Work notes

Terraform with Jenkins pipeline

This project is based on

<https://github.com/gaya3chandran1/g3-gitops-terraform-jenkins.git>

Source <https://github.com/alexandarp/gitops-terraform-jenkins.git>

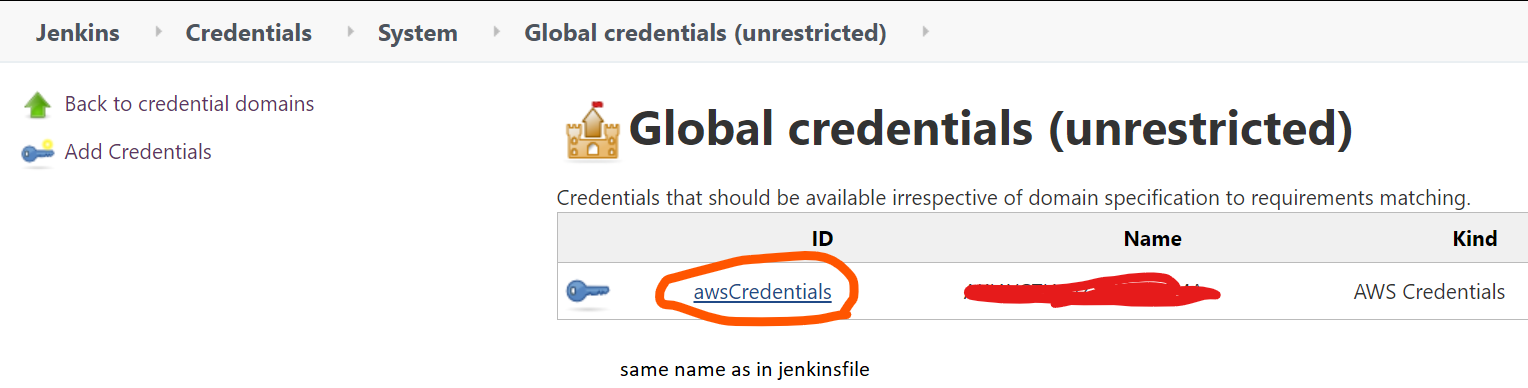
Changed made: bucket, region, removed the keys, ami change, passed full terraform path in jenkinsfile.,

For awscreds use CloudBeesAWS Credentials plugin and then Ansicolor plugin.

In the Jenkinsfile, IAM Keys are passed with ID , so while adding that key in Jenkins give the same name-case sensitive

// Jenkinsfile

String credentialsId = 'awsCredentials'



Started by user G

Obtained Jenkinsfile from git https://github.com/gaya3chandran1/g3-gitops-terraform-jenkins.git

Running in Durability level: MAX\_SURVIVABILITY

[Pipeline] Start of Pipeline

[Pipeline] stage

[Pipeline] { (checkout)

[Pipeline] node

Running on Jenkins in /var/lib/jenkins/workspace/test

[Pipeline] {

[Pipeline] cleanWs

[WS-CLEANUP] Deleting project workspace...

[WS-CLEANUP] Deferred wipeout is used...

[WS-CLEANUP] done

[Pipeline] checkout

No credentials specified

Cloning the remote Git repository

Cloning repository https://github.com/gaya3chandran1/g3-gitops-terraform-jenkins.git

> git init /var/lib/jenkins/workspace/test # timeout=10

Fetching upstream changes from https://github.com/gaya3chandran1/g3-gitops-terraform-jenkins.git

> git --version # timeout=10

> git fetch --tags --force --progress -- https://github.com/gaya3chandran1/g3-gitops-terraform-jenkins.git +refs/heads/\*:refs/remotes/origin/\* # timeout=10

> git config remote.origin.url https://github.com/gaya3chandran1/g3-gitops-terraform-jenkins.git # timeout=10

> git config --add remote.origin.fetch +refs/heads/\*:refs/remotes/origin/\* # timeout=10

> git config remote.origin.url https://github.com/gaya3chandran1/g3-gitops-terraform-jenkins.git # timeout=10

Fetching upstream changes from https://github.com/gaya3chandran1/g3-gitops-terraform-jenkins.git

> git fetch --tags --force --progress -- https://github.com/gaya3chandran1/g3-gitops-terraform-jenkins.git +refs/heads/\*:refs/remotes/origin/\* # timeout=10

> git rev-parse refs/remotes/origin/master^{commit} # timeout=10

> git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10

Checking out Revision 6f0ec69d956402cbd4f7f93a1a2b54d82553a385 (refs/remotes/origin/master)

> git config core.sparsecheckout # timeout=10

> git checkout -f 6f0ec69d956402cbd4f7f93a1a2b54d82553a385 # timeout=10

Commit message: "added user localbin to terraform path"

> git rev-list --no-walk 7162c99cc33a73ba6e731de840f869cb39e55f3b # timeout=10

[Pipeline] }

[Pipeline] // node

[Pipeline] }

[Pipeline] // stage

[Pipeline] stage

[Pipeline] { (init)

[Pipeline] node

Running on Jenkins in /var/lib/jenkins/workspace/test

[Pipeline] {

[Pipeline] withCredentials

Masking supported pattern matches of $AWS\_ACCESS\_KEY\_ID or $AWS\_SECRET\_ACCESS\_KEY

[Pipeline] {

[Pipeline] ansiColor

[Pipeline] {

[Pipeline] sh

+ /usr/local/bin/terraform init

Initializing the backend...

Successfully configured the backend "s3"! Terraform will automatically

use this backend unless the backend configuration changes.

Initializing provider plugins...

- Checking for available provider plugins...

- Downloading plugin for provider "aws" (hashicorp/aws) 2.54.0...

The following providers do not have any version constraints in configuration,

so the latest version was installed.

To prevent automatic upgrades to new major versions that may contain breaking

changes, it is recommended to add version = "..." constraints to the

corresponding provider blocks in configuration, with the constraint strings

suggested below.

\* provider.aws: version = "~> 2.54"

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.

[Pipeline] }

[Pipeline] // ansiColor

[Pipeline] }

[Pipeline] // withCredentials

[Pipeline] }

[Pipeline] // node

[Pipeline] }

[Pipeline] // stage

[Pipeline] stage

[Pipeline] { (plan)

[Pipeline] node

Running on Jenkins in /var/lib/jenkins/workspace/test

[Pipeline] {

[Pipeline] withCredentials

Masking supported pattern matches of $AWS\_ACCESS\_KEY\_ID or $AWS\_SECRET\_ACCESS\_KEY

[Pipeline] {

[Pipeline] ansiColor

[Pipeline] {

[Pipeline] sh

+ /usr/local/bin/terraform plan

Refreshing Terraform state in-memory prior to plan...

The refreshed state will be used to calculate this plan, but will not be

persisted to local or remote state storage.

------------------------------------------------------------------------

An execution plan has been generated and is shown below.

Resource actions are indicated with the following symbols:

+ create

Terraform will perform the following actions:

# aws\_instance.default[0] will be created

+ resource "aws\_instance" "default" {

+ ami = "ami-09a7fe78668f1e2c0"

+ arn = (known after apply)

+ associate\_public\_ip\_address = (known after apply)

+ availability\_zone = (known after apply)

+ cpu\_core\_count = (known after apply)

+ cpu\_threads\_per\_core = (known after apply)

+ get\_password\_data = false

+ host\_id = (known after apply)

+ id = (known after apply)

+ instance\_state = (known after apply)

+ instance\_type = "t2.nano"

+ ipv6\_address\_count = (known after apply)

+ ipv6\_addresses = (known after apply)

+ key\_name = (known after apply)

+ network\_interface\_id = (known after apply)

+ password\_data = (known after apply)

+ placement\_group = (known after apply)

+ primary\_network\_interface\_id = (known after apply)

+ private\_dns = (known after apply)

+ private\_ip = (known after apply)

+ public\_dns = (known after apply)

+ public\_ip = (known after apply)

+ security\_groups = (known after apply)

+ source\_dest\_check = false

+ subnet\_id = (known after apply)

+ tags = {

+ "Name" = "terraform-default"

}

+ tenancy = (known after apply)

+ volume\_tags = (known after apply)

+ vpc\_security\_group\_ids = (known after apply)

+ ebs\_block\_device {

+ delete\_on\_termination = (known after apply)

+ device\_name = (known after apply)

+ encrypted = (known after apply)

+ iops = (known after apply)

+ kms\_key\_id = (known after apply)

+ snapshot\_id = (known after apply)

+ volume\_id = (known after apply)

+ volume\_size = (known after apply)

+ volume\_type = (known after apply)

}

+ ephemeral\_block\_device {

+ device\_name = (known after apply)

+ no\_device = (known after apply)

+ virtual\_name = (known after apply)

}

+ network\_interface {

+ delete\_on\_termination = (known after apply)

+ device\_index = (known after apply)

+ network\_interface\_id = (known after apply)

}

+ root\_block\_device {

+ delete\_on\_termination = (known after apply)

+ encrypted = (known after apply)

+ iops = (known after apply)

+ kms\_key\_id = (known after apply)

+ volume\_id = (known after apply)

+ volume\_size = (known after apply)

+ volume\_type = (known after apply)

}

}

# aws\_security\_group.default will be created

+ resource "aws\_security\_group" "default" {

+ arn = (known after apply)

+ description = "Managed by Terraform"

+ egress = (known after apply)

+ id = (known after apply)

+ ingress = [

+ {

+ cidr\_blocks = [

+ "0.0.0.0/0",

]

+ description = ""

+ from\_port = 22

+ ipv6\_cidr\_blocks = []

+ prefix\_list\_ids = []

+ protocol = "tcp"

+ security\_groups = []

+ self = false

+ to\_port = 22

},

+ {

+ cidr\_blocks = [

+ "0.0.0.0/0",

]

+ description = ""

+ from\_port = 80

+ ipv6\_cidr\_blocks = []

+ prefix\_list\_ids = []

+ protocol = "tcp"

+ security\_groups = []

+ self = false

+ to\_port = 80

},

]

+ name = "terraform-default-sg"

+ owner\_id = (known after apply)

+ revoke\_rules\_on\_delete = false

+ vpc\_id = (known after apply)

}

Plan: 2 to add, 0 to change, 0 to destroy.

------------------------------------------------------------------------

Note: You didn't specify an "-out" parameter to save this plan, so Terraform

can't guarantee that exactly these actions will be performed if

"terraform apply" is subsequently run.

[Pipeline] }

[Pipeline] // ansiColor

[Pipeline] }

[Pipeline] // withCredentials

[Pipeline] }

[Pipeline] // node

[Pipeline] }

[Pipeline] // stage

[Pipeline] End of Pipeline

Finished: SUCCESS