Advanced Task Scheduling with Amazon ECS and Blox

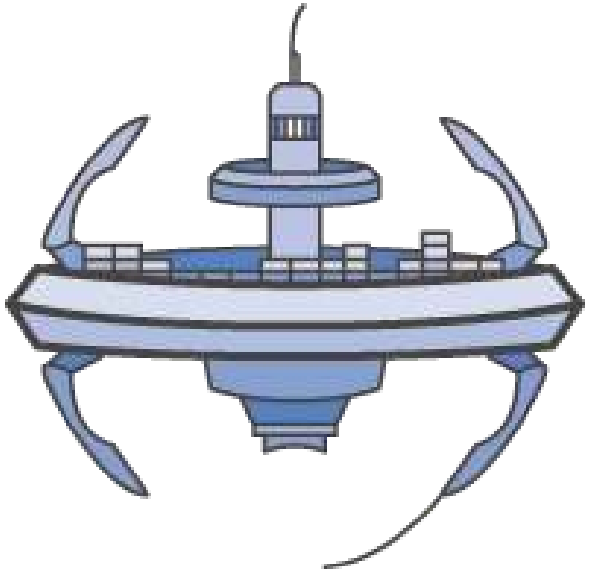
Julien Simon
Principal Technical Evangelist, AWS
@julsimon

aws

Docker on Amazon Web Services

Amazon EC2 Container Service (ECS)

- https://aws.amazon.com/ecs/
- Launched in 04/2015
- No additional charge
- Since June: integration with Spot instances
- October 2nd: per-second billing for EC2 and EBS



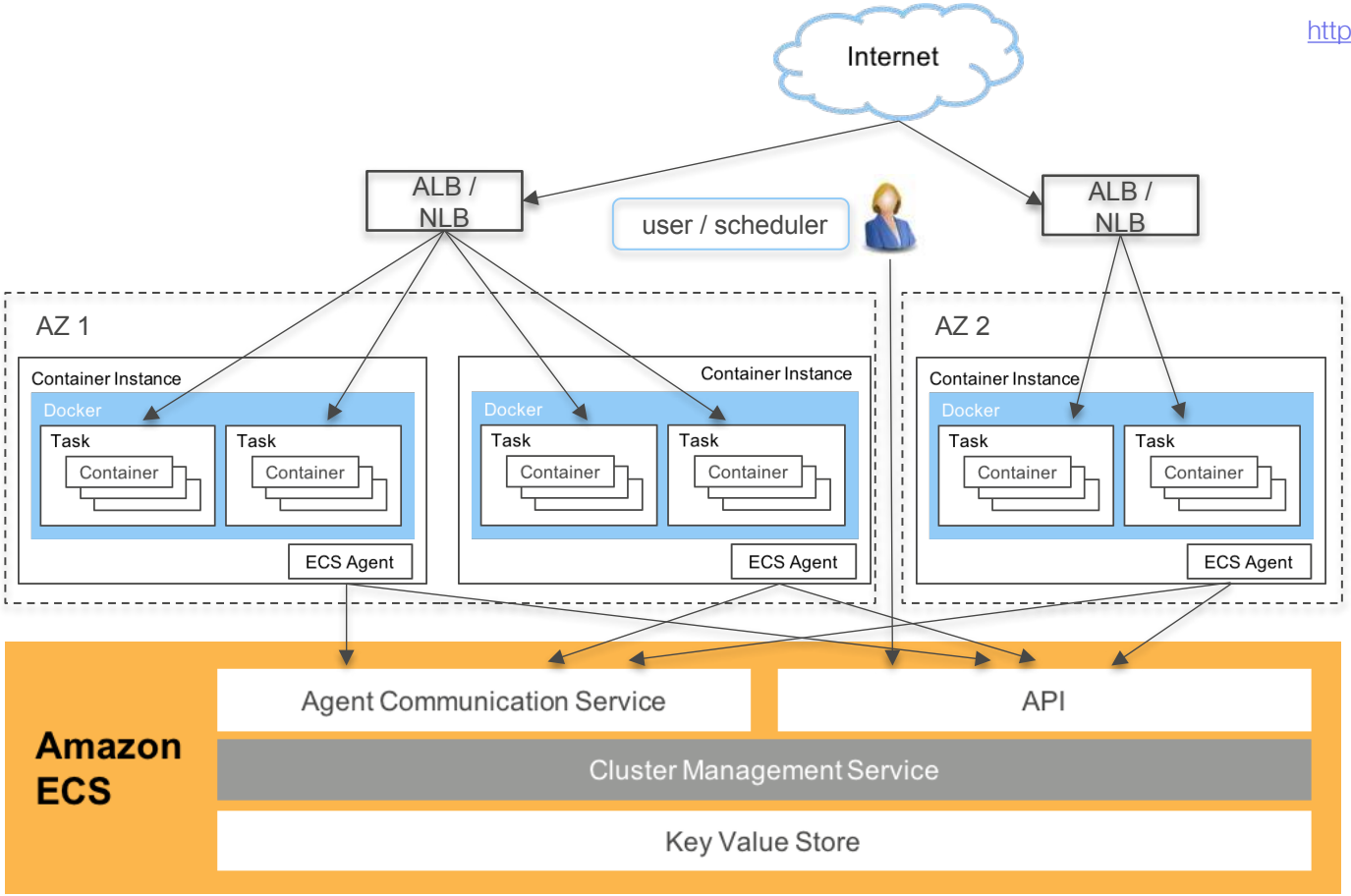
Amazon EC2 Container Registry (ECR)

- https://aws.amazon.com/ecr/
- Launched in 12/2015
- Free tier: 500MB / month for a year
- $0.10 / GB / month + outgoing traffic

ECS & ECR are available in 12 regions (US, EU, APAC, China)

Amazon ECS: Under the Hood

https://github.com/aws/amazon-ecs-agent




ENGINE
START

Selected ECS customers

Case studies and re:Invent videos: https://aws.amazon.com/ecs/

Container Partners

https://aws.amazon.com/containers/partners/

ECS Scheduling

The problem

Given a certain amount of computing power and memory,

how can we best manage an arbitrary number of apps running in Docker containers?

http://tidalseven.com

Case study: Coursera

https://www.youtube.com/watch?v=a45J6xAGUvA

Coursera deliver Massive Open Online Courses (14 million students, 1000+ courses). Their platform runs a large number of batch jobs, notably to grade programming assignments. Grading jobs need to run in near-real time while preventing execution of untrusted code inside the Coursera platform.

After trying out some other Docker solutions, Coursera have picked Amazon ECS and have even written their own scheduler.

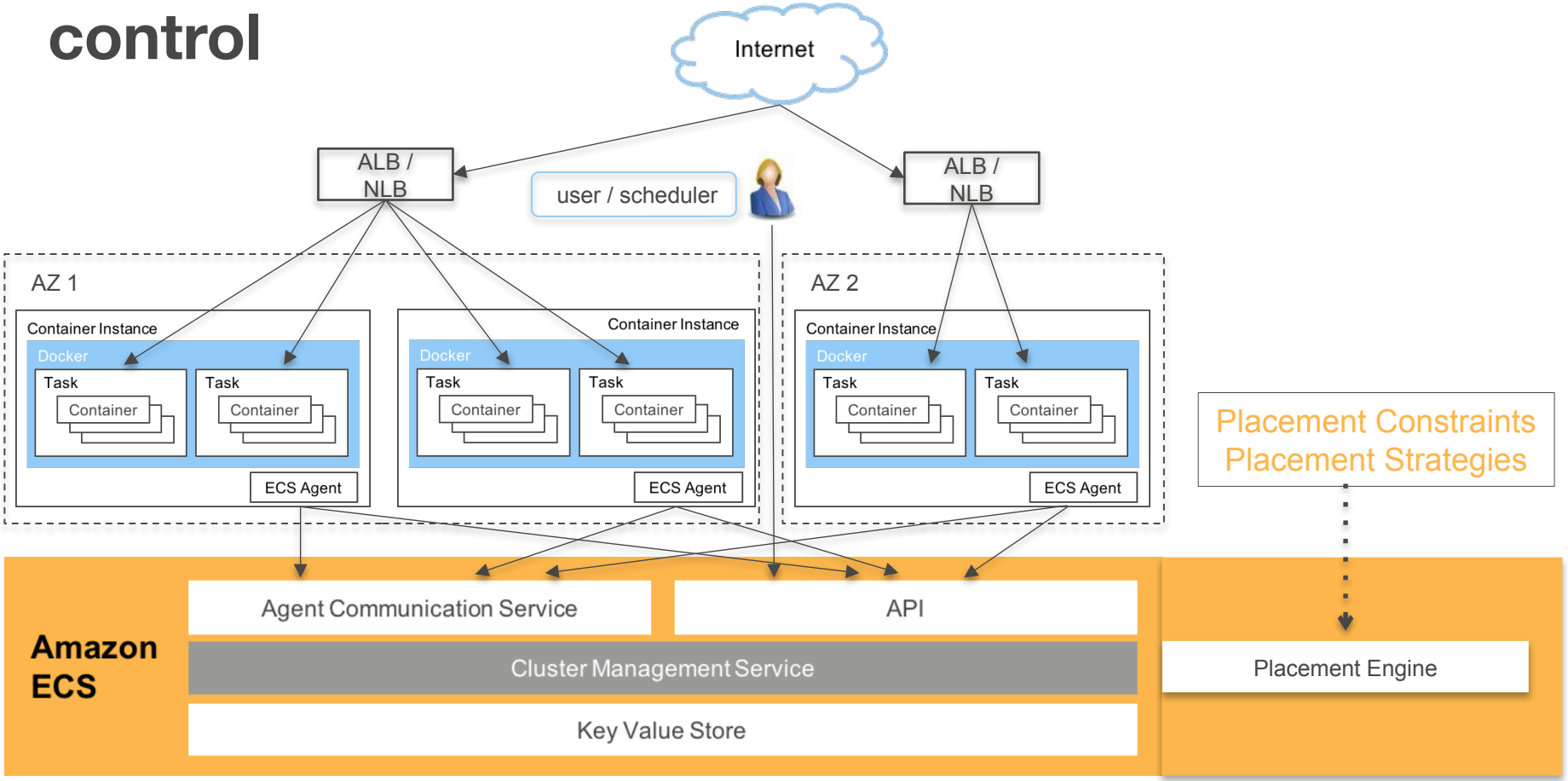
"*Amazon ECS enabled Coursera to focus on releasing new software rather than spending time managing clusters*" - Frank Chen, Software Engineer

Scheduling on ECS: two options so far

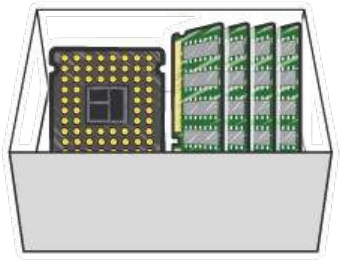
1. Let ECS handle scheduling through Services
    - Task Definition
        - ECS equivalent of the Docker Compose file
        - Versioned
        - `cpu_shares`, `mem_limit`
    - Number of containers
2. Implement a custom scheduler with the ECS API
    - Describe cluster state
    - Select a specific ECS instance according to custom logic
    - Run task on this instance

ECS Placement Engine

Placement Engine: giving developers more control



Placement Constraints



	Name	Example
✓	AMI ID	attribute:ecs.ami-id == ami-eca289fb
✓	Availability Zone	attribute:ecs.availability-zone == us-east-1a
✓	Instance Type	attribute:ecs.instance-type == t2.small
✓	Distinct Instances	type="distinctInstance"
✓	Custom	attribute:stack == prod

Example: Constraint on Instance Family/Type

```
aws ecs list-container-instances --cluster ecs-demo --filter "attribute:ecs.instance-type matches t2.*"
{
    "containerInstanceArns": [
        "arn:aws:ecs:us-east-1:123456789000:container-instance/3ced5d42-537c-40b4-9551-b9022cc13b78",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/442d988b-4b00-40bf-85ae-34e0819454f2",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/5de11ede-6c22-411e-a469-8301eeebae0f",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/6bfb12c8-1c3c-4d4a-976c-ce3c2c79b031",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/7eb87781-abab-4a6a-9a0d-602a4da59549",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/af8d48ba-73c4-409a-b40f-66596aa86c5d",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/b5c08e3e-bd25-4ec9-9a88-ce1d53640542",
    ]
}
```

```
aws ecs list-container-instances --cluster ecs-demo --filter "attribute:ecs.instance-type matches t2.small"
{
    "containerInstanceArns": [
        "arn:aws:ecs:us-east-1:123456789000:container-instance/af8d48ba-73c4-409a-b40f-66596aa86c5d",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/b5c08e3e-bd25-4ec9-9a88-ce1d53640542",
    ]
}
```

Example: Constraint on Availability Zone

```
aws ecs list-container-instances --cluster ecs-demo --filter "attribute:ecs.availability-zone matches us-east-1.*"
{
    "containerInstanceArns": [
        "arn:aws:ecs:us-east-1:123456789000:container-instance/3ced5d42-537c-40b4-9551-b9022cc13b78",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/442d988b-4b00-40bf-85ae-34e0819454f2",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/5de11ede-6c22-411e-a469-8301eeebae0f",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/6bfb12c8-1c3c-4d4a-976c-ce3c2c79b031",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/7eb87781-abab-4a6a-9a0d-602a4da59549",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/af8d48ba-73c4-409a-b40f-66596aa86c5d",
    ]
}
```

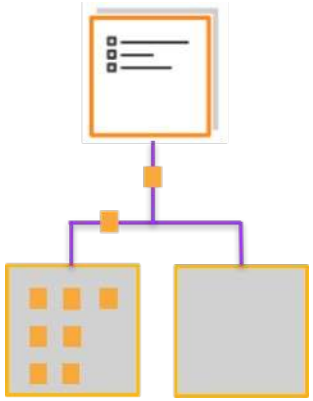
```
aws ecs list-container-instances --cluster ecs-demo --filter "attribute:ecs.availability-zone == us-east-1a"
{
    "containerInstanceArns": [
        "arn:aws:ecs:us-east-1:123456789000:container-instance/3ced5d42-537c-40b4-9551-b9022cc13b78",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/442d988b-4b00-40bf-85ae-34e0819454f2",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/5de11ede-6c22-411e-a469-8301eeebae0f",
    ]
}
```

Example: Combining Multiple Constraints

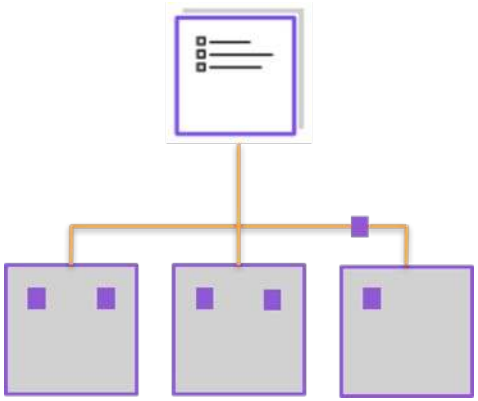
```
aws ecs list-container-instances --cluster ecs-demo --filter "attributes:ecs.instance-type matches t2.* and
attribute:ecs.availability-zone == us-east-1a"
{
    "containerInstanceArns": [
        "arn:aws:ecs:us-east-1:123456789000:container-instance/3ced5d42-537c-40b4-9551-b9022cc13b78",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/442d988b-4b00-40bf-85ae-34e0819454f2",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/5de11ede-6c22-411e-a469-8301eeebae0f",
    ]
}
```

```
aws ecs list-container-instances --cluster ecs-demo --filter "(attribute:ecs.instance-type in [t2.small, t2.medium]
or attribute:ecs.instance-type matches g2.*) and attribute:ecs.availability-zone != us-east-1d"
{
    "containerInstanceArns": [
        "arn:aws:ecs:us-east-1:123456789000:container-instance/3ced5d42-537c-40b4-9551-b9022cc13b78",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/442d988b-4b00-40bf-85ae-34e0819454f2",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/5de11ede-6c22-411e-a469-8301eeebae0f",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/6bfb12c8-1c3c-4d4a-976c-ce3c2c79b031",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/7eb87781-abab-4a6a-9a0d-602a4da59549",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/d45f5b92-4faa-44a9-a9a7-2d744566e510",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/f3d92b17-7d95-4cff-b623-390e871c6b60",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/f7858158-5806-4d8d-82ea-f0eb4680e6cf",
        "arn:aws:ecs:us-east-1:123456789000:container-instance/fc751042-590a-440d-90a7-e8ebce02d234"
    ]
}
```

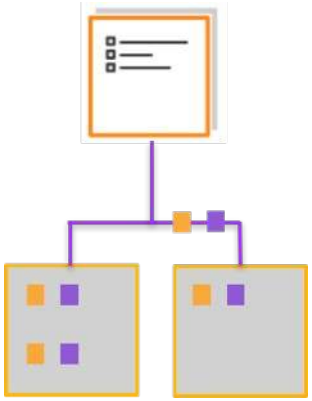
Placement Strategies



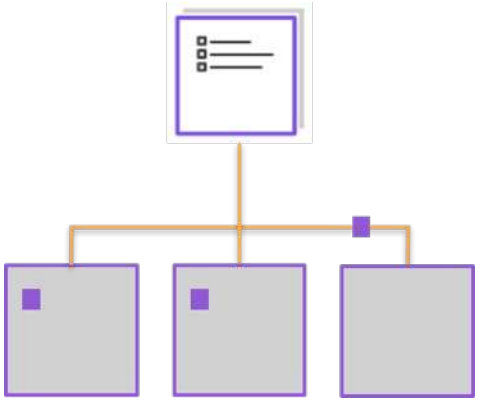
Binpacking



Spread

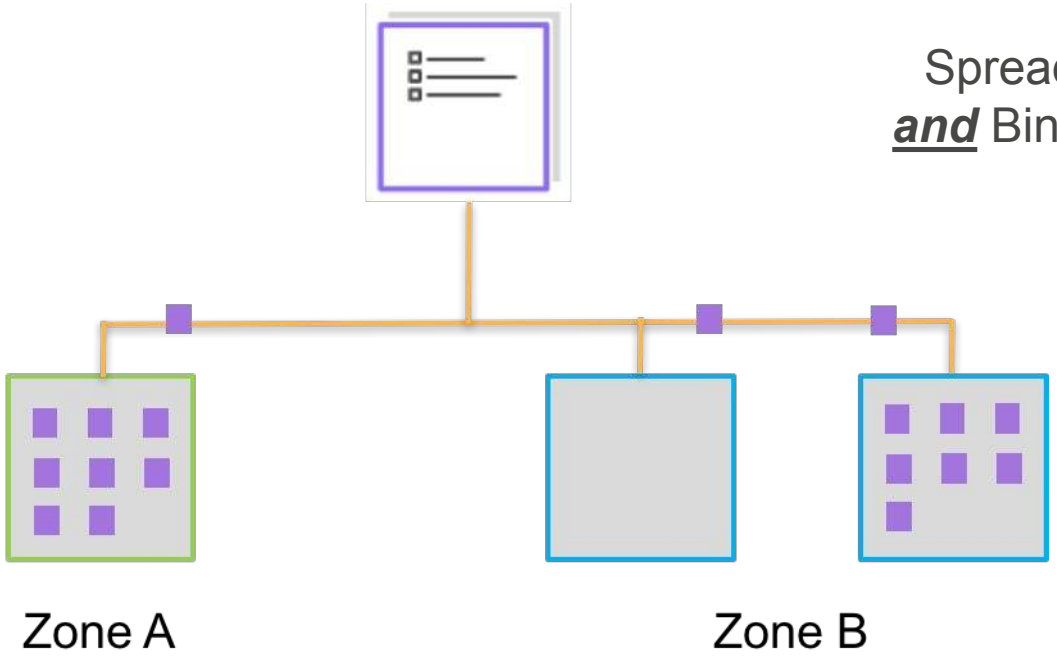


Affinity



Distinct Instance

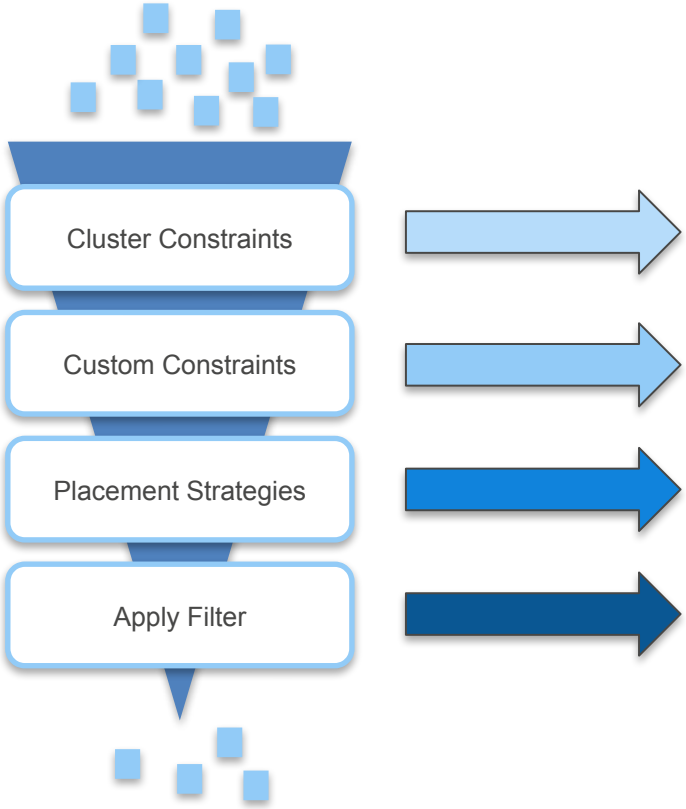
Placement Strategy Chaining



Spread tasks across Zones
and Binpack within each Zone

Placing Tasks

Anatomy of Task Placement



Satisfy CPU, memory, and port requirements

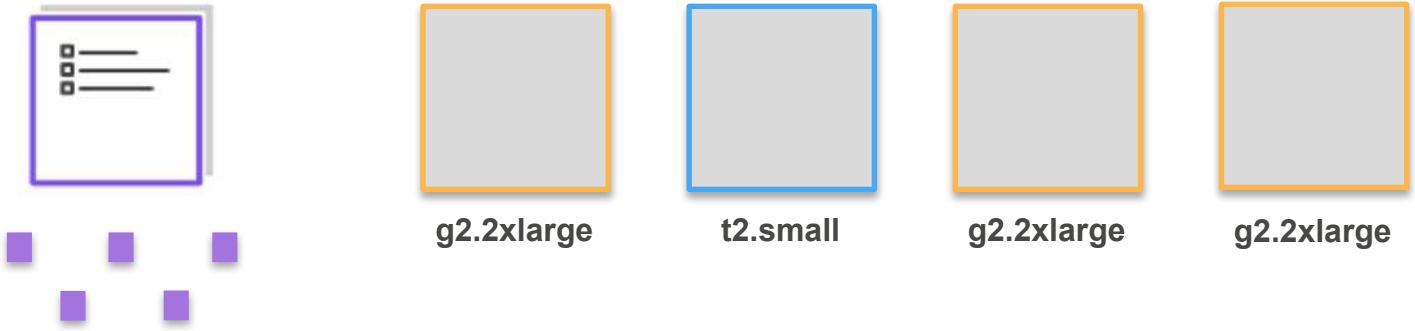
Filter for location, instance-type, AMI, or custom attribute constraints

Identify instances that meet spread or binpack placement strategy

Select final container instances for placement

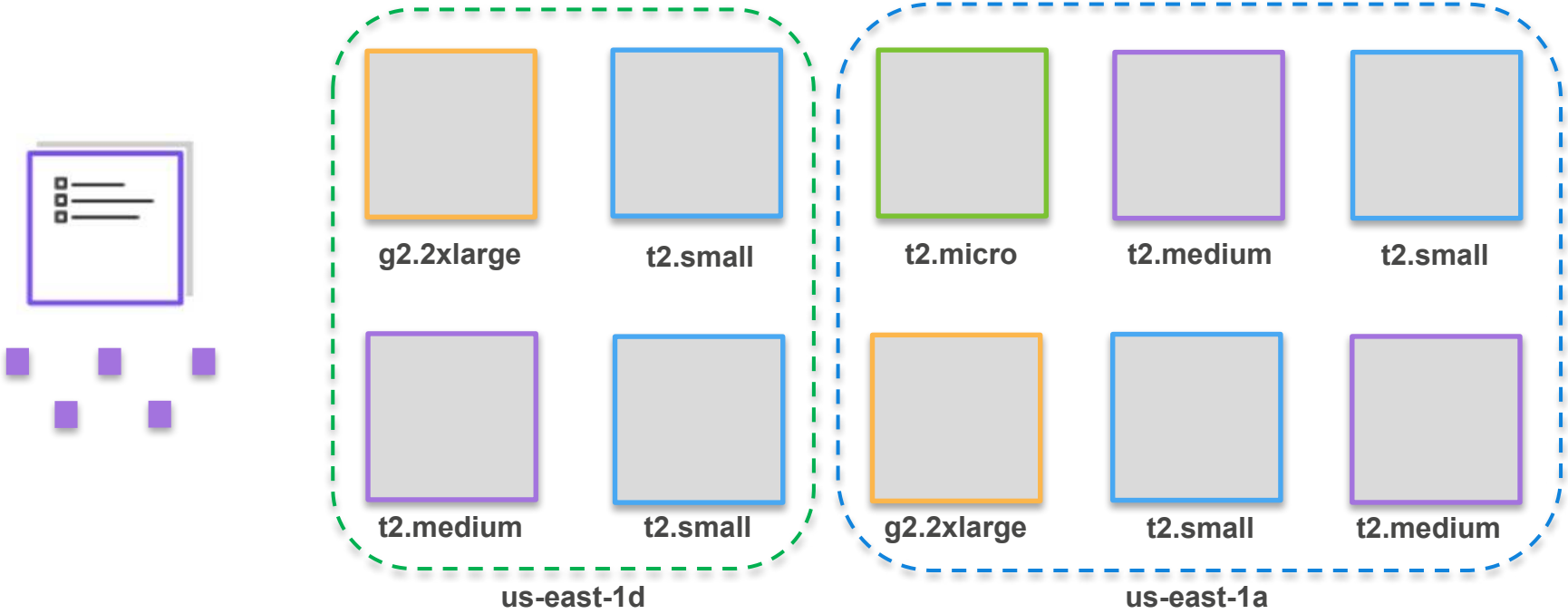
Placement: Targeting Instance Type

```
aws ecs run-task --cluster ecs-demo --task-definition myapp --count 5 --placement-constraints
type="memberOf",expression="attribute:ecs.instance-type == g2.2xlarge"
```



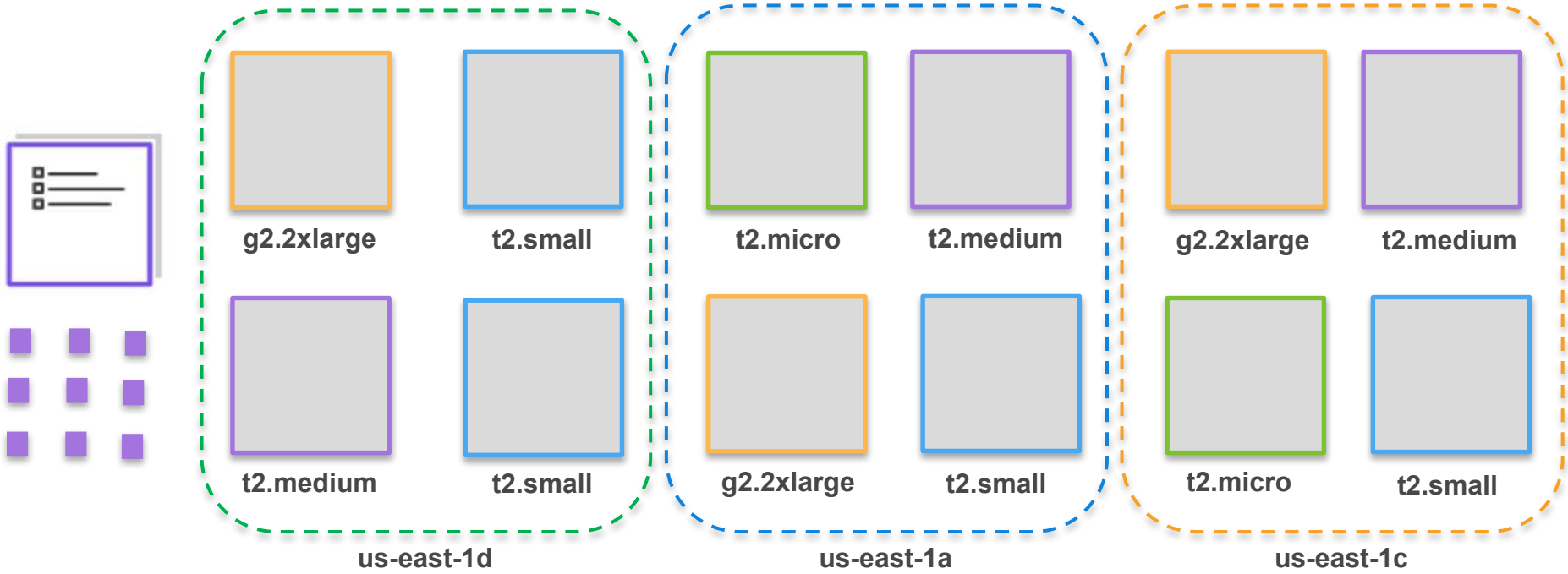
Placement: Targeting Instance Type & Zone

```
aws ecs run-task --cluster ecs-demo --task-definition myapp --count 5 --placement-constraints
type="memberOf",expression="(attribute:ecs.instance-type == t2.small or
attribute:ecs.instance-type == t2.medium) and attribute:ecs.availability-zone != us-east-1d"
```



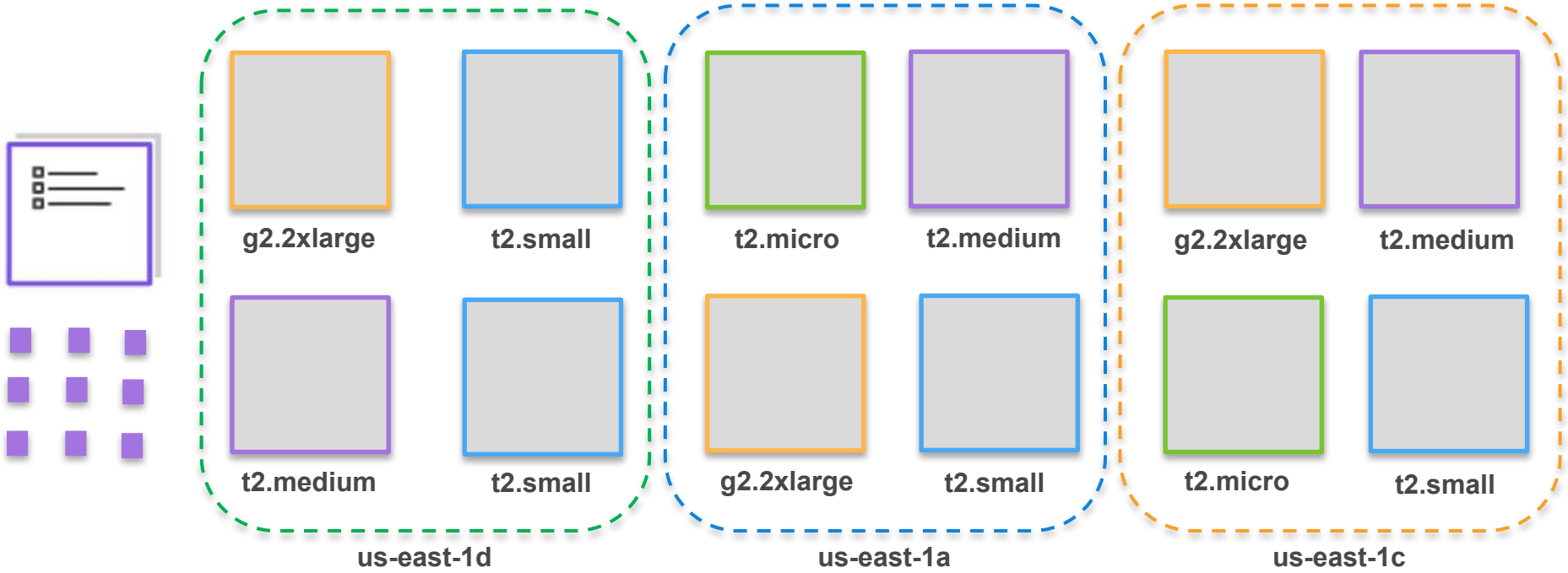
Placement: Availability Zone Spread

```
aws ecs run-task --cluster ecs-demo --task-definition myapp --count 9 --placement-strategy
type="spread",field="attribute:ecs.availability-zone"
```



Placement: Spread across Zone and Binpack

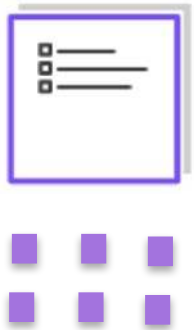
```
aws ecs run-task --cluster ecs-demo --task-definition myapp --count 9 --placement-strategy
type="spread",field="attribute:ecs.availability-zone" type="binpack",field="memory"
```

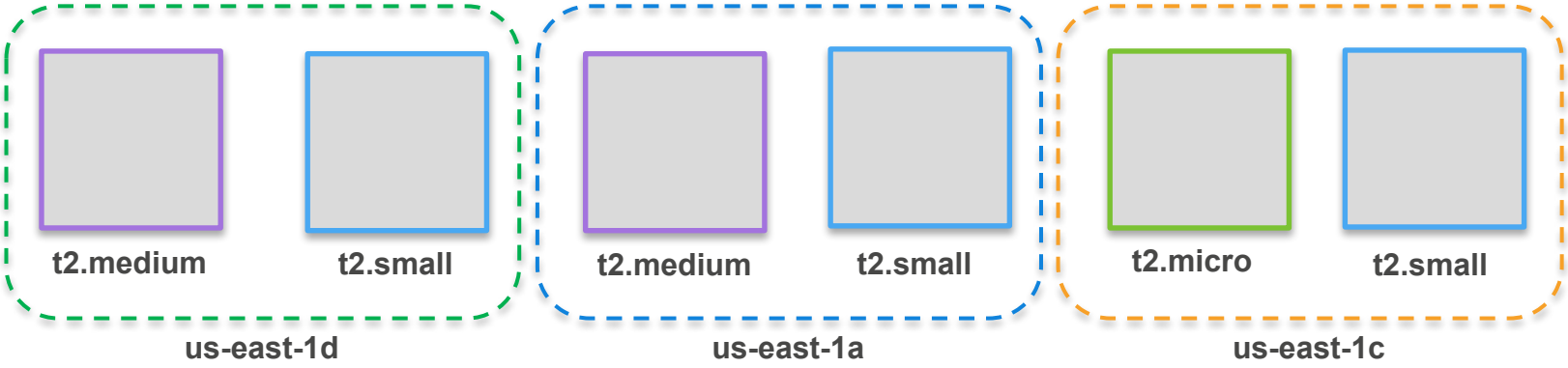


Placement: Multiple Services on a Cluster

```
aws ecs create-service --service-name srvc-binpk --cluster ecs-demo --task-definition myapp-binpk
--desired-count 5 --placement-strategy type="binpack",field="memory"
```

```
aws ecs create-service --service-name srvc-spread --cluster ecs-demo --task-definition myapp-spread
--desired-count 6 --placement-strategy type="spread",field="attribute:ecs.availability-zone"
```

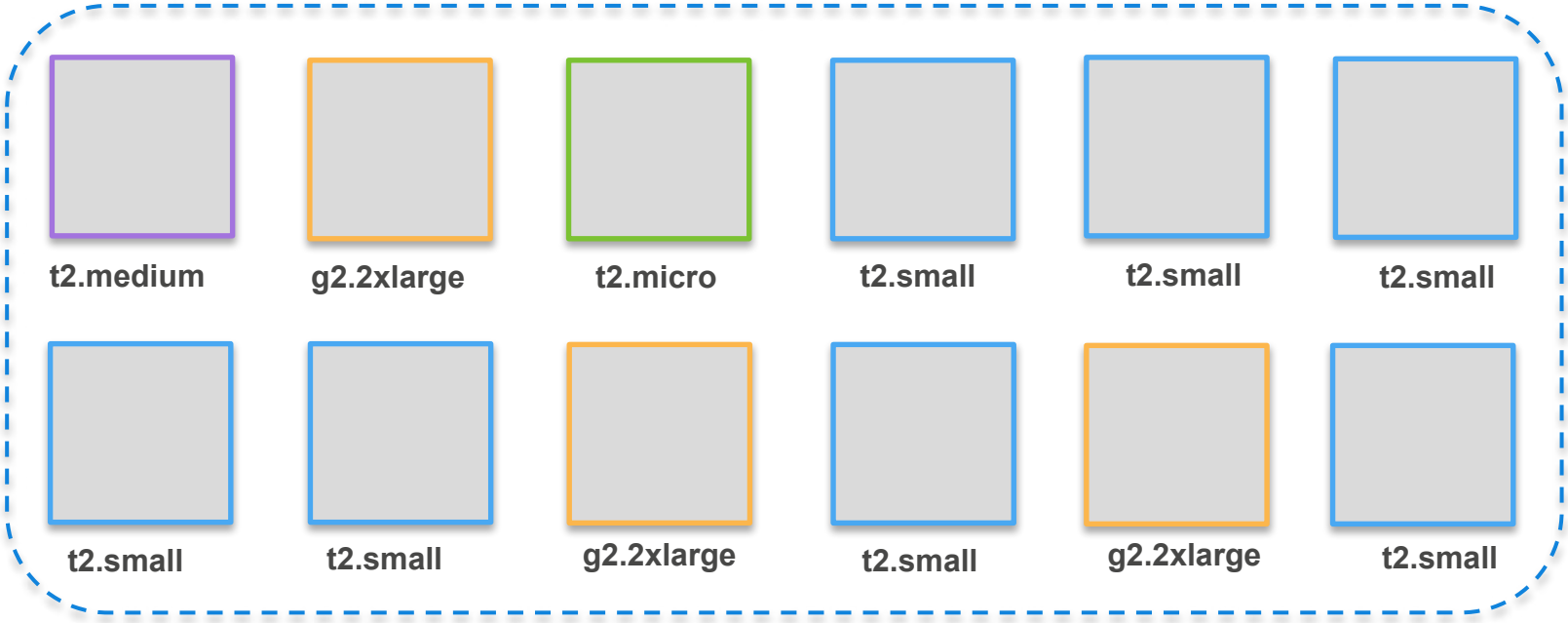




Placement: Services – Distinct Instances

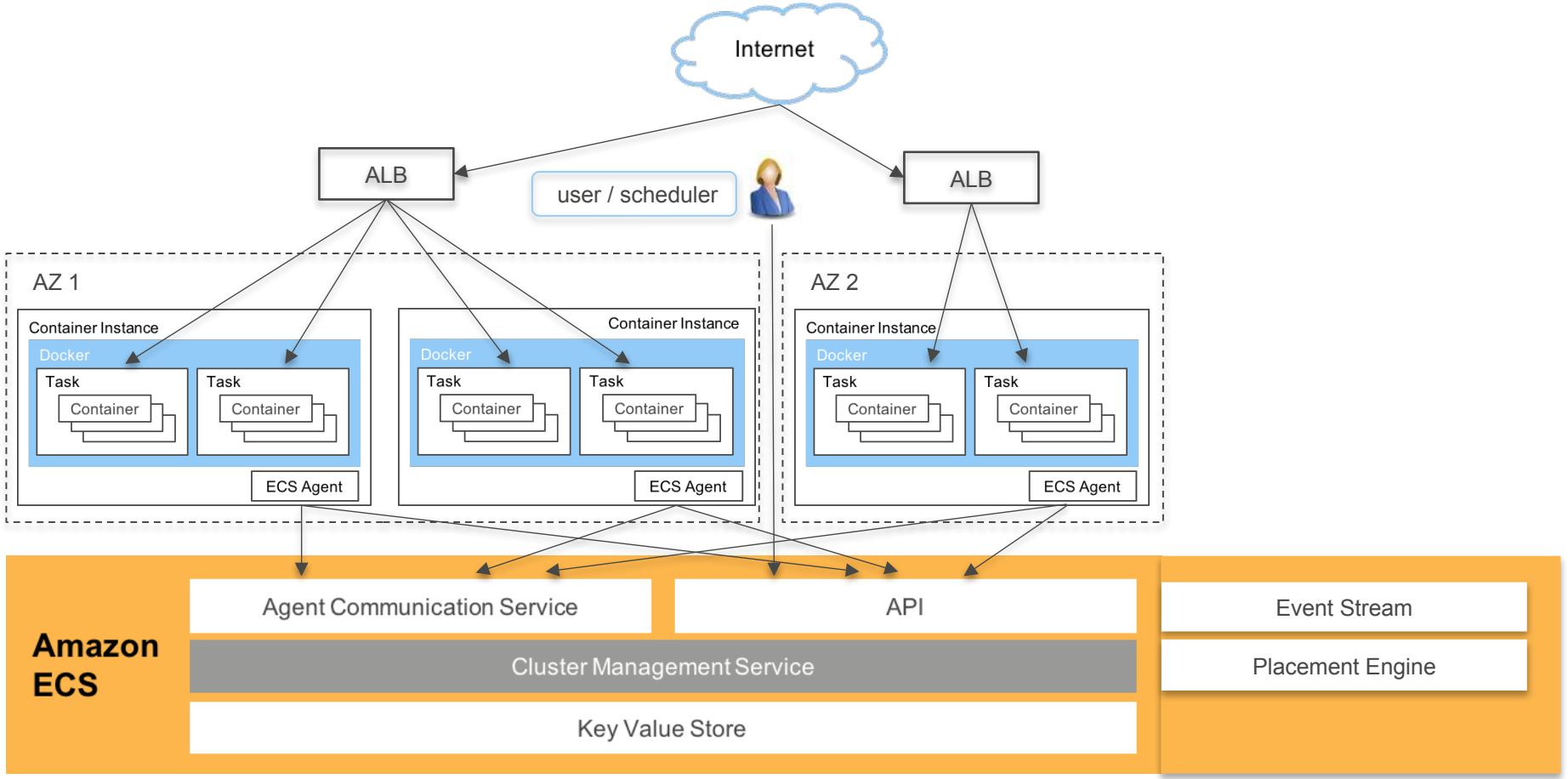
```
aws ecs create-service --service-name myapp-gpu --cluster ecs-demo --task-definition myapp-gpu --desired-count 3 --placement-constraints type="memberOf",expression="attribute:ecs.instance-type =~ g2.*" type="distinctInstance"
```

```
aws ecs create-service --service-name myapp --cluster ecs-demo --task-definition myapp --desired-count 9 --placement-constraints type="memberOf",expression="(attribute:ecs.instance-type == t2.small or attribute:ecs.instance-type == t2.medium)" type="distinctInstance"
```

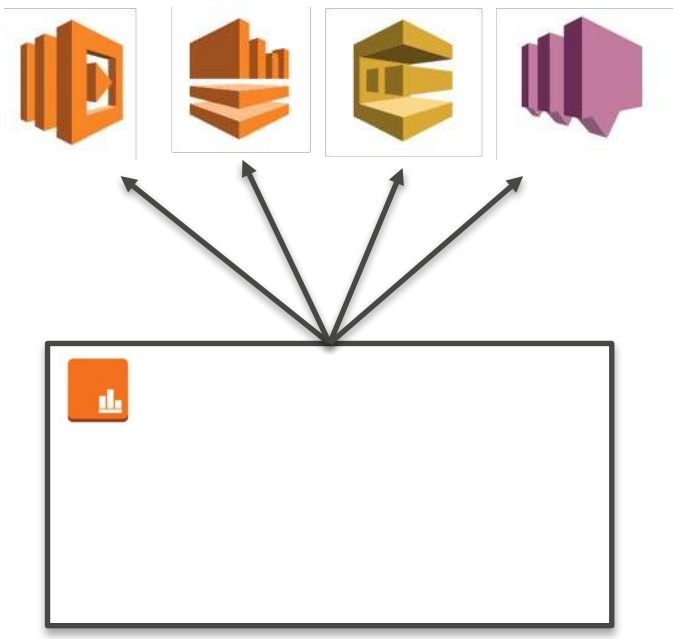


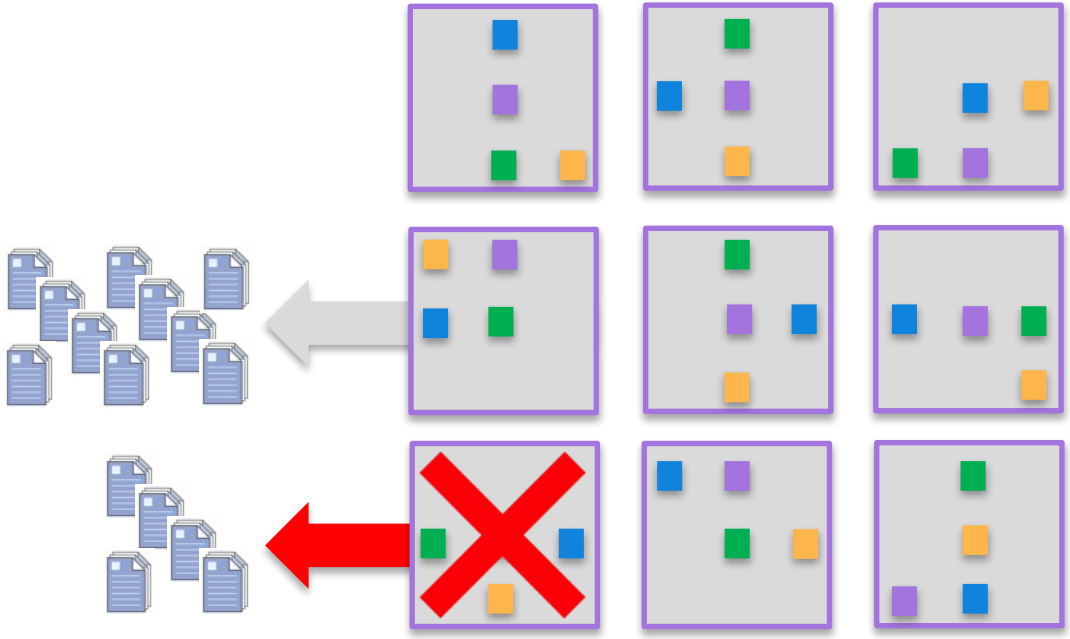
Event Stream & Blox

Amazon ECS: Under the Hood

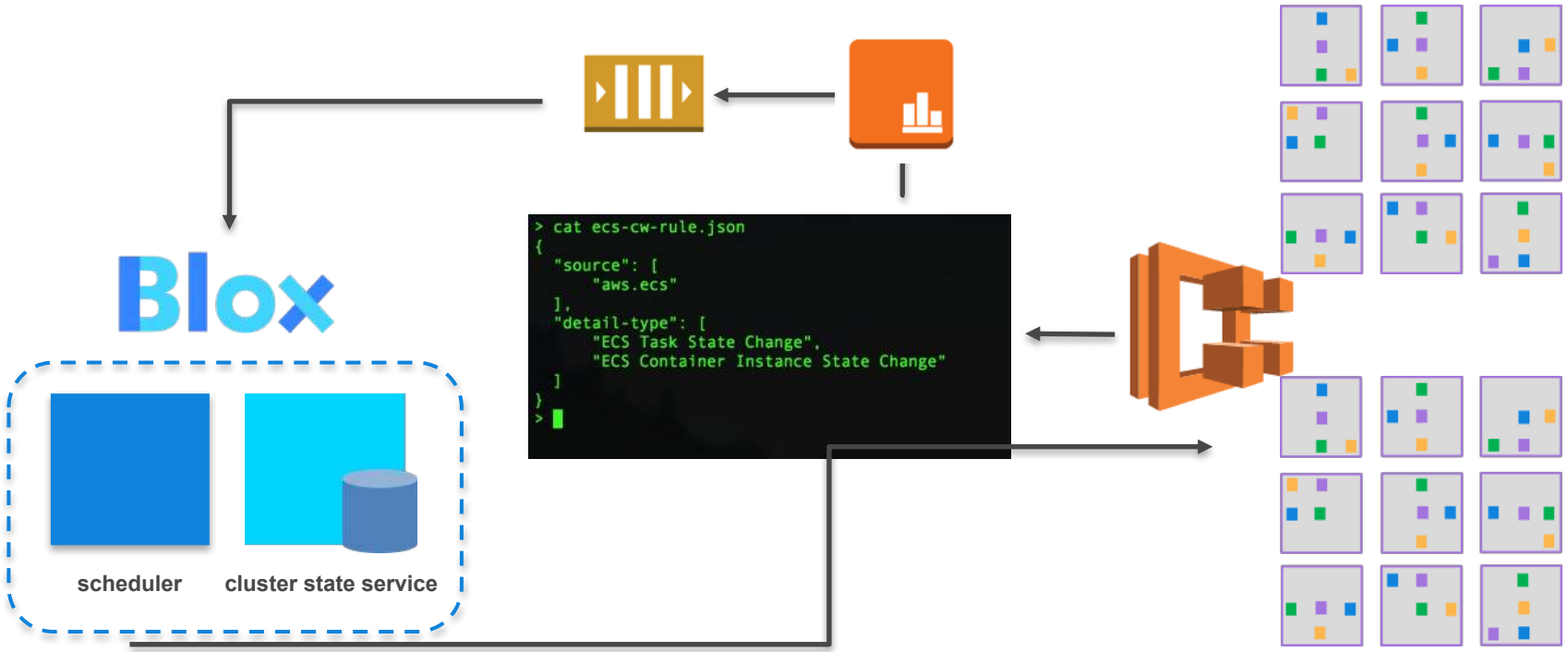


Consuming Real-time Events

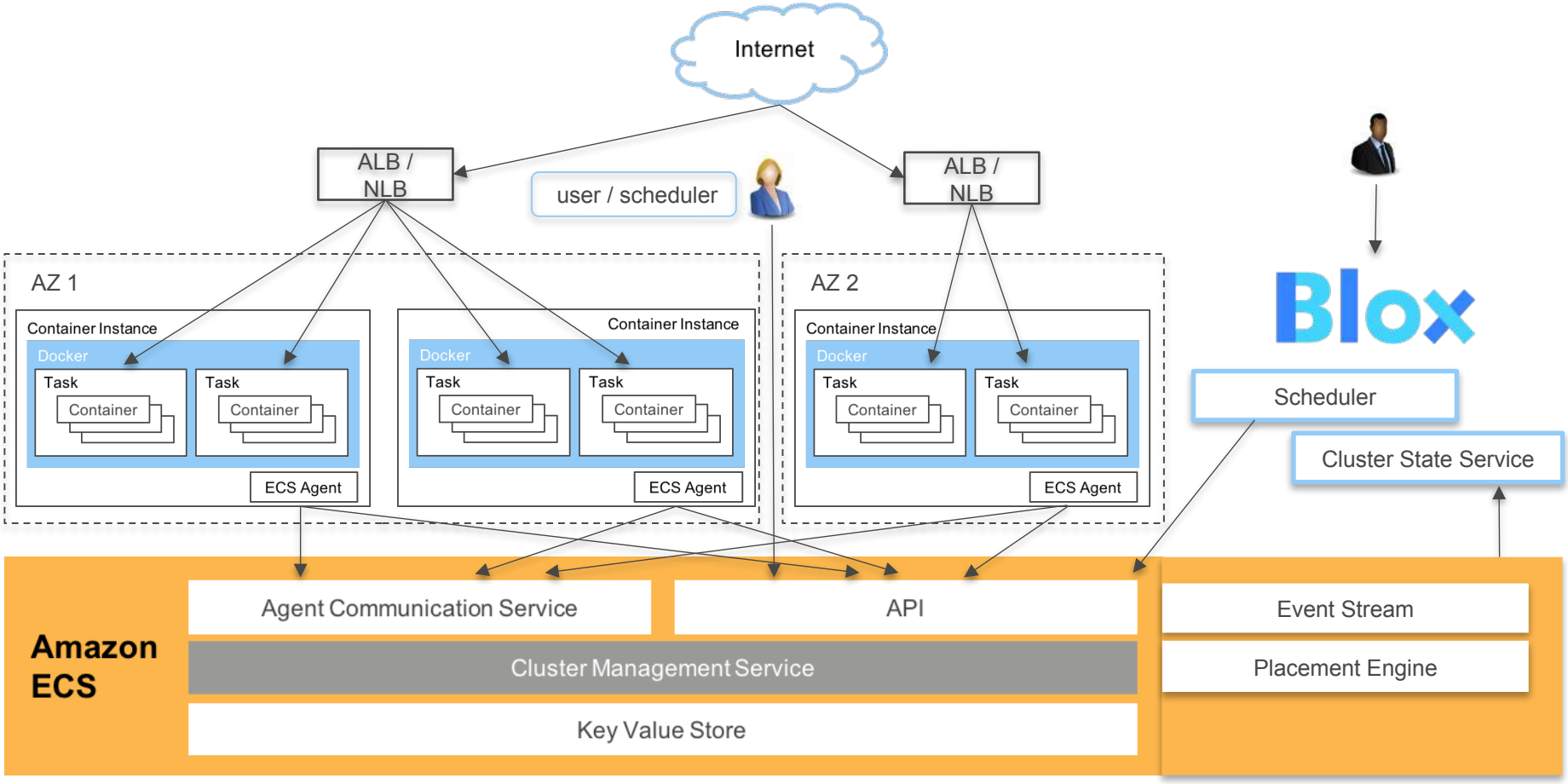




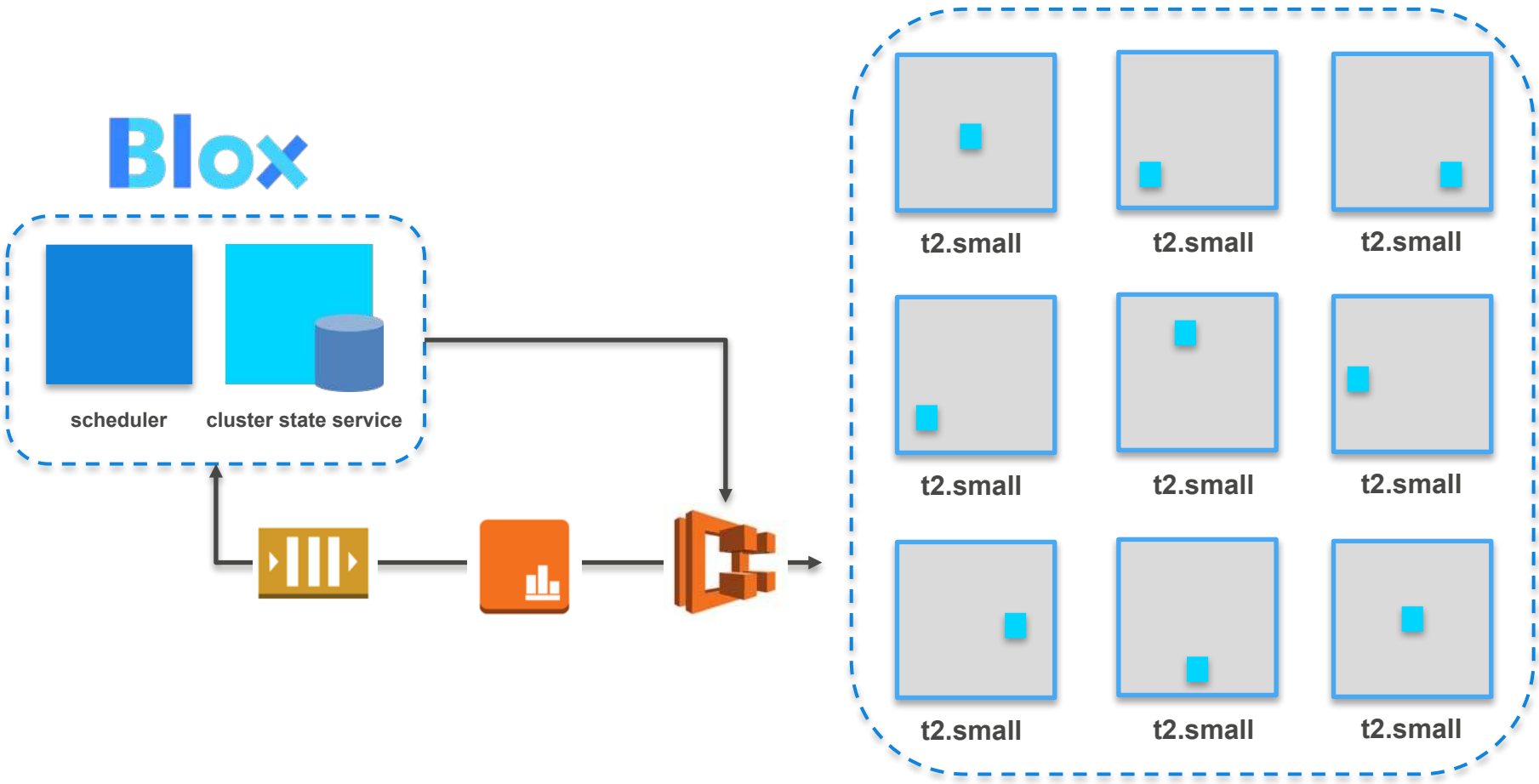
Handling ECS events with Blox



Amazon ECS: Under the Hood



Blox: Daemon Scheduler



Demo: Deploying Blox on AWS

https://github.com/juliensimon/aws/tree/master/blox

Creating Clusters

Create an ECS cluster for Blox

CF template: https://github.com/blox/blox/blob/dev/deploy/aws/conf/cloudformation_template.json

→ CloudWatch Event Rule + SQS queue

→ Daemon Scheduler + Cluster State Service + etcd

→ REST API exposing the Daemon Scheduler API

Create another ECS cluster managed by Blox

```
$ ecs-cli configure --cluster WebCluster --region ap-southeast-1
$ ecs-cli up --keypair admin --capability-iam --size 3 --instance-type t2.micro
```

Invoke the scheduler API

'demo-cli' tool: https://github.com/blox/blox/tree/dev/deploy/demo-cli

Listing Task Definitions

Grab the ARN for an *nginx* Task Definition, which the Daemon Scheduler will manage on ‘WebCluster’.

```
$ ./list-task-definitions.py --region ap-southeast-1

== Blox Demo CLI - List Task Definitions ==
{
  "taskDefinitionArns": [
    "arn:aws:ecs:ap-southeast-1:ACCOUNT:task-definition/BloxFramework:2",
    "arn:aws:ecs:ap-southeast-1:ACCOUNT:task-definition/nginx:1",
    "arn:aws:ecs:ap-southeast-1:ACCOUNT:task-definition/nginx:2"
  ]
}
```

Creating an Environment

```
$ ./blox-create-environment.py --environment WebEnvironment --cluster
WebCluster --task-definition "arn:aws:ecs:ap-southeast-1:ACCOUNT:task-
definition/nginx:2" --stack Blox --apigateway --region ap-southeast-1

== Blox Demo CLI - Create Blox Environment ==
HTTP Response Code: 200
{
  "taskDefinition":
"arn:aws:ecs:ap-southeast-1:ACCOUNT:task-definition/nginx:2",
  "deploymentToken": "17248257-08ec-4438-888f-e0ac28397653",
  "health": "healthy",
  "name": "WebEnvironment",
  "instanceGroup": {
    "cluster": "arn:aws:ecs:ap-southeast-1:ACCOUNT:cluster/WebCluster"
  }
}
```

Listing Environments

```
$ ./blox-list-environments.py --stack Blox --apigateway --region ap-
southeast-1
== Blox Demo CLI - List Blox Environments ==
HTTP Response Code: 200
{
  "items": [
    {
      "taskDefinition":
"arn:aws:ecs:ap-southeast-1:ACCOUNT:task-definition/nginx:2",
      "deploymentToken": "17248257-08ec-4438-888f-e0ac28397653",
      "health": "healthy",
      "name": "WebEnvironment",
      "instanceGroup": {
        "cluster": "arn:aws:ecs:ap-southeast-1:ACCOUNT:cluster/WebCluster"
      }
    }
  ]
}
```

Creating a Deployment

```
$ ./blox-create-deployment.py --environment WebEnvironment --deployment-token
"17248257-08ec-4438-888f-e0ac28397653" --stack Blox --apigateway --region ap-
southeast-1

== Blox Demo CLI - Create Blox Deployment ==
HTTP Response Code: 200
{
  "status": "pending",
  "environmentName": "WebEnvironment",
  "id": "7a05ea99-27a9-4339-a7a6-f4120065aea3",
  "failedInstances": [],
  "taskDefinition":
"arn:aws:ecs:ap-southeast-1:613904931467:task-definition/nginx:2”
}
```

Listing Deployments

```
$ ./blox-list-deployments.py --environment WebEnvironment --stack Blox --
apigateway --region ap-southeast-1

== Blox Demo CLI - List Blox Deployments ==
HTTP Response Code: 200
{
  "items": [
    {
      "status": "completed",
      "environmentName": "WebEnvironment",
      "id": "7a05ea99-27a9-4339-a7a6-f4120065aea3",
      "failedInstances": [],
      "taskDefinition":
"arn:aws:ecs:ap-southeast-1:ACCOUNT:task-definition/nginx:2"
    }
  ]
}
```

Scaling a Deployment

```
$ ecs-cli ps
Name                                          State    Ports  TaskDefinition
26313cbe-d929-49de-9cc3-873bf5f32a91/nginx  RUNNING         nginx:2
98442432-fd5c-434d-b93c-0737bd06aaab/nginx  RUNNING         nginx:2
ce9bf217-4b34-4f31-9c7b-a8c3402f1ffd/nginx  RUNNING         nginx:2

$ ecs-cli scale  --size 4 --capability-iam

$ ecs-cli ps
Name                                          State    Ports  TaskDefinition
26313cbe-d929-49de-9cc3-873bf5f32a91/nginx  RUNNING         nginx:2
98442432-fd5c-434d-b93c-0737bd06aaab/nginx  RUNNING         nginx:2
c404ac9a-0948-4cc8-b5b0-2238ccdf4035/nginx  RUNNING         nginx:2
ce9bf217-4b34-4f31-9c7b-a8c3402f1ffd/nginx  RUNNING         nginx:2
```

Additional resources

Tech articles by Werner Vogels, CTO, Amazon.com

http://www.allthingsdistributed.com/2014/11/amazon-ec2-container-service.html
http://www.allthingsdistributed.com/2015/04/state-management-and-scheduling-with-ecs.html
http://www.allthingsdistributed.com/2015/07/under-the-hood-of-the-amazon-ec2-container-service.html

Blox

https://blox.github.io/

Amazon ECS videos @ AWS re:Invent 2016

https://aws.amazon.com/blogs/compute/amazon-ec2-container-service-at-aws-reinvent-2016-wrap-up/

What's new on Amazon ECS

https://aws.amazon.com/ecs/release-notes/
https://aws.amazon.com/blogs/compute/powering-your-amazon-ecs-cluster-with-amazon-ec2-spot-instances/

Thank you!

@julsimon
http://aws.amazon.com/evangelists/julien-simon