Terraform Provisioners

End-to-End Infrastructure Deployment with Terraform

- Till now we have been working only on creation and destruction of infrastructure scenarios.
- **Let’s take an example:**
- We want to create a virtual machine & installed web server in it with Terraform.
- **Problem**: It is only a VM, it does not have any software installed.
    - **What if we want a complete end to end solution?**

Understanding Provisioners

•Provisioners are used to execute scripts on a local or remote machines as part of resource creation or destruction.

Let’s take an example:

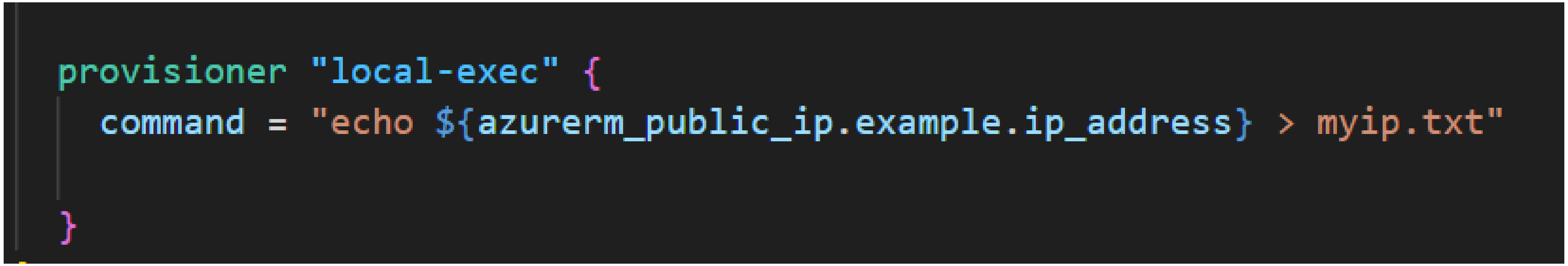
On creation of Instance, execute a script which installs NGINX webserver.

•Terraform has capability to turn provisioners both at the time of resource creation as well as destruction.

- **There are three main types of provisioners:**
  - Local-exec provisioner
  - Remote-exec provisioner
  - File provisioner

Local Exec Provisioner

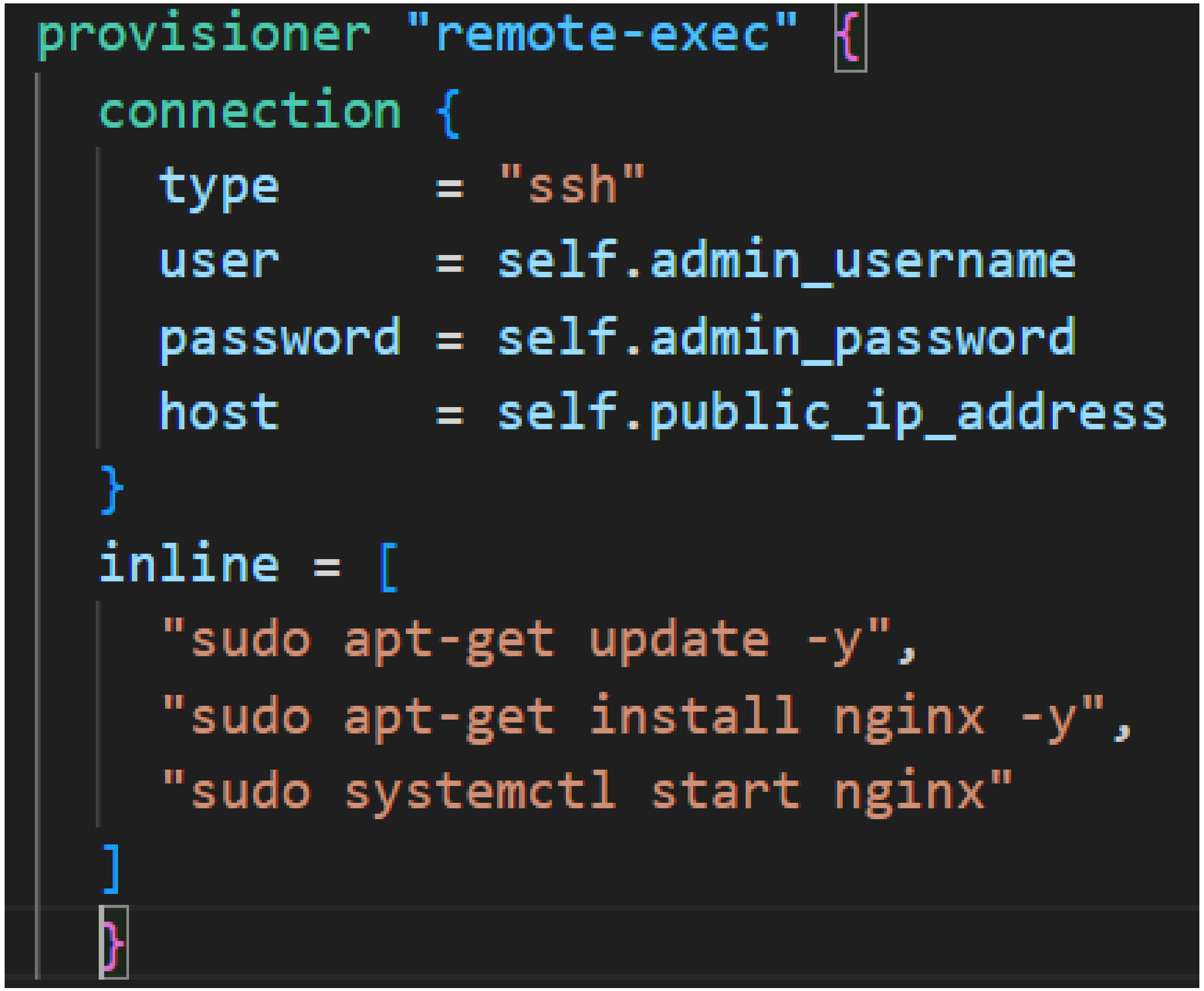
- Local-exec provisioners allow us to invoke local executable after resource is created.
- Use-case : After the VM is created, this provisioner runs a command to write the VM's public IP address to a file named myip.txt.



```
provisioner "local-exec" {
  command = "echo ${azurerm_public_ip.example.ip_address} > myip.txt"

}
```

Remote Exec Provisioners

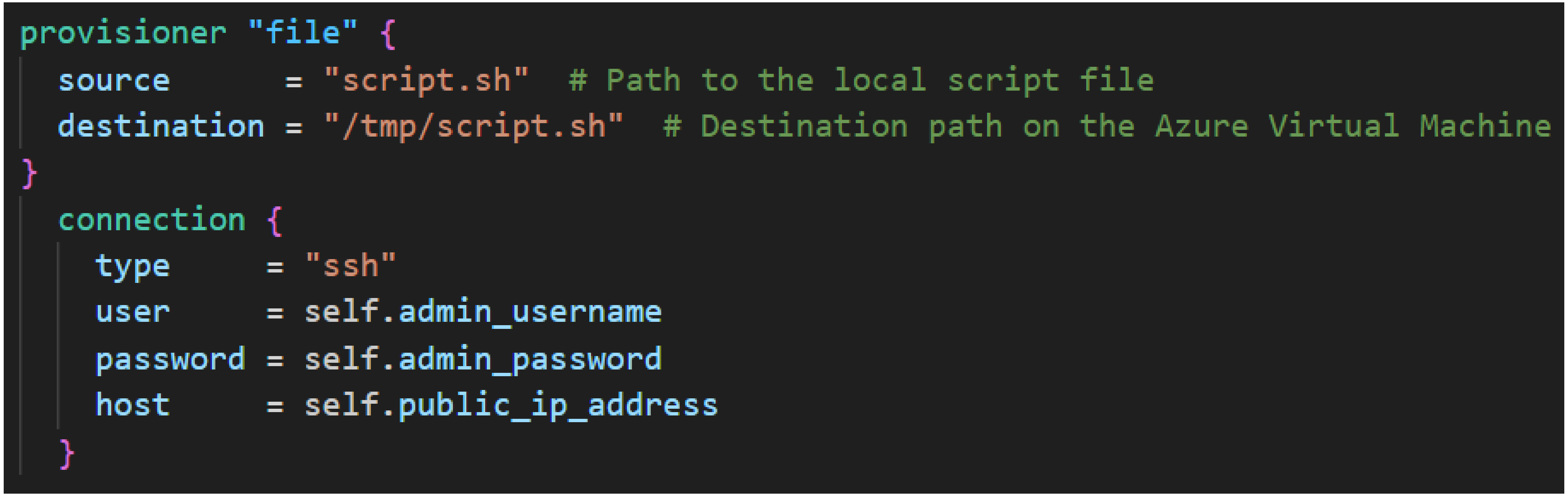
- Remote-exec provisioners allows us to invoke scripts directly on the remote server. it's useful for tasks that need to be executed after the server is created
- **Let's take an example:** Once you create a VM, you want to install nginx & start the nginx service.



```
provisioner "remote-exec" {
  connection {
    type     = "ssh"
    user     = self.admin_username
    password = self.admin_password
    host     = self.public_ip_address
  }
  inline = [
    "sudo apt-get update -y",
    "sudo apt-get install nginx -y",
    "sudo systemctl start nginx"
  ]
  }
```

File Provisioners

- File Provisioners is a tool used to copy files or directories from the local machine to a remote server.
- **Let's take an example:** Once you create a VM, you want to copy the files.



```
provisioner "file" {
  source      = "script.sh"  # Path to the local script file
  destination = "/tmp/script.sh"  # Destination path on the Azure Virtual Machine
}
  connection {
    type     = "ssh"
    user     = self.admin_username
    password = self.admin_password
    host     = self.public_ip_address
  }
```

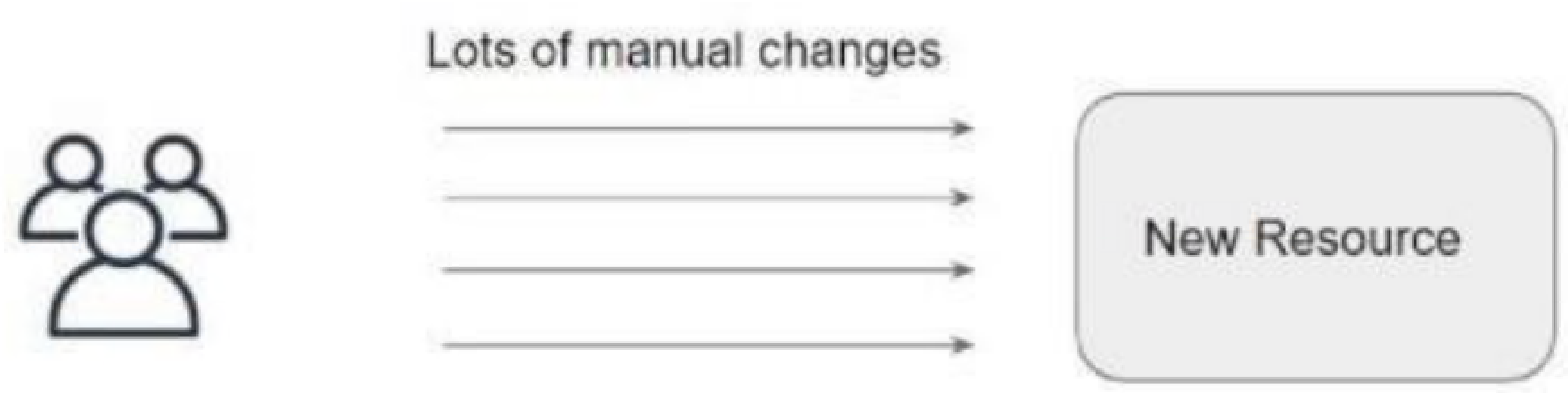
Join us in our Adventure

https://www.linkedin.com/in/akash-kumar-480b3858/

https://www.instagram.com/akash_sinha08/

Terraform taint

Overview of Taint



Youhave created a new resource via Terraform

Users have made a lot of manual changes (both infrastructure and inside the server)

Twoways to deal with this: Import the changes to Terraform/ **Delete & Recreate the resource**

Taint

. The terraform taint command manually marks a Terraform-managed resource as tainted, forcing it to   be destroyed and recreated on the next apply.

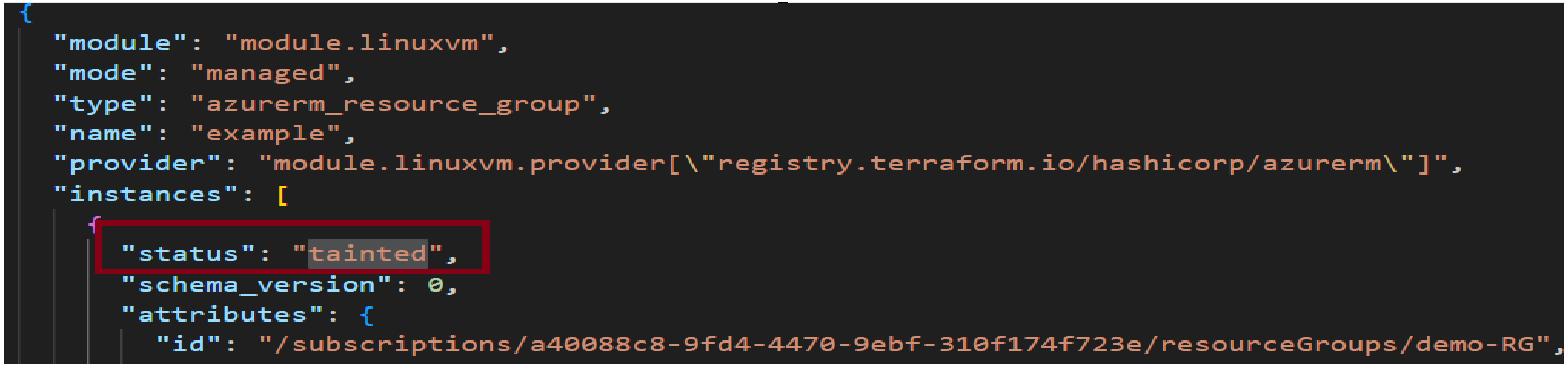
Terraform Taint Command :

- terraform taint <resource_type>.<resource_name>

EG: terraform taint `azurerm_linux_virtual_machine`.demo_vm

- This command will not modify infrastructure, but does modify the state file in order to mark a resource as tainted.

- Once a resource is marked as tainted, the next plan will show that the resource will be destroyed and recreated and the next apply will implement the change.



```
{
  "module": "module.linuxvm",
  "mode": "managed",
  "type": "azurerm_resource_group",
  "name": "example",
  "provider": "module.linuxvm.provider[\"registry.terraform.io/hashicorp/azurerm\"]",
  "instances": [
    {
      "status": "tainted",
      "schema_version": 0,
      "attributes": {
        "id": "/subscriptions/a40088c8-9fd4-4470-9ebf-310f174f723e/resourceGroups/demo-RG",
```

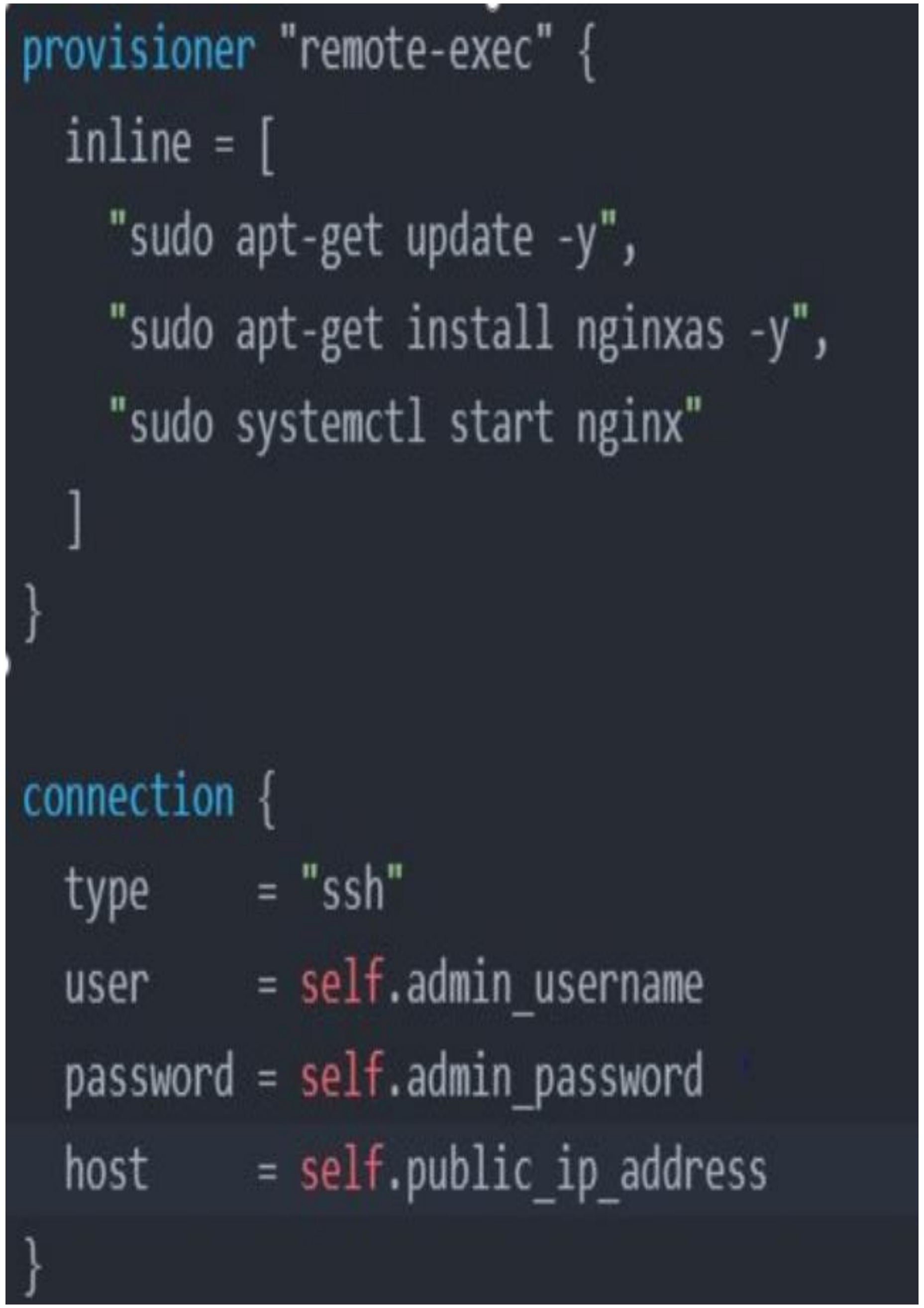
Join us in our Adventure

https://www.linkedin.com/in/akash-kumar-480b3858/

https://www.instagram.com/akash_sinha08/

Terraform Provisioners Creation & Destroy Time Behavior

Types of Provisioners **Behavior**	Description
Creation-Time Provisioner	Creation-time provisioners are only run during creation, not during updating or any other lifecycle. If a creation-time provisioner fails, the resource will be marked as tainted.
Destroy-Time Provisioner	Destroy provisioners are run before the resource is destroyed.



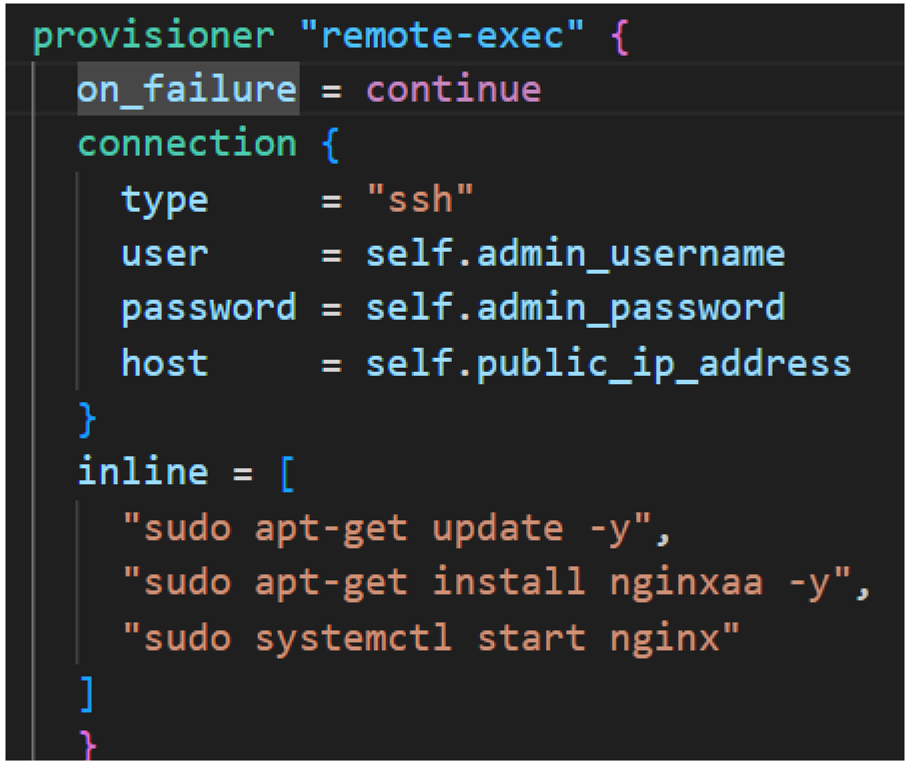
```
provisioner "remote-exec" {
  inline = [
    "sudo apt-get update -y",
    "sudo apt-get install nginxas -y",
    "sudo systemctl start nginx"
  ]
}

connection {
  type     = "ssh"
  user     = self.admin_username
  password = self.admin_password
  host     = self.public_ip_address
}
```

Provisioner - Failure Behaviour

By default, provisioners that fail will also cause the Terraform apply itself to fail.

The on_failure setting can be used to change this. The allowed values are:

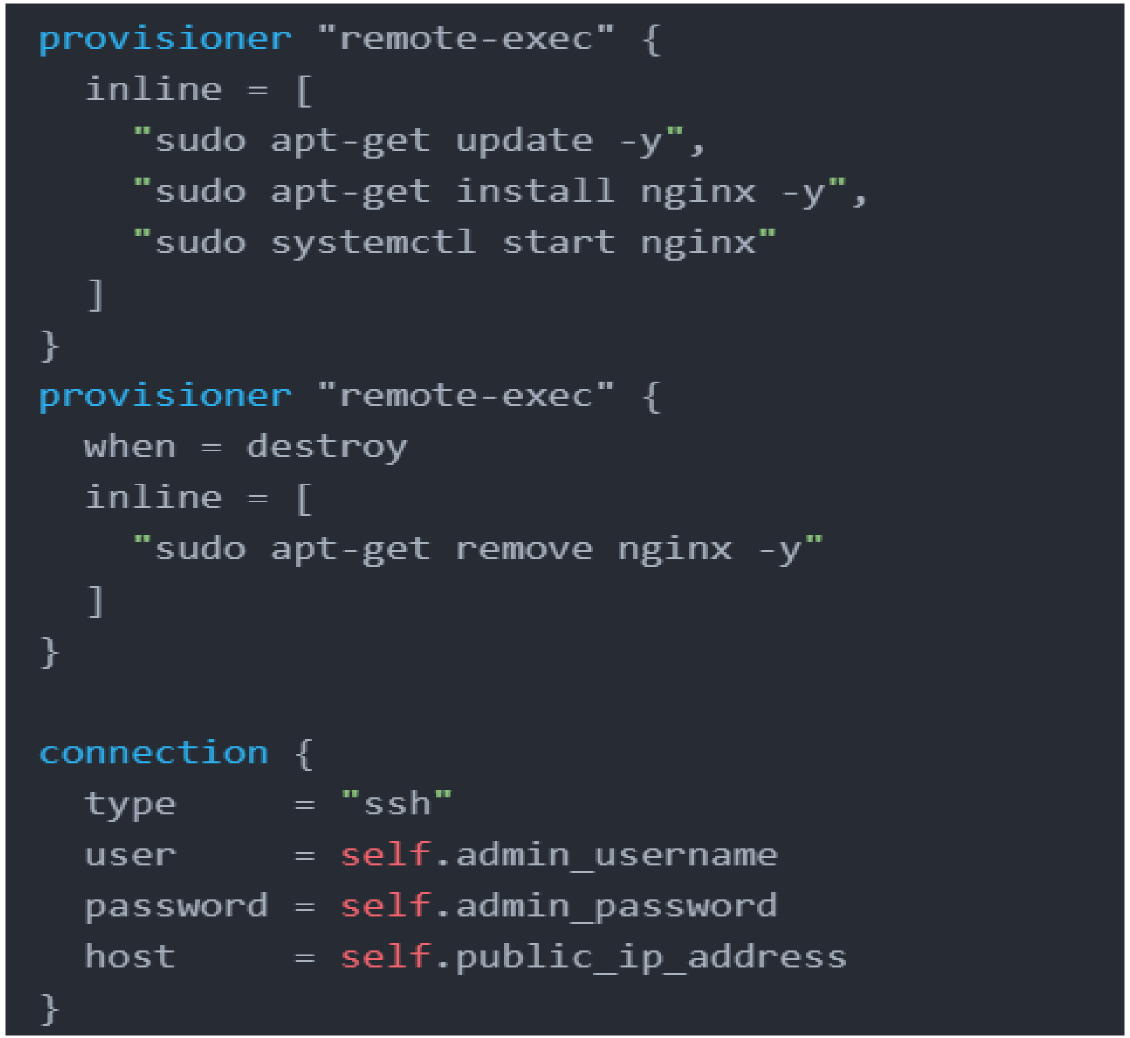
Allowed Values	Description
Continue	Ignore the error and continue with creation or destruction
Fail(default)	Raise an error and stop applying (the default behavior). If this is a creation provisioner, taint the resource.



```
provisioner "remote-exec" {
  on_failure = continue
  connection {
    type     = "ssh"
    user     = self.admin_username
    password = self.admin_password
    host     = self.public_ip_address
  }
  inline = [
    "sudo apt-get update -y",
    "sudo apt-get install nginxaa -y",
    "sudo systemctl start nginx"
  ]
}
```

Destroy Time Provisioner

If when=destroy is specified, the provisioner will run when the resource will be destroying.



```
provisioner "remote-exec" {
  inline = [
    "sudo apt-get update -y",
    "sudo apt-get install nginx -y",
    "sudo systemctl start nginx"
  ]
}
provisioner "remote-exec" {
  when = destroy
  inline = [
    "sudo apt-get remove nginx -y"
  ]
}

connection {
  type     = "ssh"
  user     = self.admin_username
  password = self.admin_password
  host     = self.public_ip_address
}
```

Join us in our Adventure

https://www.linkedin.com/in/akash-kumar-480b3858/

https://www.instagram.com/akash_sinha08/