Lab: Extending Terraform - Non Cloud Providers

In this challenge, you will use non cloud specific providers to assist in containing common tasks within Terraform.

- Task 1: Random Generator
- Task 2: SSH Public/Private Key Generator
- Task 3: Cleanup

Create Folder Structure

Change directory into a folder specific to this challenge.

For example: `/workstation/terraform/extending-terraform`.

Create a `main.tf` file, we will add to this file as we go.

Task 1: Random Generator

Create a random password:

```
resource "random_password" "password" {
  length  = 16
  special = true
}

output "password" {
  value = random_password.password.result
  sensitive = true
}
```

Run `terraform init`,

Run `terraform apply -auto-approve`.

```
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

Outputs:

password = <sensitive>
```

Add the following configuration:

```
resource "random_uuid" "guid" {
}

output "guid" {
  value = random_uuid.guid.result
}
```

```
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

Outputs:

guid = 84502be0-13fc-4a49-ce02-4ab4e1b493ca
password = <sensitive>
```

> Note: The password was NOT regenerated, why is this?

Now **update** the `random_guid` resource to use a "keepers" arguement:

```
resource "random_uuid" "guid" {
  keepers = {
    datetime = timestamp()
  }
}
```

Run `terraform apply -auto-approve` several times in a row, what happens to the guid output?

Task 2: SSH Public/Private Key Generator

Use Terraform to generate public/private SSH keys dynamically.

```
resource "tls_private_key" "tls" {
  algorithm = "RSA"
}
```

Run `terraform init`,

Run `terraform apply -auto-approve`.

```
Apply complete! Resources: 2 added, 0 changed, 2 destroyed.

Outputs:

guid = ad4efda0-d17d-559c-5cfe-1ed3ab9ce86b
password = <sensitive>
```


This is great, but I wan the keys as files, how?

Add the following config:

```
resource "local_file" "tls-public" {
  filename = "id_rsa.pub"
  content  = tls_private_key.tls.public_key_openssh
}

resource "local_file" "tls-private" {
  filename = "id_rsa.pem"
  content  = tls_private_key.tls.private_key_pem

  provisioner "local-exec" {
    command = "chmod 600 id_rsa.pem"
  }
}
```

What is this “local-exec”?

If you have too wide of permissions on a private SSH key, you can see the following error when trying to use that key to access a remote system:

```
Permissions 0777 for './id_rsa.pem' are too open.
It is recommended that your private key files are NOT accessible by others
    .
This private key will be ignored.
```

This local exec will run a `chmod` on the file after it is created.

Run `terraform init`,

Run `terraform apply -auto-approve`.

You should now have two new files in your current working directory.

Now delete one of the files (i.e. `rm id_rsa.pem`).

Run a `terraform plan`, what changes (if any) are needed? Is this what you expected?

Run `terraform apply -auto-approve` to restore any deleted files.

Task 3: Clean up

When you are done, run `terraform destroy` to remove everything (including the private and public key files).