Use Case: Running Docker Containers

1. Introduction

A new trend in software development is to break down applications into microservices and deploy them using containers. Containerization has many benefits over the traditional approach of using Virtual Machines. When adopting a microservices strategy, using containers is often the only way to success.

The benefits of running containers are (but are not limited to):

- Faster bootup (seconds instead of minutes with just virtualization)
- Better Continuous Integration (repeatable binary builds)
- Closer dev-prod parity (same binary for dev, qa and prod)
- Immutable infrastructure (container images can't be changed)
- Improved scaling (scale on a per container basis)

To make this architecture design work, containers must be orchestrated within a cluster of machines. When using AWS, EC2 Container Service (ECS) can be used to manage the lifecycle of the docker containers. The EC2 Container Registry (ECR), another AWS service, is used to store the images of the containers.

ECS can take care of starting containers from ECR and run the desired amount of containers. An Elastic Load Balancer (ELB) can be added for every microservice that needs to be deployed and that exposes some API or frontend to the customer or other microservices.

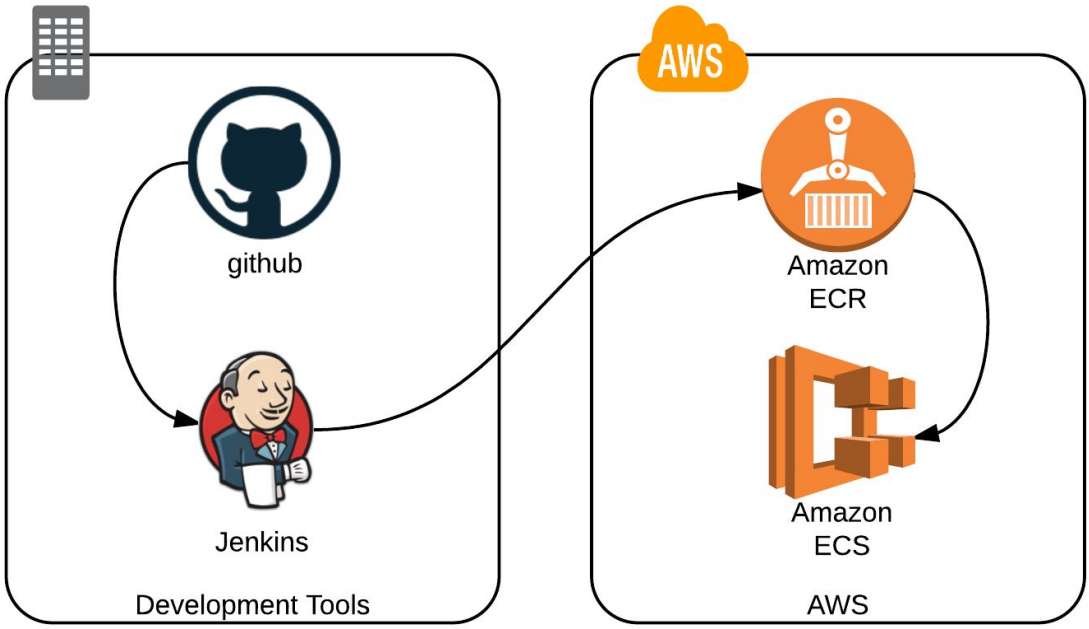
To achieve auto-discovery of microservices, consul can be added to the mix. A consul cluster consists of 3 or 5 nodes, which gives you a consistent, distributed key value store. By running an agent, all new microservices can be registered automatically to the consul cluster. A separate load balancer can be set-up to automatically detect new services in consul. At this point, when deploying a new service on the ECS cluster, it becomes immediately available through the load balancer on its own endpoint.

2. Architecture Diagram

2.1 The Delivery Pipeline

The delivery pipeline enables the developer to focus on what's really important: the app. Any change to any of the microservices can be picked up by Jenkins. Jenkins then triggers the rebuild of the Docker image. If the rebuild succeeds, the image can be pushed to Amazon ECR. To deploy the new changes on the dev / qa / production environment, Jenkins can change the ECS service definitions to make the new microservice available.

If the newly deploy application would not succeed its health checks, ECS will automatically roll back the older version.



2.2 Publicly facing microservices

Publicly facing Docker containers can be made reachable using the standard Elastic Load Balancer from AWS. The load balancer checks the health of the containers in different availability zones (different datacenters in AWS). If one of the containers would become unhealthy (e.g. datacenter failure), the load balancer will redirect all traffic to the healthy container.

Any number of docker containers can be added to EC2 instances. More EC2 instances can be added to allow more container capacity in a certain availability zone. Typically 2 availability zones are used to provide high availability.