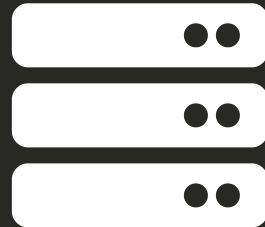


Infrastructure as Code - Terraformujeme cloud

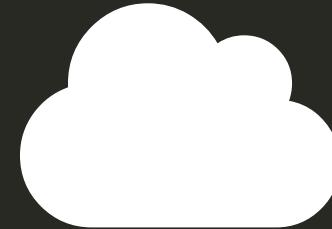
Viliam Púčik
DevOps Tech Lead

ZOOM International

On Premise vs Cloud



On Premise



Cloud

Major Cloud Providers



Amazon
Web
Services

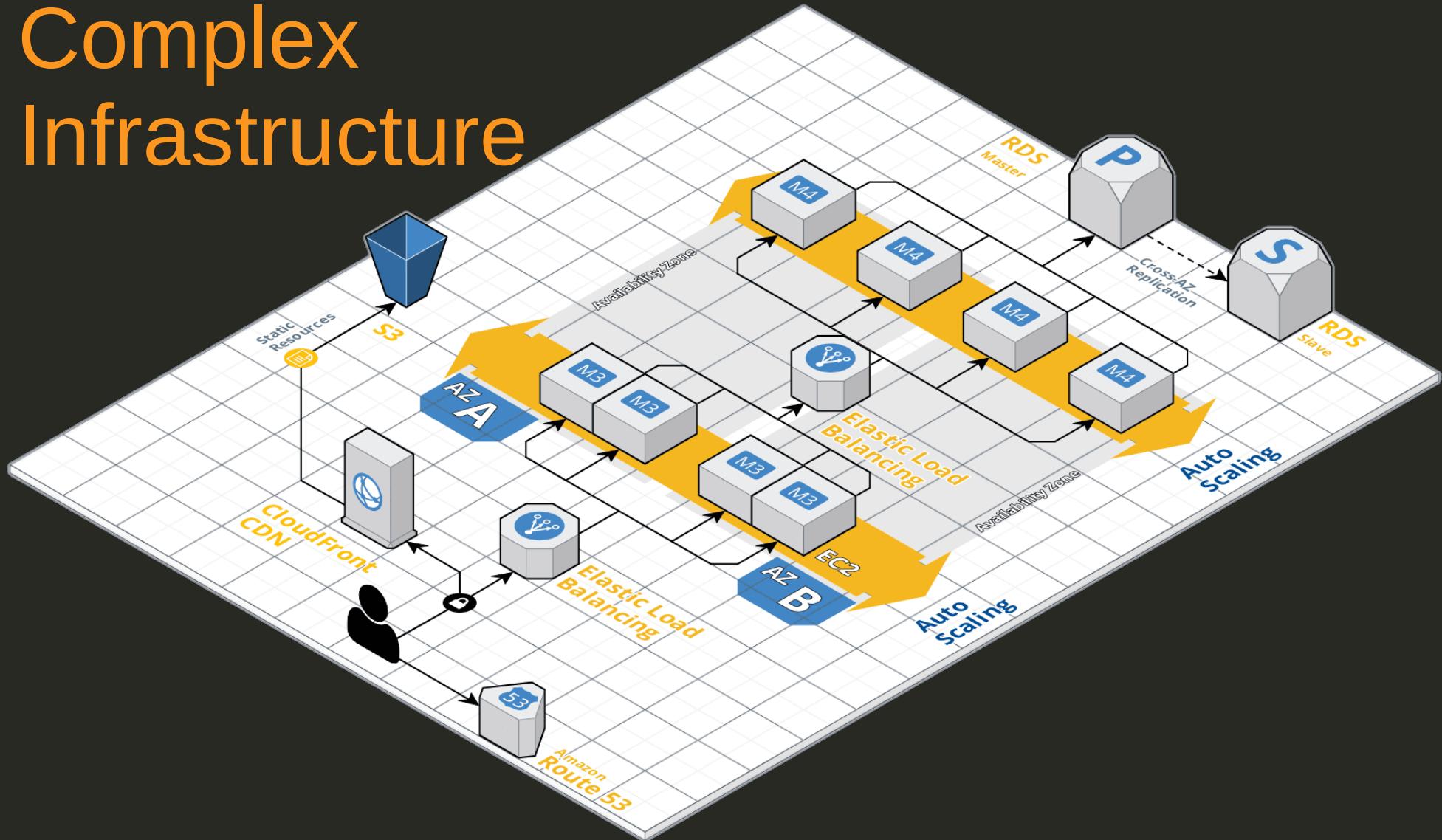


Google
Cloud
Platform

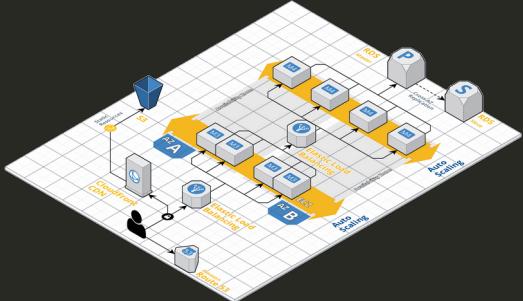


Microsoft
Azure

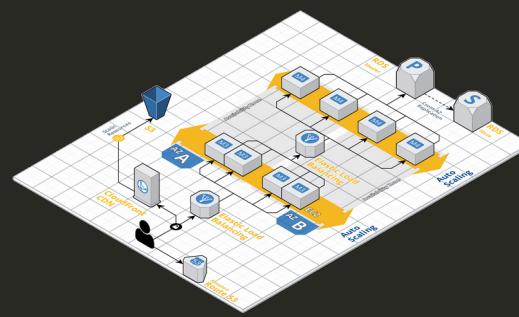
Complex Infrastructure



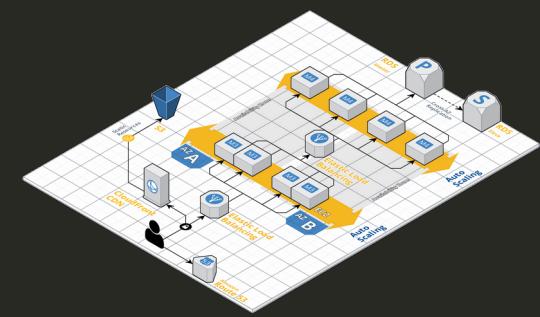
Complex Infrastructure



Development
Environment



Staging
Environment



Production
Environment

AWS Web Console

The screenshot shows the AWS EC2 Instances page. On the left, a sidebar lists various services: EC2 Dashboard, Events, Tags, Limits, Instances (selected), Launch Templates, Spot Requests, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images (AMIs, Bundle Tasks), Elastic Block Store (Volumes, Snapshots, Lifecycle Manager), Network & Security (Security Groups, Elastic IPs, Placement Groups, Key Pairs, Network Interfaces), Load Balancing (Load Balancers, Target Groups), and Auto Scaling (Launch Configurations, Auto Scaling Groups). The main content area displays a table of instances. A single instance is listed:

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Public IP	IPv6 IPs	Key Name	Monitoring	Launch Time
manual	i-0eb67652196039d0d	t2.micro	eu-central-1b	running	Initializing	None	ec2-3-122-59-71.eu-cen...	3.122.59.71	-	manual	disabled	October 31, 2019

Below the table, the details for the selected instance (i-0eb67652196039d0d) are shown. The "Description" tab is active, displaying the following information:

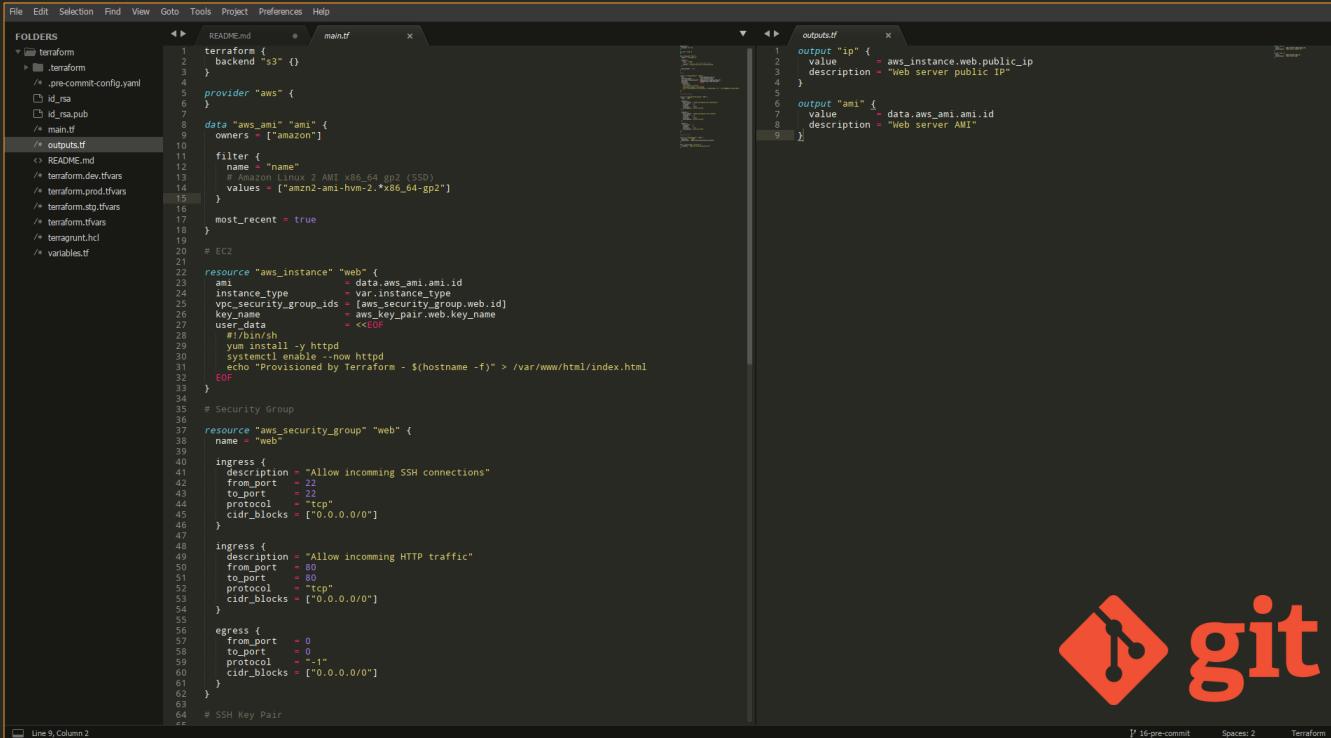
- Instance ID: i-0eb67652196039d0d
- Instance state: running
- Instance type: t2.micro
- Elastic IPs: eu-central-1b
- Availability zone: eu-central-1b
- Security groups: manual-sg, view inbound rules, view outbound rules
- Scheduled events: No scheduled events
- AMI ID: amzn2-ami-hvm-2.0.20190823.1-x86_64-gp2 (ami-00aa4671cbf840d82)
- Platform: -
- IAM role: -
- Key pair name: manual
- Owner: 850415196837
- Launch time: October 31, 2019 at 4:14:24 PM UTC+1 (less than one hour)
- Termination protection: False
- Lifecycle: normal
- Monitoring: basic
- Alarm status: None
- Kernel ID: -
- RAM disk ID: -
- Placement group: -
- Partition number: -
- Virtualization: hvm

The "Status Checks" tab shows the following status:

- Public DNS (IPv4): ec2-3-122-59-71.eu-central-1.compute.amazonaws.com
- IPv4 Public IP: 3.122.59.71
- IPv6 IPs: -
- Private DNS: ip-172-31-43-220.eu-central-1.compute.internal
- Private IPs: 172.31.43.220
- Secondary private IPs: -
- VPC ID: vpc-58a44432
- Subnet ID: subnet-968589e2
- Network interfaces: eth0
- Source/dest. check: True
- T2/T3 Unlimited: Disabled
- EBS-optimized: False
- Root device type: ebs
- Root device: /dev/xvda
- Block devices: /dev/xvda
- Elastic Graphics ID: -

At the bottom, there are links for Feedback, English (US), and a footer with copyright information: © 2008 - 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use.

Infrastructure as Code



The screenshot shows a code editor interface with two main panes. The left pane displays a file named `main.tf`, which contains Terraform configuration code for creating an AWS instance. The right pane displays a file named `outputs.tf`, which defines outputs for the instance's public IP and AMI ID. Below the code editor, a GitHub commit message is visible.

```
File Edit Selection Find View Goto Tools Project Preferences Help
```

```
FOLDERS
└─ terraform
    ├─ README.md
    └─ main.tf
```

```
backend "s3" {}  
provider "aws" {}  
  
data "aws_ami" "ami" {  
    owners = ["amazon"]  
  
    filter {  
        name = "name"  
        values = ["amzn2-ami-hvm-2.*x86_64-gp2"]  
    }  
    most_recent = true  
}  
  
resource "aws_instance" "web" {  
    ami = data.aws_ami.ami.id  
    instance_type = var.instance_type  
    vpc_security_group_ids = [aws_security_group.web.id]  
    key_name = aws_key_pair.web.key_name  
    user_data = <<EOF  
#!/bin/sh  
yum install -y httpd  
systemctl enable --now httpd  
echo "Provisioned by Terraform - $(hostname -f)" > /var/www/html/index.html  
EOF  
}  
  
# Security Group  
  
resource "aws_security_group" "web" {  
    name = "web"  
  
    ingress {  
        description = "Allow incoming SSH connections"  
        from_port = 22  
        to_port = 22  
        protocol = "tcp"  
        cidr_blocks = ["0.0.0.0/0"]  
    }  
  
    ingress {  
        description = "Allow incoming HTTP traffic"  
        from_port = 80  
        to_port = 80  
        protocol = "tcp"  
        cidr_blocks = ["0.0.0.0/0"]  
    }  
  
    egress {  
        from_port = 0  
        to_port = 0  
        protocol = "-1"  
        cidr_blocks = ["0.0.0.0/0"]  
    }  
}  
  
# SSH Key Pair
```

```
outputs.tf
```

```
output "ip" {  
    value = aws_instance.web.public_ip  
    description = "Web server public IP"  
}  
  
output "ami" {  
    value = data.aws_ami.ami.id  
    description = "Web server AMI"  
}
```

Line 9, Column 2

16-pre-commit

Spaces: 2

Terraform

git

Infrastructure as Code



HashiCorp
Terraform

(Open Source)



AWS
Cloud
Formation



GCP
Cloud
Deployment
Manager



Azure
Resource
Manager



Terraform

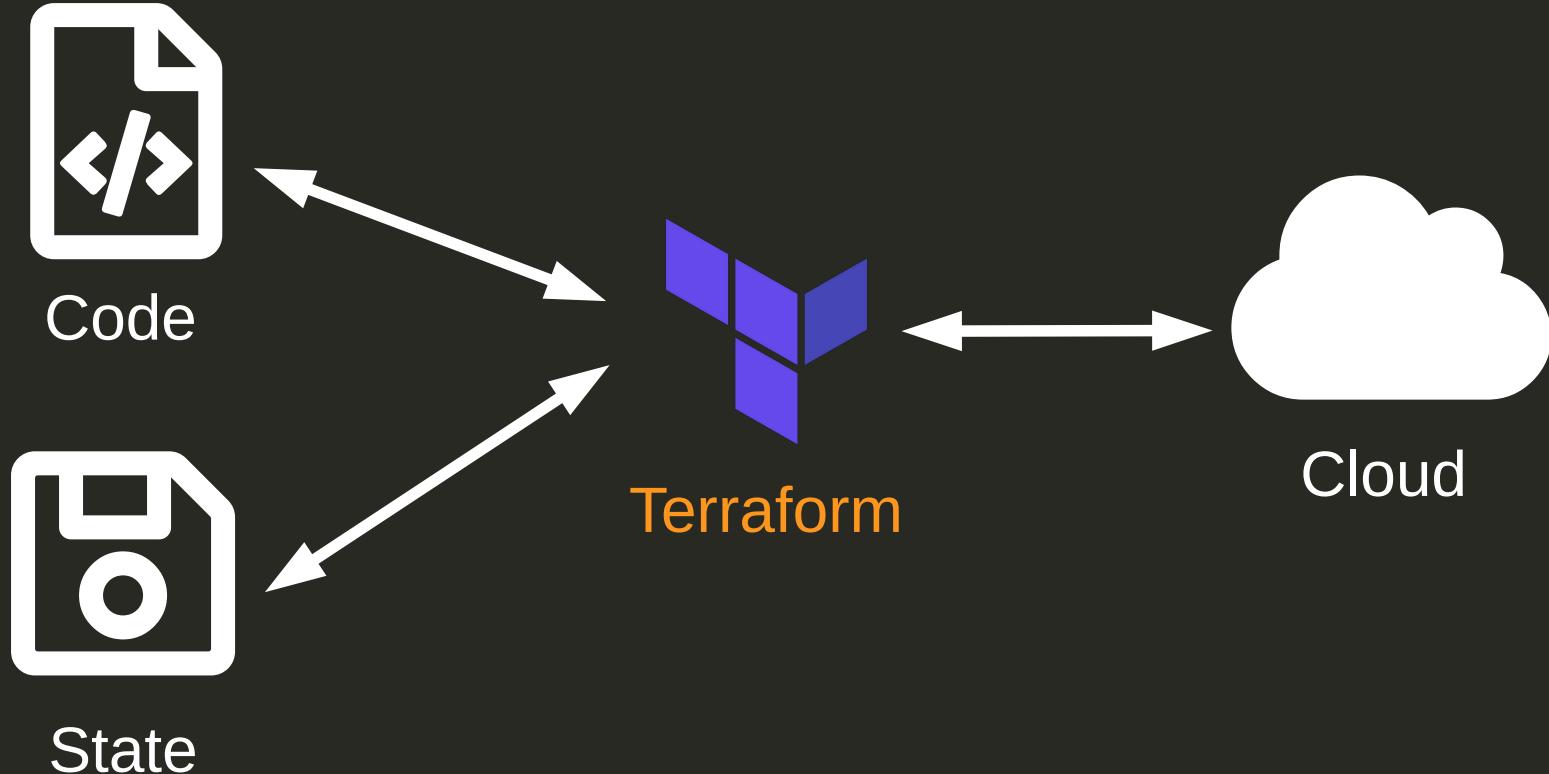
A tool for building, changing, and versioning infrastructure safely and efficiently. Building blocks:

- **Providers** (AWS, GCP, Azure, MySQL, PostgreSQL, Kubernetes, Helm, GitHub and hundred of others)
- **Resources, Data Sources** (read-only)
- **Input, Local and Output variables**
- **Expressions and Functions**

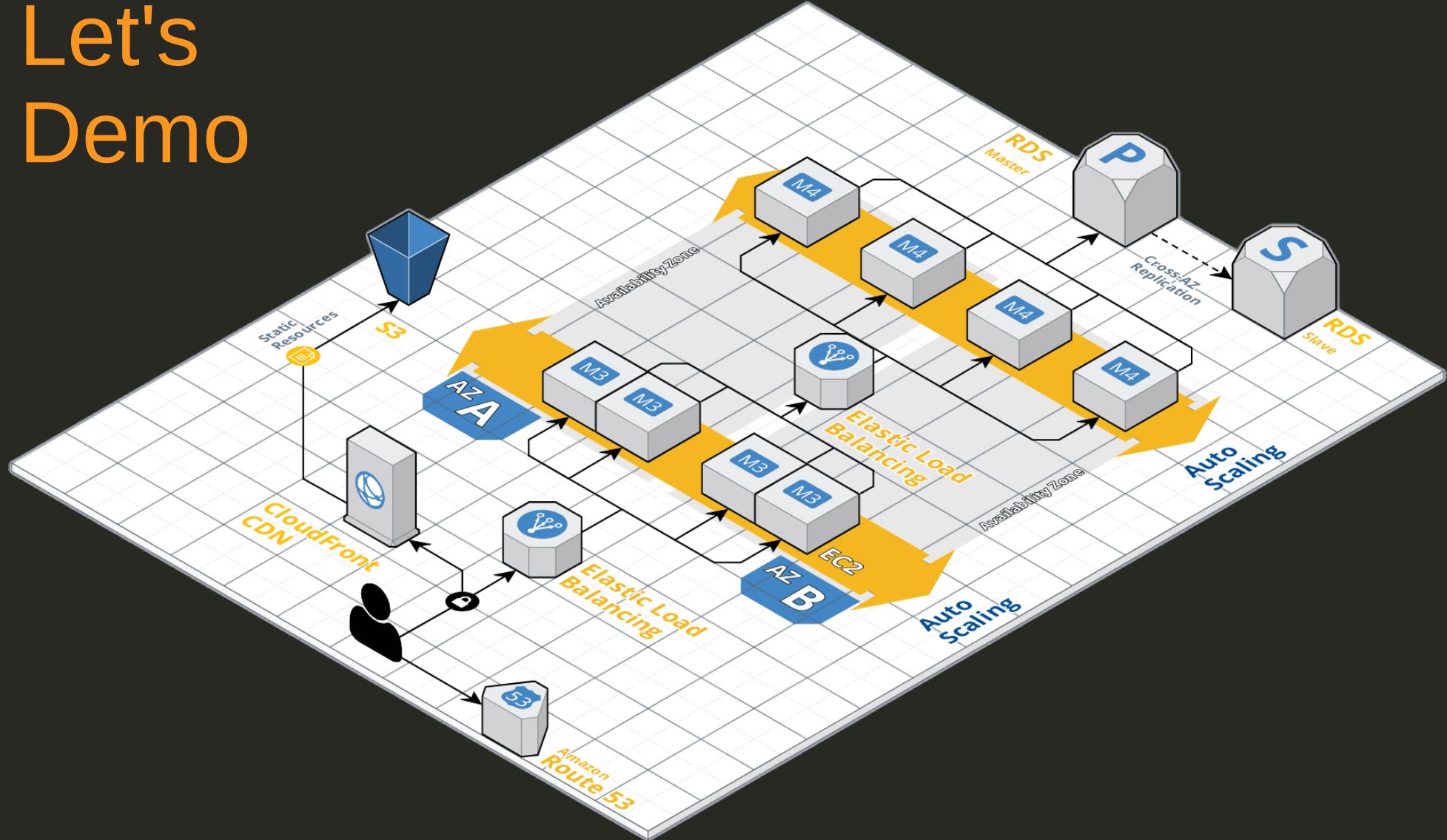
<https://www.terraform.io/>



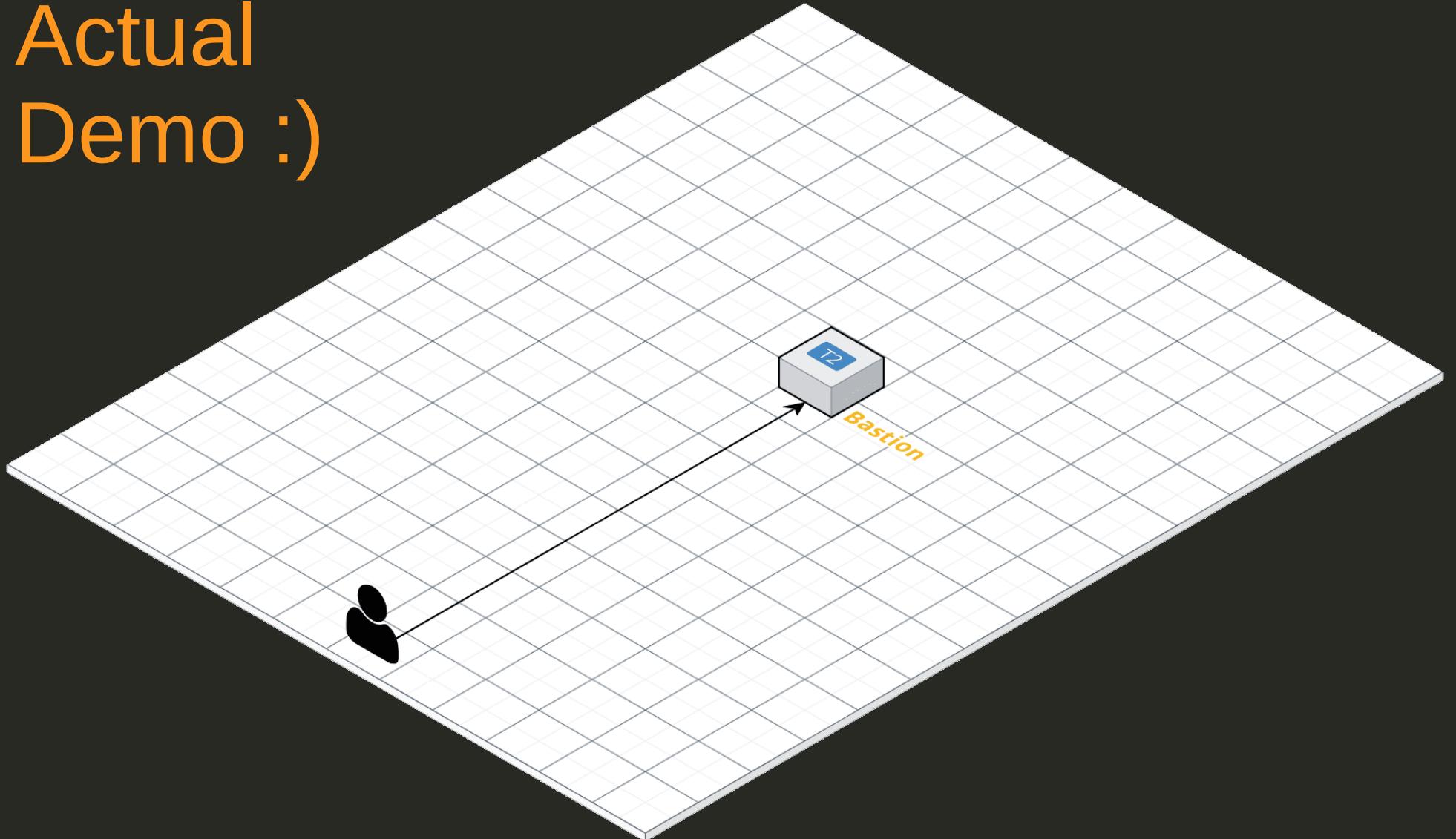
Terraform State



Let's Demo



Actual Demo :)



Terraform Meta Arguments

- `depends_on` - for specifying hidden dependencies
- `count` - for creating multiple resource instances according to a count
- `for_each` - to create multiple instances according to a map
 - or set of strings
- `provider` - for selecting a non-default provider configuration
- `lifecycle` - for lifecycle customizations
- `provisioner` and `connection` - for taking extra actions after resource creation

depends_on

```
resource "aws_instance" "bastion" {  
    vpc_security_group_ids = [  
        aws_security_group.bastion.id,  
    ]  
    depends_on = [  
        aws_instance.web,  
    ]  
}
```

count

```
resource "aws_instance" "bastion" {  
    count = 10  
}
```

count

```
variable "web_enabled" {  
    type    = bool  
    default = false  
}  
  
resource "aws_instance" "web" {  
    count = var.web_enabled == true ? 1 : 0  
}
```

for_each

```
variable "users" {  
    type      = list(string)  
    default   = ["admin", "developer", "manager"]  
}  
  
resource "aws_iam_user" "user" {  
    for_each = toset(var.users)  
    name     = each.key  
}
```

provider

```
provider "aws" {}
```

```
provider "aws" {  
  alias = "staging"  
}
```

```
resource "aws_instance" "web" {  
  provider = aws.staging  
}
```

lifecycle

```
resource "aws_instance" "web" {  
  lifecycle {  
    create_before_destroy = true  
    ignore_changes       = [tags]  
  }  
}
```

lifecycle

```
resource "aws_instance" "web" {  
  lifecycle {  
    prevent_destroy = true  
  }  
}
```

provisioner

```
resource "null_resource" "id_rsa" {  
  provisioner "local-exec" {  
    working_dir = path.module  
    command     = "ssh-keygen -N '' -f id_rsa"  
  }  
}
```

provisioner

```
resource "aws_instance" "web" {  
  provisioner "remote-exec" {  
    inline = [  
      "sudo systemctl disable httpd",  
    ]  
  }  
}
```

provisioner

```
resource "aws_instance" "web" {  
  provisioner "file" {  
    source      = "${path.module}conf/httpd.conf"  
    destination = "/etc/httpd/conf/httpd.conf"  
  }  
}
```

provisioner

```
resource "aws_instance" "web" {  
  provisioner "file" {  
    ...  
    connection {  
      type = "ssh"  
      user = "developer"  
      port = 2022  
    }  
  }  
}  
}
```

Terraform Modules

```
module "rds" {  
    source  = "terraform-aws-modules/rds/aws"  
    version = "2.5.0"  
    # insert the 11 required variables here  
}
```

<https://registry.terraform.io/>



Terragrunt

A thin wrapper for Terraform that provides extra tools for working with multiple Terraform modules. For example:

- Creates remote state and locking resources automatically
- Passes extra CLI arguments every time you run certain **terraform** commands

<https://github.com/gruntwork-io/terragrunt>



Questions?

Thank you!

<https://a.openalt.cz/53>



aws-vault

A tool to securely store and access AWS credentials in (development) environments.

- Encrypts AWS keys
- Provides temporary, one time credentials

<https://github.com/99designs/aws-vault>



Terraform Pre-Commit Framework

Automatically, before each commit:

- Formats Terraform code
- Updates `README.md` with the description of:
 - `terraform input` variables
 - `terraform output` variables

<https://github.com/antonbabenko/pre-commit-terraform>

