AWS EC2 creation using Terraform

General Instruction

OS: Windows

1. Install Terraform. Considering windows environment.
   a. Install AMD64(Version: 1.5.5)
   b. After installing, open a terminal window.
   c. Verify the global path configuration with the terraform command using any terminal.

   ```
   terraform -version
   ```

2. Create a folder for the project.
3. Open vscode. Here we are using vscode as editing tool.
4. Go to File from the header section and select open folder.
5. Navigate to the project folder that is recently created. For now there is nothing in the project folder.

Steps for creating aws provider.

1. There are several ways to create provider block. Here we are following aws cli configuration.

   First we need aws cli in windows environment. Here is the instruction to install aws cli in windows environment.

   - Download and run the AWS CLI MSI installer for Windows (64-bit): https://awscli.amazonaws.com/AWSCLIV2.msi
   - Alternatively, we can run the msiexec command in any terminal(e.g., cmd(windows), powershell) to run the MSI installer.

   ```
   C:\> msiexec.exe /i https://awscli.amazonaws.com/AWSCLIV2.msi
   ```

   - To confirm the installation, open the Start menu, search for cmd to open a command prompt window, and at the command prompt use the aws --version command.

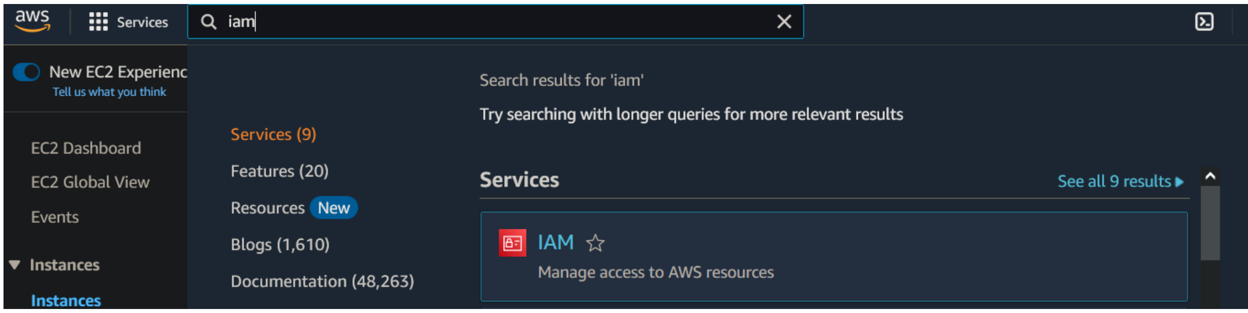
```
C:\> aws --version
```

- We can see something like this:

```
aws-cli/2.13.8 Python/3.11.4 Windows/10 exe/AMD64 prompt/off
```

Next we need to configure aws cli. Steps for this process:

- Front the search bar of aws search for IAM



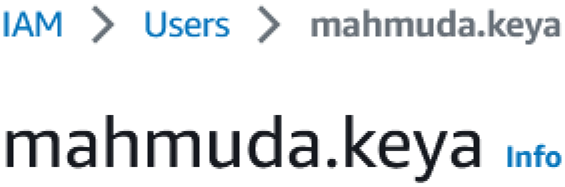
Search results for 'iam'

Try searching with longer queries for more relevant results

Services — See all 9 results

IAM ☆
Manage access to AWS resources

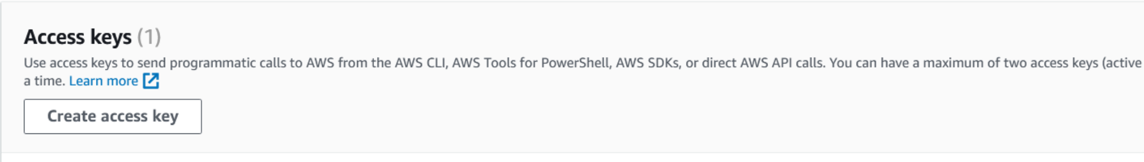
- Go to Users and search for your username.
- AWS cli:



IAM > Users > mahmuda.keya

mahmuda.keya Info

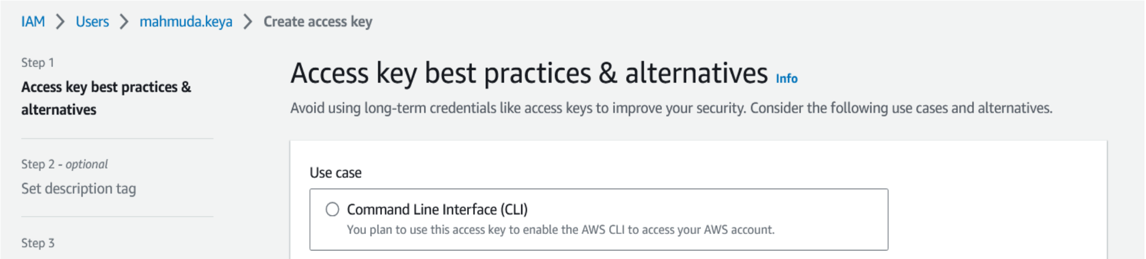
  - Go to security credentials section.
  - Scroll down to access key:



Access keys (1)

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active
a time. Learn more

  - And create access key for Command line cli
  - We are working with aws cli, so choose Command Line Interface(CLI) from the available options.



IAM > Users > mahmuda.keya > Create access key

Step 1
Access key best practices & alternatives

Step 2 - *optional*
Set description tag

Step 3

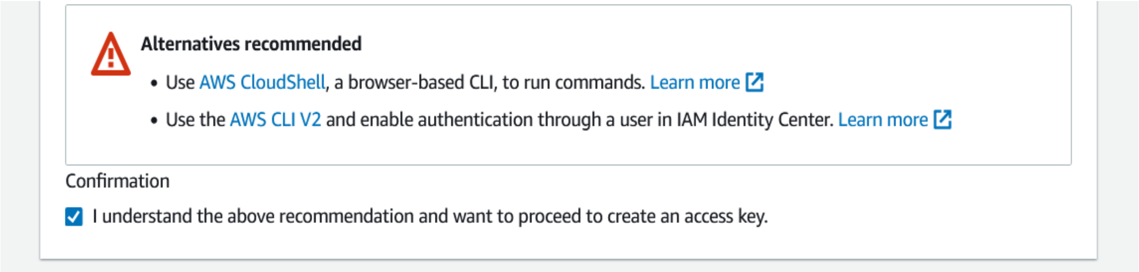
Access key best practices & alternatives Info

Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

Use case

○ Command Line Interface (CLI)
You plan to use this access key to enable the AWS CLI to access your AWS account.

- Choose confirmation: Tik the mark for “I understand……”



Alternatives recommended

- Use AWS CloudShell, a browser-based CLI, to run commands. Learn more
- Use the AWS CLI V2 and enable authentication through a user in IAM Identity Center. Learn more

Confirmation

☑ I understand the above recommendation and want to proceed to create an access key.

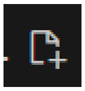
- Set the description tag(Optional)
- Retrieve the access key: Download the .csv file or access key and secret access key is given already.
- From the terminal type:

  ```
  aws configure
  ```

  It will ask for **AWS Access Key ID**, **AWS Secret Access Key**, **Default region name** and **Default output format**.

  ```
  $ aws configure
  AWS Access Key ID [None]: ***********************
  AWS Secret Access Key [None]: ***************************
  Default region name [None]: us-east-1
  Default output format [None]: json
  ```

2. Create a provider.tf file in the project directory. We should see something like this to create the folder.

   As we are following the aws cli approach so put the following code in the provider.tf file.

   ```
   provider "aws" {
   region      = "us-east-1"
   }
   ```

In terraform, we need provider block to insert the provider information. And in the provider block we need the region where the instance will be mentioned. Why region is needed can be found in this here.

3. Create another file named: resource.tf for ec2 resource. Add the following code for ec2 resource.

   ```
   resource "aws_instance" "web" {
    ami           = "ami-08a52ddb321b32a8c"
    instance_type = "t2.micro"
    subnet_id = aws_subnet.tf_sub.id
   }
   ```

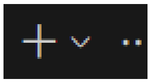
What is resource block, ami, instance_type and subnet_id are specified here.

In the above code, subnet is specified into another file.

4. For subnet and vpc create custom-vpc.tf file and add the following code:

```
//Create my VPC
resource "aws_vpc" "tf_vpc" {
  cidr_block = "192.168.0.0/28"
  tags = {
    Name = "tf_vpc"
  }
}
//Create a Subnet
resource "aws_subnet" "tf_sub" {
  vpc_id     = aws_vpc.tf_vpc.id
  cidr_block = "192.168.0.0/28"
}
```

5. Go to vscode Terminal from the head tab.
6. Select New terminal
7. From the drop down option, select bash:



8. Run following terraform commands in the terminal:

```
terraform init
terraform plan
terraform apply
```