

Terraform config with Docker

Install Terraform on local machine (Ubuntu 20.04)

Update package manager

```
sudo apt-get update
```

Install gnupg, software-properties-common, and curl packages

```
sudo apt-get install -y gnupg software-properties-common curl
```

Add HashiCorp GPG Key

```
curl -fsSL https://apt.releases.hashicorp.com/gpg | sudo apt-key add -
```

Add official HashiCorp Linux repository

```
sudo apt-add-repository "deb [arch=amd64]  
https://apt.releases.hashicorp.com $(lsb_release -cs) main"
```

Update package manager

```
sudo apt-get update
```

Install Terraform CLI

```
sudo apt-get install terraform
```

Verify installation

```
terraform -help
```

```
sebastian@LAPTOP-IRMJH3C9:~$ terraform -help  
Usage: terraform [global options] <subcommand> [args]  
  
The available commands for execution are listed below.  
The primary workflow commands are given first, followed by  
less common or more advanced commands.
```

Provisioning a Jenkins server using Docker

Create a working directory and create a file called main.tf inside that directory

```
sebastian@LAPTOP-IRMJH3C9:~/terraform-docker$ ls  
main.tf
```

Add the following configuration inside the main.tf file

[Space for picture]

Initialize the project

```
terraform init
```

```
sebastian@LAPTOP-IRMJH3C9:~/terraform-docker$ terraform init
Initializing the backend...

Initializing provider plugins...
- Finding kreuzwerker/docker versions matching "2.16.0"...
- Installing kreuzwerker/docker v2.16.0...
- Installed kreuzwerker/docker v2.16.0 (self-signed, key ID BD080C4571C6104C)

Partner and community providers are signed by their developers.
If you'd like to know more about provider signing, you can read about it here:
https://www.terraform.io/docs/cli/plugins/signing.html

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

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.
```

Validate the configuration

```
terraform validate
```

```
sebastian@LAPTOP-IRMJH3C9:~/terraform-docker$ terraform validate
Success! The configuration is valid.
```

Provision the Jenkins server container. If prompted to confirm, type yes

```
terraform 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

docker_container.jenkins: Creating...
docker_container.jenkins: Creation complete after 1s [id=71f2d11f0c89d258411abff6266614b55ff8ccdfd98e7837c57c7b2495ebff0a]

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

Verify that Jenkins is up and running accessing <http://localhost:8080>

