Server to Cloud Migrations
How we run server products on stateless cloud services

@CRAIGKNOTT92
Server to Cloud Migrations

How we run server products on stateless cloud services

@CRAIGKNOTT92
We believe behind every great human achievement, there is a team. Our mission is to unleash the potential of every team.
More than 100 teams
~600 core services
Hundreds of changes/day
Multiple products
Many, many, many, many customers
Customer Support Technology
Customer Support Technology
(SRE)
What is Jira Datacenter
Once upon a time...
“Sorry, we can’t support this anymore. You’ll have to run it yourself... Somewhere else

VM PLATFORM MANAGER
Architecture
Architecture
Architecture
WHAT’S THE PROBLEM?
WHAT’S THE PROBLEM?
INDIVIDUALLY MANAGED
WHAT'S THE PROBLEM?
INDIVIDUALLY MANAGED CLUSTER INCONSISTENCIES
WHAT’S THE PROBLEM?

INDIVIDUALLY MANAGED

CLUSTER INCONSISTENCIES

UNSTABLE
WHAT’S THE PROBLEM?
INDIVIDUALLY MANAGED
CLUSTER INCONSISTENCIES
UNSTABLE

WHAT ARE WE GOING TO DO?
Migrate to Cloud Services!
Core Requirements

- Easy to use
  - Maintain and improve upon

- Auditing
  - All changes are peer reviewed or have some audit trail

- Image Management
  - Instead of server management
TOOLING
What tools do we need

Orchestration

Image Creation

Configuration
Orchestration

The automated arrangement, coordination, and management of computer systems, middleware, and services
CLOUDFORMATION

- Support for all AWS resources on release
- State Management

TERRAFORM

- Support for multiple providers
- Open Source
- Modularization
- Change Verification
CLOUDFORMATION

- Support for all AWS resources on release
- State Management

TERRAFORM

- Support for multiple providers
- Open Source
- Modularization
- Change Verification
What tools do we need

- Terraform
- Image Creation
- Configuration
Packer!

Multiple Providers
Just like terraform supports multiple providers, Packer can create images for multiple providers.

Machine Images
The perfect tool for creating machine images in a repeatable, consistent and peer reviewed way.

Open Source
Missing a feature? Work with the open source community to implement it.
What tools do we need

Terraform

Packer

Configuration
What tools do we need

- Terraform
- Packer
- Ansible
{  
  "description": "Our Application",
  "variables": {},
  "builders": [{
    "type": "amazon-ebs",
    "access_key": "{{user `aws_access_key`}"
    "secret_key": "{{user `aws_secret_key`}"
    "region": "{{user `region`}"
    ...
  }]
}
AMI Creation With Packer

`packer.io`

```json
{
  "provisioners": [
    {
      "type": "ansible",
      "playbook_file": "../ansible/base.yml",
      ...
    }
  ],
}
```
How does this look in Terraform?
Terraform Structure

Overview

Module Layout

Environments

- environments
  - development
  - staging
  - production
  - management

Modules

- app_tier
- db_tier
- vpc
- vpc_peering
Terraform Structure

Overview

Module Layout

Environments

app_tier
- load_balancing.tf
- autoscaling.tf
- iam.tf
- security_groups.tf
- outputs.tf
- variables.tf

vpc
- main.tf
- outputs.tf
- variables.tf
Terraform Structure

Overview

Module Layout

Environments

```terraform
app_tier
- load_balancing.tf
- autoscaling.tf
- iam.tf
- security_groups.tf
- outputs.tf
- variables.tf

code:

output "vpc_id" {
  value = "${module.vpc.id}" 
}

variable "vpc_cidr" {
  description = "VPC CIDR"
}
```
Terraform Structure

Overview

Module Layout

Environments

development
- productA
  - main.tf
  - terraform.tfvars
- productB
  - main.tf
  - terraform.tfvars
management
- vpc
  - main.tf
  - terraform.tfvars
Terraform Structure

Overview

Module Layout

Environments

development

- productA
  - main.tf
  - terraform.tfvars
- productB
  - main.tf
  - terraform.tfvars

management

- vpc
  - main.tf
  - terraform.tfvars

module “app_tier” {
  ...
}

module “vpc”{
  ...
}

vpc_cidr = “10.1.0.0/21”
Terraform CI/CD with Bitbucket Pipelines

1. Start
2. Terraform PR
3. Build/Deploy
4. Approve/Merge
5. 1Click Promote Stg/Prod
6. End
PACKER CI/CD WITH BITBUCKET PIPELINES

START

Packer PR

Approve/Merge

1Click Promote Stg/Prod

Build/Deploy

Redeploy Terraform

CHECKMARK
Monitoring may seem confusing
Realities of a growing business

- Number of Services
- Number of Monitors
Scaling up Monitoring

\[
\text{Amount of services} \uparrow + \text{Amount of monitoring} \uparrow = \text{Time spent on building monitoring?}
\]
Scaling up Monitoring

Amount of services \uparrow + \text{Amount of monitoring} \uparrow = \text{Time spent on building monitoring?} \uparrow \quad \text{Probably but you want it} \downarrow
You remember terraform?
We can use that with Datadog too....
<table>
<thead>
<tr>
<th>Method</th>
<th>Effort Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manual</td>
<td>$y = n \cdot \log(n)$</td>
</tr>
<tr>
<td>Terraform</td>
<td>$y = 10 \cdot \log(n)$</td>
</tr>
</tbody>
</table>

![Graph showing effort vs. number of monitors for Manual and Terraform methods](attachment:image.png)

**Number of Monitors**

**Effort**
TERRAFORM CI/CD WITH BITBUCKET PIPELINES

START

Edit Terraform Config

Bitbucket PR

Approve/Merge

Build/Deploy

1Click Promote to Staging/Prod
Planning Migrations
How 2 Migrate

Pre-Configure
New environment & application stack configured and ready to start

Migrate Data
Database, Shared Filesystem, Other Persistent Data

Cutover DNS, or Abort
Switch DNS to new stack, or abort migration
Migration Fails
Involve stakeholders early
Will there be downtime? How long? Can we minimise it? Is it worth the effort to minimise it?

Document, document, document
Anyone on your team should be able to pick up your migration plan, understand what needs to be done and execute it.

You can spend an infinite time planning
Try stick to some kind of deadline, but be flexible you won’t get to the perfect plan.
Pre-configure New Environment
Prepare anything that can be done before the migration starts

Secure Data
Ensure customer data will be protected during the migration
Automate Everything
If you feel like you need to ssh and manually run some command, script it instead

Don’t be afraid to abort
If something unexpected occurs you’re likely better off aborting and trying again another day

Have a test plan
Thoroughly test your migration plan in the pre-production environments
We did it!
Improvements
WE THOUGHT, GREAT!
WE THOUGHT, GREAT! BUT HOW CAN WE MAKE THIS EVEN BETTER!?
Core Capabilities

Ticket Creation

Updating Tickets
Wrapping Ticket Creation API
Quick Recap
TOOLING
Cloud Services are not for everyone, but they are for almost everyone.

Use the right tooling to make your life easier in the long run.

Invest time in planning before pushing on with migrations.

Iterate and Improve to reduce impact of failure & improve resiliency.
Thank you!

@CRAIGKNOTT92