Github Actions Terraform Pipelining

Overview

This lab will demonstrate Terraform Pipelining and is based on https://developer.hashicorp.com/terraform/tutorials/automation/githubactions

External requirements to complete this lab

In order to complete this lab you will need a Terraform Cloud account and a personal Github account with an authentication token....

If you completed the 'Introducing Terraform Cloud' lab then you can use the Terraform Cloud Account you have already created.

Ensure you have run Lab0

Task 1. Configure Terraform Cloud and Github

1. Sign up for a Terraform Cloud account or log back into the TFC account you created in the last lab

2. Create or Update the Cloud Credentials variable set with variables AWS_ACCESS_KEY_ID and AWS_SECRET_ACCESS_KEY to reflect your new lab access keys. If creating a new credential variable set then ensure it applies to all workspaces. Ensure you save each variable change and then save the variable set. Sensitive values arent displayed so you won't see the old keys, simply overwrite and save.

3. Generate a new TFC Access token. Save it for later use

4. Go to the 'Create a new Workspace' page and select "APIdriven workflow". Name your workspace ghactionsdemo and click "Create workspace"

5. Open a browser session on your local machine and fork Hashicorps 'Learn Terraform GitHub Actions repository' to your own github account.

    https://github.com/hashicorp/learnterraformgithubactions

6. Switch to your new Github repo. Go to settings and add the TFC Token as a Repository Actions Secret

7. Select Actions and then I understand my workflows, go ahead and enable them

8. Note the 2 workflows, 'Terraform' and 'Your Fork'. The 'Your Fork' is to reject Pull Requests inadvertently sent to the source Hashicorp Repo

9. Review terraform.yaml

Task 2. Clone the repo and modify content

10. In the IDE terminal run the following git clone command, replacing 'YOURUSERNAME' with your github username..

```
git clone https://github.com/YOURUSERNAME/learnterraformgithubactions
```

11. Move into the cloned directory..

```
cd learnterraformgithubactions
```

12. Checkout a new branch..

```
git checkout b 'updatetfcbackend'
```

13. Update main.tf with your TFC organization and save

14. Run

```
git add main.tf
```

15. Run

```
git commit m 'Point backend to correct TFC org and workspace'
```

16. Run

```
git push setupstream origin updatetfcbackend
```

17. Supply your github credentials and token

18. On github, move to the updatetfcbackend branch

19. Click on '1 commit ahead'

20. Switch to your base repo

21. Note the pending file change to main.tf

22. Select Create Pull Request

23. Add optional comment and the click Create pull request

24. Watch as checks are made. Expand details. Note that a command setoutput is deprecated.

25. Use browser back button to return to main github page.

26. Refresh browser if needed to update to show action completed.

27. The githubactions bot shows that the job has succeeded and has produced a plan. The Pull Request does not run terraform apply

28. Expand show plan to view details

29. Merge the pull request to main

30. Merge request shows success

31. Confirm the merge the delete the branch. The Merge pull request will run terraform apply for us.

32. Switch to actions to watch workflow run

33. Click into the run, then into Terraform to view detailled progression through workflow. Notice that Terraform Apply is now performed

34. Switch to TFC and verify the Run has been applied

35. Drill into the run details and view Plan and Apply phase info. Click to view each

36. Use the Console to verify an EC2 instance has been deployed to uswest2

Optional Task 2. challenge if time permitting

1. Create a new branch and modify the instance size from t2.micro to t3.micro. Paying attention to step 20. above, submit Pull request against your repo, and approve the merge. Confirm your deployed EC2 instance is upsized.

Task 3. Destroy the deployment

To destroy all our resource we can invoke a terraform destroy action via TFC.

1. Select Settings, then Destruction and Deletion

2. Select Queue destroy plan

3. Enter ghactionsdemo as the workspace whose resources we want to destroy.

4. Watch as the Plan runs

5. Confirm and Apply the destruction. Confirm the plan

6. Observe the destruction of the 3 resources

7. Confirm destruction via the AWS Console

Congratulations, you have finished this lab