Terraform at Adobe
Kelvin Jasperson
Introduction
Systems Engineer @ Adobe Audience Manager (AAM)
Been with Adobe for 18 months

AAM was acquired by Adobe in 2011, and is 100% in AWS

Twitter- @zxjinn
Raise your hands

• Who knows what Terraform is?
• Who uses Terraform?
• ... in production?
Terraform

• Infrastructure as code
• Supports many providers
  • AWS
  • Azure
  • Digital Ocean
  • Google Cloud
  • Heroku
  • OpenStack
  • VMware vSphere/vCloud Director
  • others...
Why Terraform?

• Fun to write
• Easy to extend with modules
• Shows the execution plan (no-op)
• State stored in a committable file
Basic Terraform Example
Basic Terraform Example

```bash
$ cat main.tf
resource "aws_instance" "app" {
  ami = "ami-d1f482b1"
  count = 5
  instance_type = "t2.micro"
}
$ terraform plan
+ aws_instance.app.0...
+ aws_instance.app.1...
$ terraform apply
aws_instance.app.0: Creating...
Apply complete! Resources: 5 added, 0 changed, 0 destroyed.
$
It worked! Parallel, takes ~1 min
Basic Terraform Destroy

$ terraform destroy
Do you really want to destroy?
  Terraform will delete all your managed infrastructure.
  There is no undo. Only 'yes' will be accepted to confirm.
  Enter a value: yes

aws_instance.app.0: Destroying...

Apply complete! Resources: 0 added, 0 changed, 5 destroyed.
$
It worked! Parallel, takes ~1 min

<table>
<thead>
<tr>
<th>Instance ID</th>
<th>Instance Type</th>
<th>Availability Zone</th>
<th>Instance State</th>
<th>Status Checks</th>
</tr>
</thead>
<tbody>
<tr>
<td>i-81f33334</td>
<td>t2.micro</td>
<td>us-west-1c</td>
<td>terminated</td>
<td></td>
</tr>
<tr>
<td>i-1df333a8</td>
<td>t2.micro</td>
<td>us-west-1c</td>
<td>terminated</td>
<td></td>
</tr>
<tr>
<td>i-1ef333ab</td>
<td>t2.micro</td>
<td>us-west-1c</td>
<td>terminated</td>
<td></td>
</tr>
<tr>
<td>i-33f23286</td>
<td>t2.micro</td>
<td>us-west-1c</td>
<td>terminated</td>
<td></td>
</tr>
<tr>
<td>i-34f23281</td>
<td>t2.micro</td>
<td>us-west-1c</td>
<td>terminated</td>
<td></td>
</tr>
</tbody>
</table>
More than just EC2 instances

• S3- Simple Storage Service
• CloudFormation
• VPC- Virtual Private Cloud
• SQS- Simple Queue Service
• Route53- Hosted DNS
• RDS- Relational Database Service
• IAM- Identity and Access Management
• ECS- EC2 Container Service
• others...
Modules, Compositions, and Clusters
Modules

• Self-contained reusable code
• Behavior changes based on inputs
• Terraform code
Compositions

• Pre-defined collections of modules
• Passes parameters to many modules
• Terraform + Jinja
Clusters

• Passes params to one composition
• Ultimate source of truth
• YAML
For example

• Module
  • VPC module- NAT and Bastion instances, security groups, etc
  • App1 module- App1 Instances, SQS queues, S3 buckets, subnets
  • DB1 module- RDS instances, security groups
  • Admin module- Instances- config management, monitoring, etc

• Composition
  • Edge composition- VPC, App1, DB1, Admin
  • DataProcessing composition- VPC, App2, DB2, Admin
  • Delivery composition- VPC, App3, Admin
Analogous to modern Puppet design

• Terraform Modules = Puppet Modules
• Compositions = Roles and Profiles
• Clusters = ENC and Hiera
Ops wrapper

• Reads cluster YAML variables
• Reads composition (.tf.jijna2), writes Terraform (.tf) files with cluster variables injected
Demo!
The Future

• Jenkins runs Terraform and commits statefile
• Web interface to generate cluster YAML files for self service
• Pending discussion: ops wrapper generates Terraform JSON instead of parsing jinja
Lessons Learned, Best Practices

• A springboard for Terraform (ops wrapper for us) is invaluable
• Terraform HCL + Jinja templates are easier to write and read than Terraform JSON
• Make 1 cluster = 1 vpc = 1 environment = 1 purpose
  • Reproducible environments
  • Separated Terraform statefiles per cluster
• Version user data in a map variable
• Symlink shared Terraform files in modules
• Separate “common” infrastructure like- S3 buckets, SQS, IAM to its own cluster
Don’t

• Get impatient with Terraform
• Go in guns blazing and use it in production on day 1
• Skip reading the Terraform docs
Woohoooo!

• 85% of our production infrastructure is managed with Terraform!
Questions?