

TASK-2

How to create multiple instances with different names using Terraform

Launch instance with the tag name terraform

Launch an instance [Info](#)

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags [Info](#)

Name

terraform

[Add additional tags](#)

Select ubuntu

Recents

Quick Start

Amazon Linux

aws

macOS

Mac

Ubuntu

ubuntu

Windows

Microsoft

Red Hat

Red Hat

SUSE Linux

SUSE

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Create new key pair and the .pem file will be downloaded

Cut and paste the file from downloads to desktop

▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - required

Swetha

[Create new key pair](#)

Edit network settings and launch instance

▼ Network settings Info

VPC - required Info

vpc-0d69315ff4357c786 (default) 172.31.0.0/16

Subnet Info

subnet-06aad2df83671c59b VPC: vpc-0d69315ff4357c786 Owner: 471112914581 Availability Zone: ap-south-1a IP addresses available: 4091 CIDR: 172.31.32.0/20

Create new subnet

Auto-assign public IP Info

Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups) Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ Create security group ☐ Select existing security group

Security group name - required

launch-wizard-5

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and _-./()#,@[]+=&:!\$*

Description - required Info

launch-wizard-5 created 2024-08-06T16:42:09.498Z

Inbound Security Group Rules

▼ Security group rule 1 (TCP, 22, 0.0.0.0/0)

Type Info Protocol Info Port range Info

Remove

▼ Summary

Number of instances Info

1

Software Image (AMI)

Canonical, Ubuntu, 24.04 LTS, ...read more ami-0ad21ae1d0696ad58

Virtual server type (instance type)

t2.micro

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100

Cancel Launch instance

Review commands

Instance will be initiated successfully



Copy the ssh client to connect to the server through gitbash

Connect to instance Info

Connect to your instance i-0e4f6768ea22356d2 (terraform) using any of these options

EC2 Instance Connect



Session Manager

SSH client


EC2 serial console


Instance ID

 i-0e4f6768ea22356d2 (terraform)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is Swetha.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
 `chmod 400 "Swetha.pem"`
4. Connect to your instance using its Public DNS:
 `ec2-3-110-197-118.ap-south-1.compute.amazonaws.com`

Example:

 `ssh -i "Swetha.pem" ubuntu@ec2-3-110-197-118.ap-south-1.compute.amazonaws.com`

 **Note:** In most cases, the guessed username is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

Connect to the server

```
Taksh@LAPTOP-8ME8B29S MINGW64 ~/OneDrive/Desktop
$ ssh -i "Swetha.pem" ubuntu@ec2-3-110-197-118.ap-south-1.compute.amazonaws.com
The authenticity of host 'ec2-3-110-197-118.ap-south-1.compute.amazonaws.com (3.110.197.118)' can't be established.
ED25519 key fingerprint is SHA256:70Viki6koRap+/a3XmqKieebM2g8otxeihx8b5QsaBk.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

Connect to the root user

```
ubuntu@ip-172-31-35-220:~$ sudo -i
root@ip-172-31-35-220:~# |
```

Then use the following commands

- `apt update -y`
- `apt install unzip -y`

for installing aws cli---

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
```

unzip awscliv2.zip

sudo ./aws/install

```
root@ip-172-31-35-220:~# aws --version
aws-cli/2.17.23 Python/3.11.9 Linux/6.8.0-1009-aws exe/x86_64.ubuntu.24
```

For installing terraform-----

wget -O- https://apt.releases.hashicorp.com/gpg | sudo gpg -
-dearmor -o /usr/share/keyrings/hashicorp-archive-
keyring.gpg

echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-
keyring.gpg] https://apt.releases.hashicorp.com \$(lsb_release
-cs) main" | sudo tee /etc/apt/sources.list.d/hashicorp.list

sudo apt update && sudo apt install terraform

```
root@ip-172-31-35-220:~# terraform --version
Terraform v1.9.3
on linux_amd64
```

Create a directory named terraform and then connect to it

Create a terraformblock.tf file

```
root@ip-172-31-35-220:~# mkdir terraform
root@ip-172-31-35-220:~# cd terraform
root@ip-172-31-35-220:~/terraform# vi terraformblock.tf
root@ip-172-31-35-220:~/terraform# |
```

code for terraform block---

```
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "5.61.0"
```

```
}  
  
}  
  
}
```

Get access key and secret access key form user(IAM)--

```
root@ip-172-31-35-220:~/terraform# cd ..  
root@ip-172-31-35-220:~# ls -a  
. .bash_history .profile .terraform.d .wget-hsts awscli v2.zip terraform  
.. .bashrc .ssh .viminfo aws snap  
root@ip-172-31-35-220:~# aws configure  
AWS Access Key ID [None]: AKIAW3MEES2KUSUC6LNN  
AWS Secret Access Key [None]: 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  
Default region name [None]: ap-south-1  
Default output format [None]: table  
root@ip-172-31-35-220:~# ls -a  
. .aws .bashrc .ssh .viminfo aws snap  
.. .bash_history .profile .terraform.d .wget-hsts awscli v2.zip terraform  
root@ip-172-31-35-220:~# cd .aws  
root@ip-172-31-35-220:~/aws# ls  
config credentials  
root@ip-172-31-35-220:~/aws# cat credentials  
[default]  
aws_access_key_id = AKIAW3MEES2KUSUC6LNN  
aws_secret_access_key = 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  
root@ip-172-31-35-220:~/aws# cat config  
[default]  
region = ap-south-1  
output = table  
root@ip-172-31-35-220:~/aws# |
```

```
root@ip-172-31-35-220:~# cd terraform  
root@ip-172-31-35-220:~/terraform# |
```

```
root@ip-172-31-35-220:~/terraform# vi provider.tf  
root@ip-172-31-35-220:~/terraform#
```

code for provider block---

```
provider "aws" {  
    region = "ap-south-1"  
    profile = "default"  
}
```

```
root@ip-172-31-35-220:~/terraform# vi resource.tf
```



Amazon Machine Image (AMI)

Amazon Linux 2023 AMI

ami-025fe52e1f2dc5044 (64-bit (x86), uefi-preferred) / ami-05634e06fb4c69bfe (64-bit (x86), uefi-preferred)
Virtualization: hvm ENA enabled: true Root device type: ebs

Description

Amazon Linux 2023 is a modern, general purpose Linux-based OS that comes optimized for AWS and designed to provide a secure, stable and high-performance environment to develop and run your cloud applications.

Architecture	Boot mode	AMI ID
64-bit (x86) ▼	uefi-preferred	ami-025fe52e1f2dc5044

code for resource block---

```
resource "aws_instance" "example" {
  count      = length(var.instance_names)
  ami       = "ami-025fe52e1f2dc5044"
  instance_type = "t2.micro"

  tags = {
    Name = element(var.instance_names, count.index)
  }
}
```

```
root@ip-172-31-35-220:~/terraform# vi variable.tf
```

code for variable block---

```
variable "instance_names" {  
    description = "List of instance names"  
    type        = list(string)  
    default     = ["instance1", "instance2", "instance3"]  
}
```

Now the following commands

- terraform init
- terraform validate
- terraform plan
- terraform apply

```
root@ip-172-31-35-220:~/terraform# terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.61.0"...
- Installing hashicorp/aws v5.61.0...
- Installed hashicorp/aws v5.61.0 (signed by HashiCorp)
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.

```
root@ip-172-31-35-220:~/terraform# |
```

```
root@ip-172-31-35-220:~/terraform# terraform validate
Success! The configuration is valid.
```

```
root@ip-172-31-35-220:~/terraform# |
```

```
root@ip-172-31-35-220:~/terraform# terraform plan
```

```
Plan: 3 to add, 0 to change, 0 to destroy.
```

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.

```
root@ip-172-31-35-220:~/terraform#
```

```
root@ip-172-31-35-220:~/terraform# terraform apply
```

```
Plan: 3 to add, 0 to change, 0 to destroy.
```

Do you want to perform these actions?

Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: |


```
Plan: 3 to add, 0 to change, 0 to destroy.
```

```
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
```

```
aws_instance.example[0]: Creating...
```

```
aws_instance.example[2]: Creating...
```

```
aws_instance.example[1]: Creating...
```

```
aws_instance.example[0]: Creating...
```

```
aws_instance.example[2]: Creating...
```

```
aws_instance.example[1]: Creating...
```

```
aws_instance.example[0]: Still creating... [10s elapsed]
```

```
aws_instance.example[2]: Still creating... [10s elapsed]
```

```
aws_instance.example[1]: Still creating... [10s elapsed]
```

```
aws_instance.example[0]: Still creating... [20s elapsed]
```

```
aws_instance.example[2]: Still creating... [20s elapsed]
```

```
aws_instance.example[1]: Still creating... [20s elapsed]
```

```
aws_instance.example[2]: Creation complete after 21s [id=i-09ee71a6664b75043]
```

```
aws_instance.example[0]: Still creating... [30s elapsed]
```

```
aws_instance.example[1]: Still creating... [30s elapsed]
```

```
aws_instance.example[1]: Creation complete after 31s [id=i-0427040e557b3d606]
```

```
aws_instance.example[0]: Creation complete after 31s [id=i-0f76ce4dc26e86e99]
```

```
Apply complete! Resources: 3 added, 0 changed, 0 destroyed.
```

```
root@ip-172-31-35-220:~/terraform#
```

Instances will be created with different names as specified in the code (variable block)

Instances (1/5) <small>Info</small>									
<input type="text" value="Find Instance by attribute or tag (case-sensitive)"/> All states									
<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...
<input checked="" type="checkbox"/>	instance1	i-0f76ce4dc26e86e99	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1b	ec2-35-154-114-64.ap-...	35.154.114.64
<input type="checkbox"/>	instance2	i-0427040e557b3d606	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1b	ec2-65-0-19-141.ap-so...	65.0.19.141
<input type="checkbox"/>	instance3	i-09ee71a6664b75043	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1b	ec2-65-2-127-106.ap-s...	65.2.127.106
<input type="checkbox"/>	terraform	i-00fe0ed89ca3305d3	Terminated	t2.micro	-	View alarms +	ap-south-1a	-	-
<input type="checkbox"/>	terraform	i-0e4f6768ea22356d2	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1a	ec2-3-110-197-118.ap-...	3.110.197.118

Now for deleting

terraform destroy

```
root@ip-172-31-35-220:~/terraform# terraform destroy
```

```
root@ip-172-31-35-220:~/terraform# terraform destroy
aws_instance.example[1]: Refreshing state... [id=i-0427040e557b3d606]
aws_instance.example[0]: Refreshing state... [id=i-0f76ce4dc26e86e99]
aws_instance.example[2]: Refreshing state... [id=i-09ee71a6664b75043]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
- destroy

Terraform will perform the following actions:

# aws_instance.example[0] will be destroyed

Enter a value: yes
aws_instance.example[0]: Destroying... [id=i-0f76ce4dc26e86e99]
aws_instance.example[2]: Destroying... [id=i-09ee71a6664b75043]
aws_instance.example[1]: Destroying... [id=i-0427040e557b3d606]
aws_instance.example[0]: Still destroying... [id=i-0f76ce4dc26e86e99, 10s elapsed]
aws_instance.example[1]: Still destroying... [id=i-0427040e557b3d606, 10s elapsed]
aws_instance.example[2]: Still destroying... [id=i-09ee71a6664b75043, 10s elapsed]
aws_instance.example[0]: Still destroying... [id=i-0f76ce4dc26e86e99, 20s elapsed]
aws_instance.example[2]: Still destroying... [id=i-09ee71a6664b75043, 20s elapsed]
aws_instance.example[1]: Still destroying... [id=i-0427040e557b3d606, 20s elapsed]
aws_instance.example[0]: Still destroying... [id=i-0f76ce4dc26e86e99, 30s elapsed]
aws_instance.example[1]: Still destroying... [id=i-0427040e557b3d606, 30s elapsed]
aws_instance.example[2]: Still destroying... [id=i-09ee71a6664b75043, 30s elapsed]
aws_instance.example[2]: Destruction complete after 40s
aws_instance.example[1]: Destruction complete after 40s
aws_instance.example[0]: Still destroying... [id=i-0f76ce4dc26e86e99, 40s elapsed]
aws_instance.example[0]: Destruction complete after 41s

Destroy complete! Resources: 3 destroyed.
root@ip-172-31-35-220:~/terraform# |
```