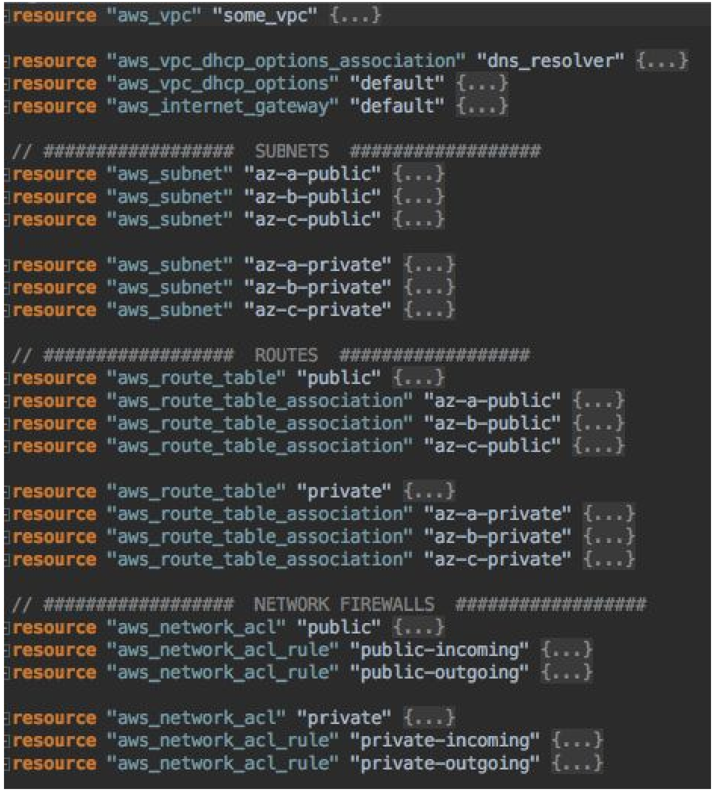
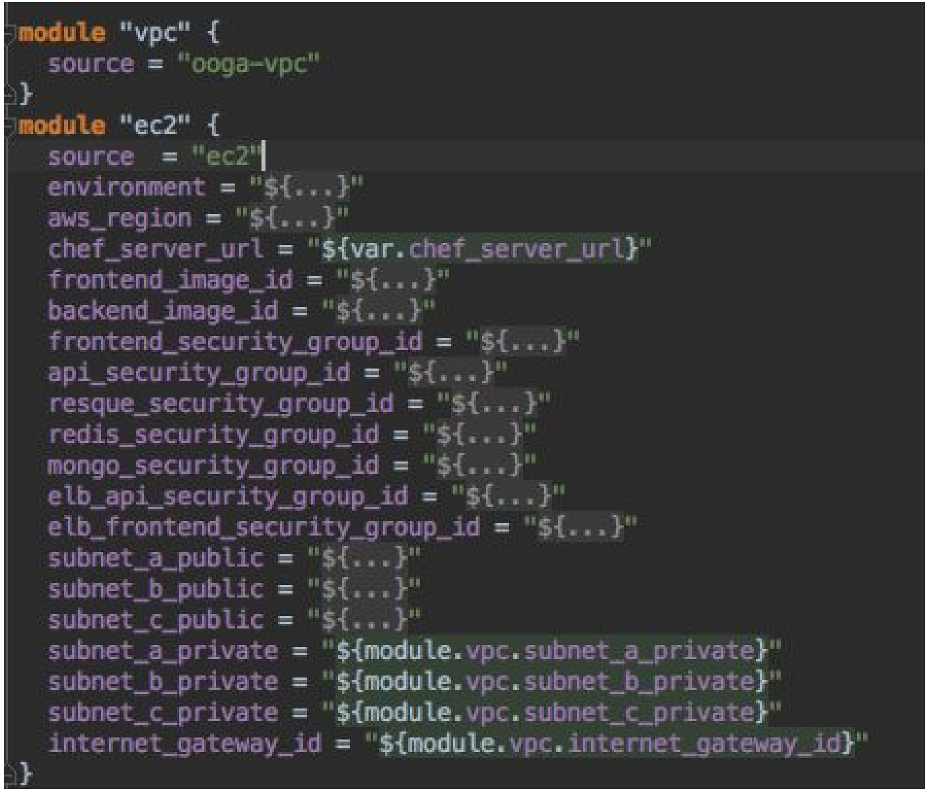
Terraform

An Approach to Infrastructure as Code

Cloud as code: Organized. Versioned. Automated.



```
resource "aws_vpc" "some_vpc" {...}

resource "aws_vpc_dhcp_options_association" "dns_resolver" {...}
resource "aws_vpc_dhcp_options" "default" {...}
resource "aws_internet_gateway" "default" {...}

// ##################  SUBNETS  ##################
resource "aws_subnet" "az-a-public" {...}
resource "aws_subnet" "az-b-public" {...}
resource "aws_subnet" "az-c-public" {...}

resource "aws_subnet" "az-a-private" {...}
resource "aws_subnet" "az-b-private" {...}
resource "aws_subnet" "az-c-private" {...}

// ##################  ROUTES  ##################
resource "aws_route_table" "public" {...}
resource "aws_route_table_association" "az-a-public" {...}
resource "aws_route_table_association" "az-b-public" {...}
resource "aws_route_table_association" "az-c-public" {...}

resource "aws_route_table" "private" {...}
resource "aws_route_table_association" "az-a-private" {...}
resource "aws_route_table_association" "az-b-private" {...}
resource "aws_route_table_association" "az-c-private" {...}

// ##################  NETWORK FIREWALLS  ##################
resource "aws_network_acl" "public" {...}
resource "aws_network_acl_rule" "public-incoming" {...}
resource "aws_network_acl_rule" "public-outgoing" {...}

resource "aws_network_acl" "private" {...}
resource "aws_network_acl_rule" "private-incoming" {...}
resource "aws_network_acl_rule" "private-outgoing" {...}
```

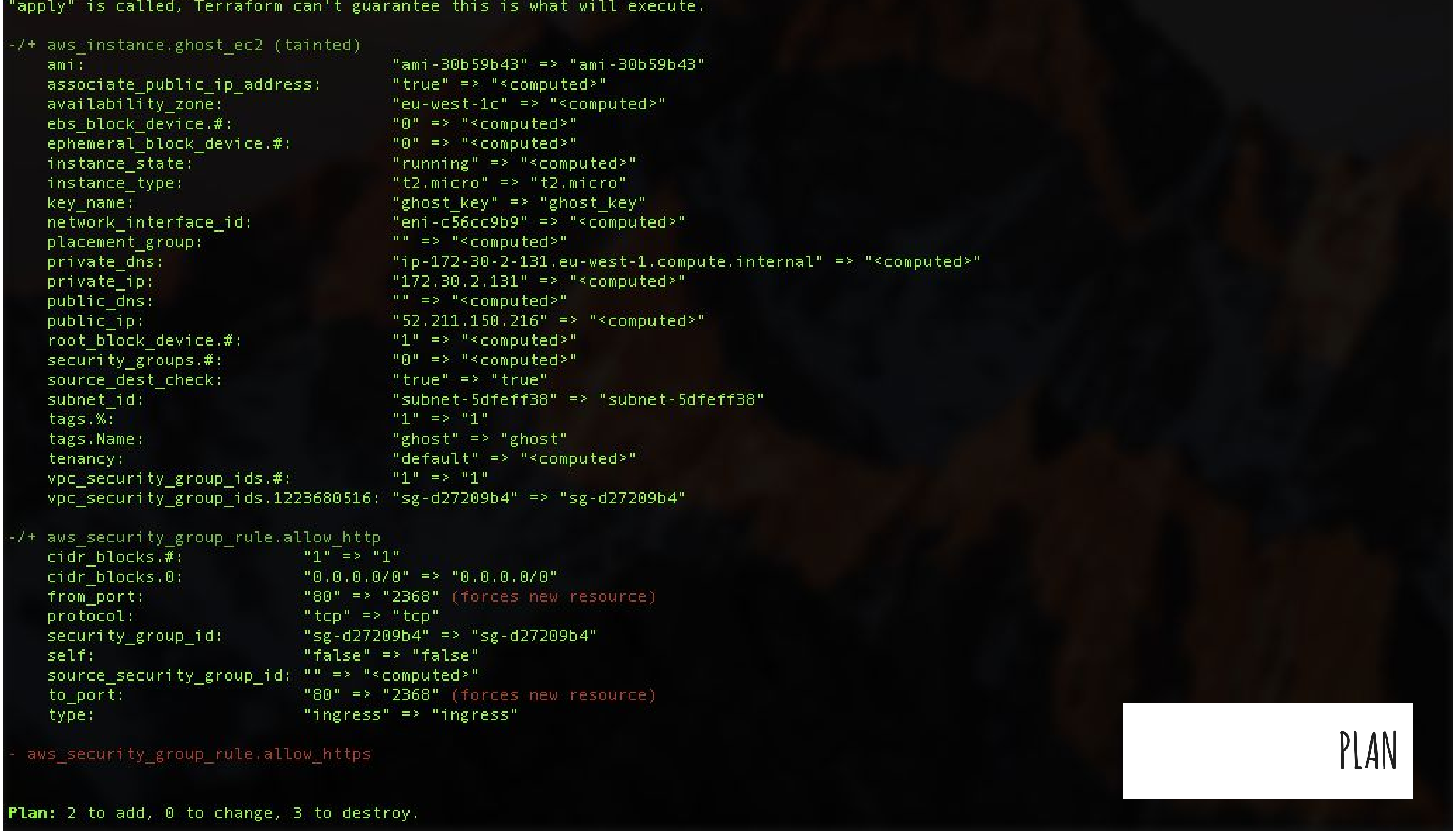


```
module "vpc" {
  source = "ooga-vpc"
}
module "ec2" {
  source   = "ec2"
  environment = "${...}"
  aws_region = "${...}"
  chef_server_url = "${var.chef_server_url}"
  frontend_image_id = "${...}"
  backend_image_id = "${...}"
  frontend_security_group_id = "${...}"
  api_security_group_id = "${...}"
  resque_security_group_id = "${...}"
  redis_security_group_id = "${...}"
  mongo_security_group_id = "${...}"
  elb_api_security_group_id = "${...}"
  elb_frontend_security_group_id = "${...}"
  subnet_a_public = "${...}"
  subnet_b_public = "${...}"
  subnet_c_public = "${...}"
  subnet_a_private = "${module.vpc.subnet_a_private}"
  subnet_b_private = "${module.vpc.subnet_b_private}"
  subnet_c_private = "${module.vpc.subnet_c_private}"
  internet_gateway_id = "${module.vpc.internet_gateway_id}"
}
```

Write

Plan

Apply



```
"apply" is called, Terraform can't guarantee this is what will execute.

-/+ aws_instance.ghost_ec2 (tainted)
    ami:                               "ami-30b59b43" => "ami-30b59b43"
    associate_public_ip_address:       "true" => "<computed>"
    availability_zone:                 "eu-west-1c" => "<computed>"
    ebs_block_device.#:                "0" => "<computed>"
    ephemeral_block_device.#:          "0" => "<computed>"
    instance_state:                    "running" => "<computed>"
    instance_type:                     "t2.micro" => "t2.micro"
    key_name:                          "ghost_key" => "ghost_key"
    network_interface_id:              "eni-c56cc9b9" => "<computed>"
    placement_group:                   "" => "<computed>"
    private_dns:                       "ip-172-30-2-131.eu-west-1.compute.internal" => "<computed>"
    private_ip:                        "172.30.2.131" => "<computed>"
    public_dns:                        "" => "<computed>"
    public_ip:                         "52.211.150.216" => "<computed>"
    root_block_device.#:               "1" => "<computed>"
    security_groups.#:                 "0" => "<computed>"
    source_dest_check:                 "true" => "true"
    subnet_id:                         "subnet-5dfeff38" => "subnet-5dfeff38"
    tags.%:                            "1" => "1"
    tags.Name:                         "ghost" => "ghost"
    tenancy:                           "default" => "<computed>"
    vpc_security_group_ids.#:          "1" => "1"
    vpc_security_group_ids.1223680516: "sg-d27209b4" => "sg-d27209b4"

-/+ aws_security_group_rule.allow_http
    cidr_blocks.#:            "1" => "1"
    cidr_blocks.0:            "0.0.0.0/0" => "0.0.0.0/0"
    from_port:                "80" => "2368" (forces new resource)
    protocol:                 "tcp" => "tcp"
    security_group_id:        "sg-d27209b4" => "sg-d27209b4"
    self:                     "false" => "false"
    source_security_group_id: "" => "<computed>"
    to_port:                  "80" => "2368" (forces new resource)
    type:                     "ingress" => "ingress"

- aws_security_group_rule.allow_https


Plan: 2 to add, 0 to change, 3 to destroy.
```

PLAN