Terraform

devops, info

cli

```
export AWS_DEFAULT_REGION="us-east-1"
export AWS_ACCESS_KEY_ID="$(revealpass AWS_ACCESS_KEY_ID)"
export AWS_SECRET_ACCESS_KEY="$(revealpass AWS_SECRET_ACCESS_KEY)"
```

```
revealpass() {
    local DATA_PATH="/secure/storage/PASSWORDS/"
    local SERVICE=$1

    [[ ! -z ${SERVICE} ]] && [[ -f ${DATA_PATH}${SERVICE} ]] && cat
${DATA_PATH}${SERVICE}
}
```

básicos

- **init** : Initialize a new or existing Terraform configuration
- **validate** : Validates the Terraform files
- **plan** : Generate and show an execution plan
- **fmt** : Rewrites config files to canonical format
- **apply** : Builds or changes infrastructure
- **graph** : Create a visual graph of Terraform resources
- **output** : Read an output from a state file
- **destroy** : Destroy Terraform-managed infrastructure

otros

- **refresh** : Update local state file against real resources
- **show** : Inspect Terraform state or plan
- **taint** : Manually mark a resource for recreation
- **untaint** : Manually unmark a resource as tainted
- **debug** : Debug output management (experimental)
- **force-unlock** : Manually unlock the terraform state
- **state** : Advanced state management

autocomplete

- `terraform -install-autocomplete` : bash/zsh
- `terraform -uninstall-autocomplete`

enlaces documentación

0.11

- https://www.terraform.io/docs/configuration-0-11/interpolation.html: interpolación (referenciar variables,

atributos de recursos, llamar funciones)

de interés

output

- volcar información a fichero:

```
resource "local_file" "foo" {
    content  = "${tls_private_key.vm_adwriter.private_key_pem}"
    filename = "${path.cwd}/vm_adwriter.key"
}
```

- desencriptar password windows:

```
output "ec2_password" {
  value = "${rsadecrypt(aws_instance.vm_adwriter.password_data,
file("${path.cwd}/vm_adwriter.key"))}"
}
```

mapas

- usando **var.aws_region** como índice del mapa para seleccionar la imagen adecuada en función de la región

terraform.tfvars

```
aws_region = "eu-west-3" # París
```

main.tf

```
variable "aws_region" {
  type = "string"
}

variable "amis-aws-windows2016base" {
  # Windows_Server-2016-English-Full-Base-2019.02.13
  type = "map"

  default = {
    us-east-1 = "ami-0bf148826ef491d16" # Virigina
    eu-west-3 = "ami-0e3f0a08a6950f3e2" # París
  }
}
```

```
resource "aws_instance" "bastion_ad" {
  ami             = "${lookup(var.amis-aws-windows2016base, var.aws_region)}"
  instance_type   = "t2.micro"
  ...
}
```

data templates

template.tpl

```
Install-WindowsFeature -Name GPMC,RSAT-AD-PowerShell,RSAT-AD-AdminCenter,RSAT-ADDS-Tools,RSAT-DNS-Server
New-ADOrganizationalUnit -Name "${container_OU}" -Path "${base_path_AD}"
New-ADGroup -Name "${admin_vpn_group}" -SamAccountName ${admin_vpn_group} -GroupCategory Security -GroupScope Global -DisplayName ${admin_vpn_group} -Path "${vpn_OU_AD}${base_path_AD}"
$Attributes = @{
    Enabled = $true
    ChangePasswordAtLogon = $false
    Name = "${user}"
    AccountPassword = "${password}" | ConvertTo-SecureString -AsPlainText -Force
}
New-ADUser @Attributes
```

```
data "template_file" "vm_adwriter" {
  template = "${file("templates/aws_instance.vm_adwriter.tpl")}"

  vars {
    container_OU = "VPNGroups"
    base_path_AD = "OU=myapp,DC=myapp,DC=com"
    admin_vpn_group = "VPNAdmins"
    vpn_OU_AD = "OU=VPNGroups,"
    user = "FirstUser",
    password = "Password123"
  }
}
```

```
...
user_data = ${data.template_file.vm_adwriter.rendered}
...
```

ejemplos

- POC estructura AD
- terraform win-instance