SCALING DRUPAL TO THE CLOUD WITH DOCKER AND AWS

Dr. Djun Kim

Camp Pacific

OUTLINE

- Overview
- Quick Intro to Docker
- Intro to AWS
- Designing a scalable application
- Connecting Drupal to AWS services
- Intro to Amazon ECS (Elastic Container Service)

OVERVIEW

We’re OK launching a site with Acquia, Pantheon, Platform.sh or another hosted service.

OR…

We’re OK setting up a site on a hosted server (on AWS, Rackspace, Linode, Digital Ocean, …)

BUT…

We’d like to understand how to build a scalable cloud site using AWS services.

THE PLAN…

1. **Build a simple containerized site**
   1.1. Build an all-in-one Drupal site in a Docker container
   1.2. Deploy (by hand) to an AWS server
2. **Extend to use AWS services**
   2.1. Database
   2.2. Cacheing
   2.3. File Storage

THE PLAN…

3. **Scale out**
   3.1. Set up a load balancer
   3.2. Add instances
4. **Automate**
   4.1. Set up repository
   4.2. Set up ECS cluster
   4.3. Set up Task definition

DOCKER – WHAT IS IT?

- A way to package services (e.g. web applications) as self-contained, runnable, environment agnostic **containers**, easy to manage and deploy.
- A way to manage configuration at scale (e.g., need 100 identical LAMP stacks, need to spin up 10 new ones NOW)
- Fast, lightweight compared to VM virtualized environments
- If I build a container, it should run identically on my laptop or in a big cluster on the cloud.

1. SIMPLE DOCKER EXAMPLE - LET'S BUILD A DRUPAL

Find a pre-configured Docker image that has apache, php, mysql, memcache, Drupal pre-installed. Just fire it up and browse to the URL.

Configure, add some modules, make a beautiful theme.

To deploy, load it up on an AWS instance and run.

1. SIMPLE DOCKER EXAMPLE - STEP BY STEP

```
# Get "official" docker image
# See docs: https://hub.docker.com/_/drupal/
docker pull drupal:7
# wait while it builds
docker images
# Should see “drupal”. Now run it:
docker run --name simple-docker -p 8080:80 -d drupal
# Install Drupal - use local sqlite DB
# Save changes as a new image
docker commit some-drupal simple-docker-v01
# Save images as a tarball
docker save simple-docker-v01 > simple-docker-v01.tar
```

INTRO TO AWS

Amazon Web Services

Create an account. You need a credit card. But much of what you need to do in terms of experimenting/learning is free/cheap.

The first thing we'll want is an *instance* - a virtual server. You can configure these in all shapes and sizes.

Log in to AWS and head on over to the AWS EC2 (elastic compute cloud) dashboard. Click on **launch instance**. For our present purposes, we can select an AMI image and a t2.micro instance (small/free). Make sure we assign a public IP address. *Assign appropriate security groups*.

Authentication is via SSL certs. Store the cert AWS generates for you safely. Make sure you understand how to configure SSH to use appropriate SSL keys/certs for logging into your instances.

1.2 SIMPLE DOCKER EXAMPLE - DEPLOY TO AWS

```
# Move to host
%local: scp -i your-key simple-docker-v01.tar ec2-user@xx.xx.xx.xx:/home/ec2-user/
%local: ssh -i your-key ec2-user@xx.xx.xx.xx

# Verify that docker is installed on your instance. If not, can  install it via
'sudo yum docker'.
%aws-instance: docker -v
Docker version 1.12.6, build 7392c3b/1.12.6

%aws-instance: docker load < simple-docker-v01.tar
# Verify that the image is there
%aws-instance: docker images

# Now run it
%aws-instance: docker run --name example -p 80:80 -d simple-docker
```

PROS & CONS

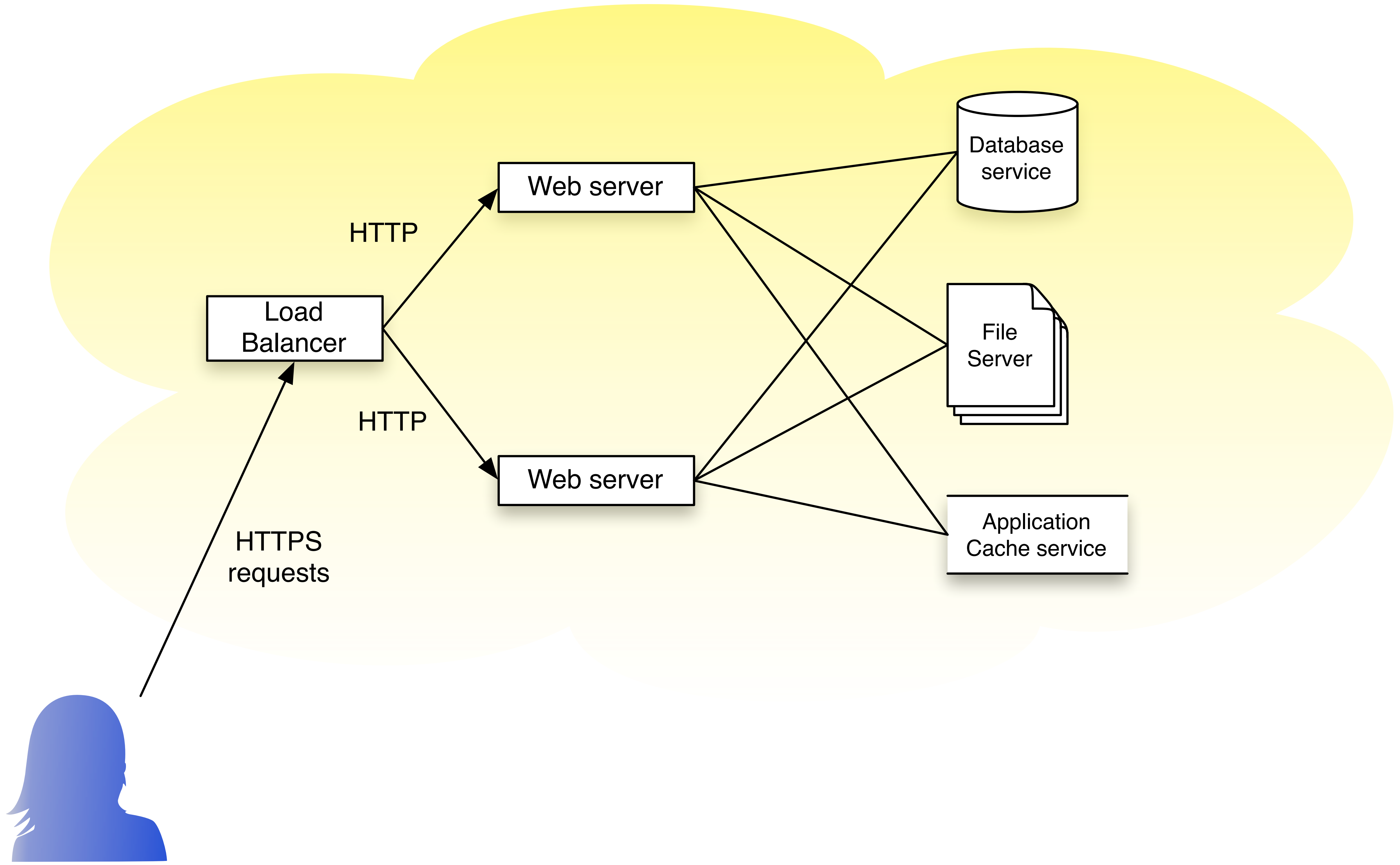
Pros

- Simple, no config
- Self-contained

Cons

- Need to use this as a base to allow any significant customization (e.g. adding code)
- Too many services in one container
- Doesn’t scale

2. USING AWS SERVICES – ARCHITECTURE



2.0 GETTING READY

- The Plan: take our site container, externalize the services:
  - Separate container for DB (more on this later)
  - Separate container for Caching service (redis)
  - Decouple the files from the container providing the web server
- Once this is working locally, we can replace the DB, Caching component, and File sharing with AWS services, without changing our Drupal/webserver container *at all*.

2.0 HOW? DOCKER-COMPOSE (FOR LOCAL)

- Setting up multi-container apps is possible with just plain docker, by linking, using shared volumes, etc. But it's not convenient.
- Enter *docker-compose*. Comes with Docker.app
- Best illustrated via an example
- Note: this example is based on the `wodby/docker4drupal` project

DOCKER-COMPOSE.YML

```yaml
version: "2"

services:

  drupal:
    image: pnwds_ecs_demo
    env_file:
      - .env
    ports:
      - "80:80"
    volumes:
      - ./docroot:/var/www/html

  mariadb:
    image: wodby/drupal-mariadb
    environment:
      MYSQL_RANDOM_ROOT_PASSWORD: 1
      # The simple way to override the mariadb config:
      MYSQL_DATABASE: ${DB_NAME}
      MYSQL_USER: ${DB_USER}
      MYSQL_PASSWORD: ${DB_PASS}

    volumes:
      - ./docker-runtime/mariadb:/var/lib/mysql
      - ./docker-runtime/mariadb-init:/docker-entrypoint-initdb.d

  redis:
    image: redis:3.2-alpine
```

2.0 DOCKER-COMPOSE FILE - COMMENTARY

- The docker-compose file defines three services
  - The web-head (`drupal`). This is a custom built docker image.
  - The database service (`mariadb`). OTS Dockerhub image
  - The caching service (`redis`). OTS Dockerhub image
- Files are shared between our local file system and the containers via *volumes*.
- The `drupal` service gets parameters passed in via an environment file (`.env`). This is a way of passing in secrets.
- The `mariadb` service gets parameters passed in from the environment (e.g. `${DB_NAME}`). There's better ways to do this in production.

2.0 RUN THE SITE LOCALLY

```
# http://docker4drupal.org/
git clone git@github.com:Wodby/docker4drupal.git
cd docker4drupal/

# Have a look at docker-compose.yml file
less docker-compose.yml

# Download drupal and move it to docroot/
drush dl drupal-7.51
mv drupal-7.51 docroot

# Create the docker-runtime directory
mkdir docker-runtime

# Copy a pre-existing Drupal DB into the mariadb-init directory
mkdir docker-runtime/mariadb-init
mv ~/pnwdsdemo.sql ../docker-runtime/mariadb-init/

# Tweak the docker-compose file
emacs docker-compose.yml
# Put our DB credentials into the Drupal settings file
emacs docroot/sites/default/files/settings.php

# Launch docker compose
docker-compose up -d

# Visit the site
open http://localhost
```

Thank you!

djun.kim@camppacific.com

@djun_kim (twitter)

GISTS

- *Note: these are from an earlier version of this presentation - they're a little more elementary in terms of assumed docker knowledge*
- **GET Docker.app**

  https://gist.github.com/djun-kim/5927705923305af1168a6bce517212f3
- **Monolithic container**

  https://gist.github.com/djun-kim/a11d6c15025019f39c805ee70ad57f35
- **Docker compose app**

  https://gist.github.com/djun-kim/e80274cd65b6edd464b73db3acba445d
- **Sample Docker-compose.yml file**

  https://gist.github.com/djun-kim/5475923e2d6e3a2a7d372b9041ced56c