Step-01: Introduction

- Understand about remote-exec Provisioner
- The `remote-exec` provisioner invokes a script on a remote resource after it is created.
- This can be used to run a configuration management tool, bootstrap into a cluster, etc.

Step-02: Create / Review Provisioner configuration

- **Usecase:**

1. We will copy a file named `file-copy.html` using `File Provisioner` to "/tmp" directory
2. Using `remote-exec provisioner`, using linux commands we will in-turn copy the file to Apache Webserver static content directory `/var/www/html` and access it via browser once it is provisioned

```
 # Copies the file-copy.html file to /tmp/file-copy.html
  provisioner "file" {
    source      = "apps/file-copy.html"
    destination = "/tmp/file-copy.html"
  }

# Copies the file to Apache Webserver /var/www/html directory
  provisioner "remote-exec" {
    inline = [
      "sleep 120",  # Will sleep for 120 seconds to ensure Apache webserver is provisioned using custom_data
      "sudo cp /tmp/file-copy.html /var/www/html"
    ]
  }
```

Step-03: Review Terraform manifests & Execute Terraform Commands

```
# Terraform Initialize
terraform init

# Terraform Validate
terraform validate

# Terraform Format
terraform fmt

# Terraform Plan
terraform plan

# Terraform Apply
terraform apply -auto-approve

# Verify
1) Login to Azure VM Instance
ssh -i ssh-keys/terraform-azure.pem azureuser@PUBLIC_IP_ADDRESSS_OF_YOUR_VM
ssh -i ssh-keys/terraform-azure.pem azureuser@54.197.54.126

2) Verify /tmp for file named file-copy.html all files copied (ls -lrt /tmp/file-copy.html)
3) Verify /var/www/html for a file named file-copy.html (ls -lrt /var/www/html/file-copy.html)
4) Access via browser http://<Public-IP>/file-copy.html
```

Step-04: Clean-Up Resources & local working directory

```
# Terraform Destroy
terraform destroy -auto-approve

# Delete Terraform files
rm -rf .terraform*
rm -rf terraform.tfstate*
```