Clustering Docker on AWS with Amazon ECR & ECS

Julien Simon, Technical Evangelist, AWS

Docker Lille, 28/01/2016

amazon web services

You’re more than welcome to tweet!

Pictures too ☺

@julsimon @aws_actus @Docker

#aws #ecs #ecr

The problem

Given a certain amount of processing power and memory,

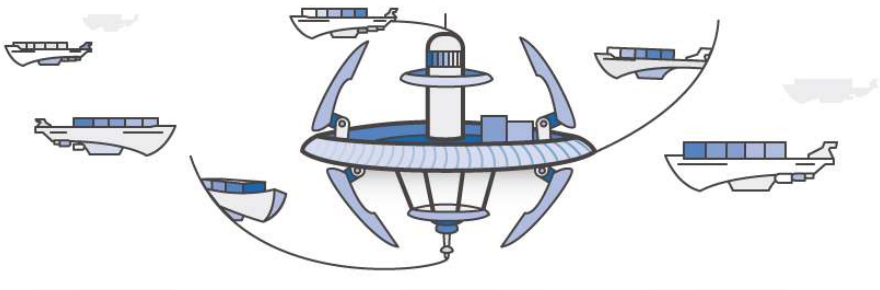
how can we best manage an arbitrary number of apps running in Docker containers?

http://tidalseven.com

Requirements for modern cluster orchestration

Reliable state management & flexible scheduling

→ Cluster must be multi-tenant and scalable
→ High availability & state management must be built-in



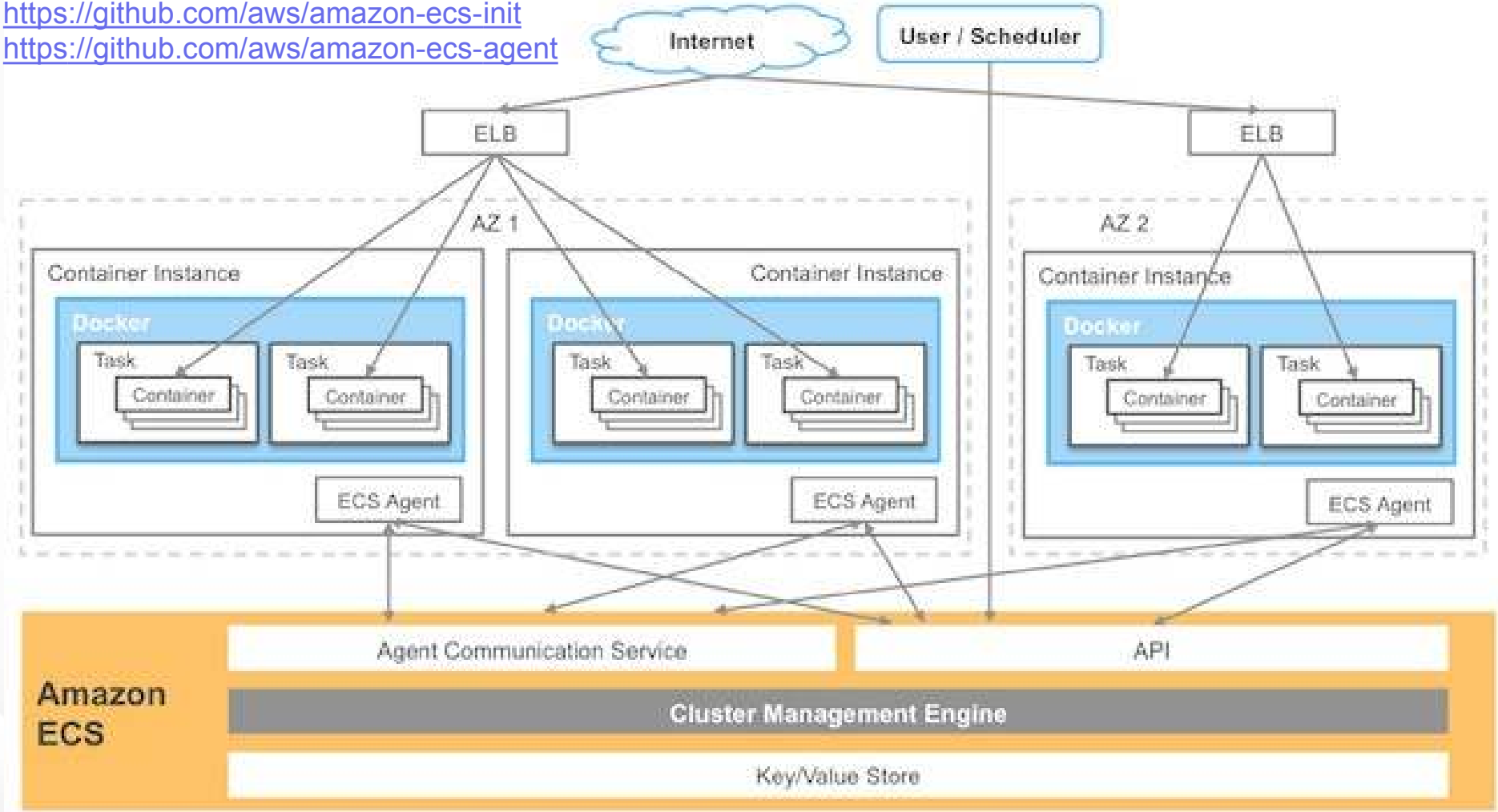
Amazon EC2 Container Service (ECS)
Launched in 04/2015
https://aws.amazon.com/ecs/
No additional charge ☺

Amazon EC2 Container Registry (ECR)
Launched in 12/2015
https://aws.amazon.com/ecr/
Free tier: 500MB / month for a year
$0.10 / GB / month + outgoing traffic

amazon web services

Start a cluster now

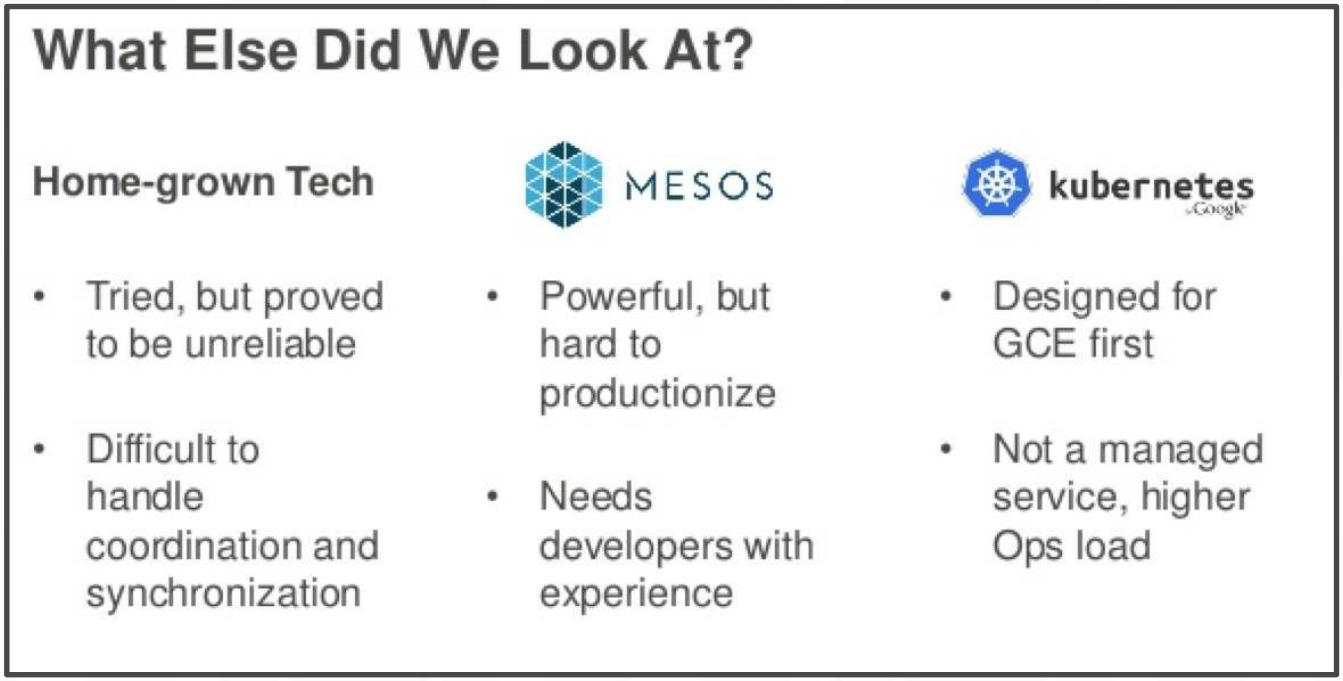
https://github.com/aws/amazon-ecs-init
https://github.com/aws/amazon-ecs-agent



http://www.allthingsdistributed.com/2015/07/under-the-hood-of-the-amazon-ec2-container-service.html

Case study: Coursera

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



"*Amazon ECS enabled Coursera to focus on releasing new software rather than spending time managing clusters*"

Frank Chen, Software Engineer

Case study: Meteor

https://www.youtube.com/watch?v=xIc3WT6kAVw

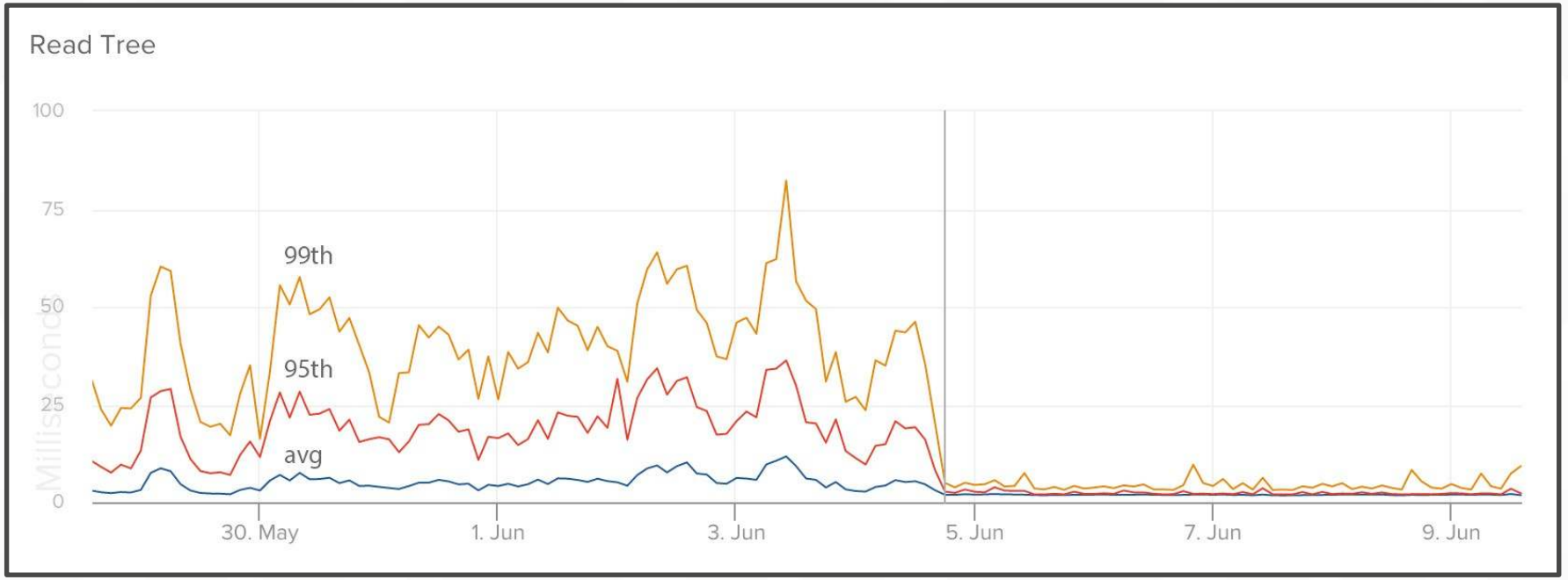
ECS container management

- Lots of exciting options here: ECS, Kubernetes, Marathon, …
- **Service argument is compelling.** Same case we make for Galaxy to our customers.
- **Integration with other parts of AWS** saves us time and code. Example: services automatically register containers with Elastic Load Balancing (ELB).
- **Support for multiple Availability Zones**.
- Bottom line: **ECS got us to market faster** than the alternatives.

"Can we scale the amount of compute resources necessary to run all our customers' apps? Can we scale the mechanics of coordinating all those pieces? Using AWS, we can answer 'yes' to both" - Matt DeBergalis, Cofounder & VP Product

Case study: Remind

https://www.youtube.com/watch?v=8zbbQkszP04



"Moving to Amazon ECS significantly improved our service performance"

Jason Fischl, VP of Engineering

amazon web services

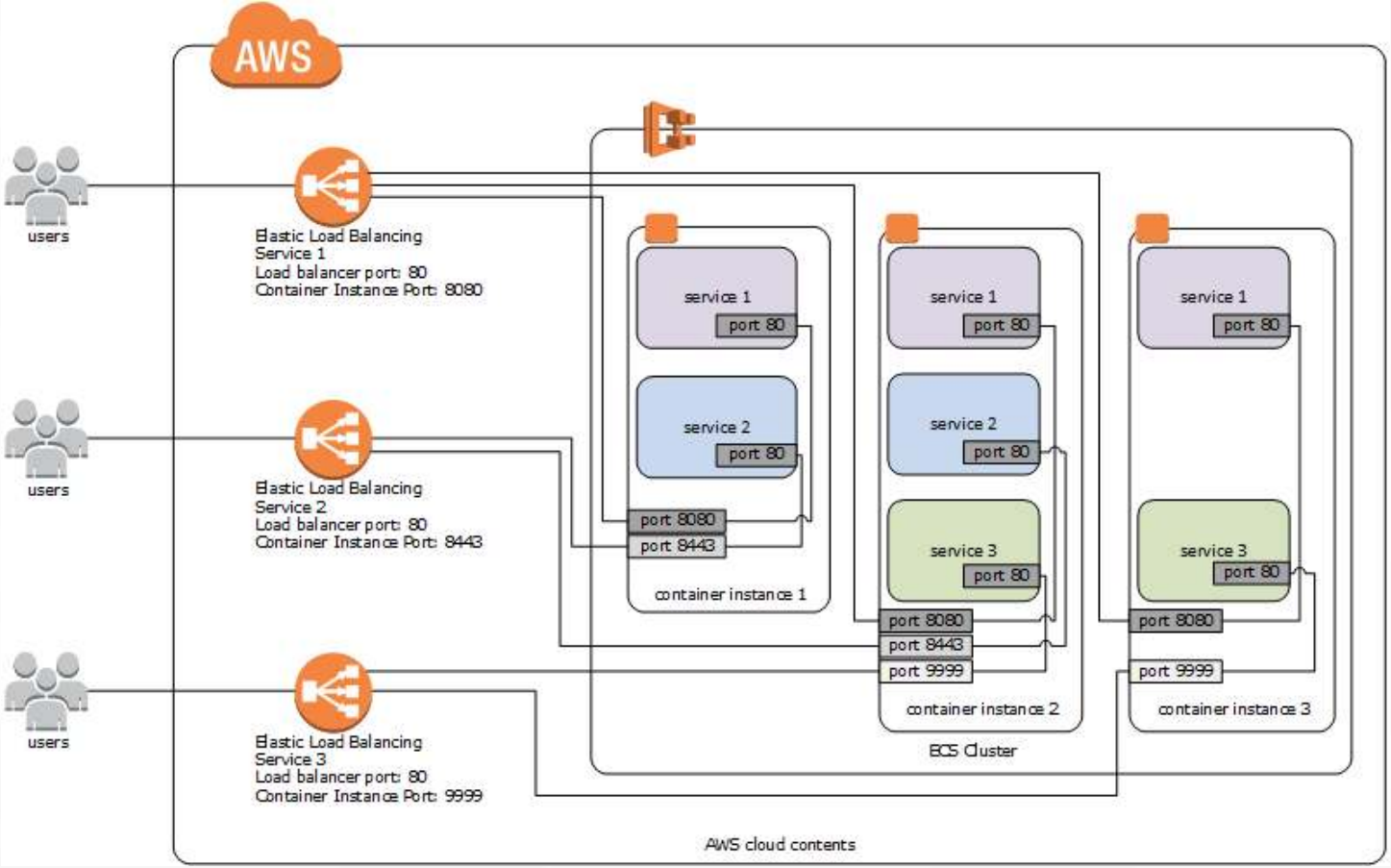
DEMO #1

Demo gods, I'm your humble servant, please be good to me

Using the 'aws' and 'ecs-cli' command lines:
Create and scale an Amazon ECS cluster
Pull an image from an Amazon ECR registry
Run, scale and load-balance a simple PHP app

… and look at EC2 Instances, Security Groups, Auto-Scaling Groups,
Elastic Load Balancers and Cloud Formation

Load balancing services on fixed ports



Micro-services: it gets worse ☺

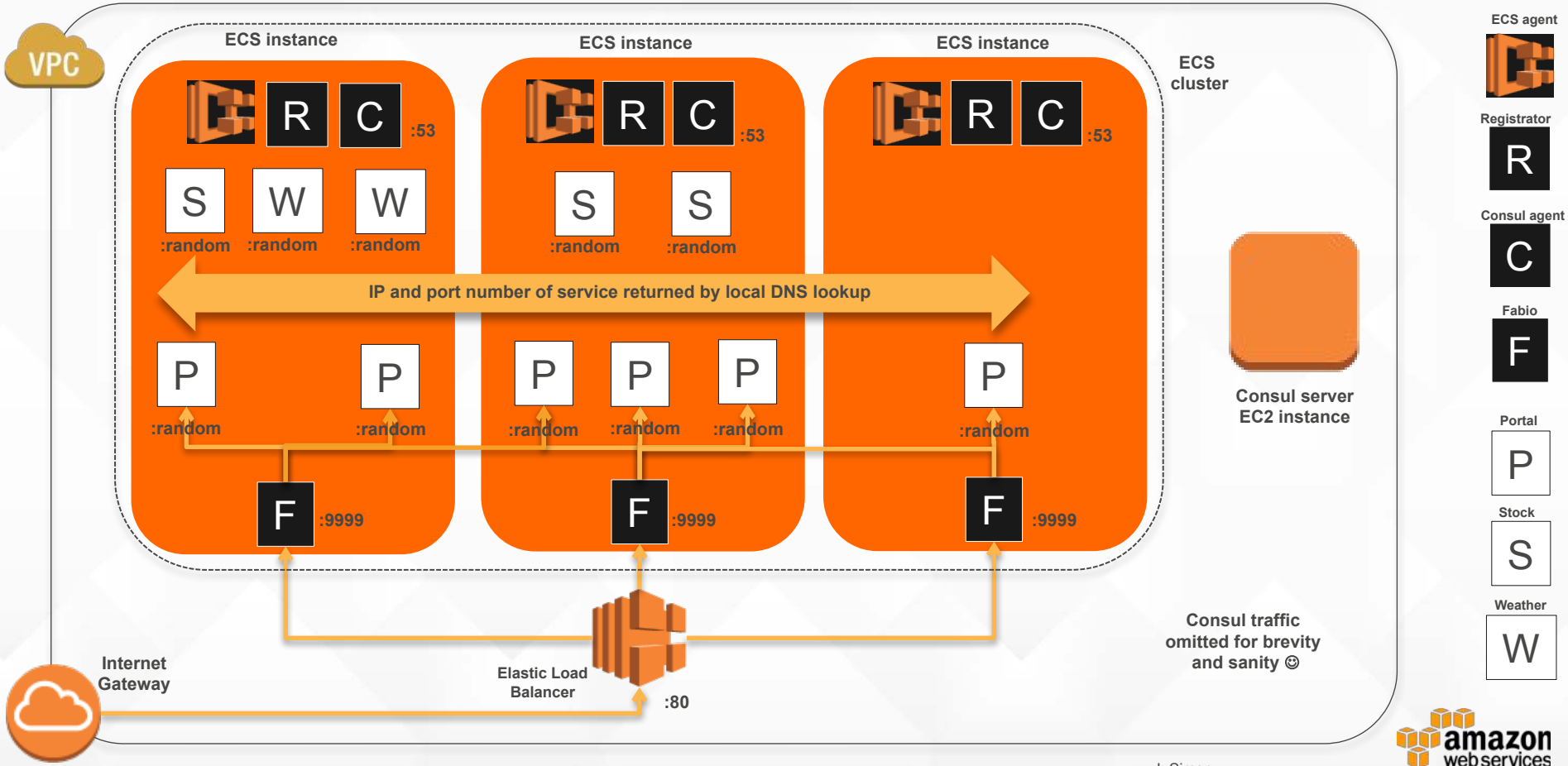
Micro-services run in an ever-moving production environment: continuous deployment, multiple versions running in parallel, servers coming and going, etc.

- Can micro-services be deployed and scaled independently?
- Can multiple copies of a micro-service run on the same server?
- Can micro-services register their name & port automatically?
- Can micro-services discover each other?
- Can traffic be load-balanced across multiple copies of a micro-service?

Yes we can!

- Can micro-services be deployed and scaled independently?
  Micro-service = Docker image + task definition + service definition
- Can multiple copies of a micro-service run on the same server?
  Let Docker assign a random port
- Can micro-services register name & port automatically?
  Use Registrator to inspect containers and register them in Consul
- Can micro-services discover each other?
  Use local Consul agent for DNS lookups
- Can traffic be load-balanced across multiple copies of a micro-service?
  User-facing service: ELB (80) → Fabio (fixed port) → service (random port)
- Internal service: DNS lookup (53) → service (random port)

Load balancing services on random ports



DEMO #2

Demo gods, I know I'm pushing it, but please don't let me down now

Run an application built from 3 micro-services (*portal*, *stock*, *weather*) running on random ports
Use a combination of ELB, Fabio and Consul (DNS lookups) for load-balancing
Scale & break stuff, see what happens ☺

Thank you! Let's keep in touch ☺

Monthly ECS/ECR news @ Docker Paris : http://www.meetup.com/fr-FR/Docker-Paris/

Many AWS events and meetups in 2016 all across France : @aws_actus @julsimon

AWS Summit Paris : 31/05/2016 (free!) https://aws.amazon.com/fr/paris16/

Want to start a local AWS User Group? https://aws.amazon.com/fr/usergroups/

BONUS SLIDES

Using Amazon ECS

https://docs.aws.amazon.com/fr_fr/AmazonECS/latest/developerguide/ECS_GetStarted.html

AWS Console
https://console.aws.amazon.com/ecs/

AWS CLI
https://github.com/aws/aws-cli
https://github.com/awslabs/aws-shell **NEW!**

AWS SDK (Java, .NET, Node.js, PHP, Python, Ruby, Go, C++)
https://github.com/aws/aws-sdk-*

Amazon ECS CLI https://github.com/aws/amazon-ecs-cli
https://www.youtube.com/watch?v=MMr78xAiZpQ

Amazon ECS resources

Building demos #1 & #2

https://docs.aws.amazon.com/fr_fr/AmazonECS/latest/developerguide/docker-basics.html
https://github.com/awslabs/ecs-demo-php-simple-app
https://aws.amazon.com/blogs/compute/service-discovery-via-consul-with-amazon-ecs/
https://github.com/awslabs/service-discovery-ecs-consul
https://www.consul.io/ - https://github.com/gliderlabs/registrator - https://github.com/eBay/fabio

Tech articles by Werner Vogels, CTO of Amazon

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

Amazon ECS video @ AWS re:Invent 2015

Amazon ECS: Distributed Applications at Scale https://www.youtube.com/watch?v=eun8CqGqdk8
Turbocharge Your Deployment Pipeline with Containers https://www.youtube.com/watch?v=o4w8opVCI-Q
From Local Docker Development to Production Deployments https://www.youtube.com/watch?v=7CZFpHUPqXw

Amazon ECR in one slide ☺

```
$ aws configure → select ‘us-east-1’
$ aws ecr get-login → run ‘docker login’ command
$ aws ecr create-repository --repository-name ecrtest

$ docker build -t ecrtest .
$ docker tag ecrtest:latest
  AWS_ACCOUNT_ID.dkr.ecr.us-east-1.amazonaws.com/ecrtest:latest
$ docker push AWS_ACCOUNT_ID.dkr.ecr.us-east-1.amazonaws.com/ecrtest:latest

$ aws ecr describe-repositories --repository-name ecrtest
$ aws ecr list-images --repository-name ecrtest
$ aws ecr batch-delete-image --repository-name ecrtest
  --image-ids imageTag=TAG",imageDigest="sha256:DIGEST”
$ aws ecr delete-repository --repository-name ecrtest
```

amazon web services

Creating, scaling and deleting an ECS cluster

$ ecs-cli **configure** -c CLUSTER_NAME -r eu-west-1
$ ecs-cli **up** --keypair *KEY_PAIR_ID* --capability-iam --size 1 --instance-type t2.micro
$ ecs-cli **scale** --size 3 --capability-iam
$ ecs-cli **ps**
$ ecs-cli **down** CLUSTER_NAME --force

Reminder:

- ‘ecs-cli up’ launches a CloudFormation template
- By default, the cluster is created in a new VPC
- By default, only port 80 is open on ECS instances
- See ‘ecs-cli up –help’ for advanced networking options

Basic ECS commands

$ aws ecs **list-clusters**

$ aws ecs **describe-clusters** --cluster CLUSTER_NAME

$ aws ecs **list-container-instances** --cluster CLUSTER_NAME

$ aws ecs **describe-container-instances** --cluster CLUSTER_NAME --container-instances *ECS_INSTANCE_ID*

Updating the Amazon ECS agent

$ aws ecs **update-container-agent** --cluster CLUSTER_NAME
--container-instance ECS_INSTANCE_ID

```
#!/bin/bash
for i in `aws ecs list-container-instances --cluster CLUSTER_NAME | grep arn | cut -b 64-99`
do
aws ecs update-container-agent --cluster CLUSTER_NAME--container-instance $i
done
```

Allowing SSH access on an ECS cluster

$ aws ecs **describe-container-instances** --cluster CLUSTER_NAME
--container-instances *ECS_INSTANCE_ID*
--query 'containerInstances[*].ec2InstanceId’

$ aws ec2 **describe-instances** --instance-ids *EC2_INSTANCE_ID*
--query "Reservations[*].Instances[*].SecurityGroups[*].GroupId"

$ aws ec2 **authorize-security-group-ingress**
--group-id *SECURITY_GROUP_ID*
--protocol tcp --port 22 --cidr 0.0.0.0/0

Amazon ECS metadata

Log into the ECS instance

Instance information

$ curl http://localhost:51678/v1/metadata

Task information

$ curl http://localhost:51678/v1/tasks

Creating and scaling an Amazon ECS service

Write a **docker-compose.yml** file

$ ecs-cli compose service **start**

$ ecs-cli compose service **up**

$ ecs-cli compose service **scale** 3

Stopping and deleting an Amazon ECS service

$ ecs-cli compose service **stop**

$ ecs-cli compose service **delete**