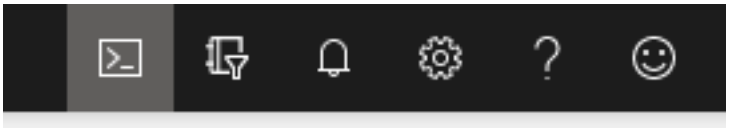
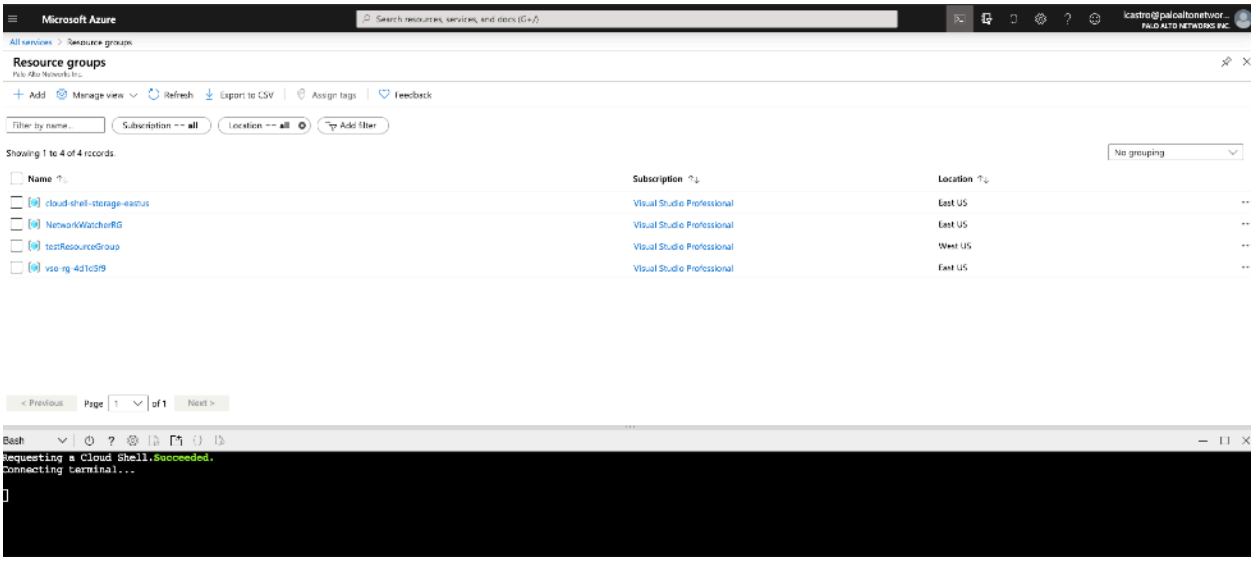
Create our virtual networks in two different regions according the instructions files.

1. Log in to the Azure portal at https://portal.azure.com.

Step 1 – Open Azure Cloud Shell

Click the right side bottom to access Cloud Shell

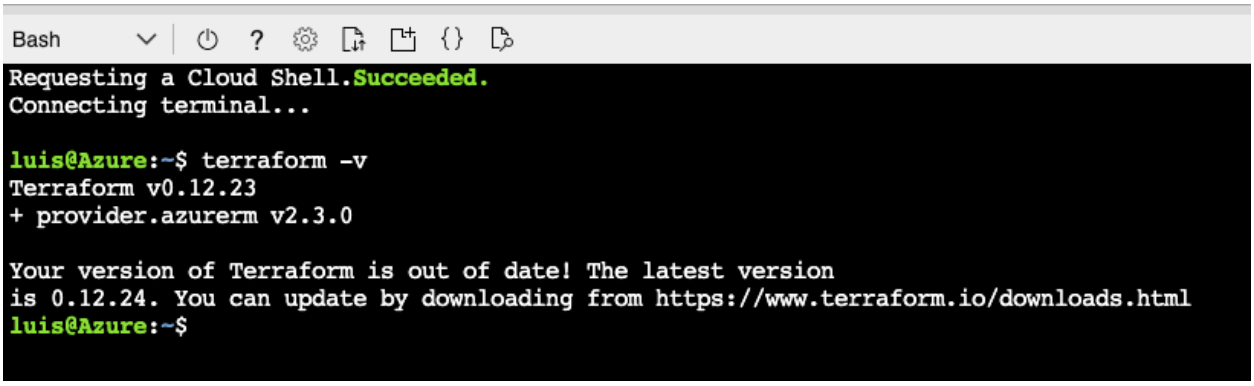




Step 2 - Verify Terraform Version

Inside the Azure Cloud Shell type the following

$ terraform -v



Step 3 - Verify Subscription ID

Run the following command

$az account list --query "[].{name:name, subscriptionId:id, tenantId:tenantId}"

Save the output in a text file

```
luis@Azure:~$ az account list --query "[].{name:name, subscriptionId:id, tenantId:tenantId}"
[
  {
    "name": "Visual Studio Professional",
    "subscriptionId": "6bcf9eb8-9994-44d8-bd69-4bad91926bb5",
    "tenantId": "66b66353-3b76-4e41-9dc3-fee328bd400e"
  }
]
luis@Azure:~$
```

Step 4 - Setup Service Principal to use with Terraform

Select the right account

$ az account set --subscription 6bcf9eb8-9994-44d8-bd69-4bad91926bb5

Run the following command

$ az ad sp create-for-rbac --role="Contributor"

Check the creation process and save the output in a text file

```
Bash
Your version of Terraform is out of date! The latest version
is 0.12.24. You can update by downloading from https://www.terraform.io/downloads.html
luis@Azure:~$ az ad sp create-for-rbac --role="Contributor"
Creating a role assignment under the scope of "/subscriptions/6bcf9eb8-9994-44d8-bd69-4bad91926bb5"
  Retrying role assignment creation: 1/36
{
  "appId": "0e933552-e011-4671-8ade-493cf99833e7",
  "displayName": "azure-cli-2020-03-29-21-16-54",
  "name": "http://azure-cli-2020-03-29-21-16-54",
  "password": "70f9e903-321d-4b34-919c-c6514031fb69",
  "tenant": "66b66353-3b76-4e41-9dc3-fee328bd400e"
}
luis@Azure:~$
```

Step 5 - Create the Terraform Variables

Copy the following commands inside the Azure Cloud Shell, and from the last step change with values of: Subscription, appID, Password and Tenant

```
#!/bin/sh
echo "Setting environment variables for Terraform"
export ARM_SUBSCRIPTION_ID=your_subscription_id
export ARM_CLIENT_ID=your_appId
export ARM_CLIENT_SECRET=your_password
export ARM_TENANT_ID=your_tenant_id
```

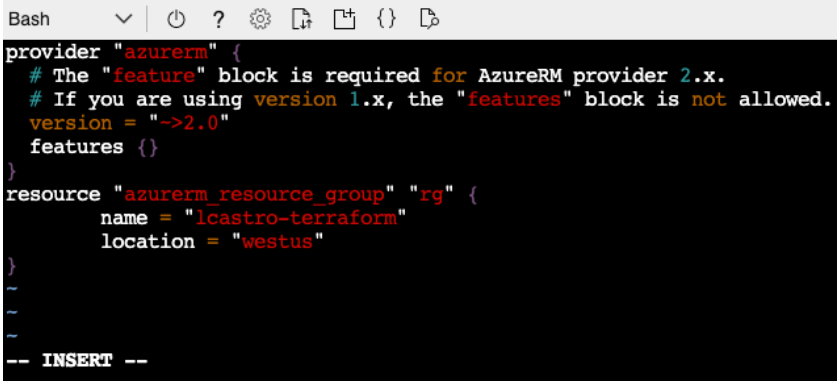
Step 6 - Create a Terraform Test File
Using Vi or Nano create a test file with Terraform extension with the following content

$ vi test.tf

Change the Name value for your username+terraform

Change location for your designated location

```
provider "azurerm" {
  # The "feature" block is required for AzureRM provider 2.x.
  # If you are using version 1.x, the "features" block is not allowed.
  version = "~>2.0"
  features {}
}
resource "azurerm_resource_group" "rg" {
        name = "lcastro-terraform"
        location = "westus"
}
```



Exit and Save from vi with the following commands: ESC + : + WQ

Step 7 - Initialize the Terraform Deployment

$ terraform init

Verify is successfully initiated

```
luis@Azure:~$ vi test.tf
luis@Azure:~$ terraform init

Initializing the backend...

Initializing provider plugins...

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.
luis@Azure:~$
```

Step 8 - Preview Terraform Actions

$ terraform plan

```
Bash
luis@Azure:~$ terraform plan
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not be
persisted to local or remote state storage.

azurerm_resource_group.rg: Refreshing state... [id=/subscriptions/6bcf9eb8-9994-44d8-bd69-4bad91926bb5/resourceGroups/testResourceGroup]

------------------------------------------------------------------------

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
-/+ destroy and then create replacement

Terraform will perform the following actions:

  # azurerm_resource_group.rg must be replaced
-/+ resource "azurerm_resource_group" "rg" {
      ~ id       = "/subscriptions/6bcf9eb8-9994-44d8-bd69-4bad91926bb5/resourceGroups/testResourceGroup" -> (known after apply)
        location = "westus"
      ~ name     = "testResourceGroup" -> "lcastro-terraform" # forces replacement
      - tags     = {} -> null
    }

Plan: 1 to add, 0 to change, 1 to destroy.

------------------------------------------------------------------------

Note: You didn't specify an "-out" parameter to save this plan, so Terraform
can't guarantee that exactly these actions will be performed if
"terraform apply" is subsequently run.

luis@Azure:~$
```

Step 9 - Execute Terraform Plan

$ terraform apply

When prompted ask “yes” and hit Enter

Verify steps and output

```
luis@Azure:~$ terraform apply
azurerm_resource_group.rg: Refreshing state... [id=/subscriptions/6bcf9eb8-9994-44d8-bd69-4bad91926bb5/resourceGroups/testResourceGroup]

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
-/+ destroy and then create replacement

Terraform will perform the following actions:

  # azurerm_resource_group.rg must be replaced
-/+ resource "azurerm_resource_group" "rg" {
      - id       = "/subscriptions/6bcf9eb8-9994-44d8-bd69-4bad91926bb5/resourceGroups/testResourceGroup" -> (known after apply)
        location = "westus"
      - name     = "testResourceGroup" -> "lcastro-terraform" # forces replacement
      - tags     = {} -> null
    }

Plan: 1 to add, 0 to change, 1 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

azurerm_resource_group.rg: Destroying... [id=/subscriptions/6bcf9eb8-9994-44d8-bd69-4bad91926bb5/resourceGroups/testResourceGroup]
azurerm_resource_group.rg: Still destroying... [id=/subscriptions/6bcf9eb8-9994-44d8-bd69-...26bb5/resourceGroups/testResourceGroup, 10s elapsed]
azurerm_resource_group.rg: Still destroying... [id=/subscriptions/6bcf9eb8-9994-44d8-bd69-...26bb5/resourceGroups/testResourceGroup, 20s elapsed]
azurerm_resource_group.rg: Still destroying... [id=/subscriptions/6bcf9eb8-9994-44d8-bd69-...26bb5/resourceGroups/testResourceGroup, 30s elapsed]
azurerm_resource_group.rg: Destruction complete after 33s
azurerm_resource_group.rg: Creating...
azurerm_resource_group.rg: Creation complete after 1s [id=/subscriptions/6bcf9eb8-9994-44d8-bd69-4bad91926bb5/resourceGroups/lcastro-terraform]

Apply complete! Resources: 1 added, 0 changed, 1 destroyed.
luis@Azure:~$
```

Step 10 - Verify Azure Resource Groups inside the Web Console

Click **Resource Groups** and validate the Name and Location