Managing State in Terraform

Ned Bellavance

FOUNDER, NED IN THE CLOUD LLC

@ned1313 www.nedinthecloud.com

Overview

State data exploration

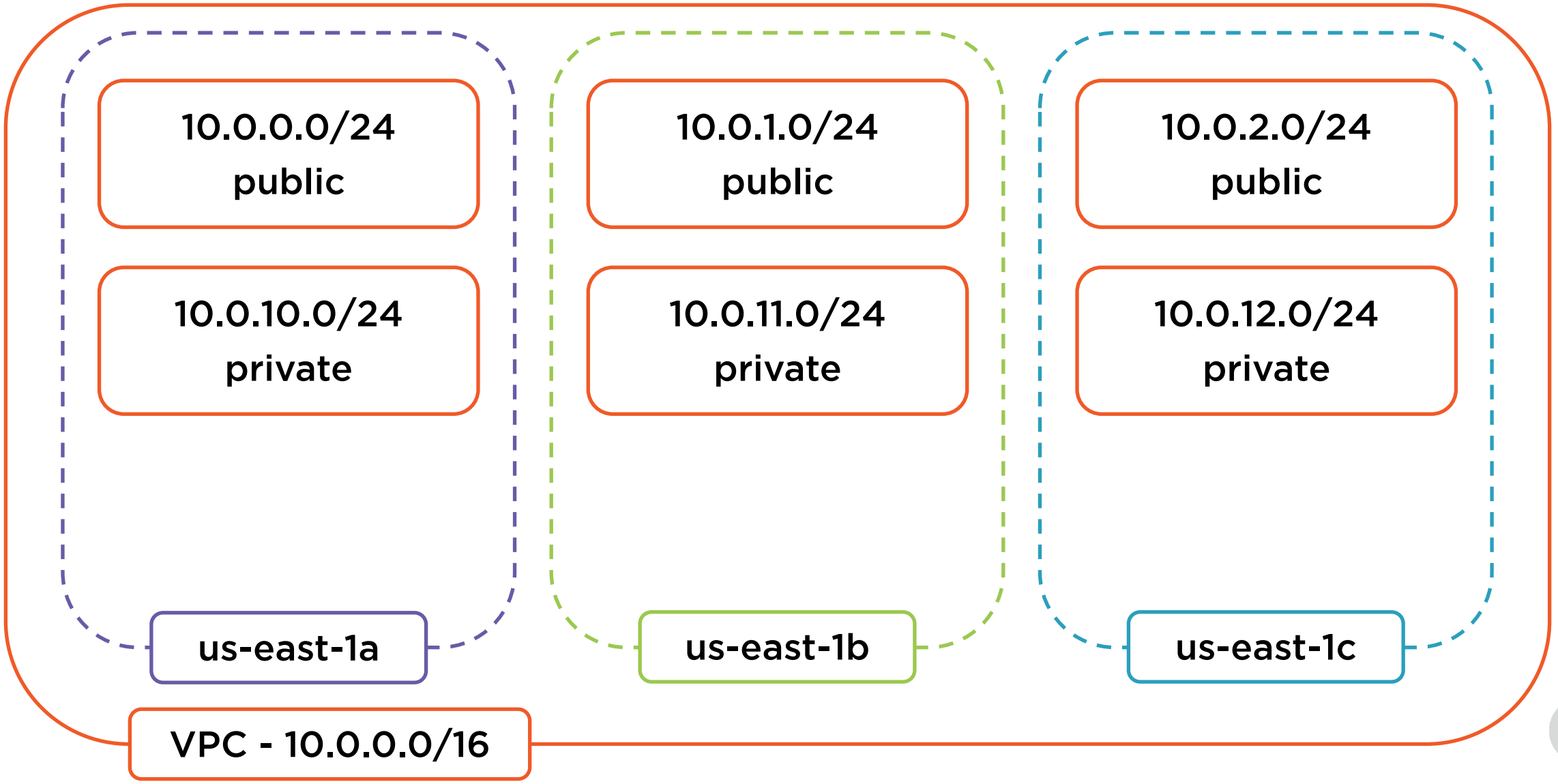
Backend options for state data

Migrating state data

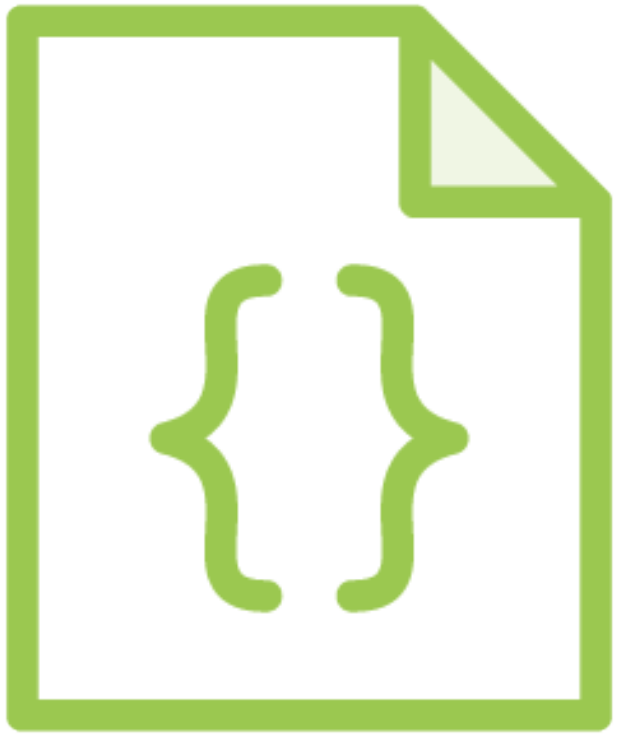
Globomantics Environment

- Work with the larger team
- Create infrastructure for other teams
- Enable collaboration through remote state
- Restrict access for other teams

Current Environment



Terraform State



JSON format (Do not touch!)

Resources mappings and metadata

Inspect through CLI

Refreshed during operations

Stored in backends

- Standard & enhanced
- Locking & workspaces

Terraform State Commands

list – list objects in state data

show – show details about an object

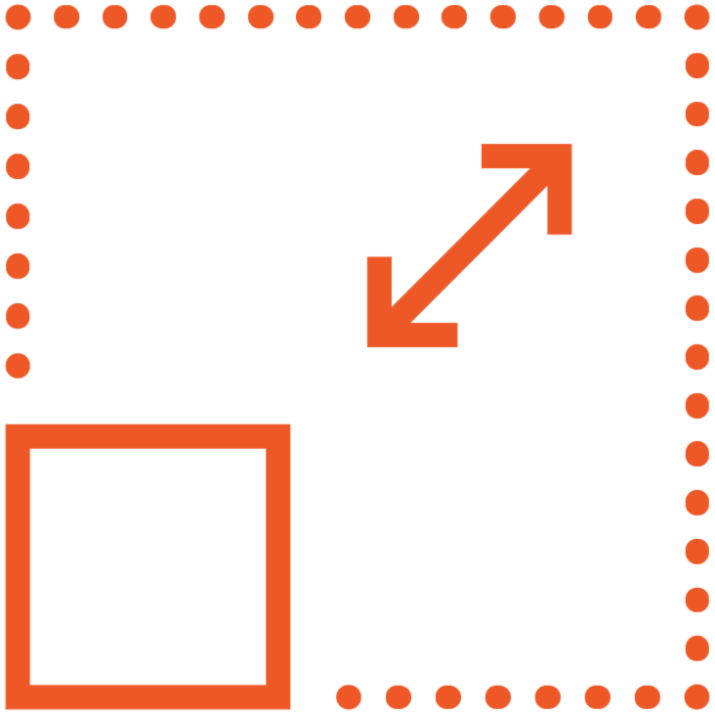
mv – move an item in state

rm – remove an item from state

pull – output current state to stdout

push – update remote state from local

Backends



State data is stored in backends

Backends must be initialized

Partial configurations are recommended

Interpolation is not supported

Backend Example

#Basic backend configuration

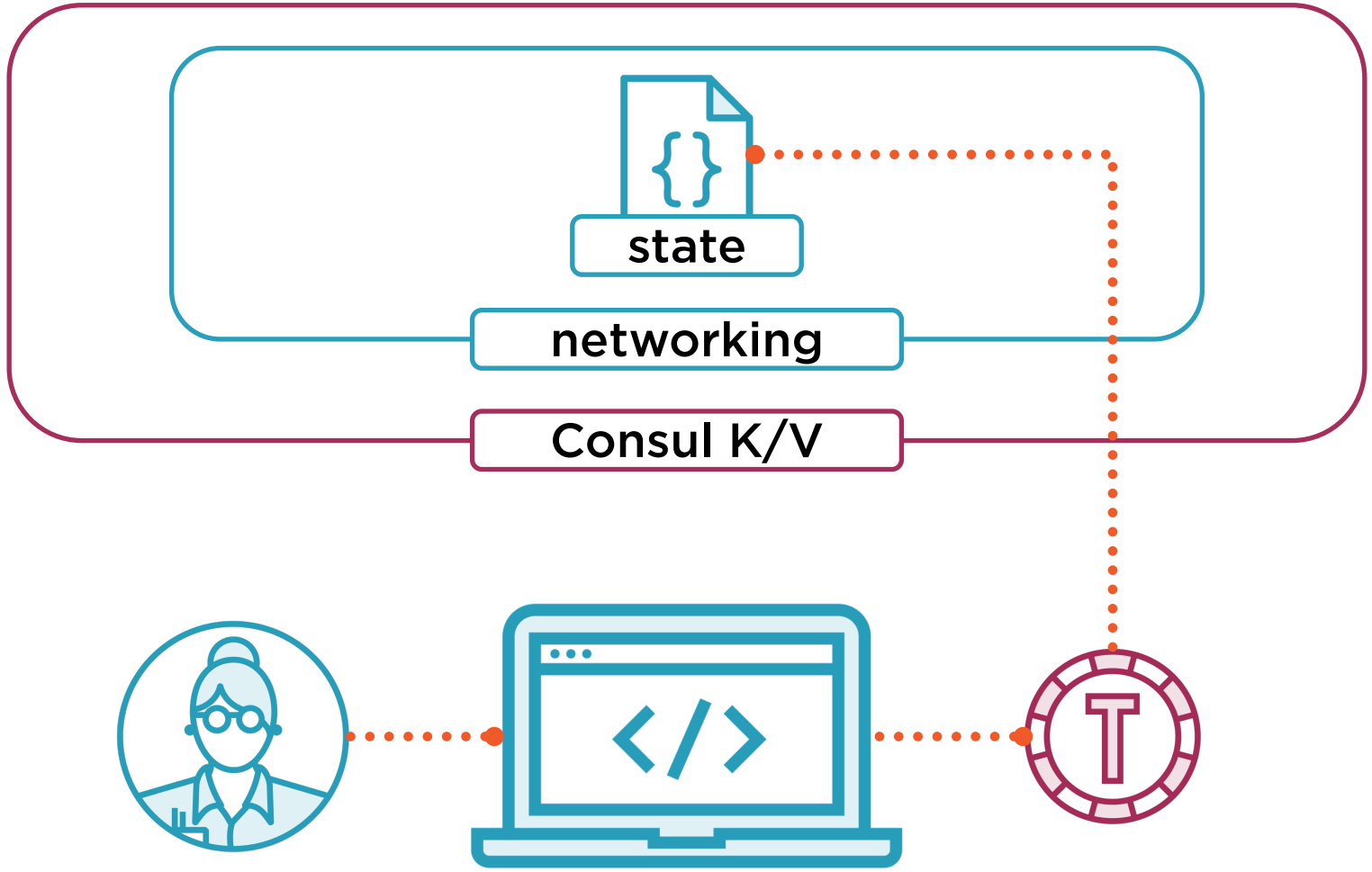
```
terraform {

  backend "type" {
    # backend info
    # authentication info
  }

}
```

#Backends: Consul, AWS S3, Azure Storage, Google Cloud Storage

Globomantics



Migrating Terraform State

Update backend configuration

Run terraform init

Confirm state migration

Summary

Terraform state is kinda important

State management through CLI

Remote state is preferred

Coming up:

- Using data sources and templates