

terraform practical task

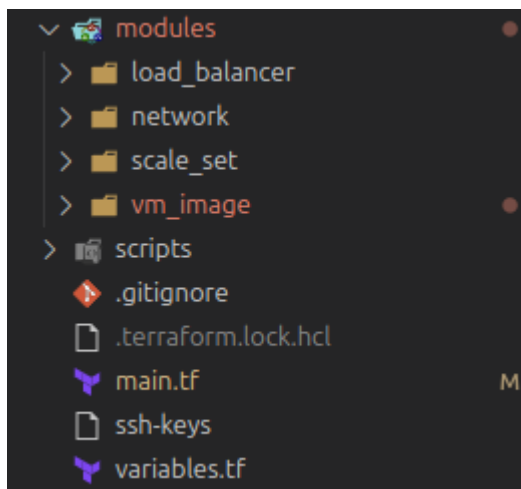
Practical task

Create a simple cloud infrastructure using Terraform by following the steps provided below:

1. Everything should be inside Terraform configuration files, manual changes are not allowed.
2. Try to use terraform modules to segregate resources by its types (compute, network).
3. Create a temporary VM with metadata script installing HTTP server (Apache) with a simple website inside.
4. Create an image from a temporary VM.
5. Terraform should create a scale set of 3 instances (use predefined image as source image for VMs), including external load balancer with health checks, everything should be done via terraform tf/tfstate files.
6. Every host should display server number/hostname to ensure that load balancer is working.
7. Users should be able to connect to the website in High Availability mode via external load balancer IP.
8. Add firewall for accessing external load balancer from limited IP addresses range and only for certain ports.
9. Use Public Cloud storage service as backend for Terraform state.

Modules

In the project there are 4 modules to decouple overall code structure.



Inputs common to every module are:

- Resource Group name
- Location
- Resources names prefix

Network module

Creates:

- Vnet
- Subnet
- Network Security Group (NSG) with rules
- NSG association to Subnet

VM Image module

Runs Bash script on temporary VM and after that creates an Image out of it. Outputs Image ID.

To create a new Image other temporary resources need to be created (VM, disk...), so there is a boolean `regenerate_image` variable to handle that. When it is set to `false` (default), all the temporary resources that were used to create an image are deleted and only image persist. If it is set to `true` all the temporary resources are created again (but not deleted after) to generate new Image.

So in order to create new Image from blank project using this terraform module we would run terraform apply twice:

```
terraform apply -var 'regenerate_image=true' # Create Image and temp resources
terraform apply # default regenerate_image=false. Deletes temp resources
```

Load Balancer module

Creates Azure Load Balancer PaaS resource with:

- Public IP
- Load Balancer backend pool
- Health Check probe
- Load Balancer rule

Scale set module

Creates VMs scale set out of given Image with option to run additional Bash script on each instance. Adds scale set to provided Load Balancer backend pool.

Example usage

First create resource group with appropriate location and name:

```
resource "azurerm_resource_group" "rg" {
  name      = "grid-terraform"
  location  = "westeurope"
}
```

In the network module add rules to allow SSH and HTTP.

```
module "network_module" {
  source          = "../modules/network"
  resource_group_name = azurerm_resource_group.rg.name
  location        = azurerm_resource_group.rg.location
  nsg_rules = [{
    name              = "ssh"
    priority           = 1001
    direction         = "Inbound"
    access            = "Allow"
    protocol          = "Tcp"
    source_port_range = "*"
    destination_port_range = "22"
    source_address_prefix = "*"
    destination_address_prefix = "*"
  },
  {
    name              = "http"
    priority           = 1002
    direction         = "Inbound"
    access            = "Allow"
    protocol          = "Tcp"
    source_port_range = "*"
    destination_port_range = "80"
    source_address_prefix = "*"
    destination_address_prefix = "*"
  }
]
```

In the VM Image module set required params and create Bash script to correctly configure Image.

```

module "vm_image" {
  source           = "./modules/vm_image"
  location         = azurerm_resource_group.rg.location
  resource_group_name = azurerm_resource_group.rg.name
  provision_script_path = "./scripts/initial.sh"
  temp_vm_subnet_id  = module.network_module.subnet_id
  regenerate_image   = var.regenerate_image
}

```

scripts >  initial.sh

```

1  #!/bin/bash
2  apt update
3  apt install -y apache2
4  echo "<html><body><h1>$(hostname)</h1></body></html>" > /home/azureuser/page.html
5  cp /home/azureuser/page.html /var/www/html/index.html
6

```

In the Load Balancer module set up ports used by backend services and port on which to access Load Balancer.

```

module "load_balancer" {
  source           = "./modules/load_balancer"
  resource_group_name = azurerm_resource_group.rg.name
  location         = azurerm_resource_group.rg.location
  lb_backend_port   = 80
  lb_frontend_port  = 80
  health_check_path = "/"
  health_check_port = 80
}

```


In the Scale Set module fill params outputted by previous modules and optionally add Bash script to run on every instance. Here I run script to create HTML with machine's hostname in h1 tag to be able to easily check that load balancer works.

```

module "vms_scale_set" {
  source                = "./modules/scale_set"
  location              = azurerm_resource_group.rg.location
  resource_group_name  = azurerm_resource_group.rg.name
  public_key            = file("./ssh-keys")
  subnet_id            = module.network_module.subnet_id
  image_id              = module.vm_image.image_id
  nsg_id               = module.network_module.nsg_id
  lb_backend_pool_id   = module.load_balancer.backend_pool_id
  provision_script_path = "./scripts/landing-page.sh"
  instances_count      = 2
}

```

```

scripts >  landing-page.sh
1  #!/bin/bash
2  echo "<html><body><h1>$(hostname)</h1></body></html>" > /home/azureuser/page.html
3  cp /home/azureuser/page.html /var/www/html/index.html
4

```

Running configuration

```

01:30:49 adrwal@olek-desktop-pc terraform-task ±|main x|→ terraform apply -var 'regenerate_image=true'

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following
symbols:
+ create

Terraform will perform the following actions:

# azurerm_resource_group.rg will be created
+ resource "azurerm_resource_group" "rg" {
+ id           = (known after apply)
+ location     = "westeurope"

Plan: 19 to add, 0 to change, 0 to destroy.

Changes to Outputs:
+ lb_frontend_ip = (known after apply)

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

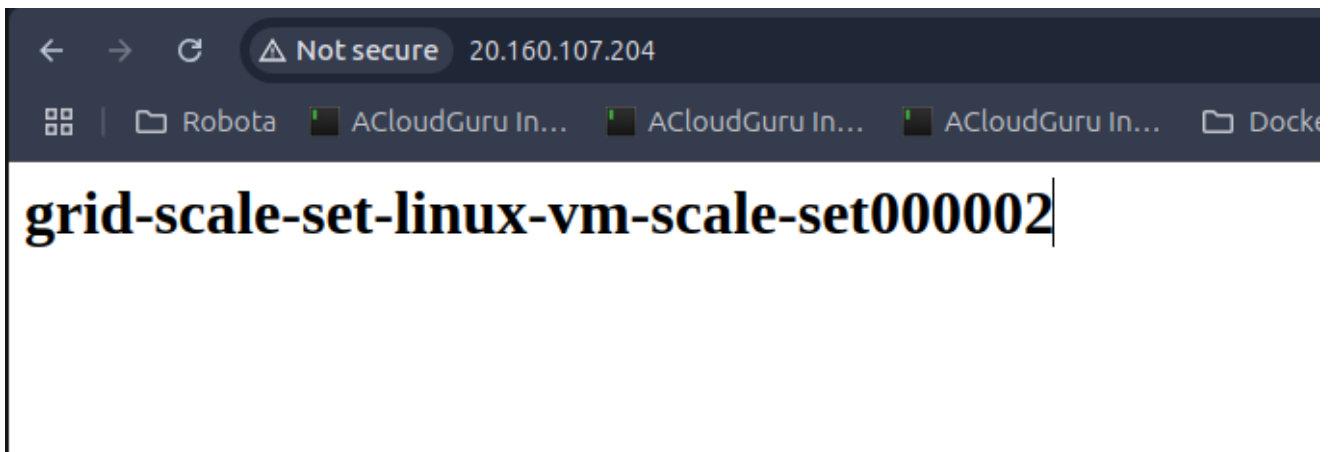
```

```
module.vms_scale_set.azurelinux_virtual_machine_scale_set.linux_vm_scale_set: Still creating... [50s elapsed]
module.vms_scale_set.azurelinux_virtual_machine_scale_set.linux_vm_scale_set: Still creating... [1m0s elapsed]
module.vms_scale_set.azurelinux_virtual_machine_scale_set.linux_vm_scale_set: Still creating... [1m10s elapsed]
module.vms_scale_set.azurelinux_virtual_machine_scale_set.linux_vm_scale_set: Creation complete after 1m13s [id=/subscriptions/9/resourceGroups/grid-terraform/providers/Microsoft.Compute/virtualMachineScaleSets/grid-scale-set-linux-vm-scale-set]

Apply complete! Resources: 19 added, 0 changed, 0 destroyed.

Outputs:

lb_frontend_ip = "20.160.107.204"
01:36:03 adrwal@olek-desktop-pc terraform-task ±|main x|→
```



Azure Load Balancer uses hash based routing of 5 elements tuple (source IP, source port, dest IP, dest port, protocol). Because of that, after I refresh this page I will get the same response. To see that Load Balancer works I have to get response from other instance and to do that I can modify source port by opening this page from other browser.



In Azure we created 19 resources:

grid-terraform Resource group

Search

Create Manage view Delete resource group Refresh Export to CSV Open query Assign tags Move Delete Export

Overview

- Activity log
- Access control (IAM)
- Tags
- Resource visualizer
- Events
- Settings
 - Deployments
 - Security
 - Deployment stacks
 - Policies
 - Properties
 - Locks
- Cost Management
 - Cost analysis
 - Cost alerts (preview)
 - Budgets
 - Advisor recommendations
- Monitoring
 - Insights (preview)

Essentials

Subscription (move) : [Azure for Students](#) Deployments : [No deployments](#)

Subscription ID : c539c22d-81df-4900-93d0-f5d20ccc64b9 Location : West Europe

Tags (edit) : [Add tags](#)

Resources Recommendations (1)

Filter for any field... Type equals all Location equals all Add filter

Showing 1 to 11 of 11 records. ☐ Show hidden types

Name ↑↓	Type ↑↓	Location ↑↓
<input type="checkbox"/> base-temp-osdisk	Disk	West Europe
<input type="checkbox"/> grid-image-base-temp-vm	Virtual machine	West Europe
<input type="checkbox"/> grid-image-base-vm-disk	Disk	West Europe
<input type="checkbox"/> grid-image-base-vm-image	Image	West Europe
<input type="checkbox"/> grid-image-base-vm-snapshot	Snapshot	West Europe
<input type="checkbox"/> grid-image-base-vm-temp-nic	Network Interface	West Europe
<input type="checkbox"/> grid-lb-lb	Load balancer	West Europe
<input type="checkbox"/> grid-lb-lb-ip	Public IP address	West Europe
<input type="checkbox"/> grid-network-nsg	Network security group	West Europe
<input type="checkbox"/> grid-network-vnet	Virtual network	West Europe
<input type="checkbox"/> grid-scale-set-linux-vm-scale-set	Virtual machine scale set	West Europe

Some of them were used only to create `grid-image-base-vm-image` resource. To remove them I can run `terraform apply` again, with `regenerate_image` variable set to false (default).

```
lb_frontend_ip = "20.160.107.204"
01:36:03 adrwal@olek-desktop-pc terraform-task ±|main x|→ terraform apply
azurerm_resource_group.rg: Refreshing state... [id=/subscriptions/c539c22d-81df-4900-93d0-f5d20ccc64b9/resourceGroups/grid-terraform]
module.network_module.azure_rm_network_security_group.nsg: Refreshing state..
```

Plan: 0 to add, 0 to change, 5 to destroy.

Do you want to perform these actions?

Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

```

module.vm_image.azure_rm_network_interface.temp_nic[0]: Still de
elapsed]
module.vm_image.azure_rm_network_interface.temp_nic[0]: Destruct

Apply complete! Resources: 0 added, 0 changed, 5 destroyed.

Outputs:

lb_frontend_ip = "20.160.107.204"
01:49:35 adrwal@olek-desktop-pc terraform-task ±|main x|→

```

Home >

grid-terraform ☆ ☆ ...
Resource group

Search

+ Create Manage view Delete resource group Refresh Export to CSV Open query Assign tags Move Delete

Overview

- Activity log
- Access control (IAM)
- Tags
- Resource visualizer
- Events
- Settings
 - Deployments
 - Security
 - Deployment stacks
 - Policies
 - Properties
 - Locks
- Cost Management
 - Cost analysis
 - Cost alerts (preview)
 - Budgets
 - Advisor recommendations
- Monitoring

Essentials

Subscription (move): [Azure for Students](#) Deployments: [No deployments](#)

Subscription ID: c539c22d-81df-4900-93d0-f5d20ccc64b9 Location: West Europe

Tags (edit): [Add tags](#)

Resources Recommendations (1)

Filter for any field... Type equals all Location equals all Add filter

Showing 1 to 6 of 6 records. Show hidden types

Name ↑↓	Type ↑↓
grid-image-base-vm-image	Image
grid-lb-lb	Load balancer
grid-lb-lb-ip	Public IP address
grid-network-nsg	Network security group
grid-network-vnet	Virtual network
grid-scale-set-linux-vm-scale-set	Virtual machine scale set

Limiting access from certain ports and IPs

To allow access only from certain ports and IPs change rule in network module allowing http traffic in the subnet.

```

36     {
37         name                = "http"
38         priority             = 1002
39         direction           = "Inbound"
40         access               = "Allow"
41         protocol             = "Tcp"
42         source_port_range    = "25565-25570"
43         destination_port_range = "80"
44         source_address_prefix = "*"
45         destination_address_prefix = "*"
46     }

```

Above change limits access to clients with ports in range 25565-25570.

Blocking access for certain IPs is limited when using NSG rules only, so instead of exposing

Azure Load Balancer to public I had to create Azure Firewall and make Load Balancer Internal.

In this new setup only firewall has public IP and before traffic is routed to now internal Load Balancer it is filtered by firewall rules.

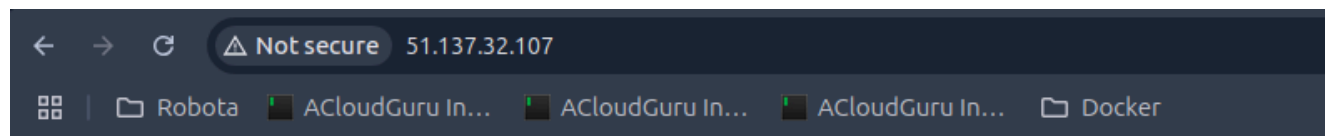
```
57 module "firewall" {
58   source = "./modules/firewall"
59   location = azurerm_resource_group.rg.location
60   resource_group_name = azurerm_resource_group.rg.name
61   virtual_network_name = module.network_module.vnet_name
62   load_balancer_ip = module.load_balancer.private_ip
63   resources_subnet_ip = module.network_module.subnet_id
64 }
```

After this change when running `terraform apply`, firewall IP is outputted instead:

```
Apply complete! Resources: 27 added, 0 changed, 0 destroyed.

Outputs:

firewall_public_ip = "51.137.32.107"
01:48:39 adrwal@olek-desktop-pc terraform-task ±|main x|→
```



grid-scale-set-linux-vm-scale-set000003

Public Cloud storage service as backend for Terraform state.

In Terraform update `terraform` block configuration:

```

1 terraform {
2   required_version = ">=1.0.0"
3   required_providers {
4     azurerm = {
5       source = "hashicorp/azurerm"
6       version = "~>3.0"
7     }
8   }
9   backend "azurerm" {
10    resource_group_name = "StorageAccount-Grid"
11    storage_account_name = "gridterraform"
12    container_name = "tfstate"
13  }
14 }

```

On Azure create new resource group separate from resource group used to manage the infrastructure.

In the resource group add `Storage account` and in it, create a container which will store the state file.

The screenshot displays the Azure portal interface. At the top, the 'StorageAccount-Grid' resource group is selected. The left sidebar shows the 'Overview' tab, with a search bar and various management options like 'Activity log', 'Access control (IAM)', 'Tags', 'Resource visualizer', 'Events', 'Settings', 'Deployments', 'Security', 'Deployment stacks', 'Policies', and 'Properties'.

The main content area shows the 'Essentials' section for the resource group, including subscription information (Subscription ID: c539c22d-81df-4900-93d0-f5d20ccc64b9, Location: West) and a list of resources. The 'Resources' tab is active, showing a table with one resource: 'gridterraform' (Storage account).

Below this, the 'gridterraform | Containers' section is shown. It includes a search bar and a table of containers. The table has columns for 'Name', 'Last modified', and 'Anonymous access'. Two containers are listed: '\$logs' and 'tfstate', both created on 12/18/2024.

Name	Last modified	Anonymous access
\$logs	12/18/2024, 2:02:27 AM	Private
tfstate	12/18/2024, 2:03:43 AM	Private

Copy storage account access key. It will be needed when running `terraform init`

Home > Resource groups > StorageAccount-Grid > gridterraform

gridterraform | Access keys

Storage account

Search

Set rotation reminder Refresh Give feedback

Access keys authenticate your applications' requests to this storage account. Keep your keys in a secure location like Azure Key Vault, and replace them often with new keys. The two keys allow you to replace one while still using the other.

Remember to update the keys with any Azure resources and apps that use this storage account.
[Learn more about managing storage account access keys](#)

Storage account name
gridterraform

key1 Rotate key
Last rotated: 12/18/2024 (0 days ago)
Key
.....
Connection string
.....

key2 Rotate key
Last rotated: 12/18/2024 (0 days ago)
Key
.....
Connection string
.....

02:07:53 adrwal@olek-desktop-pc terraform-task ±|main x|→ terraform init
Initializing the backend...
key
The blob key.
Enter a value: YytwPT8QwY/bT/+uKF8pk84C3j0BKcTlbIGxQb/Br0sLR8LtIgHZgEY2tIVQ7z7yc9WBZGygct5X+ASt3V4lJQ==
Successfully configured the backend "azurerm"! Terraform will automatically use this backend unless the backend configuration changes.
Initializing modules...
Initializing provider plugins...
- Reusing previous version of hashicorp/azurerm from the dependency lock file
- Using previously-installed hashicorp/azurerm v3.117.0
Terraform has been successfully initialized!
You may now begin working with Terraform. Try running "terraform plan" to see any changes that are required for your infrastructure. All Terraform commands should now work.
If you ever set or change modules or backend configuration for Terraform, rerun this command to reinitialize your working directory. If you forget, other commands will detect it and remind you to do so if necessary.
02:08:23 adrwal@olek-desktop-pc terraform-task ±|main x|→

After running `terraform apply` state file has been updated and stored in Azure container.

```
• ^[[A02:13:15 adrwal@olek-desktop-pc terraform-task ±|main x|→ terraform apply -var 'regenerate_image=true'
```

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:

+ create

Terraform will perform the following actions:

```
# azure_rm_resource_group.rg will be created
+ resource "azure_rm_resource_group" "rg" {
  + id          = (known after apply)
  + location    = "westeurope"
  + name        = "grid-terraform"
}
```

```
module.vms_scale_set.azure_rm_linux_virtual_machine_scale_set.linux_vm
subscriptions/c539c22d-81df-4900-93d0-f5d20ccc64b9/resourceGroups/grid-ter
chineScaleSets/grid-scale-set-linux-vm-scale-set]
```

Apply complete! Resources: 19 added, 0 changed, 0 destroyed.

```
02:18:41 adrwal@olek-desktop-pc terraform-task ±|main x|→
```

Home > gridterraform | Containers > tfstate >

tfstate
Container

Search

Upload Change access level

Overview

Diagnose and solve problems

Access Control (IAM)

Settings

Shared access tokens

Access policy

Properties

Metadata

Authentication method: Access key (Switch to Microsoft Entra user account)
Location: tfstate / YytwPT8QwY / bT / +uKF8pk84C3j0BKcTlBtGxQb

Search blobs by prefix (case-...)

Show deleted blobs

Add filter

Name

BrOsLR8LtlgHZgEY2tIVQ7z7yc...

YytwPT8QwY/bT/+uKF8pk84C3j0BKcTlBtGxQb/BrOsLR8LtlgHZgEY2tIVQ7z7yc9W
Blob

Save Discard Download Refresh Delete

Overview Versions Snapshots Edit Generate SAS

The file "YytwPT8QwY/bT/+uKF8pk84C3j0BKcTlBtGxQb/BrOsLR8LtlgHZgEY2tIVQ7z7yc9W8ZGygct5X+ASl3V4UQ==" may not render correctly as it contains an un

```
1 {
2   "version": 4,
3   "terraform_version": "1.10.2",
4   "serial": 5,
5   "lineage": "3a76c8b3-b319-be4f-7e0d-a4d683f8e52c",
6   "outputs": {},
7   "resources": [
8     {
9       "mode": "managed",
10      "type": "azure_rm_resource_group",
11      "name": "rg",
12      "provider": "provider[\"registry.terraform.io/hashicorp/azurerm\"]",
13      "instances": [
14        {
15          "schema_version": 0,
16          "attributes": {
17            "id": "/subscriptions/c539c22d-81df-4900-93d0-f5d20ccc64b9/resourceGroups/grid-ter",
18            "location": "westeurope",
19            "managed_by": "",
20            "name": "grid-terraform",
21            "tags": null,
22            "timeouts": null
23          },
24          "sensitive_attributes": [],
25          "private": "eyJlMmMjYjczMC1lY2FhLTExZTYtOGY4ODZNDM2M2JjN2M0YzA1OmsiY3JlYXRlIjo1NDAwMl",
26        }
27      ]
28    },
29    {
30      "module": "module.load_balancer",
31      "mode": "managed",
32      "type": "azurerm_lb",
```

Preview