AWS Summit

AWS技术峰会 2015 · 上海

亚马逊 aws

Intro to Amazon EC2 Container Service

Aaron Kao

Sr. Product Marketing Manager

Agenda

Why Containers?

- Container Use Patterns
- Production Challenges

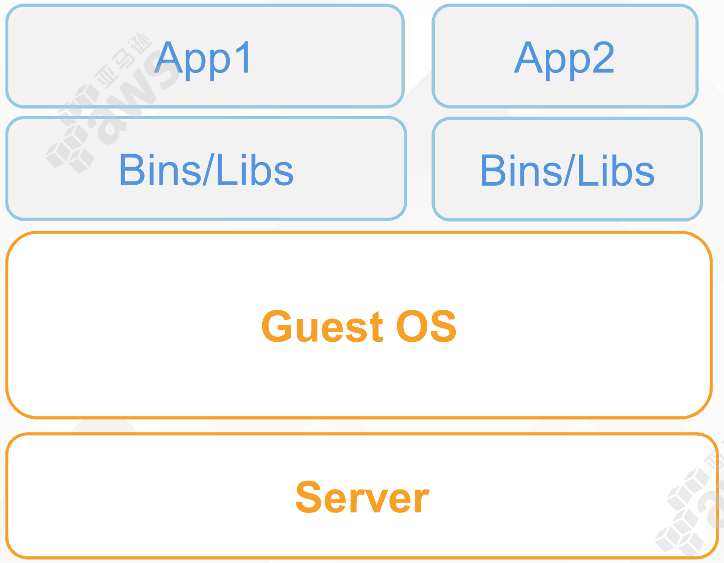
Cluster Management

Amazon EC2 Container Service

Demo

What Containers?

What are Containers?



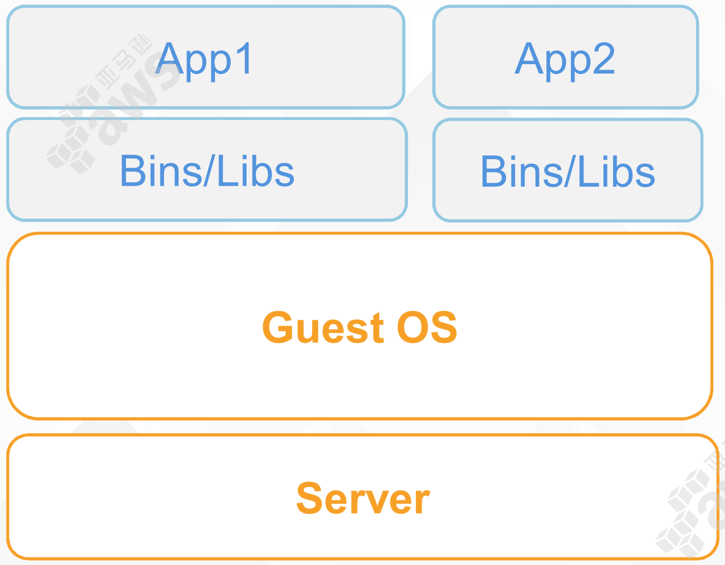
OS virtualization

Process isolation

Images

Automation

Container advantages



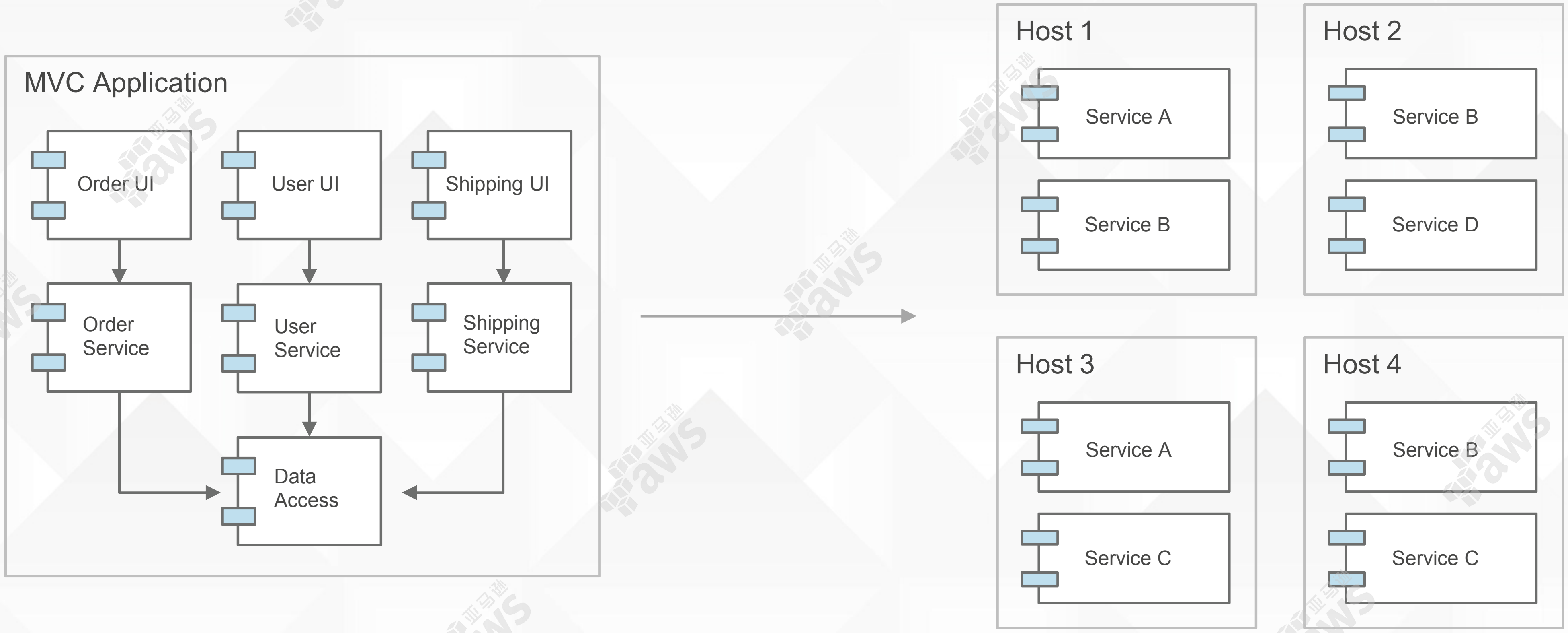
Portable

Flexible

Fast

Efficient

Services evolve to microservices



Containers are natural for services

Simple to model

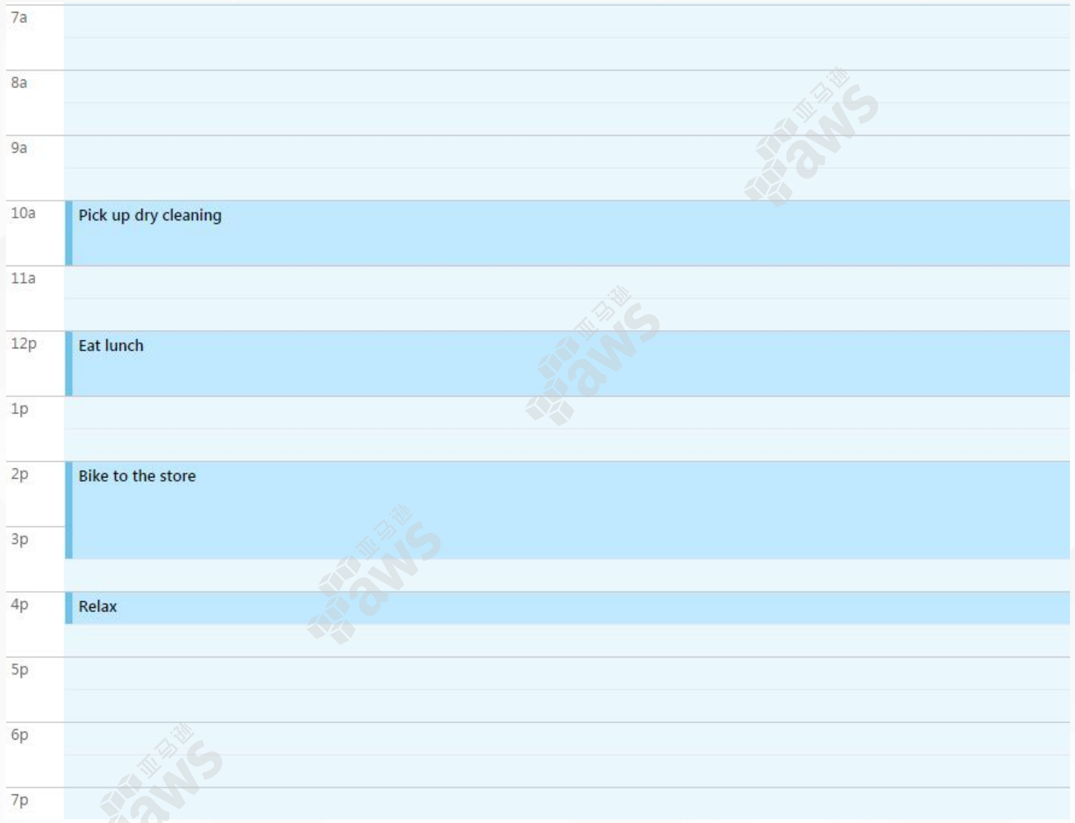
Any app, any language

Image is the version

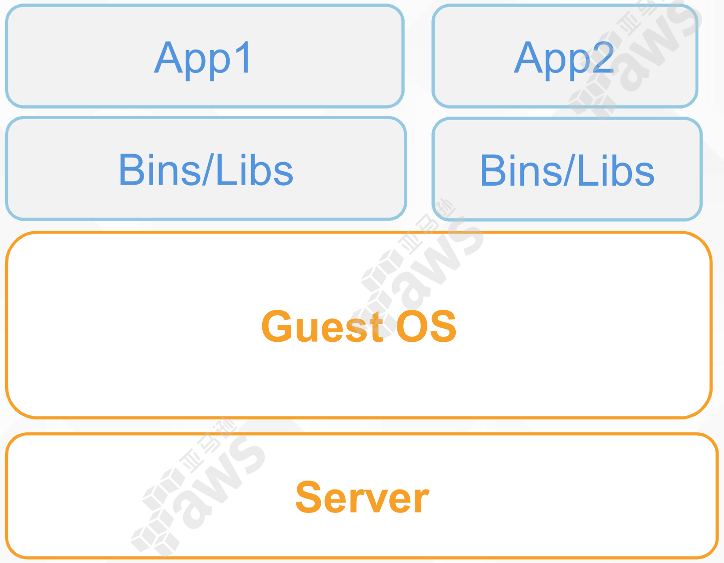
Test & deploy same artifact

Stateless servers decrease change risk

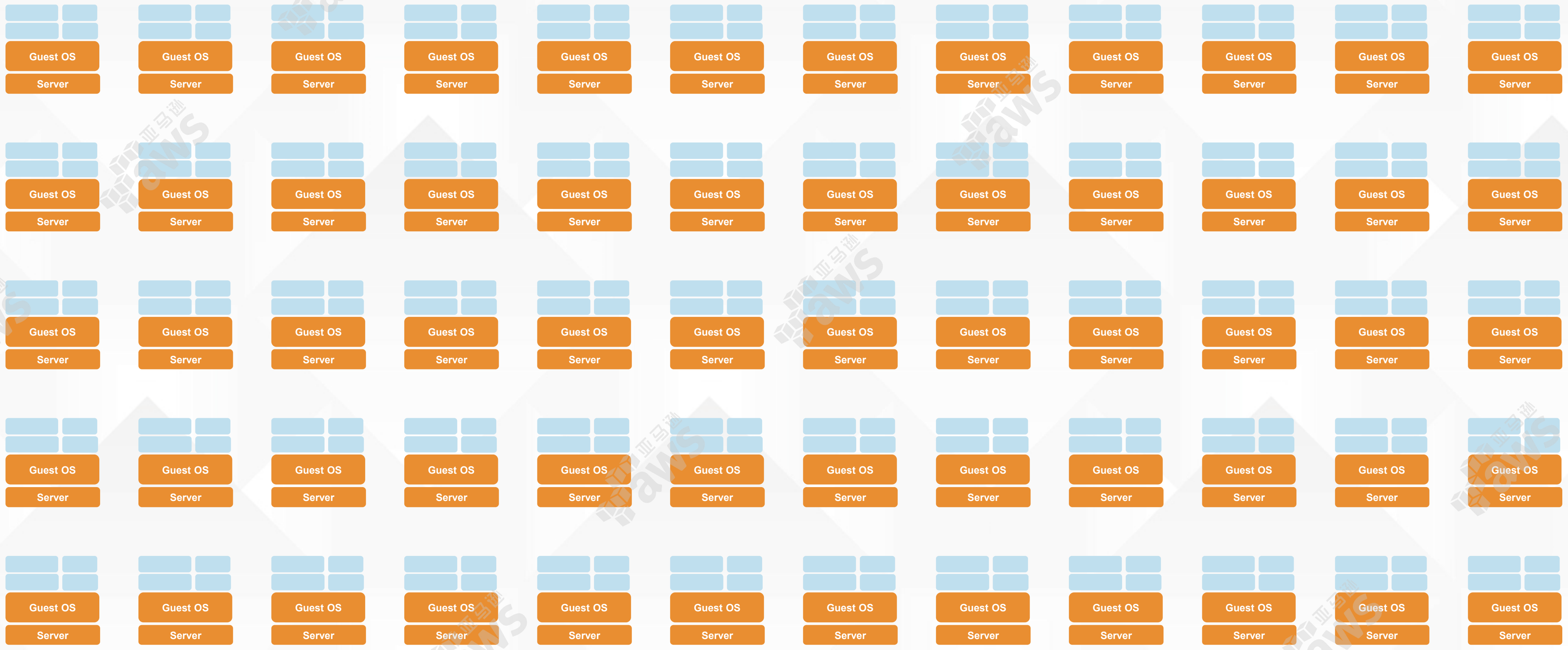
Scheduling


7a
8a
9a
10a Pick up dry cleaning
11a
12p Eat lunch
1p
2p Bike to the store
3p
4p Relax
5p
6p
7p

Scheduling one resource is straightforward



Scheduling a cluster is hard



Scheduling 101

Know your constraints

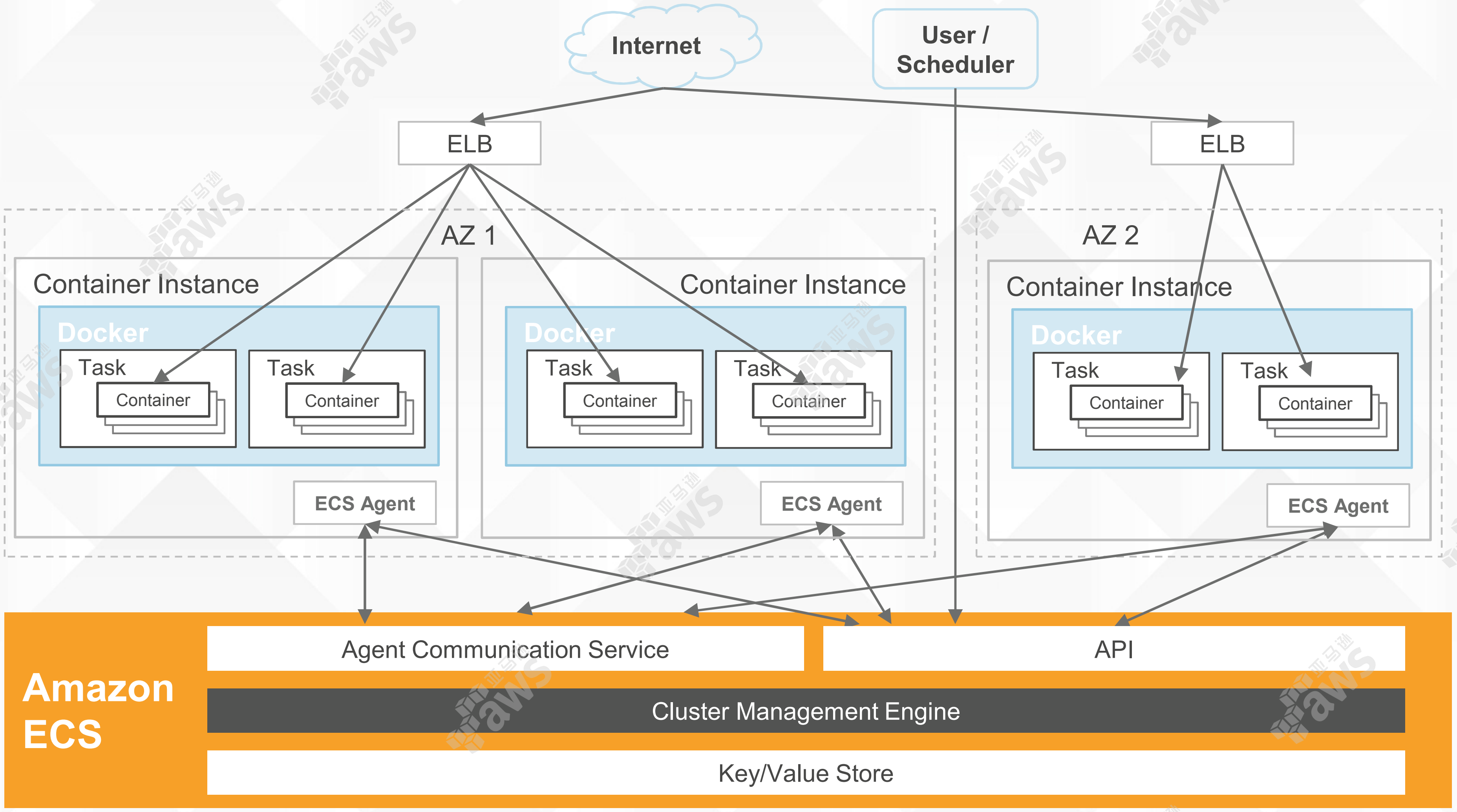
Find resources that meet the constraints

Request a resource

Confirm the resource

Cluster Management

Cluster Management



Cluster Management with Amazon ECS

Management of followers via ECS Agent

Dispatching of sub-tasks to proper location

Cluster state inspection

http://amzn.to/1jlHvnU

Cluster Management under the Hood

Paxos-based transactional journal based data store

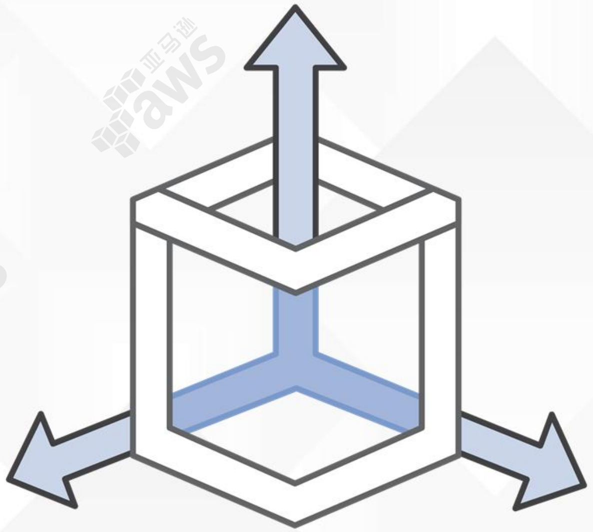
Writes are committed as transaction in the journal with order-based ID. The current value is the sum of all transactions made as recorded by the journal.

Reads are simply a snapshot in time of the journal. For a write to succeed, the write proposed must be the latest transaction since the last read.

http://bit.ly/1M9gGiv

Amazon EC2 Container Service

Easily Manage Clusters for Any Scale



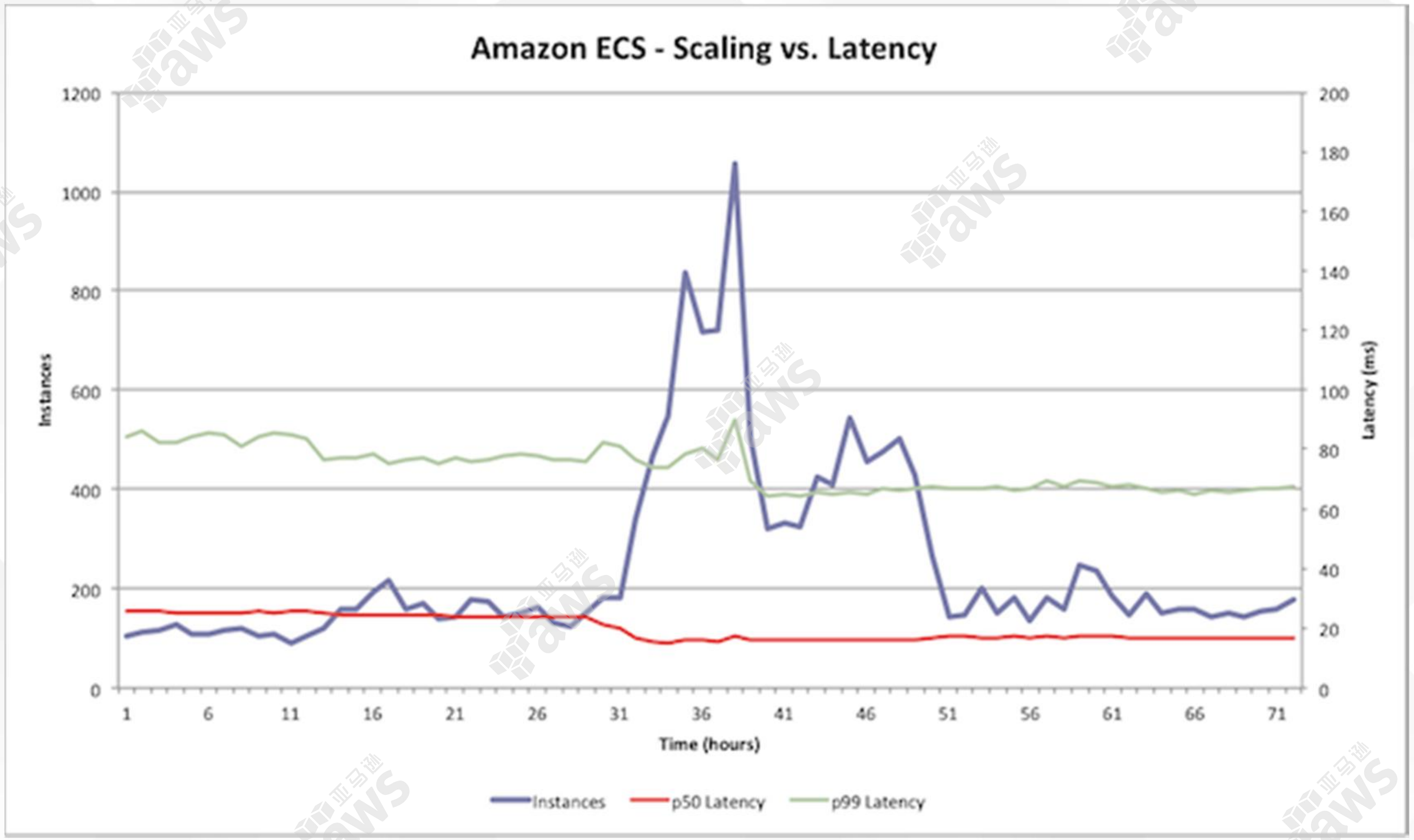
Nothing to run

Complete state

Control and monitoring

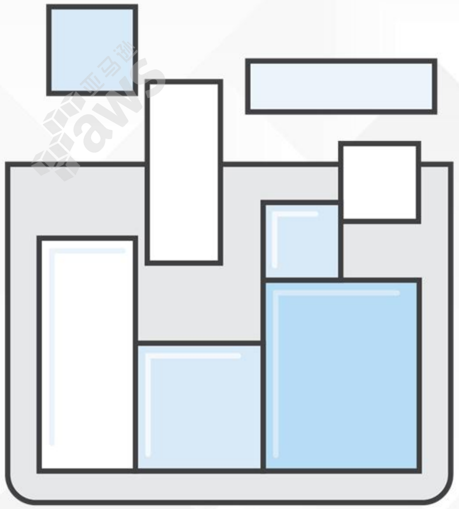
Scale

Scalable



Amazon ECS - Scaling vs. Latency
Instances
Latency (ms)
Time (hours)
0
200
400
600
800
1000
1200
0
20
40
60
80
100
120
140
160
180
200
1
6
11
16
21
26
31
36
41
46
51
56
61
66
71
Instances
p50 Latency
p99 Latency

Flexible Container Placement

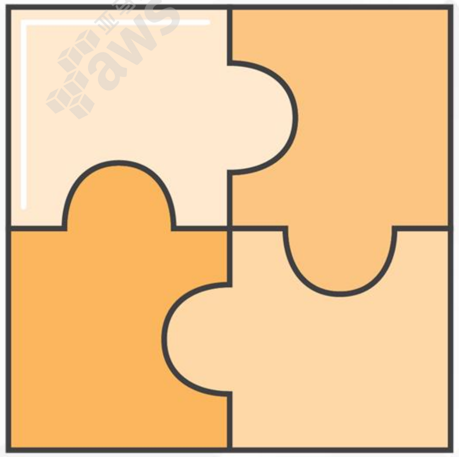


Applications

Batch jobs

Multiple schedulers

Designed for use with other AWS services



Elastic Load Balancing

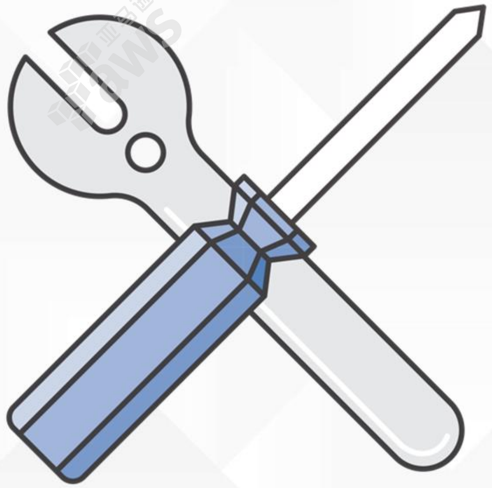
Amazon Elastic Block Store

Amazon Virtual Private Cloud

AWS Identity and Access Management

AWS CloudTrail

Extensible



Comprehensive APIs

Open source agent

Custom schedulers

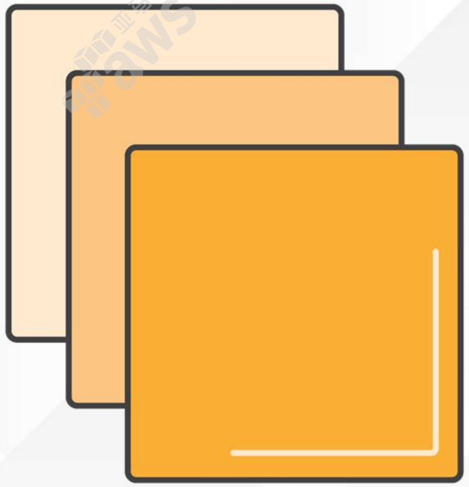
Key Components: Container Instances

Amazon EC2 instances

Docker daemon

Amazon ECS agent

Key Components: Clusters



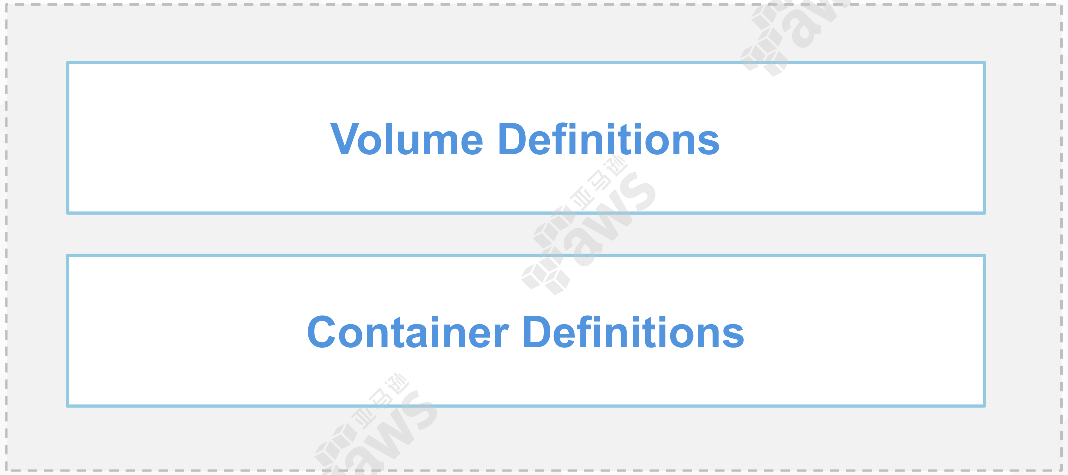
Regional

Resource pool

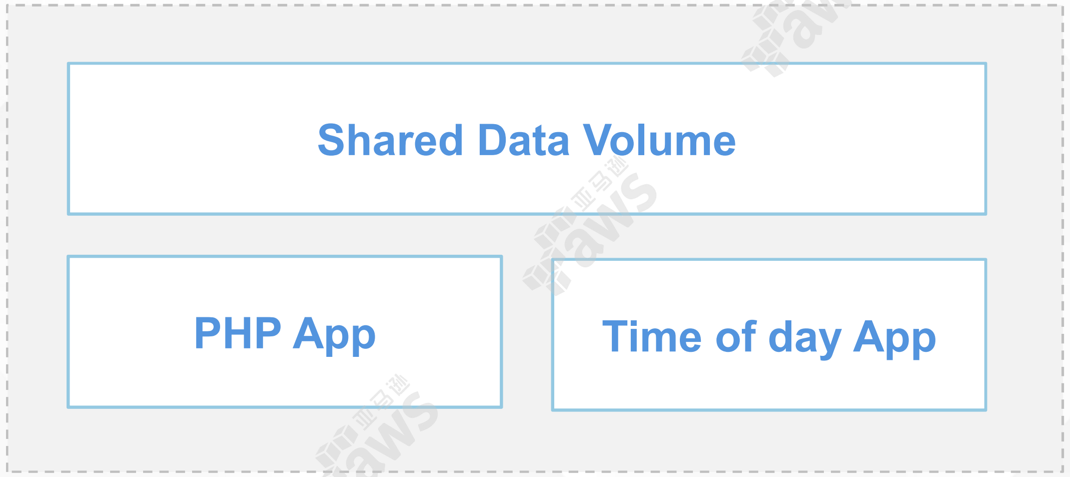
Grouping of Container Instances

Start empty, dynamically scalable

Key Components: Task Definitions



Key Components: Task Definitions



Key Components: Task Definitions

```
{
           ″environment″: [],
           ″name″: ″simple-demo″,
           ″image″: ″my-demo″,
           ″cpu″: 10,
           ″memory″: 500,
           ″portMappings″: [
               {
                   ″containerPort″: 80,
                   ″hostPort″: 80
               }
           ],
           ″mountPoints″: [
               {
                   ″sourceVolume″: ″my-vol″,
                   ″containerPath″: ″/var/www/my-vol″
               }
           ],
           ″entryPoint″: [
               ″/usr/sbin/apache2″,
               ″-D″,
               ″FOREGROUND″
           ],
           ″essential″: true
      },
```

```
{
            ″name″: ″busybox″,
            ″image″: ″busybox″,
            ″cpu″: 10,
            ″memory″: 500,
            ″volumesFrom″: [
            {
              ″sourceContainer″: ″simple-demo″
            }
            ],
            ″entryPoint″: [
                ″sh″,
                ″-c″
            ],
            ″command″: [
                ″/bin/sh -c \″while true; do /bin/date >
/var/www/my-vol/date; sleep 1; done\″″
            ],
            ″essential″: false
        }
```

Key Components: Task Definitions

```
{
            "environment": [],
            "name": "simple-demo",
            "image": "my-demo",
            "cpu": 10,
            "memory": 500,
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80
                }
            ],
            "mountPoints": [
                {
                    "sourceVolume": "my-vol",
                    "containerPath": "/var/www/my-vol"
                }
            ],
            "entryPoint": [
                "/usr/sbin/apache2",
                "-D",
                "FOREGROUND"
            ],
            "essential": true
        },
```

10 CPU Units (1024 is full CPU), 500 Megabytes of Memory

Expose port 80 in container to port 80 on host

Create and mount volumes

Essential to our Task

Key Components: Task Definitions

From Docker Hub

Mount volume from
other container

Command to exec

```
{
            "name": "busybox",
            "image": "busybox",
            "cpu": 10,
            "memory": 500,
            "volumesFrom": [
            {
              "sourceContainer": "simple-demo"
            }
            ],
            "entryPoint": [
                "sh",
                "-c"
            ],
            "command": [
                "/bin/sh -c \"while true; do /bin/date >
/var/www/my-vol/date; sleep 1; done\""
            ],
            "essential": false
        }
```

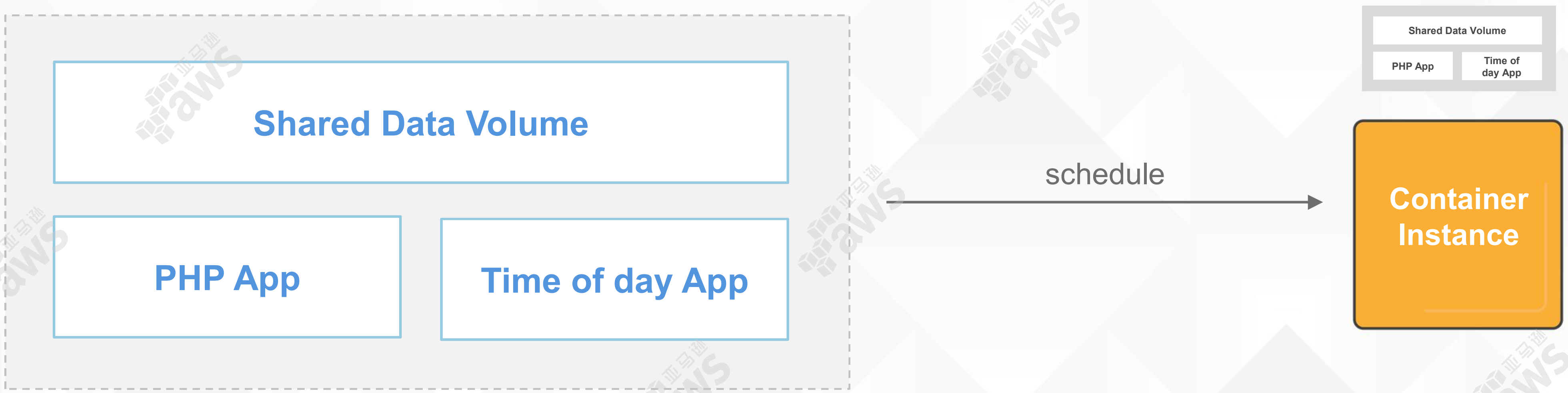
Key Components: Tasks

Unit of work

Grouping of related Containers

Run on Container Instances

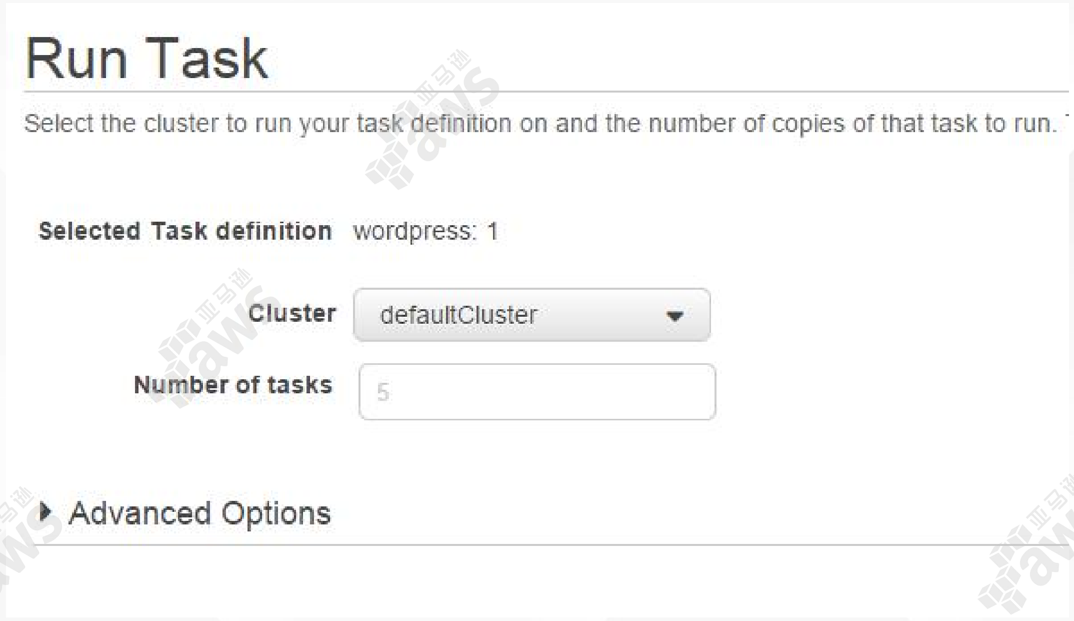
Key Components: Tasks



Running Services

Run a task

Good for short-lived containers, e.g. batch jobs



Run Task

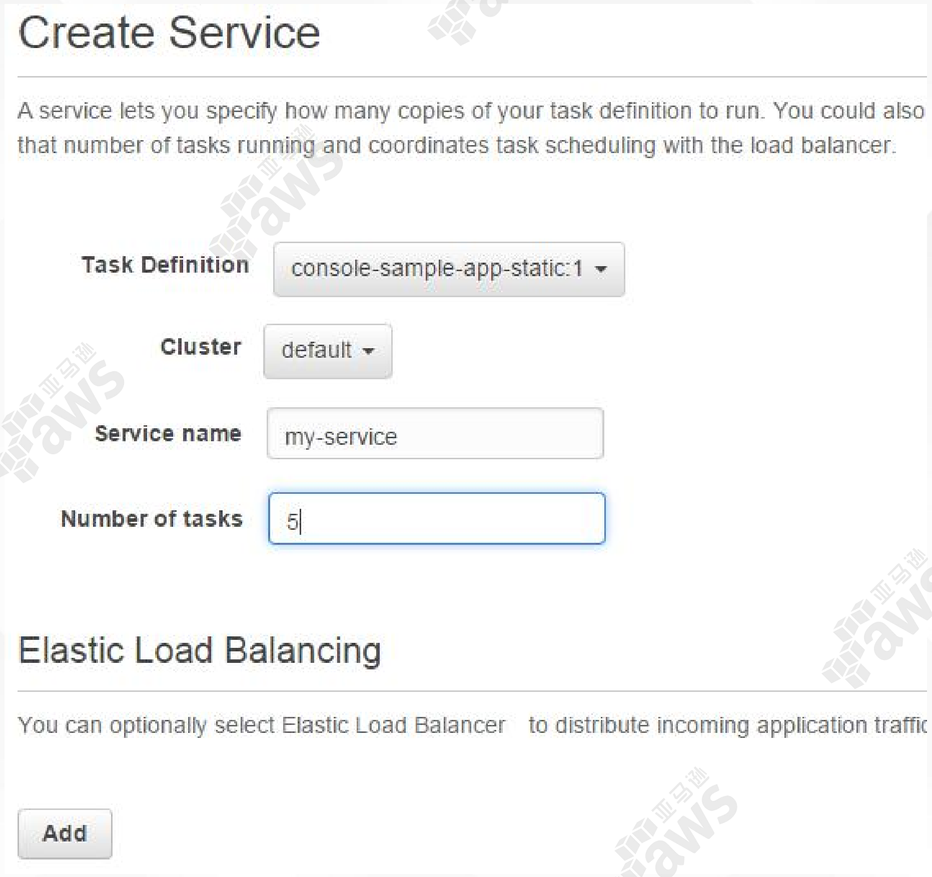
Select the cluster to run your task definition on and the number of copies of that task to run.

Selected Task definition	wordpress: 1
Cluster	defaultCluster
Number of tasks	5

▸ Advanced Options

Create a Service

Good for long-running applications and services



Create Service

A service lets you specify how many copies of your task definition to run. You could also
that number of tasks running and coordinates task scheduling with the load balancer.

Task Definition	console-sample-app-static:1
Cluster	default
Service name	my-service
Number of tasks	5

Elastic Load Balancing

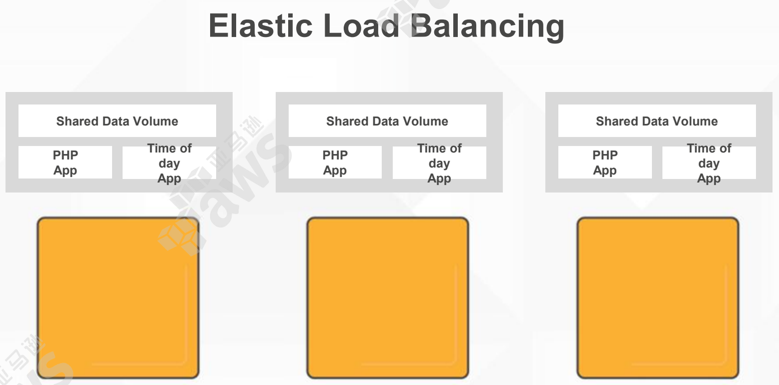
You can optionally select Elastic Load Balancer to distribute incoming application traffic

Create Service

Load Balance traffic across containers

Automatically recover unhealthy containers

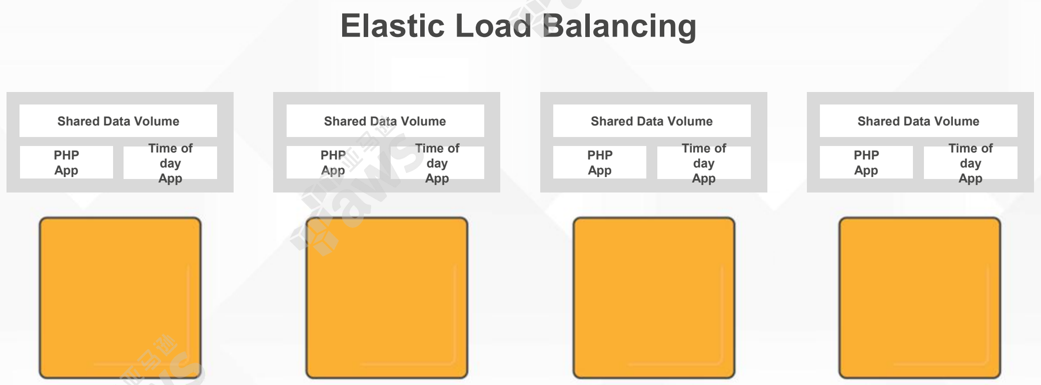
Discover services



Update Service (cont.)

Scale up

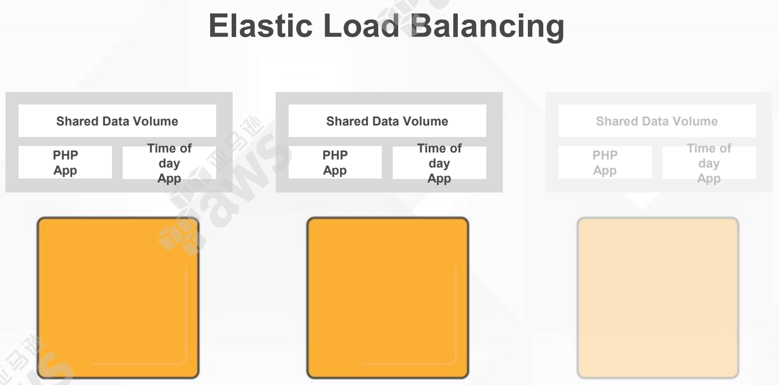
Scale down



Update Service (cont.)

Deploy new version

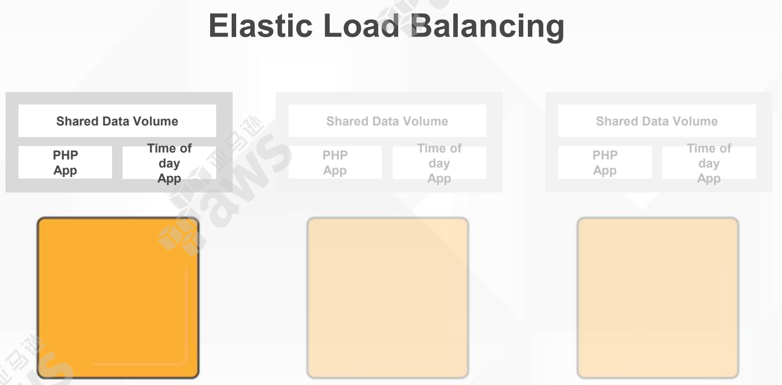
Drain connections



Update Service (cont.)

Deploy new version

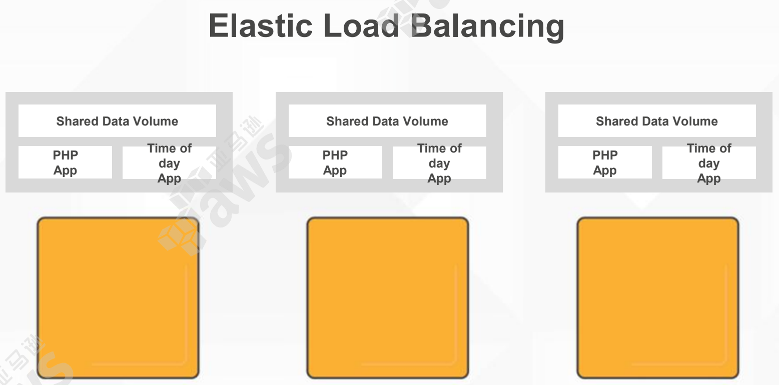
Drain connections



Update Service (cont.)

Deploy new version

Drain connections



Demo

Thank You