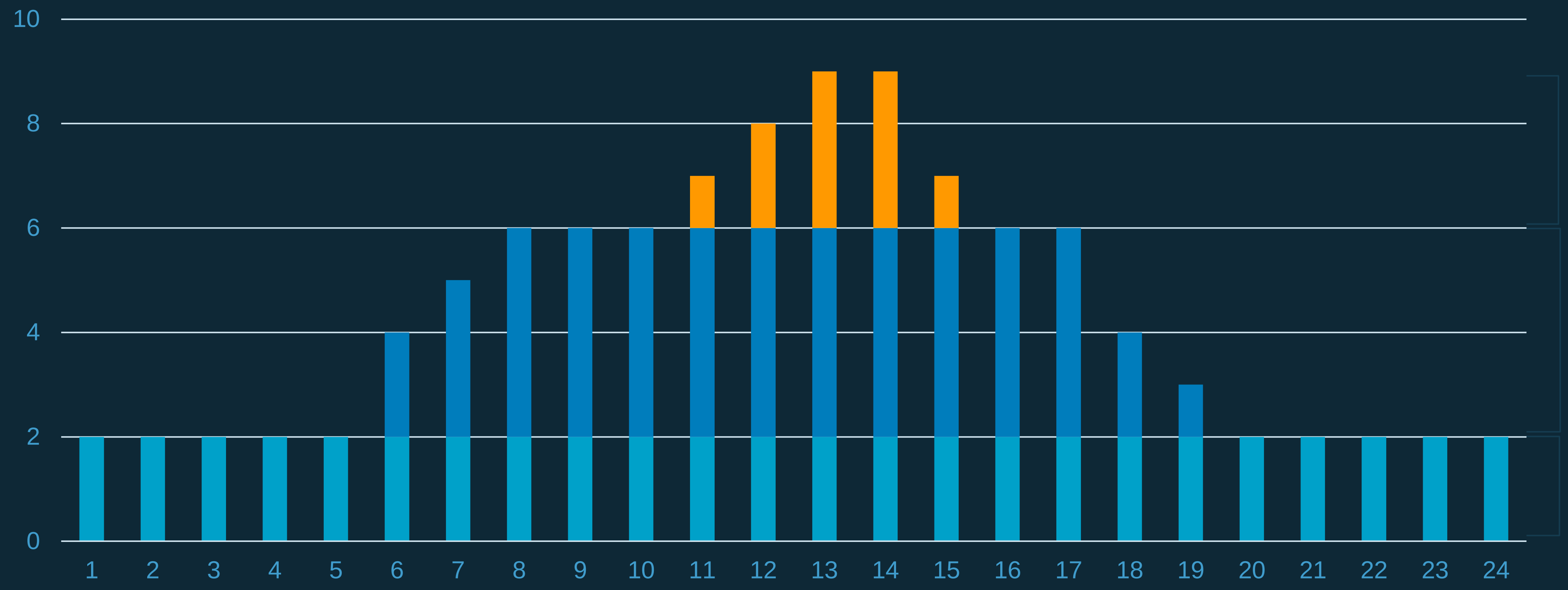
Committed & steady-state usage

Spot Instances

Spare EC2 capacity at **savings of up to 90%** off On-Demand prices

Fault-tolerant, flexible, stateless workloads

Combine purchase options to optimize at scale



Scale using **Spot**

On-Demand, or both

Use **RIs** for known/steady-state workloads

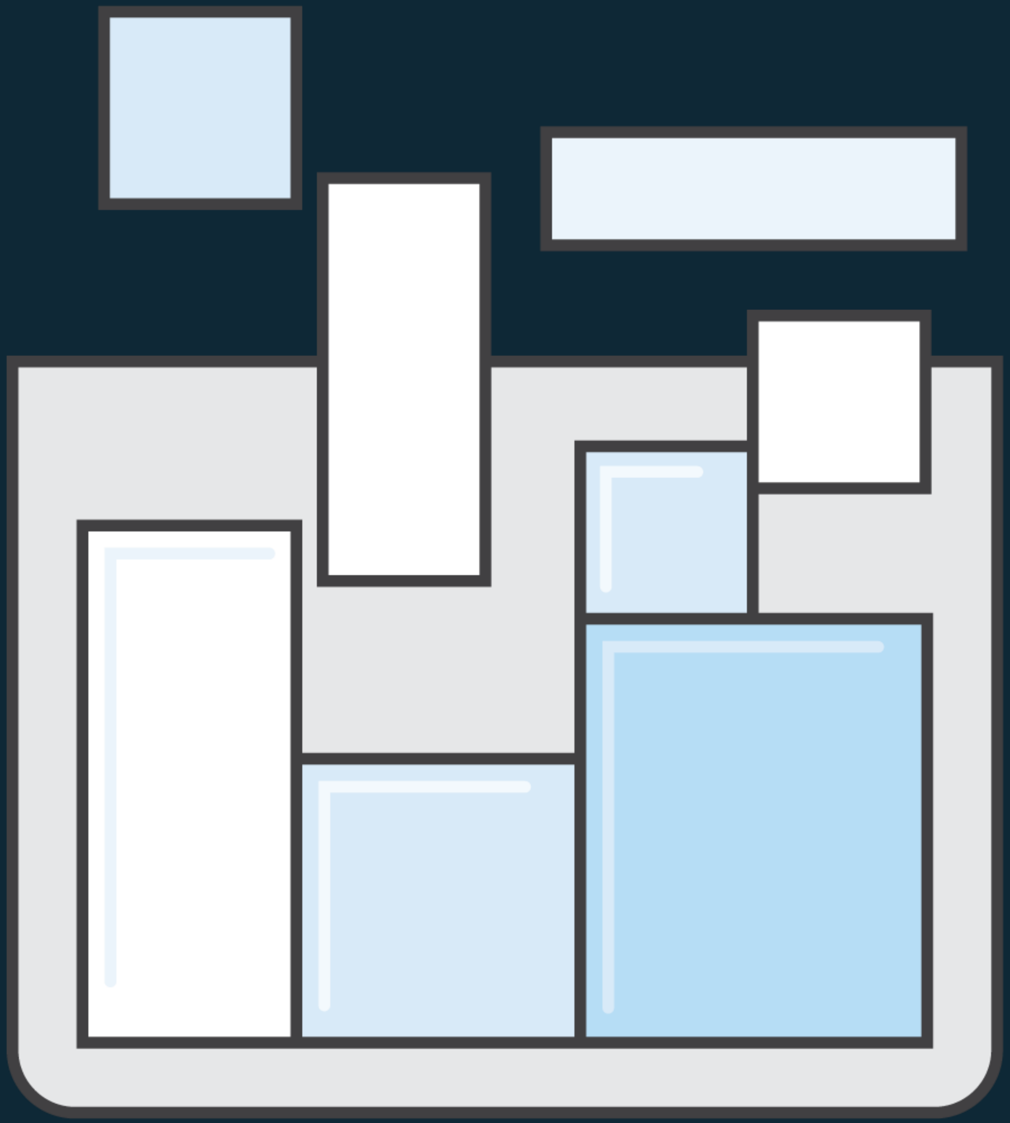
On-Demand capacity reservations for your reservation needs

Why combine instances and purchase models?

To turbo boost an application using Spot Instances

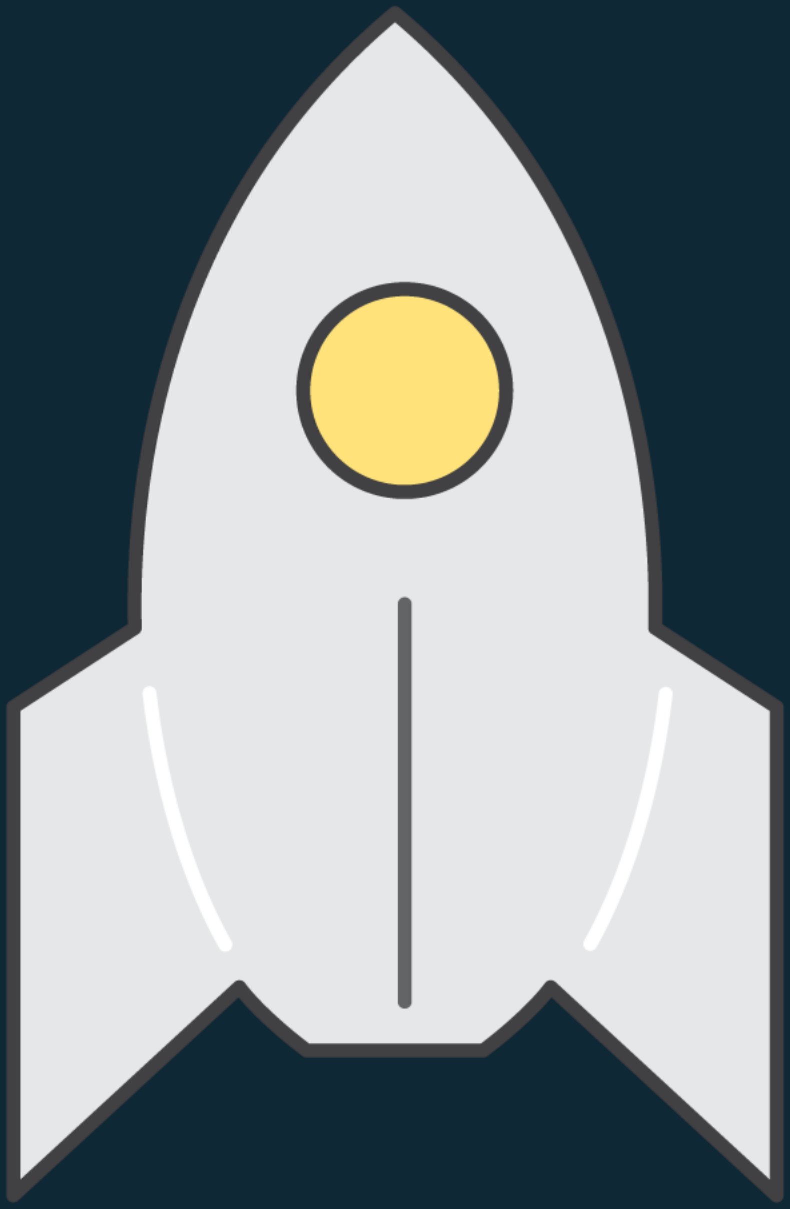
Why combine instances and purchase models?

To scale on vCPUs, memory, or containers



Why combine instances and purchase models?

To scale 1000x



The tools

Automate cost optimization & capacity management

EC2 Launch Templates

EC2 Fleet

EC2 Auto Scaling

… Let’s see how this all works together to automatically optimize scale, performance, and cost behind the scenes

EC2 Launch Templates

Use launch templates to achieve …

Increased productivity

Simplified permissions

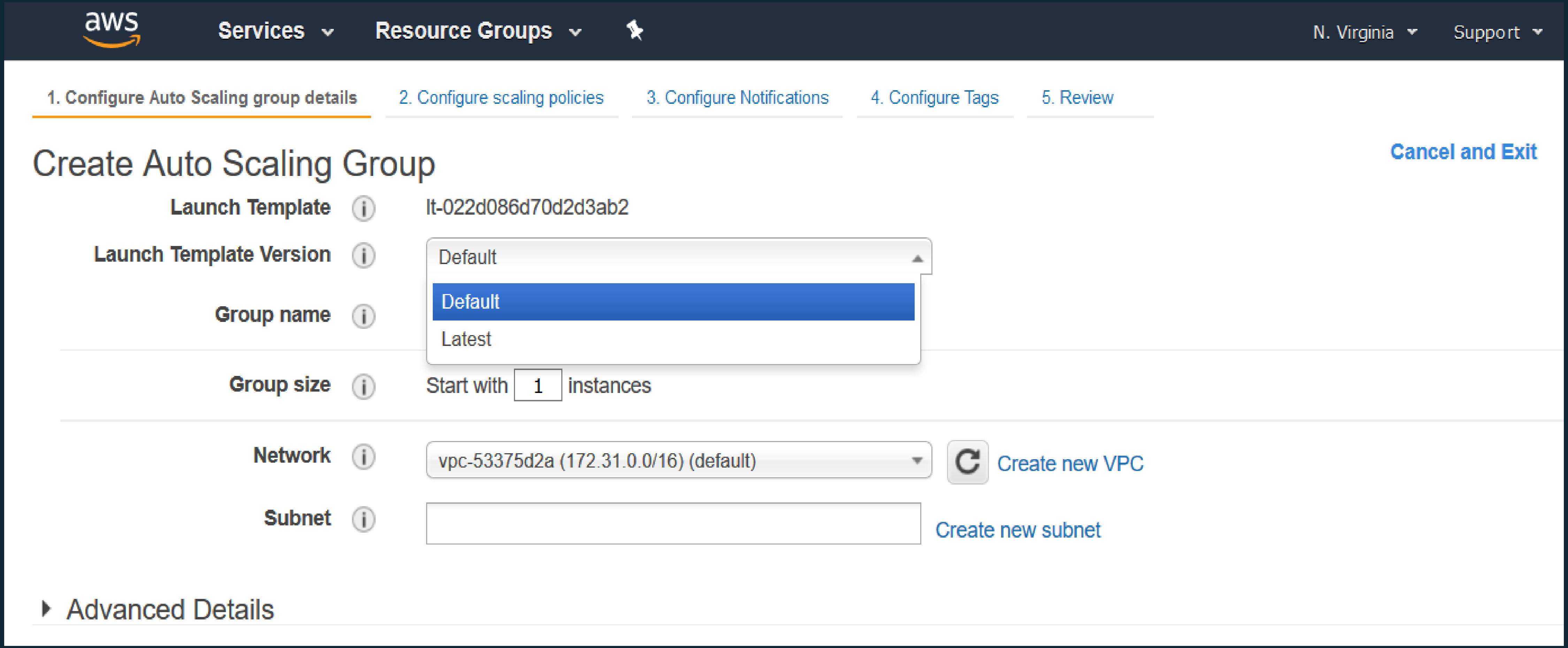
Governance & best practices

Consistent experience

Launch templates are now available in AWS CloudFormation with Auto Scaling and EC2 Fleet

Increased productivity: Automated updates

For example, push a patched AMI to EC2 Auto Scaling groups

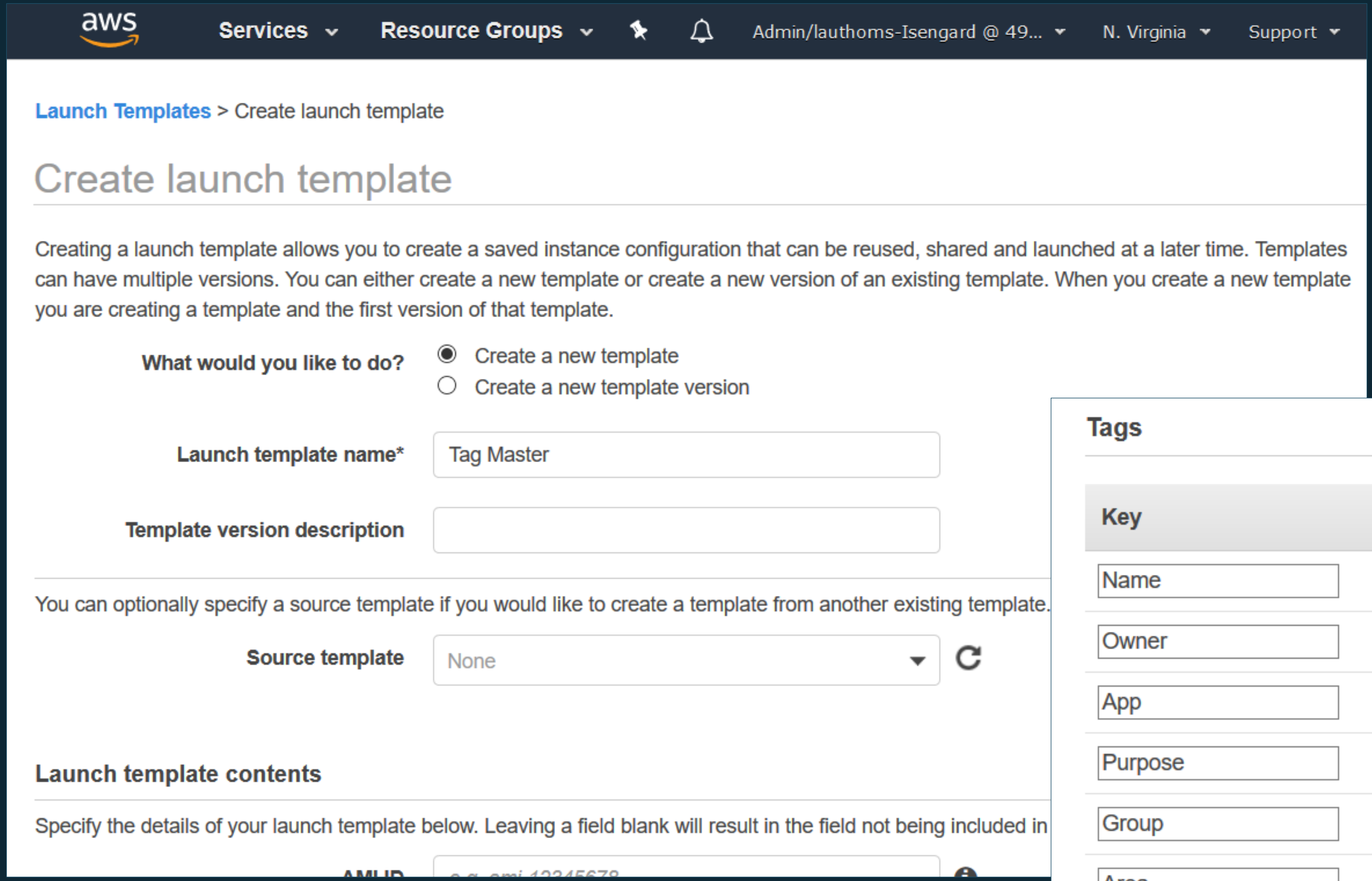


Create Auto Scaling Group

Launch Template	lt-022d086d70d2d3ab2
Launch Template Version	Default (Default / Latest)
Group name	
Group size	Start with 1 instances
Network	vpc-53375d2a (172.31.0.0/16) (default)
Subnet	

Increased productivity: Eliminate repetitive tasks

For example, save tags in a launch template



Launch Templates > Create launch template

Create launch template

Creating a launch template allows you to create a saved instance configuration that can be reused, shared and launched at a later time. Templates can have multiple versions. You can either create a new template or create a new version of an existing template. When you create a new template you are creating a template and the first version of that template.

What would you like to do? [✓ Create a new template] Create a new template version

Launch template name* Tag Master

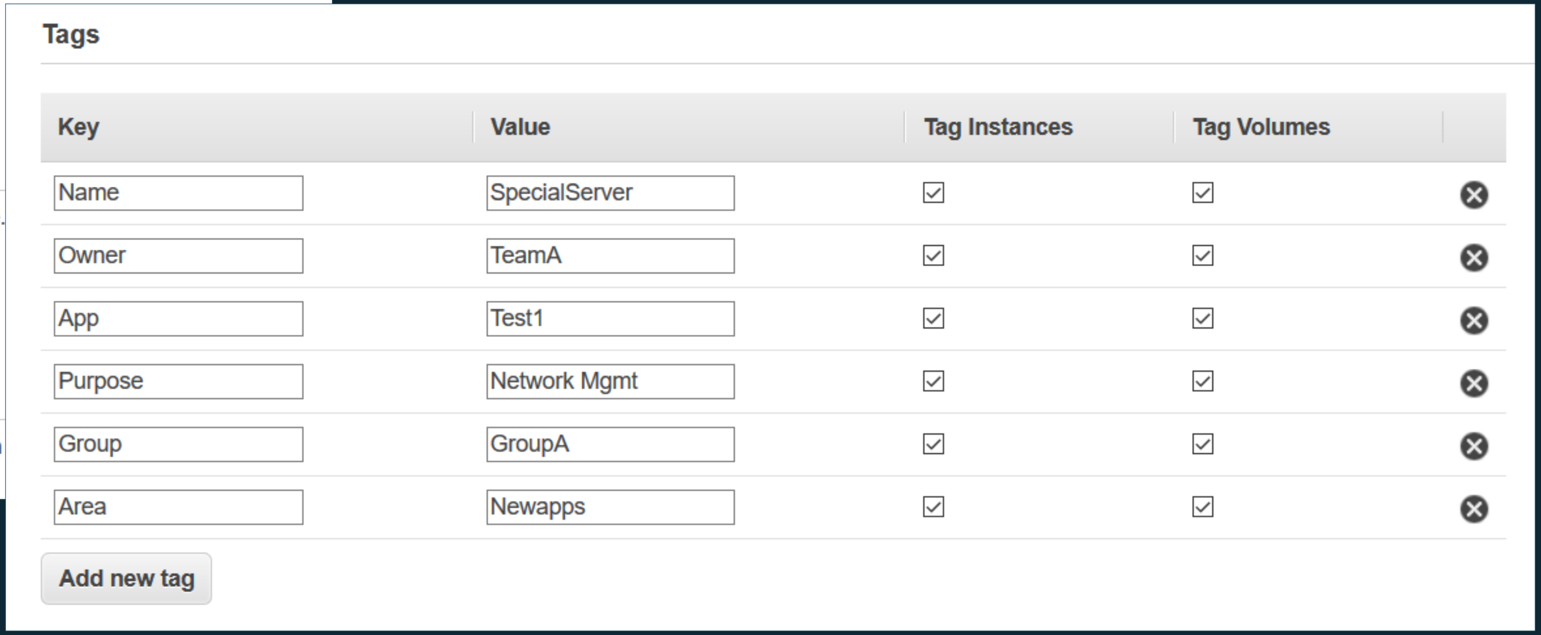
Template version description

You can optionally specify a source template if you would like to create a template from another existing template.

Source template None

Launch template contents

Specify the details of your launch template below. Leaving a field blank will result in the field not being included in



Tags

Key	Value	Tag Instances	Tag Volumes
Name	SpecialServer	✓	✓
Owner	TeamA	✓	✓
App	Test1	✓	✓
Purpose	Network Mgmt	✓	✓
Group	GroupA	✓	✓
Area	Newapps	✓	✓

Use launch templates as an Auth vehicle

Before

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
         "ec2:RunInstances"
      ],
            "Resource": [
         "arn:aws:ec2:us-east-1::image/*",
         "arn:aws:ec2:us-east-1:1234567890:subnet/*",
         "arn:aws:ec2:us-east-1:1234567890:network-interface/*",
         "arn:aws:ec2:us-east-1:1234567890:security-group/*",
         "arn:aws:ec2:us-east-1:1234567890:key-pair/*",
         "arn:aws:ec2:us-east-1:1234567890:instance/*",
         "arn:ec2:ec2:us-east-1:1234567890:snapshot",
         "arn:ec2:ec2:us-east-1:1234567890:elastic-gpu/*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
         "ec2:RunInstances"
      ],
      "Resource": [
         "arn:aws:ec2:us-east-1:1234567890:volume/*",
      ],
      "Condition": {
         "NumericLessThan": {
             "ec2:VolumeSize" : "X"
          }
       }
    },
    {
      "Effect": "Allow",
      "Action": [
         "ec2:CreateTags"
      ],
      "Resource": "arn:aws:ec2:us-east-1:1234567890:*/*",
      "Condition": {
         "StringEquals": {
             "ec2:CreateAction" : "RunInstances"
          }
       }
    }
  ]
}
```

After

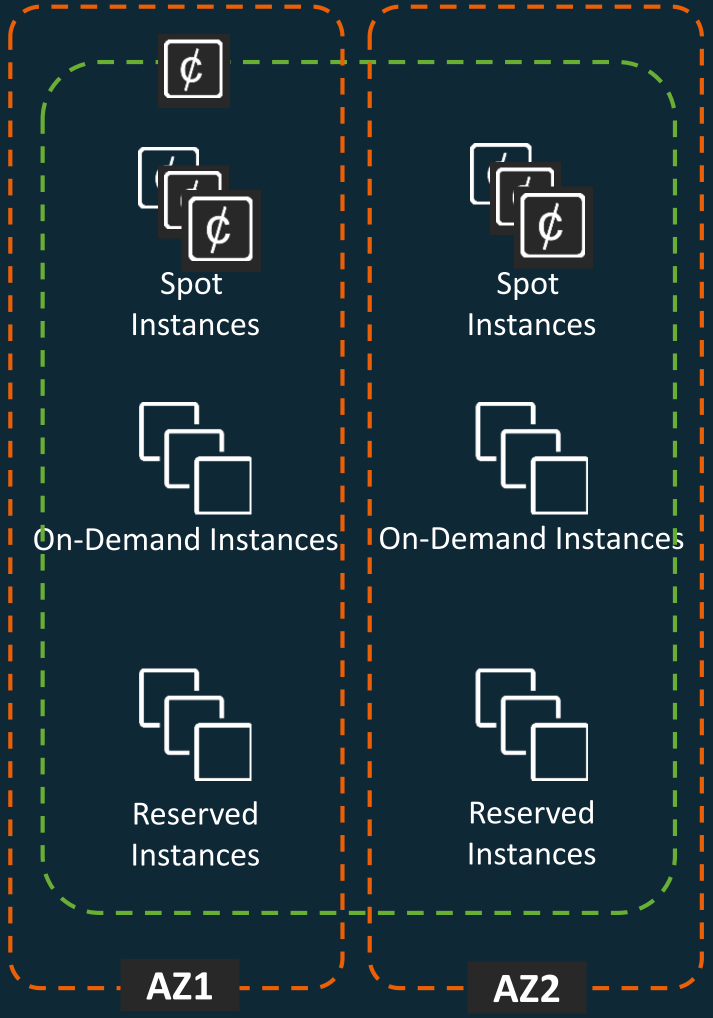
```
{
    "Version":"2012-10-17",
    "Statement":[
        {
            "Effect":"Allow",
            "Action":"ec2:RunInstances",
            "Resource":"*",
            "Condition":{
                "ArnLike":{
"ec2:LaunchTemplate":"arn:aws:ec2:region:
account:launch-template/(* or actual
template id)"
                }
            }
        }
    ]
}
```

EC2 Launch Templates - demo

EC2 Fleet

Amazon EC2 Fleet

Simplifies provisioning of EC2 capacity across different instance types, AZs, and purchase models with a single API



Use all three purchase models to optimize costs

Automatic optimization behind the scenes with software

Benefits

Reduce costs

Increase operational efficiency

Key features

Flexible capacity allocation

Massive scale

Simplified provisioning

Amazon EC2 Fleet and Allocation strategies

Amazon EC2 Fleet

- Provisions capacity across multiple instance types according to allocation strategies

Allocation strategies

- On-Demand prioritized list of instance types
- Spot Instances across the N lowest priced instance pools

Amazon EC2 Fleet API details (target capacity)

```
--target-capacity-specification
{
    "TotalTargetCapacity": integer,
    "OnDemandTargetCapacity": integer,
    "SpotTargetCapacity": integer,
    "DefaultTargetCapacityType": "spot"|"on-demand”
}
```

Amazon EC2 Fleet API details (type)

`--type`

`request`

- places an asynchronous one-time request without maintaining capacity or submitting requests in alternative capacity pools if capacity is unavailable

`maintain (default)`

- places an asynchronous request for your desired capacity, and maintains it by replenishing interrupted Spot Instances

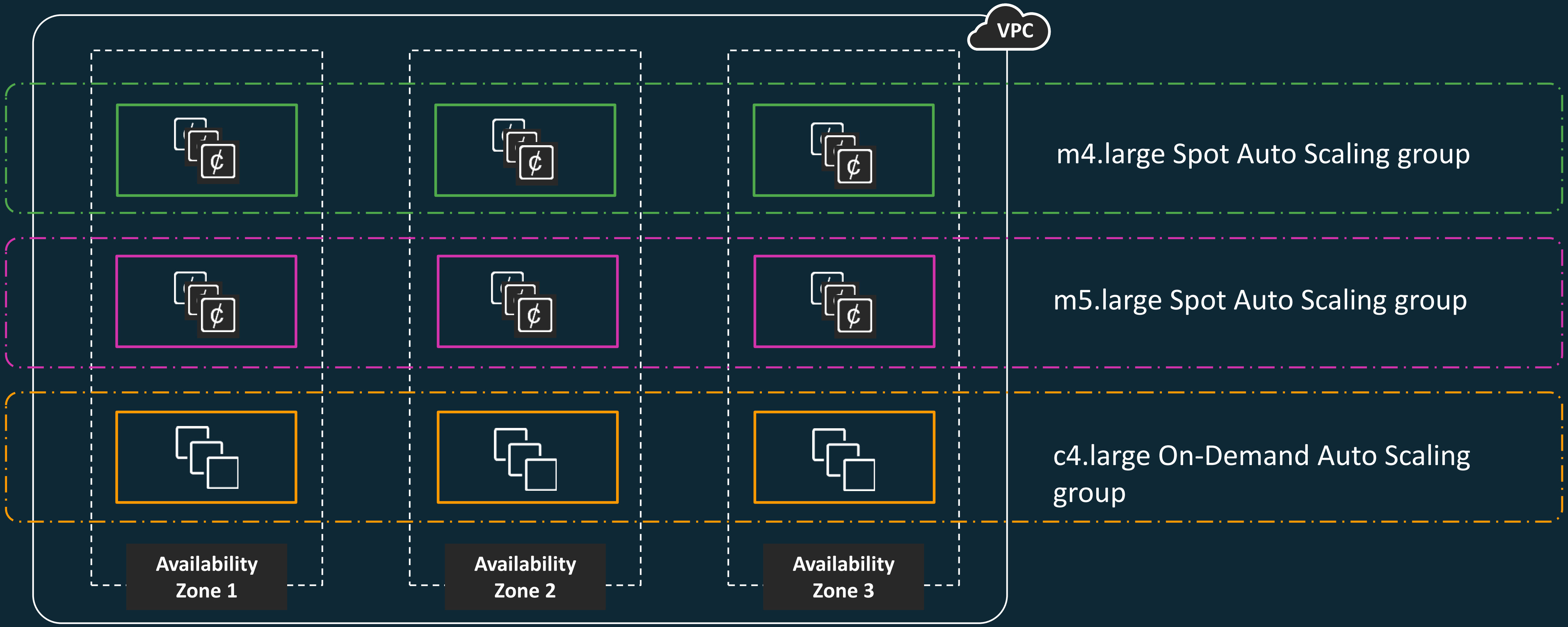
`instant (new-ish!)`

- places a synchronous one-time request, and returns errors for any instances that could not be launched

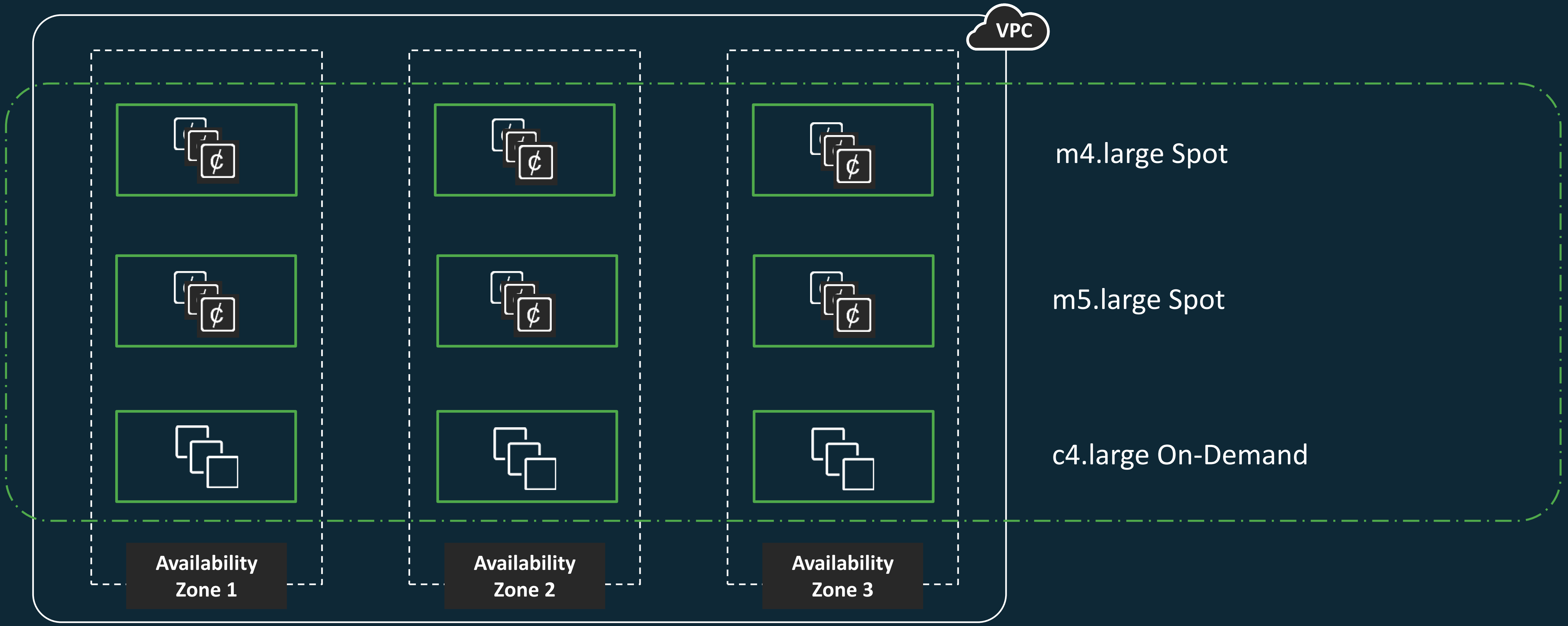
EC2 Fleet - demo

EC2 Auto Scaling (with multiple purchase options and instance types)

Before: Multiple Auto Scaling groups to use Spot, On-Demand, and RIs together


VPC
m4.large Spot Auto Scaling group
m5.large Spot Auto Scaling group
c4.large On-Demand Auto Scaling group
Availability Zone 1
Availability Zone 2
Availability Zone 3

After: Include Spot, On-Demand, and RIs in a single Auto Scaling group



Save up to 90% using EC2 Auto Scaling and EC2 Fleet

Automatically provision and scale instances across instance families and purchase models in a single Auto Scaling group

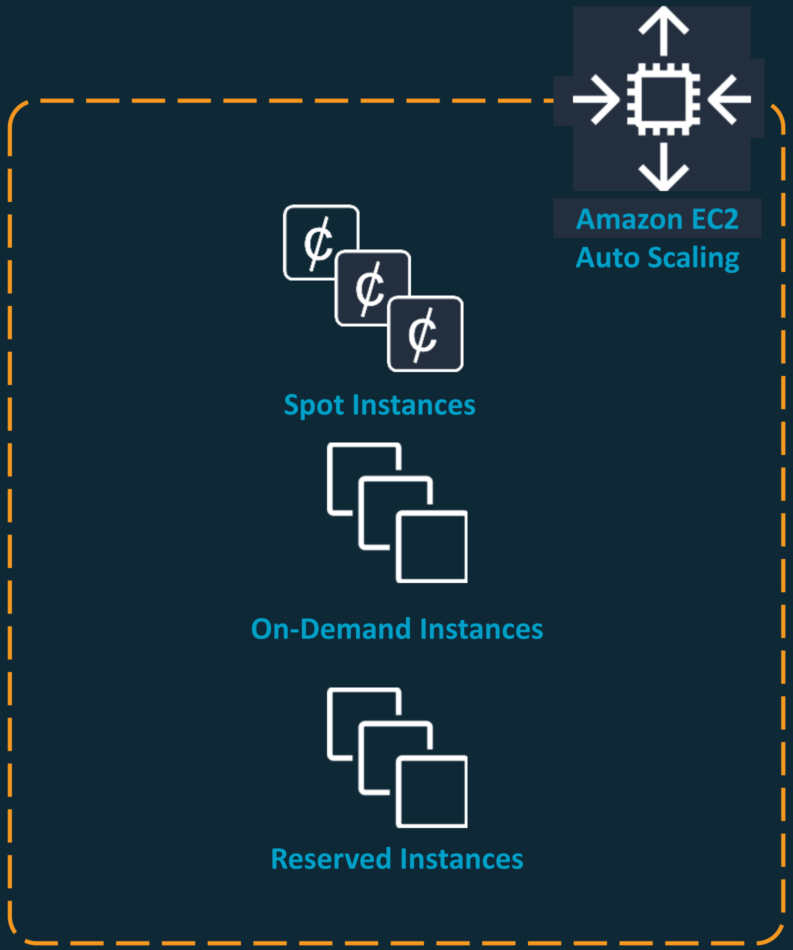
Lowest cost

Specify what percentage of your Auto Scaling group capacity should be fulfilled by On-Demand Instances and Spot Instances to optimize cost

Prioritized list

Use a prioritized list for On-Demand Instance types to scale capacity during an urgent, unpredictable event to optimize performance

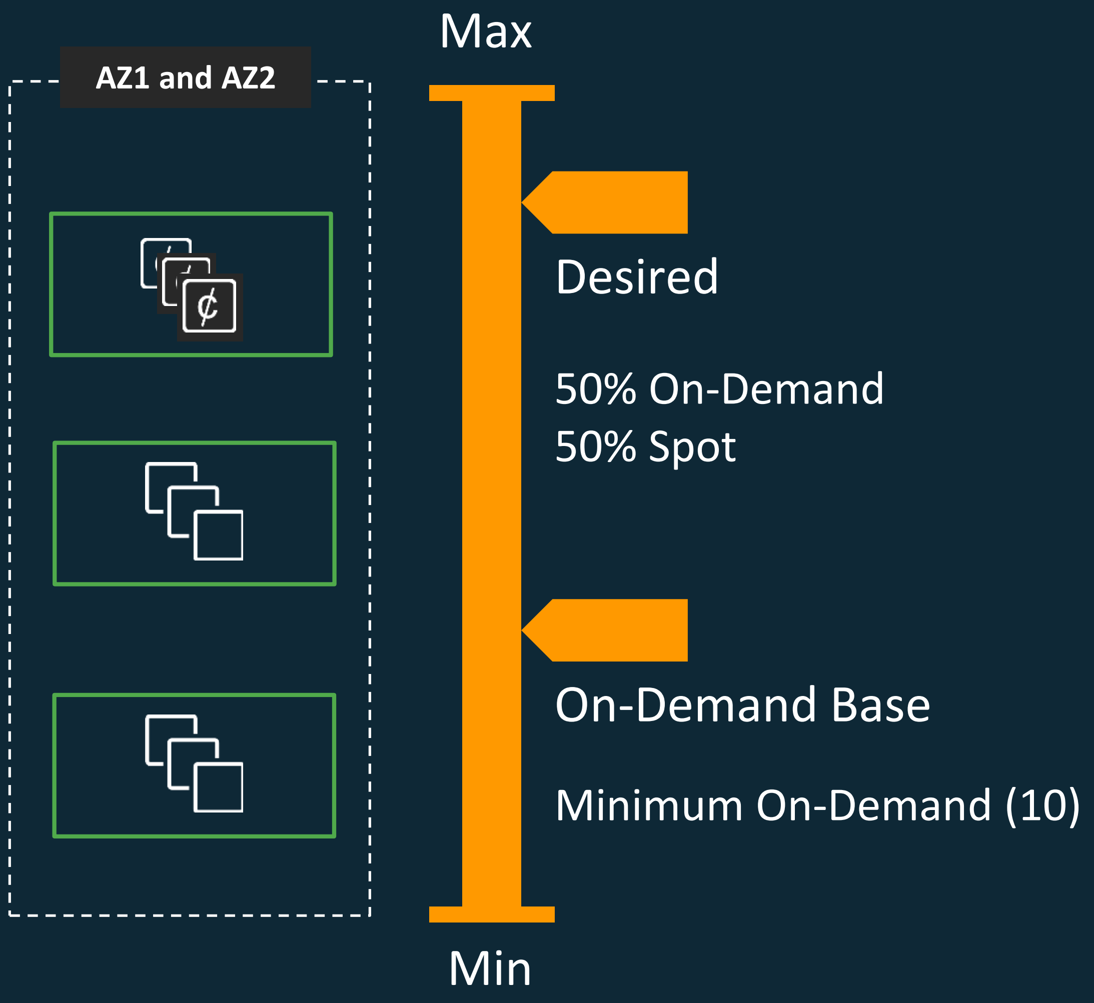
Reduce operational overhead



Reduce cost | Optimize performance | Reduce operational overhead

API Parameters

```
"MixedInstancesPolicy": {
    "LaunchTemplate": {
       "LaunchTemplateSpecification": {
         "LaunchTemplateName": ”MyLaunchTemplate"
       },
       "Overrides": [
         { "InstanceType": "c5.large" },
         { "InstanceType": "c4.large" }
       ]
     },
     "InstancesDistribution": {
       "OnDemandAllocationStrategy": "prioritized",
       "OnDemandBaseCapacity": 10,
       "OnDemandPercentageAboveBaseCapacity": 50,
       "SpotAllocationStrategy": "lowest-price",
       "SpotInstancePools": 2
     }
}
```



Instances distribution

Auto Scaling group scales up and down according to the instances distribution

- EC2 Auto Scaling fills base capacity with On-Demand Instances
- Capacity beyond base capacity is fulfilled with Spot or On-Demand Instances according to percentage split

Example:

OnDemandBaseCapacity: 10

OnDemandPercentageAboveBaseCapacity: 50

	Desired Capacity	On-Demand Instances	Spot Instances
Desired < Base	5	5	0
Desired = Base	10	10	0
Desired > Base	20	10 + 5	5

Recommendations on Mixed Instances Policy

Choose at least 2 instance type overrides

- Improves availability for On-Demand and Spot Instances

Diversify across at least N = 2 Spot Instance pools

- Reduces risk from fluctuations in Spot capacity and prices

Choose instance types of same size across families

- Maintains stability as dynamically scale up and down

Use default spot max price

- Leverages spot cost savings while defaulting to on-demand price as maximum price to pay

On-Demand and Spot Instance pools

Instances Distribution

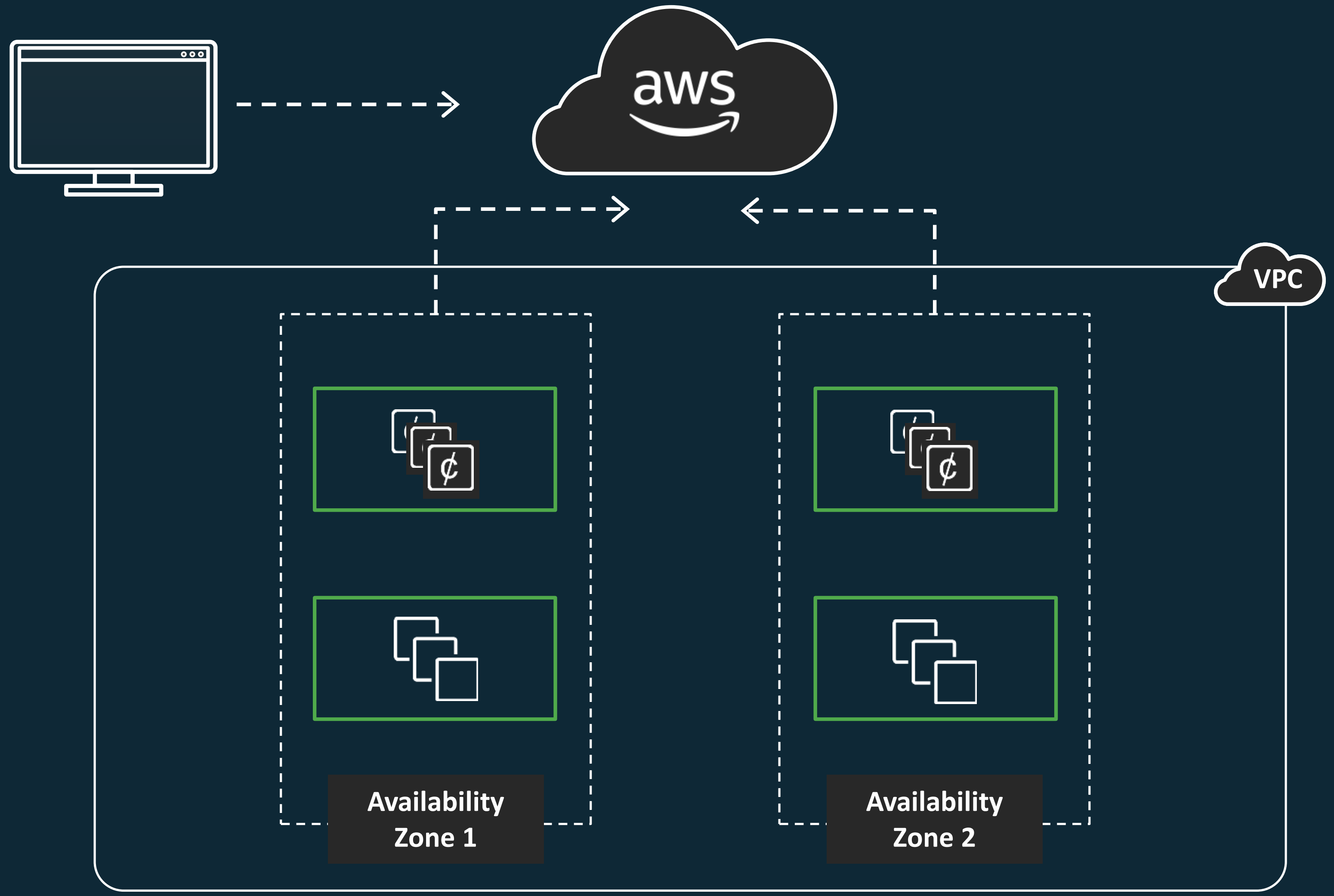
- Distribute instances evenly between Availability Zones for On-demand and Spot separately

Launch Failures

- Launch instances in another Availability Zone when launches fail for On-Demand and Spot separately

Allows On-Demand capacity to be balanced while Spot capacity is migrated to another Availability Zone due to low capacity

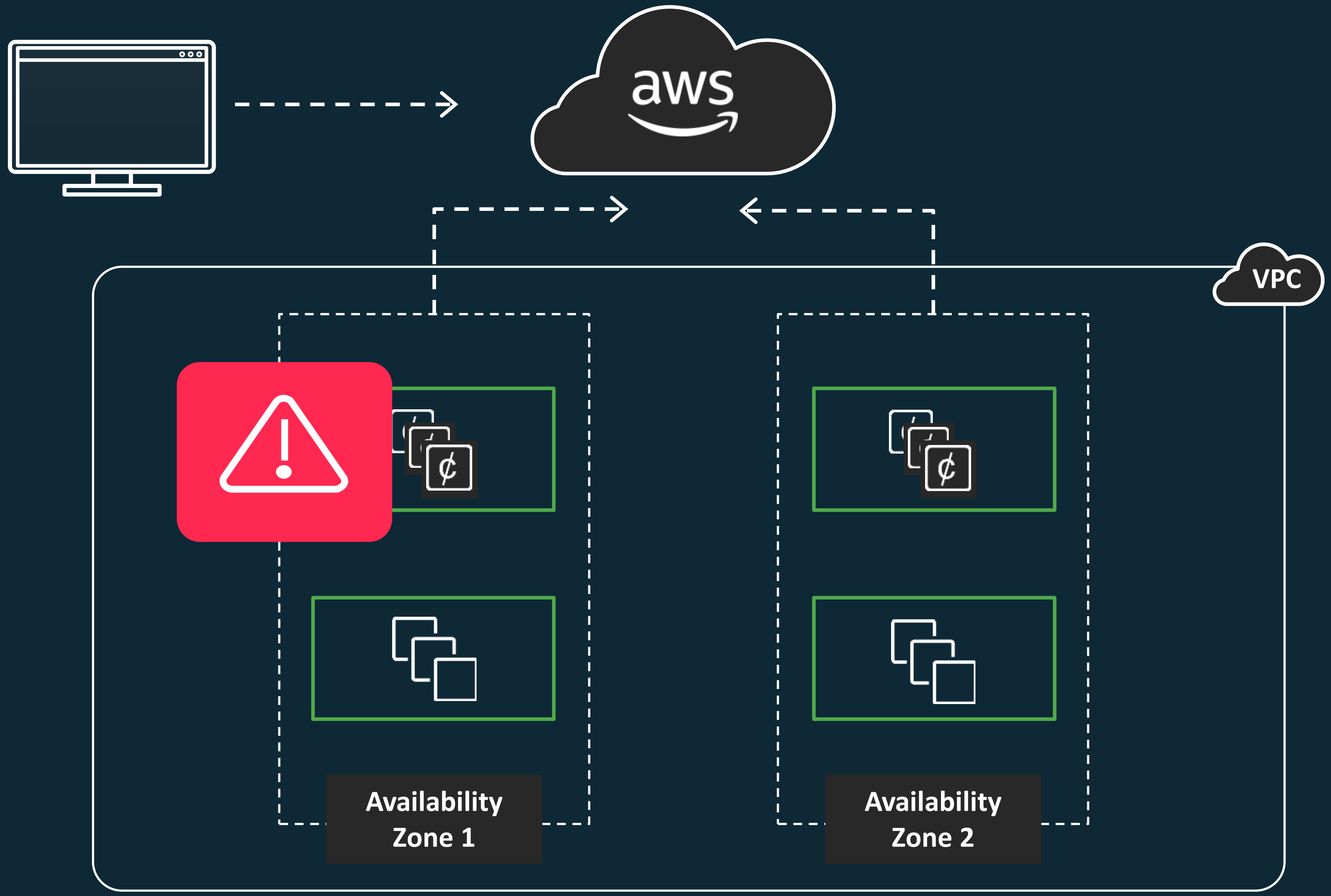
On-Demand and Spot Instance pools



Balanced capacity across Spot Instances

Balanced capacity across On-Demand Instances

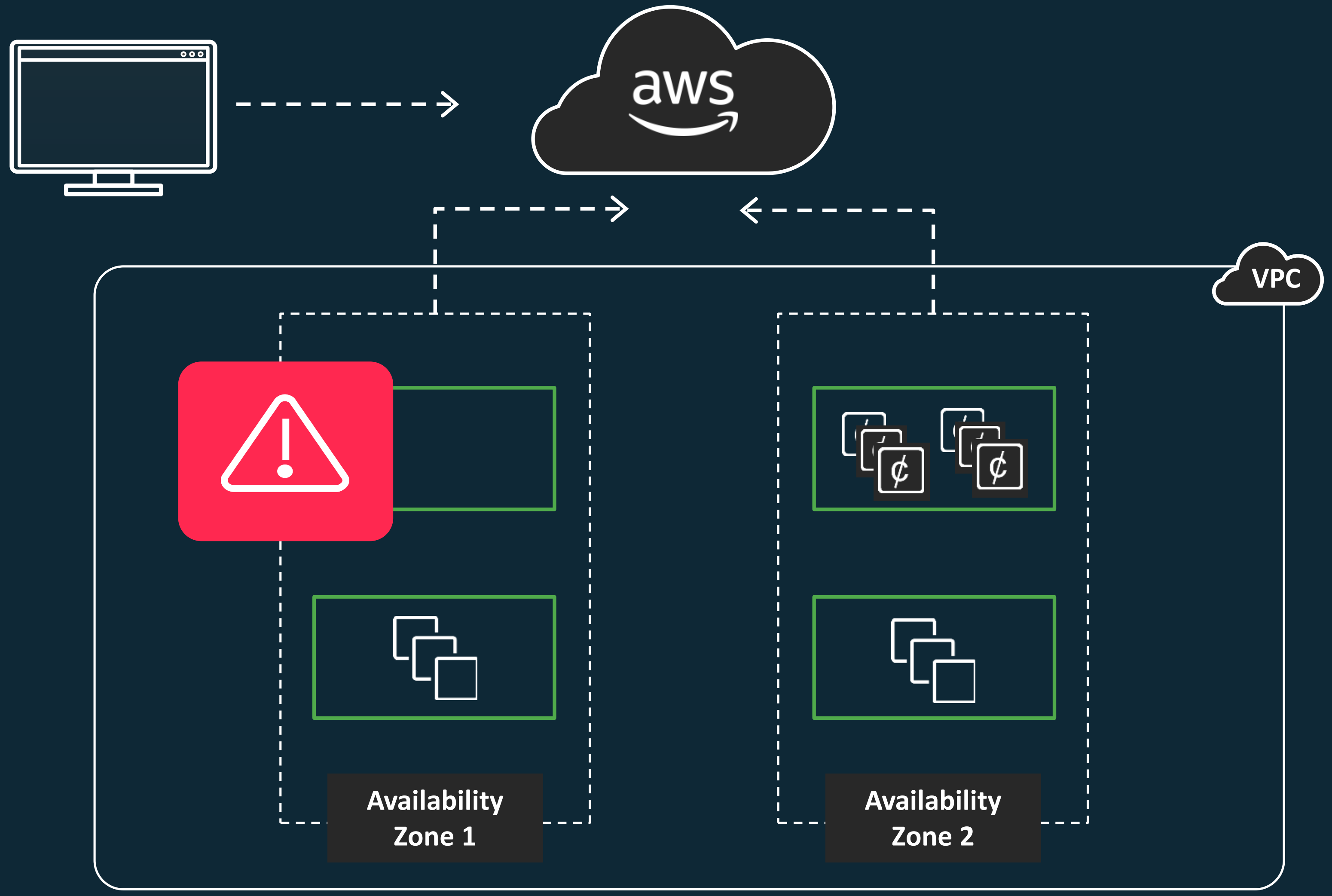
On-Demand and Spot Instance pools



Spot Instances are unavailable in AZ1

Balanced capacity across On-Demand Instances

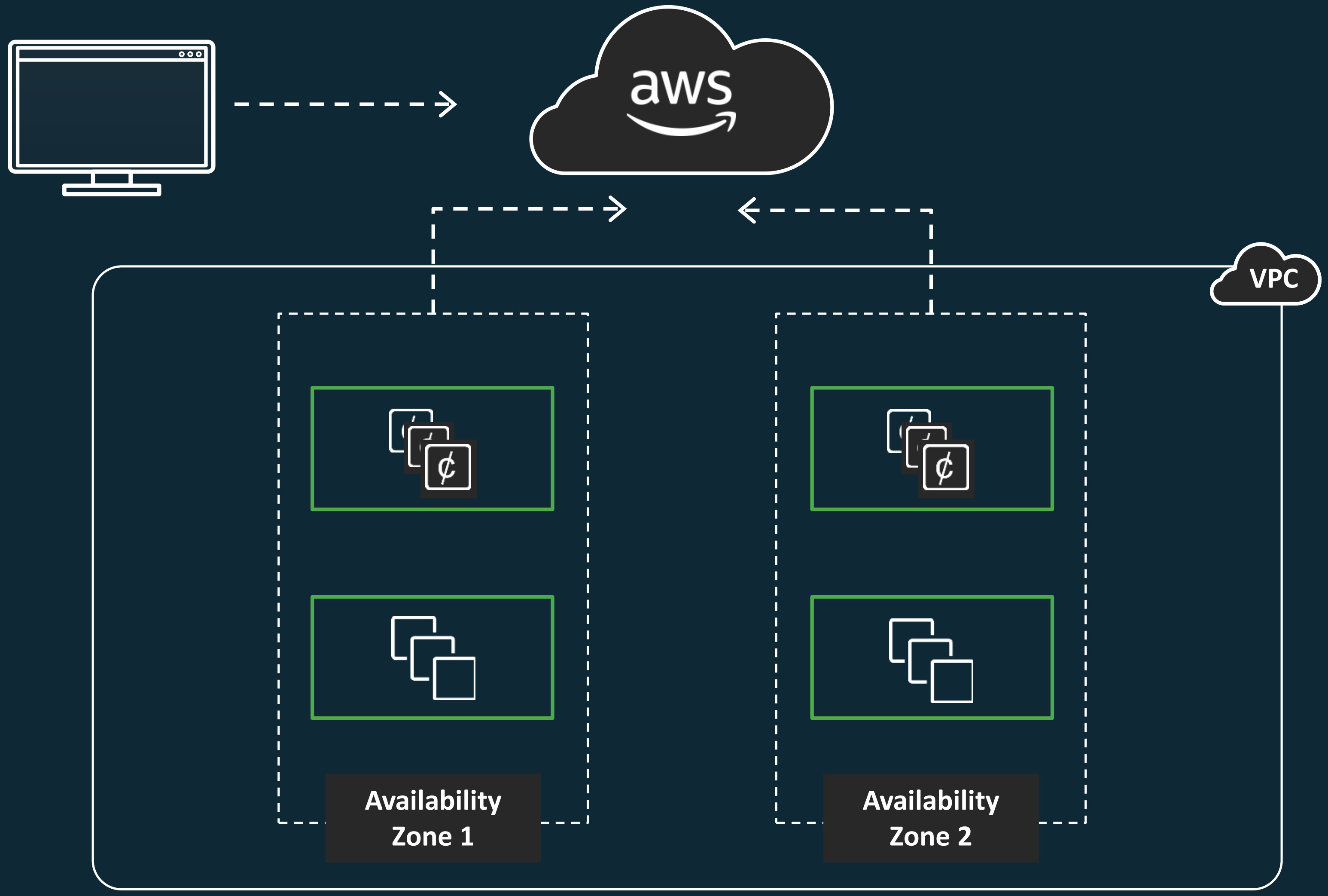
On-Demand and Spot Instance pools



Spot Instances are retargeted to AZ2

Balanced capacity across On-Demand Instances

On-Demand and Spot Instance pools



Balanced capacity across Spot Instances

Balanced capacity across On-Demand Instances

EC2 Auto Scaling (with multiple purchase options and instance types) - demo

Thank you!

Please remember to fill out the survey

https://aws.amazon.com/ec2/spot/getting-started/

@schmutze

@BoydMcgeachie