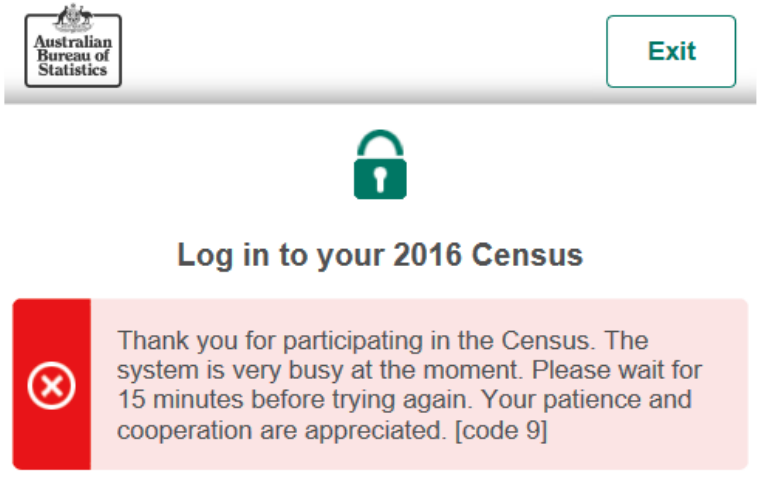
'Bad' Infrastructure



Building and Maintaining Awesome Applications on Google's Infrastructure

Presentation Maintained by

Eric Jiang

This presentation's code/slides can be found on
https://github.com/lorderikir/googlecloud-techtalk

Google Cloud

Slide Deck Version: v3.0.0-monash-esol

Talk Summary

1. Introduction to Google Cloud Platform
2. What is Google App Engine
    a. GAE Environments
    b. Databases
3. Deep-Dive
    a. Improving Turnaround Times
4. Other Tools

Introduction

What is Google Cloud Platform?

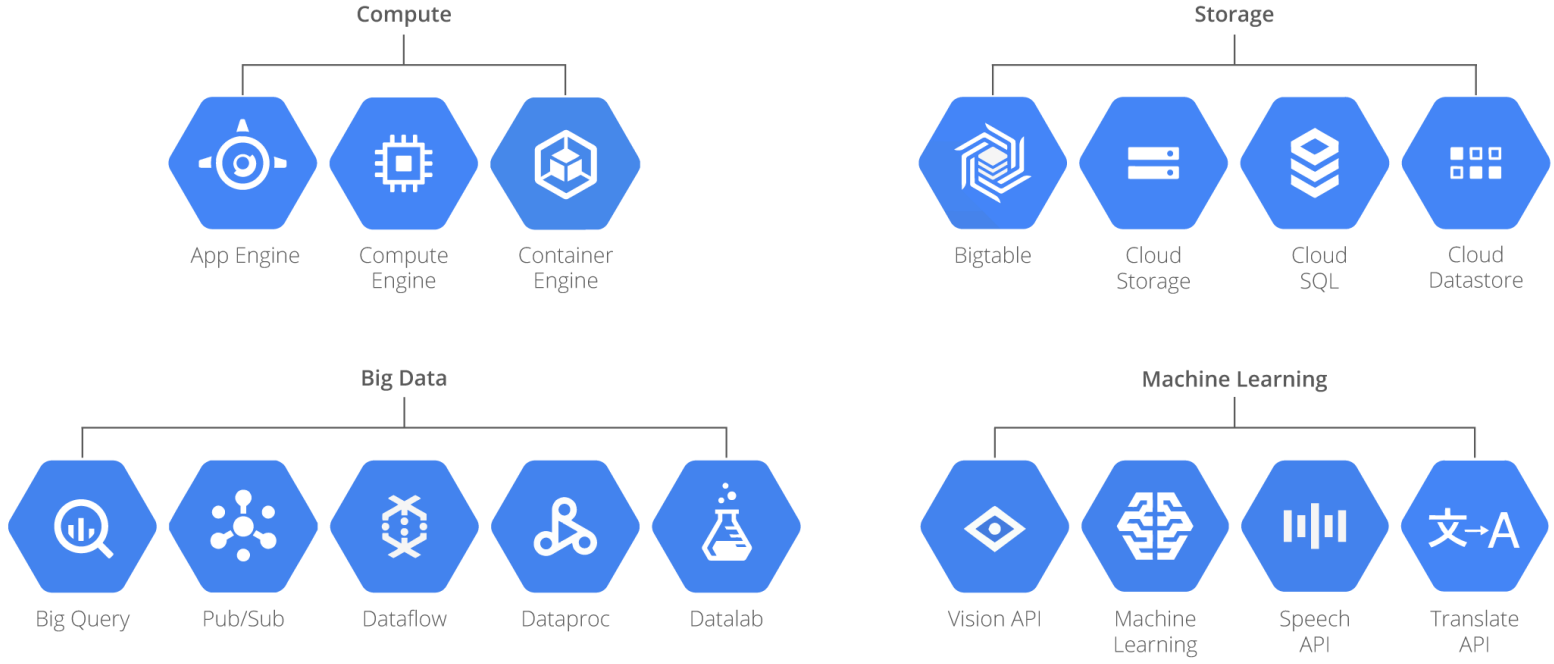
> Google Cloud Platform lets you build and host applications and websites, store data, and analyze data on Google's scalable infrastructure.

Composes of many applications, such as:

- Google App Engine (GAE)
- Google Kubernetes Engine (GKE) Previously known as Google Container Engine
- Google DataStore
- Cloud ML (built off *TensorFlow*)
- and much more

Did you know that Firebase and DialogFlow (API.AI) are both built on Google Cloud Platform

Google Cloud Platform



Google App Engine

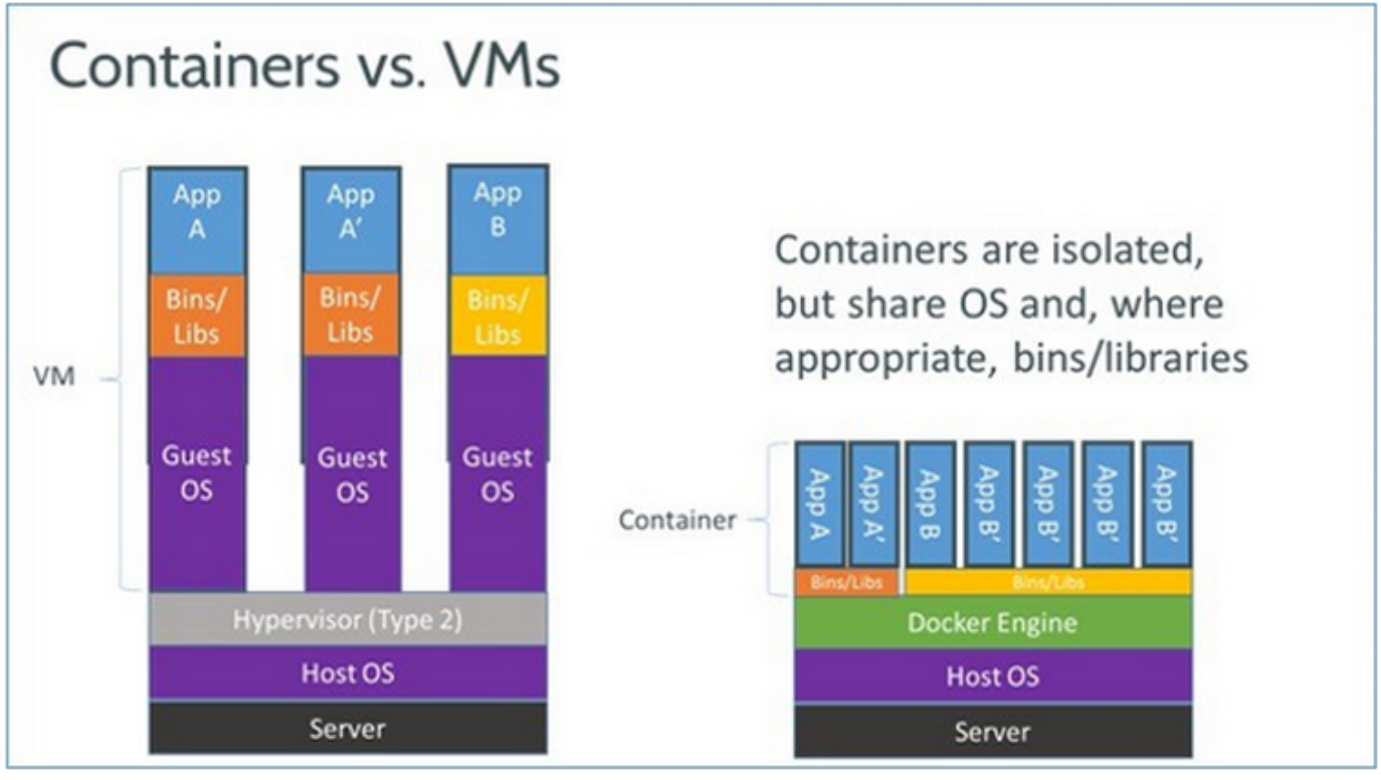
- designed around the fact that *Google just can't send everyone into their datacentre(s) and update applications across their many datacenters*
- Built off Remote Deployments

Language	Environment
Java 7 (and Kotlin[1])	Standard
Java 8	Standard (Beta)/Flexible
Node.js	Flexible
Python 2.7	Standard
Python 3.5	Flexible

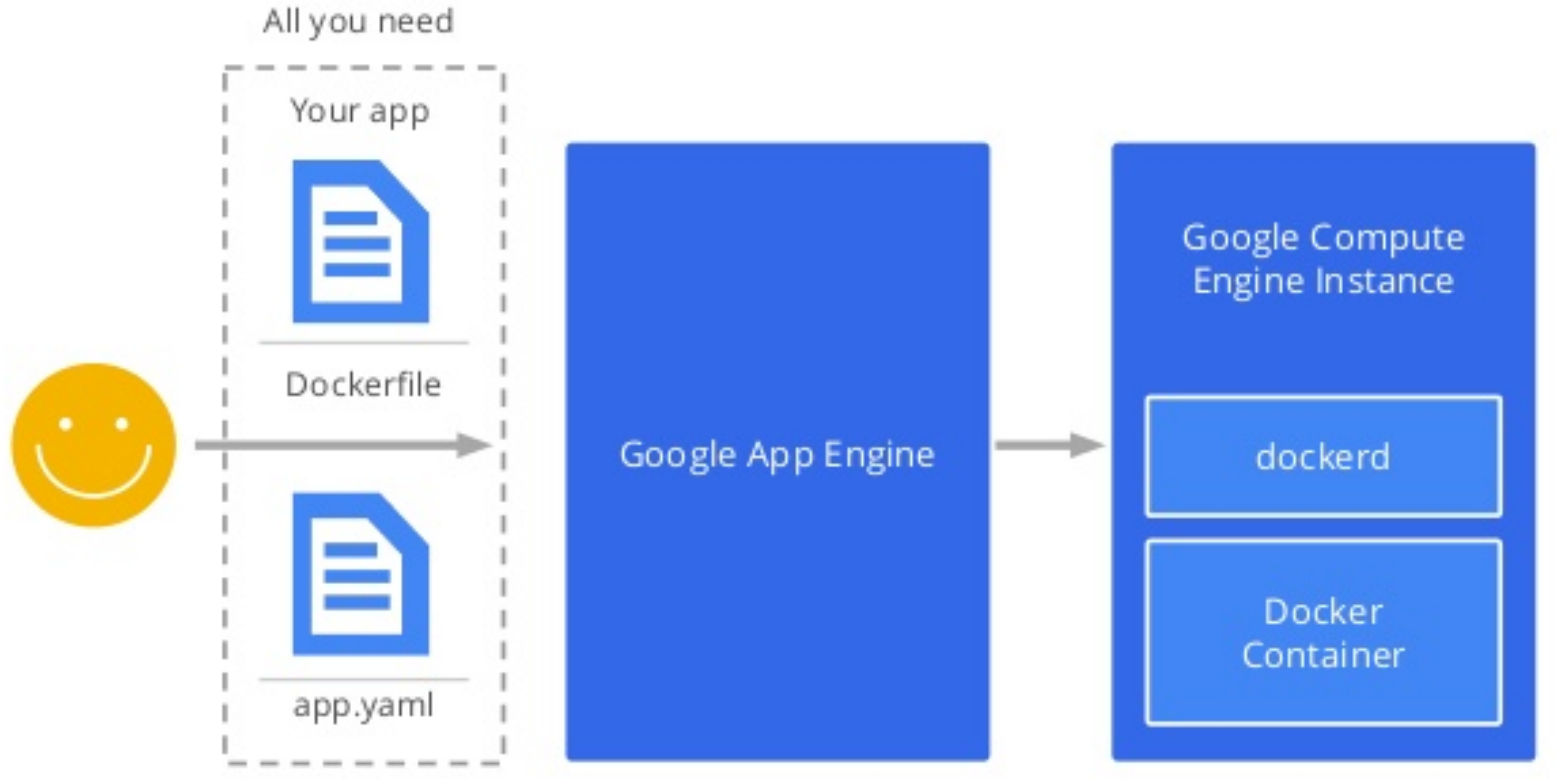
[1] This for you Kotlin fans out there

Standard Environments run in a specialised envrionment. Though building the application is more constrained then other environments, it means scaling up is faster.

Flexible Environment applications run off a Docker container, it is designed for applications that recieve constant traffic. When deployed they are Google Compute Engine as Virtual Machines*



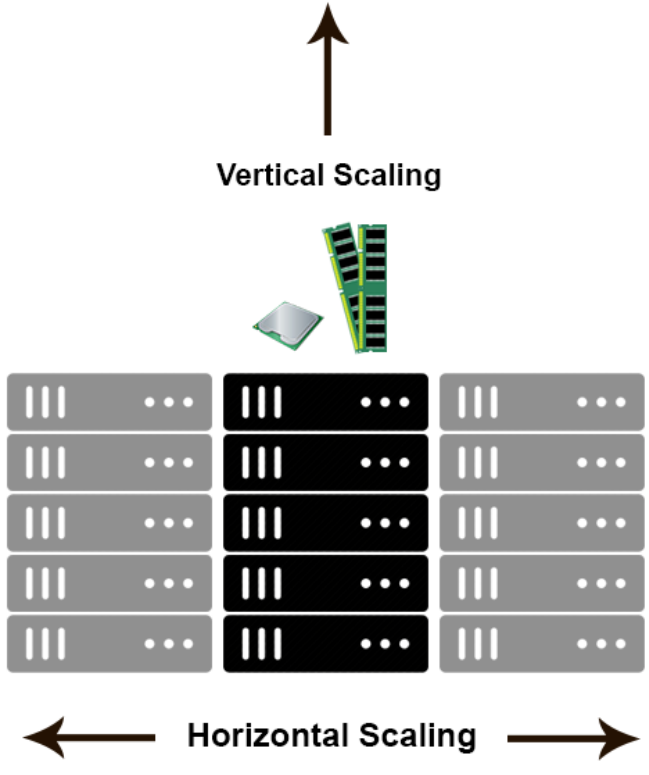
* Because they run off Docker, you can write your own Dockerfile Configuration to deploy, and deploy it anywhere, you can even move it to AWS


All you need
Your app
Dockerfile
app.yaml
Google App Engine
Google Compute Engine Instance
dockerd
Docker Container

Scaling Modern Web Applications

Me when I look at Scaling:

What is the difference between Vertical and Horizontally Scaling



Horizontal Scaling: scale by adding more machines into your pool of resources machine.

Vertical Scaling: scale by adding more power (CPU, RAM) to an existing machine.

> An easy way to remember this is to think of a machine on a server rack, we add more machines across the horizontal direction and add more resources to a machine in the vertical direction.

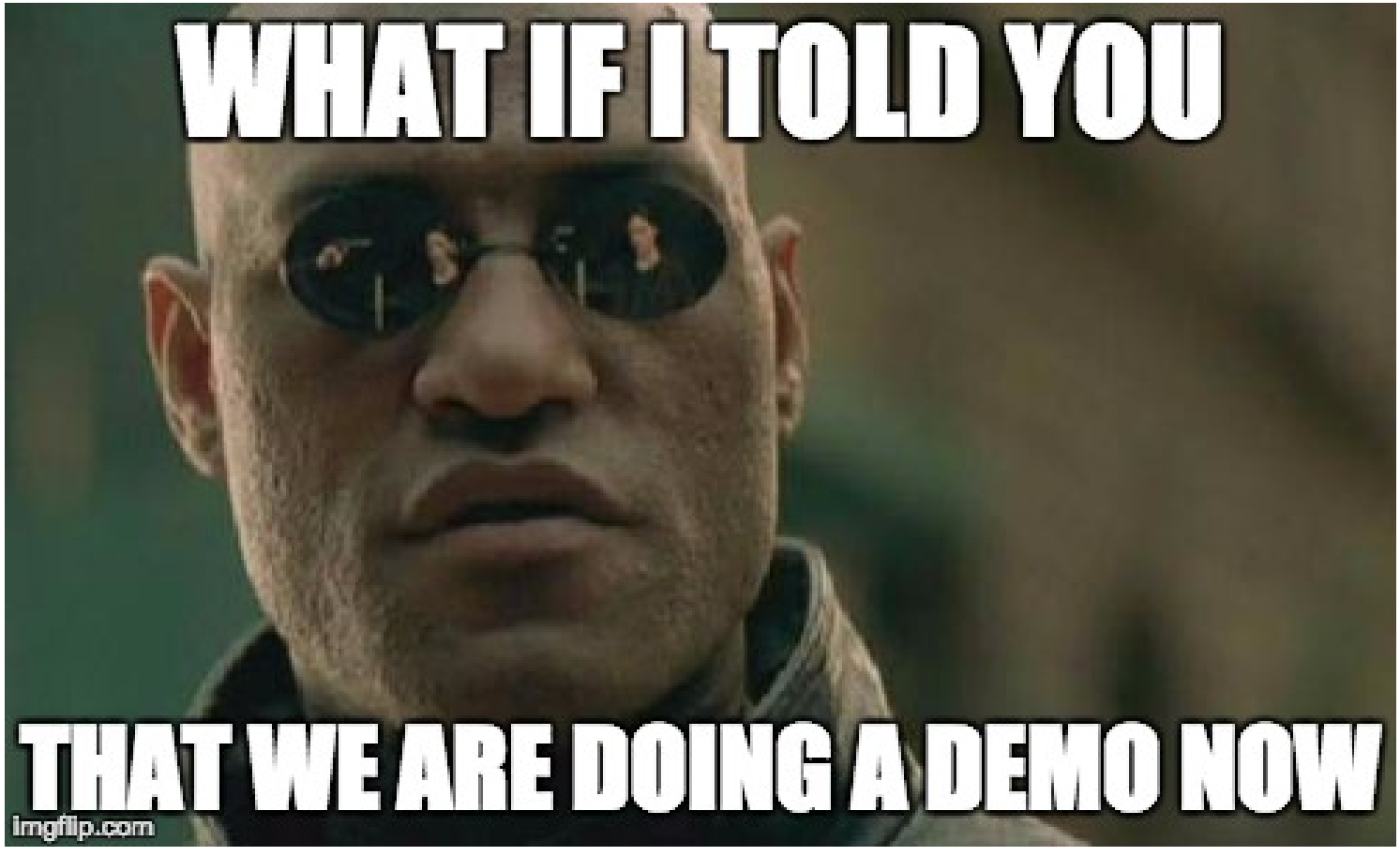
Benefits of Horizontal Scaling

- Dynamic scaling allows spinning up more instances and nodes faster, i.e. if you suddenly get a influx of traffic
- Vertical Scaling is limited to capacity of resources, simply adding more resources
- Just simplying load testing isn't good enough, examples of this include Niantic (PokemonGo) and Australian Census 2016

Scaling in Action: The monPlan Stack

- The frontend and backend has been seperated into 2 projects. The backend stores the information for the frontend,
- So when an increase in load happens, the frontend scales up, but not necessarily the backend, as the frontend requests information when necessary

Demo Section



Deploying to Google App Engine (GAE)

- Due to time constraints, I'm going to skip this section.
  You can read more about it here: https://d.pr/BDZnDr

Improving Turnaround with StackDriver Monitoring

Stackdriver

> Google Stackdriver is a freemium cloud computing systems management service offered by Google. It provides performance and diagnostics data to public cloud users.

- Allows monitoring of not only GCP Services, but also AWS, Azure and Your Own Infrastructure
- Monitoring Traffic, Performance of GAE Instances and Stacktrace allows faster turnaround time

Deep Dive

StackDriver Console: https://d.pr/y9DN9f

Improving Analytics using BigQuery and StackDriver Logging

Other Tools Available on GCP that you play with

- Cloud ML (Google Cloud Machine Learning) which is built off TensorFlow
- Compute Engine - Google VMs
- Container Engine - built off Kubernetes and allows deployment of custom applications
- Cloud Storage - CDN provider of files (like *Amazon S3*)
- Network Balancer - for Load Balancing of traffic for your applications
- Cloud ML APIs such as Natural Language Processing, Data Loss Prevention, etc.

Questions