Understanding the state file

At a high level Terraform state is a mapping of the actual state of your infrastructure which was created from your configurations (which are the desired state). The state file is a custom JSON hierarchy which contains the following metadata:

version - the protocol version of the state file
terraform_version - the version of Terraform that wrote this state file
serial - incremented on any operation that modifies the infrastructure

- version — the protocol version of the state file
- terraform_version — the version of Terraform that wrote this state file
- serial — incremented on any operation that modifies the infrastructure
- lineage — set when the state is created
- remote — used to track the metadata required to push/pull state the configured remote
- backend — used to track the configuration for the backend in use with the state
- modules — contains all the modules in a “breadth-first order”

- path — the import path from the root module
- outputs — any outputs declared by the module
- resources — a mapping of the logically name resources within that level
- depends_on — a list of things this module relies on

Another thing to consider with regards to state and Terraform in general is that it operates using the breadth-first order (BFS) algorithm. This means that it will traverse the graph data structure level-by-level. Specifically it will start with the configurations in the current working directory which it considers the “root” module and then pull in the next level of modules and so on.

Recovering a simple Terraform

This is a very simple configuration contained within one file `main.tf`

```
provider "aws" {
  region = "us-east-1"
}resource "aws_instance" "example" {
  ami           = "ami-b73b63a0"
  instance_type = "t2.micro"  root_block_device {
    delete_on_termination = "true"
  }
}
```

After applying this configuration the `terraform.tfstate` contains the following. Because the configuration is contained in the *cwd* all of the configured resources are contained in the “root” module.

```
{
    "version": 3,
```

```
    "terraform_version": "0.11.13",
    "serial": 1,
    "lineage": "0b400f3d-db43-a2e4-cfb5-5a856736739b",
    "modules": [
        {
            "path": [
                "root"
            ],
            "outputs": {},
            "resources": {
                "aws_instance.example": {
                    "type": "aws_instance",
                    "depends_on": [],
                    "primary": {
                        "id": "i-06becd0903c31ab3e",
                        "attributes": {
                            "ami": "ami-b73b63a0",
                            "arn": "arn:aws:ec2:us-east-
1:224557780148:instance/i-06becd0903c31ab3e",
                            "associate_public_ip_address": "true",
                            "availability_zone": "us-east-1a",
                            "cpu_core_count": "1",
                            "cpu_threads_per_core": "1",
                            "credit_specification.#": "1",
                            "credit_specification.0.cpu_credits":
"standard",
                            "disable_api_termination": "false",
                            "ebs_block_device.#": "0",
                            "ebs_optimized": "false",
                            "ephemeral_block_device.#": "0",
                            "get_password_data": "false",
                            "iam_instance_profile": "",
                            "id": "i-06becd0903c31ab3e",
                            "instance_state": "running",
                            "instance_type": "t2.micro",
                            "ipv6_addresses.#": "0",
                            "key_name": "",
                            "monitoring": "false",
                            "network_interface.#": "0",
                            "password_data": "",
                            "placement_group": "",
                            "primary_network_interface_id": "eni-
07d59931a3c3cab68",
                            "private_dns": "ip-172-31-51-
247.ec2.internal",
                            "private_ip": "172.31.51.247",
                            "public_dns": "ec2-3-89-65-146.compute-
```

```
1.amazonaws.com",
                            "public_ip": "3.89.65.146",
                            "root_block_device.#": "1",
"root_block_device.0.delete_on_termination": "true",
                            "root_block_device.0.iops": "0",
                            "root_block_device.0.volume_id": "vol-
0f0c73b8015ecf09e",
                            "root_block_device.0.volume_size": "8",
                            "root_block_device.0.volume_type":
"standard",
                            "security_groups.#": "1",
                            "security_groups.3814588639":
"default",
                            "source_dest_check": "true",
                            "subnet_id": "subnet-690a0942",
                            "tags.%": "0",
                            "tenancy": "default",
                            "volume_tags.%": "0",
                            "vpc_security_group_ids.#": "1",
                            "vpc_security_group_ids.324294369":
"sg-a74531c1"
                        },
                        "meta": {
                            "e2bfb730-ecaa-11e6-8f88-34363bc7c4c0":
{
                                "create": 600000000000,
                                "delete": 1200000000000,
                                "update": 600000000000
                            },
                            "schema_version": "1"
                        },
                        "tainted": false
                    },
                    "deposed": [],
                    "provider": "provider.aws"
                }
            },
            "depends_on": []
        }
    ]
}
```

In our hypothetical we no longer have this state file and thus, if we apply this Terraform, it will create a new EC2 resource. Additionally

the EC2 resource that already exists is orphaned from the Terraform that should control it.

In order to regain control of this resource we will use the `import` command to recover the metadata for this resource. Import requires a resource address and the resource ID. The resource address is a combination of the resource type and resource name which for this configuration is 'aws_instance.example'. The ID in this instance is the AWS EC2 ID 'i-06becd0903c31ab3e'.

```
$ terraform import aws_instance.example i-06becd0903c31ab3e
aws_instance.example: Importing from ID "i-06becd0903c31ab3e"...
aws_instance.example: Import complete!
  Imported aws_instance (ID: i-06becd0903c31ab3e)
aws_instance.example: Refreshing state... (ID: i-
06becd0903c31ab3e)Import successful!The resources that were
imported are shown above. These resources are now in your Terraform
state and will henceforth be managed by Terraform.
```

The `import` command creates a new state file pulling in information about the specified AWS Instance ID. Once complete the only difference between the new state and old will be the lineage hash.

```
$ diff old.tfstate new.tfstate
5c5
<     "lineage": "0b400f3d-db43-a2e4-cfb5-5a856736739b",
---
>     "lineage": "845a48d7-0ecc-e228-2d24-f96d05ff2c9a",
```

Recovering a (more) complex Terraform

For purposes of this post I'm not going to get too complex, but importing modules does require some additional understanding. The Terraform that follows makes use of modules to fully define the EC2 instances (web), Security Groups (sg) and Elastic Load Balancers (elb).

```
variable "region" { default = "us-east-1" }
variable "availability_zone" { default = "us-east-1a" }
variable "instance_type" { default = "t2.micro" }
variable "instance_count" { default = "2" }
variable "key_name" {}
variable "public_key_path" {}
variable "connection_user" { default = "ec2-user" }
variable "name" { default = "web" }
variable "environment" { default = "production" }provider "aws" {
  region = "${var.region}"
}module "web" {
  source = "../modules/web"  availability_zone =
"${var.availability_zone}"
  connection_user = "${var.connection_user}"
  instance_count = "${var.instance_count}"
  instance_type = "${var.instance_type}"
  key_name = "${var.key_name}"
  public_key_path = "${var.public_key_path}"
  region = "${var.region}"
  web_security_groups = "${module.sg.security_group_id}"
  name = "${var.name}"
  environment = "${var.environment}"}module "sg" {
  source = "../modules/sg"  name = "${var.name}"
  environment = "${var.environment}"
}module "elb" {
  source = "../modules/elb"  availability_zones =
"${var.availability_zone}"
  name = "${var.name}"
  environment = "${var.environment}"
  web_instance_ids = "${module.web.web_instance_ids}"
}output "elb_dns_name" {
  value = "${module.elb.dns_name}"
}
```

Running an `apply` on this Terraform yields the following state. Notice the root module now contains only output metadata. The next level are the web, sg, and elb modules referenced by the root module (recall breadth-first order). For this configuration these modules are where all of the AWS resources are created.

```
{
    "version": 3,
    "terraform_version": "0.11.13",
```

```
    "serial": 1,
    "lineage": "1ec1c6b6-7b32-f8bf-dbe0-f44d0697d878",
    "modules": [
        {
            "path": [
                "root"
            ],
            "outputs": {
                "elb_dns_name": {
                    "sensitive": false,
                    "type": "string",
                    "value": "web-elb-1595946901.us-east-
1.elb.amazonaws.com"
                }
            },
            "resources": {},
            "depends_on": []
        },
        {
            "path": [
                "root",
                "elb"
            ],
            "outputs": {
                "dns_name": {
                    "sensitive": false,
                    "type": "string",
                    "value": "web-elb-1595946901.us-east-
1.elb.amazonaws.com"
                }
            },
            "resources": {
                "aws_elb.elb": {
                    "type": "aws_elb",
                    "depends_on": [],
                    "primary": {
                        "id": "web-elb",
                        "attributes": {
                            "access_logs.#": "0",
                            "arn":
"arn:aws:elasticloadbalancing:us-east-
1:224557780148:loadbalancer/web-elb",
                            "availability_zones.#": "1",
                            "availability_zones.3569565595": "us-
east-1a",
                            "connection_draining": "true",
                            "connection_draining_timeout": "400",
```

```
                            "cross_zone_load_balancing": "true",
                            "dns_name": "web-elb-1595946901.us-
east-1.elb.amazonaws.com",
                            "health_check.#": "1",
                            "health_check.0.healthy_threshold":
"2",
                            "health_check.0.interval": "30",
                            "health_check.0.target": "HTTP:80/",
                            "health_check.0.timeout": "5",
                            "health_check.0.unhealthy_threshold":
"2",
                            "id": "web-elb",
                            "idle_timeout": "400",
                            "instances.#": "2",
                            "instances.1754828781": "i-
0377163446bd5a9d9",
                            "instances.3613894088": "i-
058a9c36e326add1c",
                            "internal": "false",
                            "listener.#": "1",
                            "listener.3057123346.instance_port":
"80",

"listener.3057123346.instance_protocol": "http",
                            "listener.3057123346.lb_port": "80",
                            "listener.3057123346.lb_protocol":
"http",

"listener.3057123346.ssl_certificate_id": "",
                            "name": "web-elb",
                            "security_groups.#": "1",
                            "security_groups.2781873026": "sg-
0a08d259fb40ee9e6",
                            "source_security_group":
"224557780148/default_elb_15702516-bb9f-3fbd-b08e-4ab8ab664325",
                            "source_security_group_id": "sg-
0a08d259fb40ee9e6",
                            "subnets.#": "1",
                            "subnets.3664814999": "subnet-
690a0942",
                            "tags.%": "2",
                            "tags.Environment": "production",
                            "tags.Name": "web-elb",
                            "zone_id": "Z35SXDOTRQ7X7K"
                        },
                        "meta": {},
                        "tainted": false
```

```
                    },
                    "deposed": [],
                    "provider": "provider.aws"
                }
            },
            "depends_on": []
        },
        {
            "path": [
                "root",
                "sg"
            ],
            "outputs": {
                "security_group_id": {
                    "sensitive": false,
                    "type": "string",
                    "value": "sg-075867a836925cbc1"
                }
            },
            "resources": {
                "aws_security_group.web_sg": {
                    "type": "aws_security_group",
                    "depends_on": [],
                    "primary": {
                        "id": "sg-075867a836925cbc1",
                        "attributes": {
                            "arn": "arn:aws:ec2:us-east-
1:224557780148:security-group/sg-075867a836925cbc1",
                            "description": "Security group for web
production",
                            "egress.#": "1",
                            "egress.482069346.cidr_blocks.#": "1",
                            "egress.482069346.cidr_blocks.0":
"0.0.0.0/0",
                            "egress.482069346.description": "",
                            "egress.482069346.from_port": "0",
                            "egress.482069346.ipv6_cidr_blocks.#":
"0",
                            "egress.482069346.prefix_list_ids.#":
"0",
                            "egress.482069346.protocol": "-1",
                            "egress.482069346.security_groups.#":
"0",
                            "egress.482069346.self": "false",
                            "egress.482069346.to_port": "0",
                            "id": "sg-075867a836925cbc1",
                            "ingress.#": "2",
```

```
                            "ingress.2214680975.cidr_blocks.#":
"1",
                            "ingress.2214680975.cidr_blocks.0":
"0.0.0.0/0",
                            "ingress.2214680975.description": "",
                            "ingress.2214680975.from_port": "80",

"ingress.2214680975.ipv6_cidr_blocks.#": "0",
                            "ingress.2214680975.prefix_list_ids.#":
"0",
                            "ingress.2214680975.protocol": "tcp",
                            "ingress.2214680975.security_groups.#":
"0",
                            "ingress.2214680975.self": "false",
                            "ingress.2214680975.to_port": "80",
                            "ingress.2541437006.cidr_blocks.#":
"1",
                            "ingress.2541437006.cidr_blocks.0":
"0.0.0.0/0",
                            "ingress.2541437006.description": "",
                            "ingress.2541437006.from_port": "22",

"ingress.2541437006.ipv6_cidr_blocks.#": "0",
                            "ingress.2541437006.prefix_list_ids.#":
"0",
                            "ingress.2541437006.protocol": "tcp",
                            "ingress.2541437006.security_groups.#":
"0",
                            "ingress.2541437006.self": "false",
                            "ingress.2541437006.to_port": "22",
                            "name": "web-production-sg",
                            "owner_id": "224557780148",
                            "revoke_rules_on_delete": "false",
                            "tags.%": "0",
                            "vpc_id": "vpc-16cb4b72"
                        },
                        "meta": {
                            "e2bfb730-ecaa-11e6-8f88-34363bc7c4c0":
{
                                "create": 600000000000,
                                "delete": 600000000000
                            },
                            "schema_version": "1"
                        },
                        "tainted": false
                    },
                    "deposed": [],
```

```
                    "provider": "provider.aws"
                }
            },
            "depends_on": []
        },
        {
            "path": [
                "root",
                "web"
            ],
            "outputs": {
                "web_instance_ids": {
                    "sensitive": false,
                    "type": "string",
                    "value": "i-058a9c36e326add1c,i-
0377163446bd5a9d9"
                },
                "web_public_ips": {
                    "sensitive": false,
                    "type": "string",
                    "value": "52.23.195.180,52.90.92.220"
                }
            },
            "resources": {
                "aws_instance.web.0": {
                    "type": "aws_instance",
                    "depends_on": [],
                    "primary": {
                        "id": "i-058a9c36e326add1c",
                        "attributes": {
                            "ami": "ami-b73b63a0",
                            "arn": "arn:aws:ec2:us-east-
1:224557780148:instance/i-058a9c36e326add1c",
                            "associate_public_ip_address": "true",
                            "availability_zone": "us-east-1a",
                            "cpu_core_count": "1",
                            "cpu_threads_per_core": "1",
                            "credit_specification.#": "1",
                            "credit_specification.0.cpu_credits":
"standard",
                            "disable_api_termination": "false",
                            "ebs_block_device.#": "0",
                            "ebs_optimized": "false",
                            "ephemeral_block_device.#": "0",
                            "get_password_data": "false",
                            "iam_instance_profile": "",
                            "id": "i-058a9c36e326add1c",
```

```
                            "instance_state": "running",
                            "instance_type": "t2.micro",
                            "ipv6_addresses.#": "0",
                            "key_name": "aws-us-east",
                            "monitoring": "false",
                            "network_interface.#": "0",
                            "password_data": "",
                            "placement_group": "",
                            "primary_network_interface_id": "eni-
054da3f3fed4af2aa",
                            "private_dns": "ip-172-31-48-
53.ec2.internal",
                            "private_ip": "172.31.48.53",
                            "public_dns": "ec2-52-23-195-
180.compute-1.amazonaws.com",
                            "public_ip": "52.23.195.180",
                            "root_block_device.#": "1",
                            
"root_block_device.0.delete_on_termination": "true",
                            "root_block_device.0.iops": "0",
                            "root_block_device.0.volume_id": "vol-
06a935cde57d945d0",
                            "root_block_device.0.volume_size": "8",
                            "root_block_device.0.volume_type":
"standard",
                            "security_groups.#": "1",
                            "security_groups.3272911431": "web-
production-sg",
                            "source_dest_check": "true",
                            "subnet_id": "subnet-690a0942",
                            "tags.%": "0",
                            "tenancy": "default",
                            "volume_tags.%": "0",
                            "vpc_security_group_ids.#": "1",
                            "vpc_security_group_ids.2874570390":
"sg-075867a836925cbc1"
                        },
                        "meta": {
                            "e2bfb730-ecaa-11e6-8f88-34363bc7c4c0":
{
                                "create": 600000000000,
                                "delete": 1200000000000,
                                "update": 600000000000
                            },
                            "schema_version": "1"
                        },
                        "tainted": false
```

```
                    },
                    "deposed": [],
                    "provider": "provider.aws"
                },
                "aws_instance.web.1": {
                    "type": "aws_instance",
                    "depends_on": [],
                    "primary": {
                        "id": "i-0377163446bd5a9d9",
                        "attributes": {
                            "ami": "ami-b73b63a0",
                            "arn": "arn:aws:ec2:us-east-
1:224557780148:instance/i-0377163446bd5a9d9",
                            "associate_public_ip_address": "true",
                            "availability_zone": "us-east-1a",
                            "cpu_core_count": "1",
                            "cpu_threads_per_core": "1",
                            "credit_specification.#": "1",
                            "credit_specification.0.cpu_credits":
"standard",
                            "disable_api_termination": "false",
                            "ebs_block_device.#": "0",
                            "ebs_optimized": "false",
                            "ephemeral_block_device.#": "0",
                            "get_password_data": "false",
                            "iam_instance_profile": "",
                            "id": "i-0377163446bd5a9d9",
                            "instance_state": "running",
                            "instance_type": "t2.micro",
                            "ipv6_addresses.#": "0",
                            "key_name": "aws-us-east",
                            "monitoring": "false",
                            "network_interface.#": "0",
                            "password_data": "",
                            "placement_group": "",
                            "primary_network_interface_id": "eni-
09ee1c30f1daa3268",
                            "private_dns": "ip-172-31-48-
31.ec2.internal",
                            "private_ip": "172.31.48.31",
                            "public_dns": "ec2-52-90-92-
220.compute-1.amazonaws.com",
                            "public_ip": "52.90.92.220",
                            "root_block_device.#": "1",

"root_block_device.0.delete_on_termination": "true",
                            "root_block_device.0.iops": "0",
```

```
                            "root_block_device.0.volume_id": "vol-
0b971c7751639583d",
                            "root_block_device.0.volume_size": "8",
                            "root_block_device.0.volume_type":
"standard",
                            "security_groups.#": "1",
                            "security_groups.3272911431": "web-
production-sg",
                            "source_dest_check": "true",
                            "subnet_id": "subnet-690a0942",
                            "tags.%": "0",
                            "tenancy": "default",
                            "volume_tags.%": "0",
                            "vpc_security_group_ids.#": "1",
                            "vpc_security_group_ids.2874570390":
"sg-075867a836925cbc1"
                        },
                        "meta": {
                            "e2bfb730-ecaa-11e6-8f88-34363bc7c4c0":
{
                                "create": 600000000000,
                                "delete": 1200000000000,
                                "update": 600000000000
                            },
                            "schema_version": "1"
                        },
                        "tainted": false
                    },
                    "deposed": [],
                    "provider": "provider.aws"
                }
            },
            "depends_on": []
        }
    ]
}
```

Again, using our hypothetical we have lost the state files for this configuration and have orphaned resources in “production”. The method to recover is essentially the same except we must now include the module information in the resource address. Meaning the resource type and name needs to be prefixed with “module” and the module

name. Otherwise Terraform will import the resource into the root module which will not match with the configuration.

Web module import

Starting with the web module we must account for it having an “instance_count = 2”. So we will actually need to run the import command twice to pull in information for each instance. Additionally, because we have more than one we must introduce an index to the address because we have multiple instances.

```
$ terraform import module.web.aws_instance.web[0] i-
058a9c36e326add1c
module.web.aws_instance.web: Importing from ID "i-
058a9c36e326add1c"...
module.web.aws_instance.web: Import complete!
  Imported aws_instance (ID: i-058a9c36e326add1c)
module.web.aws_instance.web: Refreshing state... (ID: i-
058a9c36e326add1c)Import successful!The resources that were
imported are shown above. These resources are now in your Terraform
state and will henceforth be managed by Terraform.$ terraform
import module.web.aws_instance.web[1] i-0377163446bd5a9d9
module.web.aws_instance.web: Importing from ID "i-
0377163446bd5a9d9"...
module.web.aws_instance.web: Import complete!
  Imported aws_instance (ID: i-0377163446bd5a9d9)
module.web.aws_instance.web: Refreshing state... (ID: i-
0377163446bd5a9d9)Import successful!The resources that were
imported are shown above. These resources are now in your Terraform
state and will henceforth be managed by Terraform.
```

Security Group module import

Importing the security group you will notice that it creates one `aws_security_group_rule` for each rule in addition to the `aws_security_group`. This is expected behavior even though the rules in the configuration are in-line in the group. Best practice is apparently

one resource per rule and I suppose I should be following best practices here, but I'm not... in the case of this configuration the extraneous rule resources will be deleted on first `apply`.

```
$ terraform import module.sg.aws_security_group.web_sg sg-
075867a836925cbc1
module.sg.aws_security_group.web_sg: Importing from ID "sg-
075867a836925cbc1"...
module.sg.aws_security_group.web_sg: Import complete!
  Imported aws_security_group (ID: sg-075867a836925cbc1)
  Imported aws_security_group_rule (ID: sgrule-4158594480)
  Imported aws_security_group_rule (ID: sgrule-1096105989)
  Imported aws_security_group_rule (ID: sgrule-1888236961)
module.sg.aws_security_group_rule.web_sg-2: Refreshing state...
(ID: sgrule-1888236961)
module.sg.aws_security_group.web_sg: Refreshing state... (ID: sg-
075867a836925cbc1)
module.sg.aws_security_group_rule.web_sg: Refreshing state... (ID:
sgrule-4158594480)
module.sg.aws_security_group_rule.web_sg-1: Refreshing state...
(ID: sgrule-1096105989)Import successful!The resources that were
imported are shown above. These resources are now in your Terraform
state and will henceforth be managed by Terraform.
```

ELB module import

And finally we import the ELB which doesn't require anything special.

```
$ terraform import module.elb.aws_elb.elb web-elb
module.elb.aws_elb.elb: Importing from ID "web-elb"...
module.elb.aws_elb.elb: Import complete!
  Imported aws_elb (ID: web-elb)
module.elb.aws_elb.elb: Refreshing state... (ID: web-elb)Import
successful!The resources that were imported are shown above. These
resources are now in your Terraform state and will henceforth be
managed by Terraform.
```

The new state file will need some manual edits due to an existing bug with Security Group imports. On import the inline rules are also each imported as an `aws_security_group_rule`. Removing the JSON for

the individual rules from the new state file brings it back to parity with the existing infrastructure as defined by the Terraform configurations.