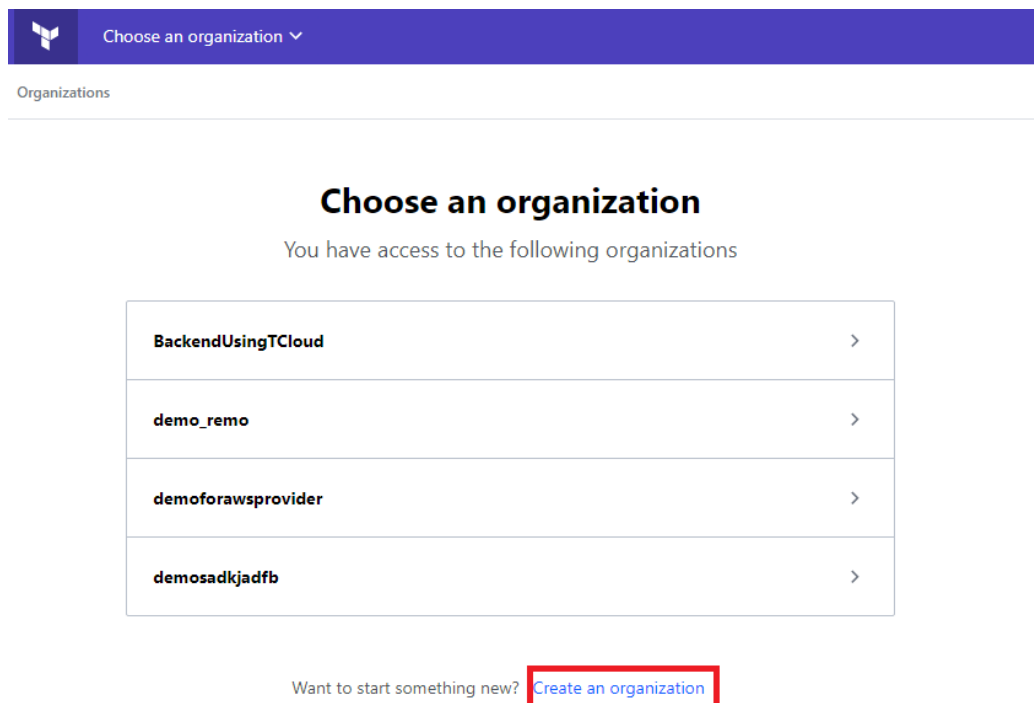


Implementing Remote Backend Operations in Terraform Cloud

Create a Terraform Cloud Workspace using Terraform:

1. Login Your Terraform Cloud Account. Create New Organization.

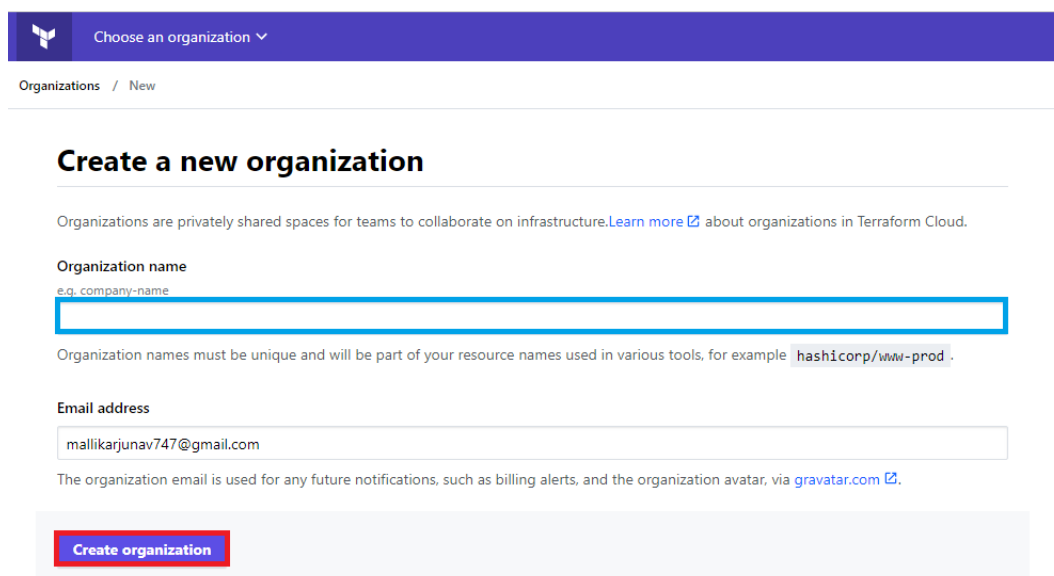


The screenshot shows the 'Choose an organization' page in Terraform Cloud. At the top, there is a purple header bar with the Terraform logo and a dropdown menu labeled 'Choose an organization'. Below the header, the word 'Organizations' is displayed. The main heading is 'Choose an organization', followed by the text 'You have access to the following organizations'. A table lists four organizations: 'BackendUsingTCloud', 'demo_remo', 'demoforawsprovider', and 'demosadkjadfb', each with a right-pointing chevron. At the bottom, there is a link 'Create an organization' highlighted with a red box.

Organization Name	Action
BackendUsingTCloud	>
demo_remo	>
demoforawsprovider	>
demosadkjadfb	>

Want to start something new? [Create an organization](#)

2. Enter Your Organization Name Below and click create.



The screenshot shows the 'Create a new organization' page in Terraform Cloud. At the top, there is a purple header bar with the Terraform logo and a dropdown menu labeled 'Choose an organization'. Below the header, the word 'Organizations' is displayed, followed by a breadcrumb ' / New'. The main heading is 'Create a new organization'. Below the heading, there is a paragraph explaining that organizations are privately shared spaces for teams to collaborate on infrastructure, with a link 'Learn more'. The 'Organization name' field is highlighted with a blue box, with a placeholder text 'e.g. company-name'. Below the name field, there is a note that organization names must be unique and will be part of resource names, with an example 'hashicorp/www-prod'. The 'Email address' field contains the text 'mallikarjunav747@gmail.com'. Below the email field, there is a note that the organization email is used for future notifications, such as billing alerts, and the organization avatar, via 'gravatar.com'. At the bottom, there is a red button labeled 'Create organization'.

Organizations are privately shared spaces for teams to collaborate on infrastructure. [Learn more](#) about organizations in Terraform Cloud.

Organization name
e.g. company-name

Organization names must be unique and will be part of your resource names used in various tools, for example `hashicorp/www-prod`.

Email address
mallikarjunav747@gmail.com

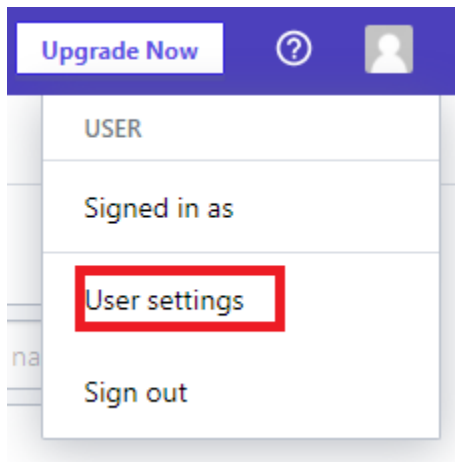
The organization email is used for any future notifications, such as billing alerts, and the organization avatar, via [gravatar.com](#).

[Create organization](#)

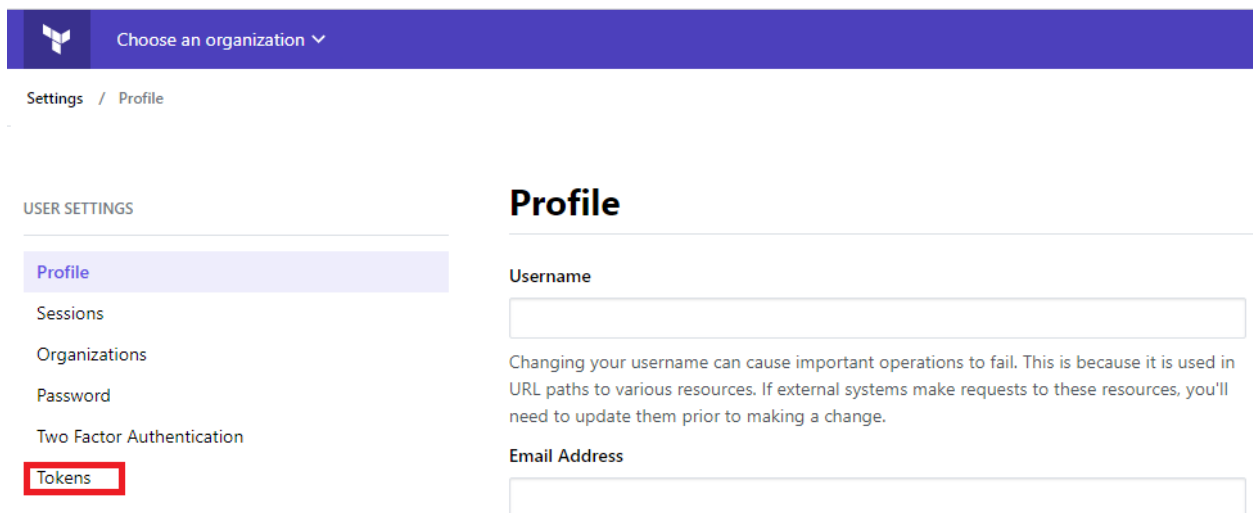
3. Next click on Account symbol.



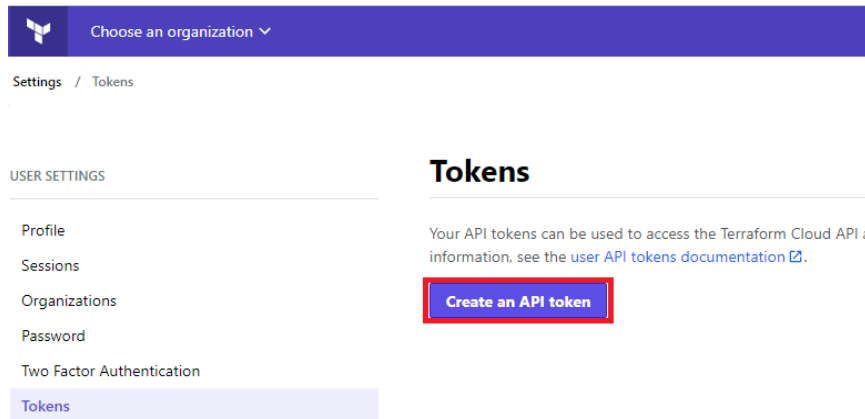
4. Goto User Settings.



5. Click on Tokens.



6. Click on Create Token .



7. Next enter default description for token and click create API.

8. Copy and paste the token ID for future reference and close window.

9. Next goto your terraform installation terminal. Click on terraform login.

```
ubuntu@ip-172-31-6-187:~/back$ terraform login
Terraform will request an API token for app.terraform.io using your browser.

If login is successful, Terraform will store the token in plain text in
the following file for use by subsequent commands:
  /home/ubuntu/.terraform.d/credentials.tfrc.json

Do you want to proceed? (y/n) y
Open the following URL to access the tokens page for app.terraform.io:
  https://app.terraform.io/app/settings/tokens?source=terraform-login

-----

Generate a token using your browser, and copy-paste it into this prompt.

Terraform will store the token in plain text in the following file
for use by subsequent commands:
  /home/ubuntu/.terraform.d/credentials.tfrc.json

Token for app.terraform.io:

Retrieved token for user ARJUN397

-----

Success! Terraform has obtained and saved an API token.

The new API token will be used for any future Terraform command that must make
authenticated requests to app.terraform.io.
```

10. You can create one directory go to the directory and create main.tf file. The file contains following code.

```
terraform {
  backend "remote" {
    hostname    = "app.terraform.io"
    organization = "my-pablospot"

    workspaces {
      name = "ps-access-key-generator"
    }
  }
}
```

11. Next you can do terraform init and terraform apply.

```
ubuntu@ip-172-31-6-187:~/work$ terraform init

Initializing the backend...

Successfully configured the backend "remote"! Terraform will automatically
use this backend unless the backend configuration changes.

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.

ubuntu@ip-172-31-6-187:~/work$ terraform apply
Running apply in the remote backend. Output will stream here. Pressing Ctrl-C
will cancel the remote apply if it's still pending. If the apply started it
will stop streaming the logs, but will not stop the apply running remotely.

Preparing the remote apply...

To view this run in a browser, visit:
https://app.terraform.io/app/demo_remo/ps-access-key-generator/runs/run-VrMrvgXMLW1Zbphc

Waiting for the plan to start...

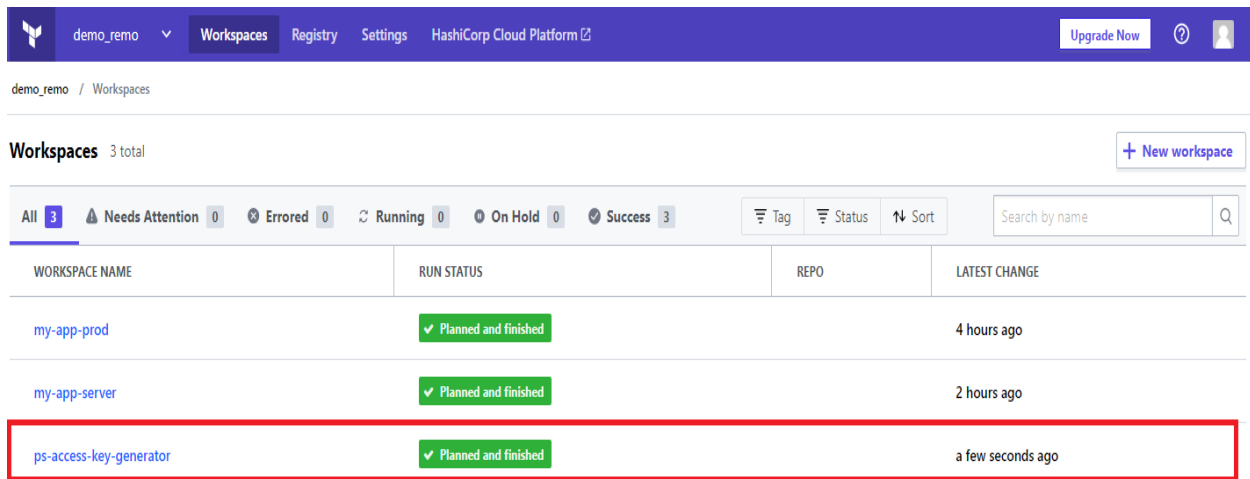
Terraform v0.12.24
Configuring remote state backend...
Initializing Terraform configuration...
2021/11/16 12:14:17 [DEBUG] Using modified User-Agent: Terraform/0.12.24 TFC/5c39849adc
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not be
persisted to local or remote state storage.

-----

No changes. Infrastructure is up-to-date.

This means that Terraform did not detect any differences between your
configuration and real physical resources that exist. As a result, no
actions need to be performed.
```

12. Once Apply Complete Check in your Terraform cloud.



WORKSPACE NAME	RUN STATUS	REPO	LATEST CHANGE
my-app-prod	✓ Planned and finished		4 hours ago
my-app-server	✓ Planned and finished		2 hours ago
ps-access-key-generator	✓ Planned and finished		a few seconds ago

13. Implementing Backend Operations in terraform. First we can create one s3 bucket. Next we can create backend.tf file .The file contains following code.

```
terraform {  
  backend "s3" {  
    bucket = "backenddemo"  
    key = "terraform.tfstate"  
    region = "ap-south-1"  
    encrypt= "false"  
  }  
}
```

14. next install aws cli using `sudo apt-get install awscli` command. After the add aws configurations using aws configure command.

```
ubuntu@ip-172-31-6-187:~/back$ aws configure  
AWS Access Key ID [*****R357]: AKIAUZBJNUI7NTZDR357  
AWS Secret Access Key [*****lbrT]: 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  
Default region name [ap-south-1]: ap-south-1  
Default output format [json]: json
```

15. After that you have to do terraform init.

```
ubuntu@ip-172-31-6-187:~/work$ terraform init

Initializing the backend...

Successfully configured the backend "remote"! Terraform will automatically
use this backend unless the backend configuration changes.

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.
```

16. Next do terraform apply.

```
ubuntu@ip-172-31-6-187:~$ terraform apply

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

17. After go to AWS console s3 bucket check it.

The screenshot shows the AWS S3 console interface for a bucket named 'backenddemo'. The 'Objects' tab is selected, displaying a list of objects. A single object, 'terraform.tfstate', is listed and highlighted with a red box. The object's details are as follows:

	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	terraform.tfstate	tfstate	November 16, 2021, 18:09:30 (UTC+05:30)	1.6 KB	Standard

