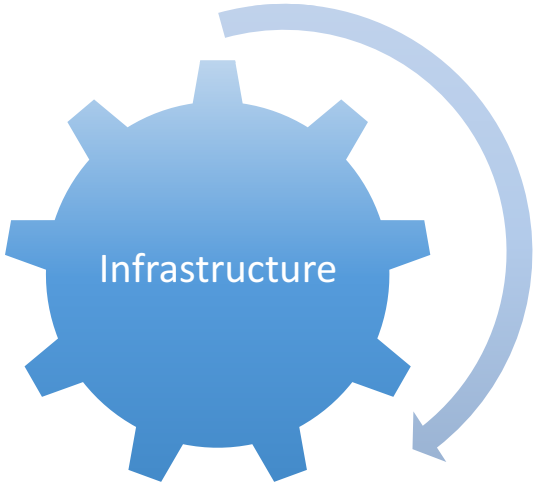
Terraform

Introduction

Gustavo Hoirisch
2016

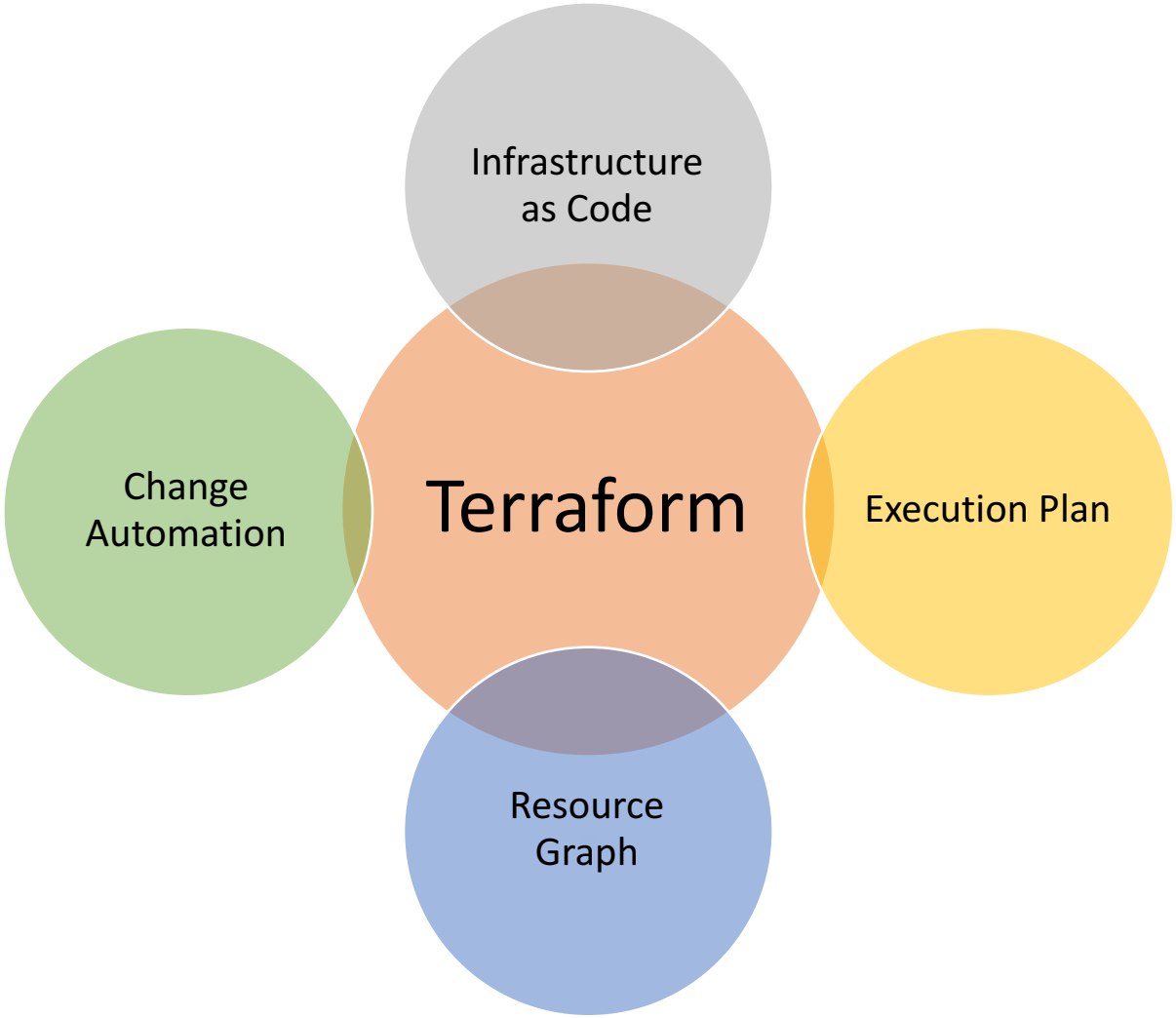
What is Terraform?

Building
Updating
Versioning



Safely
Efficiently

Key Features



Getting Started

https://www.terraform.io/intro/getting-started/install.html

Installing:

- Using Homebrew:

```
$ brew install terraform
```

OR

- Download and run the binary
  https://www.terraform.io/downloads.html

```
# Checking if terraform is installed
$ terraform –v
> Terraform v0.7.5
```

AWS and ADFS

aws-adfs

- https://github.com/venth/aws-adfs

```
$ aws-adfs login --adfs-host=adfs.myob.com.au --profile=default
```

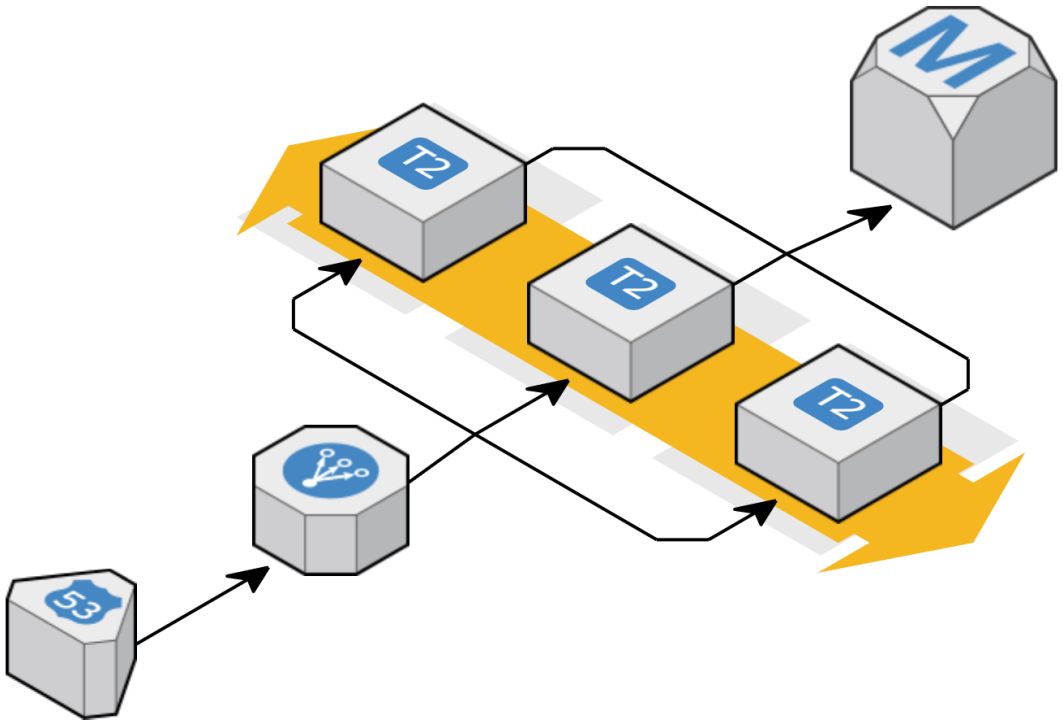
Install with Python PIP

```
$ pip install aws-adfs
```

Use Cases

- Multi-Tier Applications
- Disposable Environments
- Software Defined Networking

Hands On



Hands On

Demo

1. https://asciinema.org/a/8o061zy0heoonz8hqktahq3gq?speed=2 - apply, plan, destroy
2. https://asciinema.org/a/55z3b93s09p2r3w081p7e91wa?speed=2 - referencing components
3. https://asciinema.org/a/8wla2amsf24gxnx9khmodt90c?speed=2 - variables
4. https://asciinema.org/a/9x4rd7y51dbd1wllvb9szppfx?speed=2 - splitting into files

Variables

declaring

```
# No default value
variable "region" {
        type = "string”
        description = "AWS region”
}

# Default Value
variable "instance_type" {
        type = "string"
        description = "Instance Type"
        default = "t2.small”
}
```

```
# List – like arrays
variable “my_list” {
        type = “list”
        default = [“a-value”, “another-value”]
}

# Maps
variable “amis” {
        type = “map”
        default = {
                us-east-1 = "image-1234"
                us-west-2 = "image-4567"
        }
}
```

- > “${var.region}”, “${var.instance_type}”, “${var.my_list.0}”, “${var.amis[“us-east-1”]}”

Refactoring

- Too much repetition: “us-west-2”
  - Can become hard to change – too many places to update, easy to forget
  - Solution: extract the values into variables/constants

- Single file becomes too large to read!
  - Solution: split into logical block, i.e.: IAM.tf, VPC.tf, ASG.tf, etc…

Variables

assigning

- terraform.tfvars
  - key = value pair

    region = “us-west-2”

    instance_type = “m3.medium”

- Environment Variables

  TF_VAR_region=us-west-2 TF_VAR_instance_type=m3.medium terrafrom plan

- Command Line Flags

  terraform plan –var region=us-west-2

Debugging

https://www.terraform.io/docs/internals/debugging.html

- Dependency Graph

https://www.terraform.io/docs/commands/graph.html

```
$ brew install graphviz
$ terraform graph | dot -Tpng > graph.png
$ open graph.png
```

- Read the errors
  - Feedback will normally tell you the problem

- TF_LOG – DEBUG, INFO, WARN, TRACE

Advanced

- Remote State
  - share tfstate with other users in a central location
- Data Sources
  - allows a Terraform configuration to build on information defined outside of Terraform
    - Example: get an AMI ID from AWS
- Builtin Functions
  - https://www.terraform.io/docs/configuration/interpolation.html
  - CIDR Block calculation, string formating, merging, md5, etc…

Advanced

continued

- Modules
  - Common resource structures should be reused across projects
    - For MYOB: replicate the OPS Standards, implement sensible defaults
      - https://github.com/MYOB-Technology/platform-terraform
      - https://github.com/MYOB-Technology/EX-Terraform
    - https://github.com/hashicorp/best-practices/tree/master/terraform/providers/aws
    - https://github.com/segmentio/stack

Questions

Terraform Away

https://www.terraform.io/docs/providers/aws/index.html

- $ terraform plan
- $ terraform apply
- $ terraform destroy

Thank you

Gustavo Hoirisch
2016