Infrastructure as Code

CS398 - ACC

Prof. Robert J. Brunner

Ben Congdon
Tyler Kim

MP7

How’s it going?

Final Autograder run:

- Tonight ~8pm
- Tomorrow ~3pm

● Due tomorrow at 11:59 pm.

● Latest Commit to the repo at the time will be graded.

● Last Office Hours today after the lecture until 7pm.

Infrastructure as Code

Problem Statement

- Distributed applications...
    - Are sensitive to how they are configured
        - i.e. Needs of a database server will be different than an web server
    - Are updated continuously
        - New code and patches are deployed daily, if not hourly
    - Will be operated by teams of humans
        - i.e. Possibility of “operator error”
    - Run on tens/hundreds/thousands of nodes

How do we deploy our Cloud infrastructure?

Approaches:

- **Setup everything manually!**
  - Does this scale? Clearly no.

How do we deploy our Cloud infrastructure?

Approaches:

- **Setup everything manually!**
  - Does this scale? Clearly no.
- **Custom scripts**
  - Use your cloud provider’s API to create machines
  - Programmatically SSH into the machine to do tasks
  - Does this scale? Maybe… but why reinvent the wheel?

How do we deploy our Cloud infrastructure?

Approaches:

- **Setup everything manually!**
  - Does this scale? Clearly no.
- **Custom scripts**
  - Use your cloud provider’s API to create machines
  - Programmatically SSH into the machine to do tasks
  - Does this scale? Maybe… but why reinvent the wheel?
- **Infrastructure as Code**
  - Declare your infrastructure setup in a specific format
  - Your IaC framework deploys/updates your cloud infrastructure!
  - Does this scale? Yes!

Infrastructure as Code Ideas

- Approaches to “writing down” cloud configuration:
    - **Declarative:** Define the target state of your cloud. *What* should the eventual cloud deployment look like?
    - **Imperative:** Define how the configuration system should setup the cloud. *How* should the system deploy your application?
    - **Intelligent:** Define relationships and constraints between services, and the system will figure out *how* and *what* to update.

Infrastructure as Code Ideas

- Approaches to updating cloud configuration:
  - **Push**: A central server tells child servers their configuration
  - **Pull**: Child servers request configuration from a central server

Infrastructure as Code Solutions

- **Ansible:** Declarative/Imperative; Push
- **Puppet:** Declarative; Pull
- **Chef:** Imperative; Pull
- **Salt**: Declarative
- **Terraform:** Declarative/Intelligent; Push

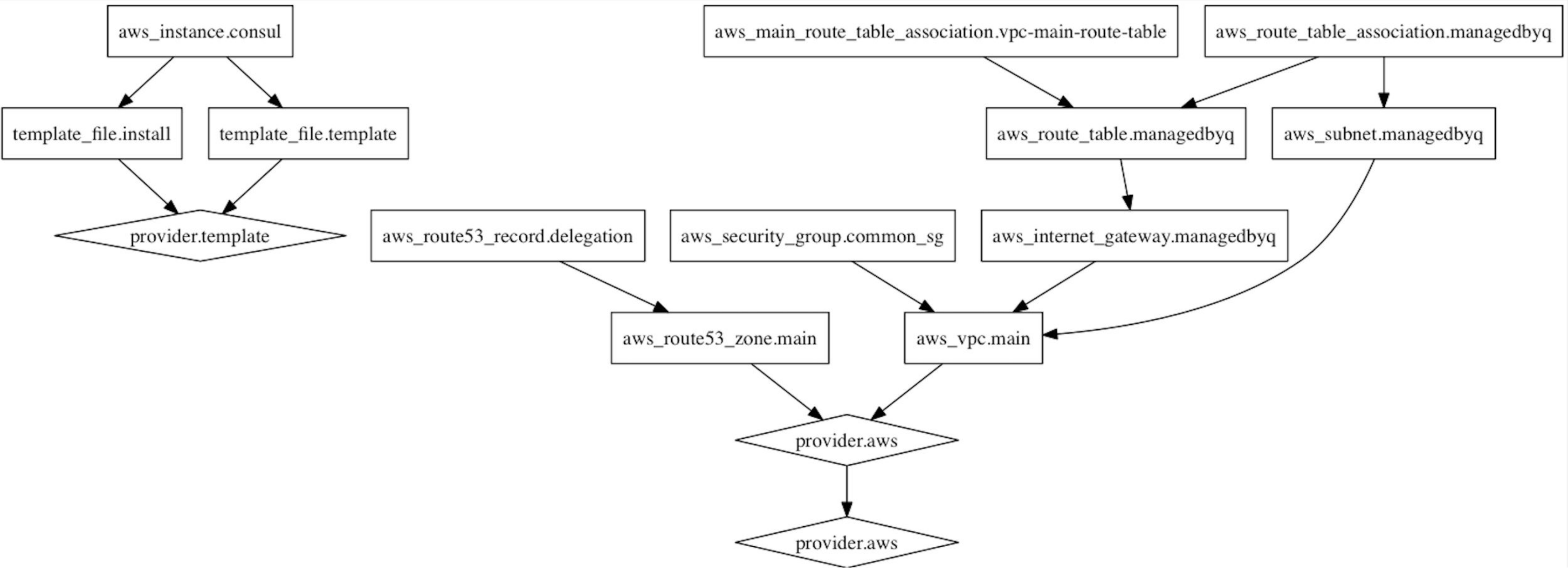
Terraform

- Created by HashiCorp; Open source

- Cloud Platform Agnostic
  - Support for AWS, GCP, Azure, Kubernetes, Heroku, and a bunch more

- Stateful and environment aware
  - Internal resource graph used to create cloud resources in the correct order
  - Internal state and configuration can be easily version-controlled

Terraform Resource Graph


aws_instance.consul
template_file.install
template_file.template
provider.template
aws_main_route_table_association.vpc-main-route-table
aws_route_table_association.managedbyq
aws_route_table.managedbyq
aws_subnet.managedbyq
aws_route53_record.delegation
aws_security_group.common_sg
aws_internet_gateway.managedbyq
aws_route53_zone.main
aws_vpc.main
provider.aws
provider.aws

https://github.com/hashicorp/terraform/issues/1997

Terraform Definitions

- Provider:
  - Interacts with a cloud service (i.e. GCP, AWS, Azure)
  - Affects change in a cloud service (i.e. creating/destroying resources) using the service’s API

- Resource:
  - An infrastructure component
  - i.e. VMs, Networks, Containers, Hard Drives, Storage Buckets

Terraform Modules

- Terraform uses *.tf files for configuration
- Common semantics:
  - variables.tf
    - Hold variables that may change over the lifetime of the configuration
    - i.e. Instance sizing, database table names, etc.
  - main.tf
    - Import variables and any necessary modules.
  - Others (i.e. ec2.tf)
    - Service-specific configuration
    - Usually 1-file-per-service (i.e. one for EC2, and another for DynamoDB)

Terraform Syntax

- Basic configuration language that supports some interpolation, but is generally declarative

- Useful to lookup and use examples
    - Many open-source Terraform templates are available

Terraform Syntax

```
# An AMI
variable "ami" {
  description = "the AMI to use"
}

/* A multi
   line comment. */
resource "aws_instance" "web" {
  ami               = "${var.ami}"
  count             = 2
  source_dest_check = false

  connection {
    user = "root"
  }
}
```

Terraform Syntax

```
# An AMI
variable "ami" {
  description = "the AMI to use"
}

/* A multi
   line comment. */
resource "aws_instance" "web" {
  ami               = "${var.ami}"
  count             = 2
  source_dest_check = false

  connection {
    user = "root"
  }
}
```

Variable

Resource

Variable Interpolation

Terraform Syntax

Resource Type
(Defined by Provider)

Resource Name
(Defined by You)

```
resource "aws_instance" "web" {
  ami               = "${var.ami}"
  count             = 2
  source_dest_check = false

  connection {
    user = "root"
  }
}
```

Terraform Commands

- terraform get
    - Downloads and updates local terraform modules
- terraform plan
    - Creates an execution plan to transform the state in your cloud to the state of your current local configuration
- terraform apply
    - Runs the execution plan, and creates/updates/deletes resources in your cloud as necessary
    - Can be a destructive action if you're not careful!

Terraform Use Cases

- Complex, Multi-Tiered Applications
  - Terraform modules are easily composable into complex architectures

- Temporary Environments
  - Useful for creating staging/testing environments
  - Can create identical infrastructure setup to production environment

- Deploying Across Multiple Cloud Providers
  - Terraform is platform agnostic
  - Similar configurations can be automatically replicated across clouds

Wednesday

Terraform Demo

Final Project Office Hours

MP Office Hours

MP8

Terraform

This MP will run on individual GCP. Please read the documentation.

Due Next Tuesday