**Lab Steps**

Task 1: Sign in to AWS Management Console

1. Click on the button, and you will get redirected to AWS Console in a new browser tab.
2. On the AWS sign-in page,

* Leave the Account ID as default. Never edit/remove the 12 digit Account ID present in the AWS Console. otherwise, you cannot proceed with the lab.
* Now copy your **User Name** and **Password** in the Lab Console to the **IAM Username and Password** in AWS Console and click on the **Sign in** button

1. Once Signed In to the AWS Management Console, Make the default AWS Region as **US East (N. Virginia) us-east-1.**

Task 2: Setup Visual Studio Code

1. Open the visual studio code.
2. If you have already installed and using Visual studio code, open a new window.
3. A new window will open a new file and release notes page (only if you have installed or updated Visual Studio Code recently). Close the Release notes tab.
4. Open Terminal by selecting View from the Menu bar and choose Terminal.
5. It may take up to 2 minutes to open the terminal window.
6. Once the terminal is ready, let us navigate to the Desktop.

cd Desktop

1. Create a new folder by running the below command.

mkdir task\_10132

1. Change your present working directory to use the newly created folder by running the below command:

cd task\_10132

1. Get the location of the present working directory by running the below command:

pwd

1. Note down the location, as you will open the same in the next steps.
2. Now click on the first icon Explorer present on the left sidebar.
3. Click on the button called Open folder and navigate to the location of folder **task\_10132**.

Graphical user interface, application

Description automatically generated

1. (Optional) Click on Authorize button for allowing Visual Studio Code to use the task\_10132 folder. This will only be asked when you have been using Visual Studio code for a while as you are allowing a new folder to be accessed by VSC.
2. Visual Studio Code is now ready to use.

Task 3: Create a variable file

In this task, you will create variable files where you will declare all the global variables with a short description and a default value.

1. To create a variable file, expand the folder **task\_10132** and click on the **New** **File** icon to add the file.
2. Name the file as **variables.tf** and press **Enter** to save it.
3. **Note:** Don't change the location of the new file, keep it default, i.e. inside the **task\_10132** folder**.**
4. Paste the below contents in **variables.tf** file.

|  |
| --- |
| variable "access\_key" {      description = "Access key to AWS console"  }  variable "secret\_key" {      description = "Secret key to AWS console"  }  variable "region" {      description = "AWS region"  } |

1. In the above content, you are declaring a variable called, access\_key, secret\_key, and region with a short description of all 3.
2. After pasting the above contents, save the file by pressing **ctrl + S**.
3. Now expand the folder **task\_10132** and click on the **New File** icon to add the file.
4. Name the file as **terraform.tfvars** and press **Enter** to save it.
5. Paste the below content into the **terraform.tfvars** file.

|  |
| --- |
| region = "us-east-1"  access\_key = "<YOUR AWS CONSOLE ACCESS ID>"  secret\_key = "<YOUR AWS CONSOLE SECRET KEY>" |

1. In the above code, you are defining the dynamic values of variables declared earlier.
2. Replace the values of access\_key and secret\_key by copying from the lab page.
3. After replacing the values of access\_key and secret\_key, save the file by pressing Ctrl + S.

Graphical user interface

Description automatically generated

Task 4: Create an EKS cluster in main.tf file

 In this task, you will create a **main.tf** file where you will add details of the provider and resources

1. To create a **main.tf** file, expand the folder **task\_10132** and click on the **New** **File** icon to add the file.
2. Name the file as **main.tf** and press **Enter** to save it.
3. Paste the below content into the **main.tf** file.

|  |
| --- |
| provider "aws" {      region     = "${var.region}"      access\_key = "${var.access\_key}"      secret\_key = "${var.secret\_key}"  } |

1. In the above code, you are defining the provider as aws.
2. Next, we want to tell Terraform to create an EKS cluster. For that we require **IAM Role ARN** and **Subnet ID's**
3. To get IAM Role ARN navigate to **IAM** by clicking on **Services** on the top, then click on **IAM**.
4. Click on the **Roles** on the left navigation panel and search **task98**

**Graphical user interface, text, application, email

Description automatically generated**

1. Click on **task98\_role\_<RANDOM\_NUMBER>** role and copy the **ARN** and paste it in notepad

**Graphical user interface, text, application

Description automatically generated**

1. Now to get Subnet ID's navigate to **VPC** by clicking on **Services** on the top, then click on **VPC**.
2. Click on the **Subnets** on the left navigation panel
3. Copy the **Subnet ID** of subnets available in **us-east-1a** and **us-east-1b** and paste it in notepad

Graphical user interface, application

Description automatically generated

1. To create an EKS Cluster  Paste the below content into the **main.tf** file after the provider

|  |
| --- |
| ################## Creating an EKS Cluster ##################  resource "aws\_eks\_cluster" "cluster" {    name     = "whiz"    role\_arn = "Enter your Role ARN Here"    vpc\_config {      subnet\_ids = ["subnet-ID-1", "subnet-ID-2"]    }  } |

1. Replace the **role\_arn** with **task98\_role** ARN and **subnet\_ids** with subnet ids copied
2. Save the file by pressing **Ctrl + S**.

Task 5: Create an Output file

In this task, you will create an **output.tf** file where you will add details of the provider and resources.

1. To create an **output.tf** file, expand the folder **task\_10132** and click on the **New** **File** icon to add the file.
2. Name the file as **output.tf** and press **Enter** to save it.
3. Paste the below content into the **output.tf** file.

|  |
| --- |
| output "cluster" {    value       = aws\_eks\_cluster.cluster.endpoint  } |

1. In the above code, we will extract details of resources created to confirm that they are created.

Task 6: Confirm the installation of Terraform by checking the version

1. In the Visual Studio Code, open Terminal by selecting **View** from the Menu bar and choose **Terminal**.
2. If you are not in the newly created folder change your present working directory by running the below command.

cd task\_10132

1. To confirm the installation of Terraform, run the below command to check the version:

terraform version

1. If you are getting output as command not found: terraform, this means that terraform is not installed on your system, To install terraform follow the official guide link provided in the Prerequisite section above.

Task 7: Apply terraform configurations

1. Initialize Terraform by running the below command,

terraform init

**Text

Description automatically generated**

**Note:** terraform init will check for all the plugin dependencies and download them if required, this will be used for creating a deployment plan

1. To generate the action plans run the below command,

terraform plan

Text

Description automatically generated with medium confidence

1. To create all the resources declared in main.tf configuration file, run the below command:

terraform apply

1. Approve the creation of all the resources by entering **yes**.

Text

Description automatically generated

1. It may take up to 10-15 minutes for the terraform apply command to create the resources.
2. Endpoint of cluster created by terraform will be visible there.

Task 8: Check the resources in AWS Console

1. Make sure you are in the **US East (N. Virginia) us-east-1** Region.
2. Navigate to **EKS** by clicking on **Services** on the top, then click on **EKS** in the **Containers** section.
3. Click on the **Clusters** on the left navigation panel. You can see the cluster created successfully.

Graphical user interface, application

Description automatically generated

Task 9: Create an Environment in CloudShell

1. Make sure you are in the N.Virginia Region.
2. Click on icon (Cloud Shell) on the top right AWS menu bar.
3. A new tab in your browser opens and if you see a welcome message to cloud shell then click on the Close button in that message.
4. You will see a creating environment message on the screen.

Graphical user interface

Description automatically generated with medium confidence

1. Wait for a few minutes to complete the environment creation. Once the environment is created, You are ready to use the terminal.

Text

Description automatically generated

Task 10: Install kubectl on AWS CloudShell

1. Once the environment is ready on CloudShell, Download the Amazon EKS vended kubectl binary for your cluster's Kubernetes version from Amazon S3. To do so, run the following command:

curl -o kubectl https://amazon-eks.s3.us-west-2.amazonaws.com/1.18.9/2020-11-02/bin/linux/amd64/kubectl

1. Apply execute permissions to the binary.

chmod +x ./kubectl

1. Copy the binary to a folder in your PATH. If you have already installed a version of kubectl, then we recommend creating a $HOME/bin/kubectl and ensuring that $HOME/bin comes first in your $PATH.

mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export PATH=$PATH:$HOME/bin

1. After you install kubectl , you can verify its version with the following command:

kubectl version --short --client

Task 11: Configure your AWS CloudShell to communicate with your cluster

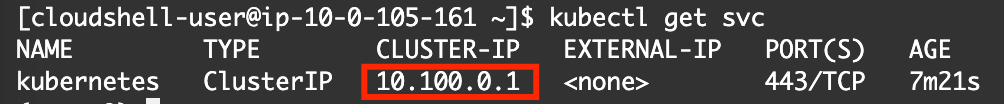
1. Once the environment is ready on CloudShell, you create a kubeconfig file for your cluster. The settings in this file enable the kubectl CLI to communicate with your cluster.
2. To create a kubeconfig file, run the following command:

aws eks update-kubeconfig --region us-east-1 --name whiz

**Note: If your cluster name is not whiz, Update your cluster name in the above command.**

1. Test your configuration, with the following command:

kubectl get svc



Task 12: Delete AWS Resources

1. To delete the resources, open Terminal again.
2. Run the below command to delete all the resources.

terraform destroy

1. Approve the creation of all the resources by entering **yes**. You can see the **Destroy complete!** message.  
   Graphical user interface, text

   Description automatically generated

**Completion and Conclusion**

* You have successfully set up the Visual Studio Code editor.
* You have successfully created variables.tf and terraform.tfvars files.
* You have successfully created an EKS cluster using terraform
* You have successfully created output.tf
* You have successfully executed the terraform configuration commands to create the resources.
* You have successfully checked all the resources created by opening the Console.
* You have successfully installed Kubectl in AWS Cloudshell.
* You have successfully configured AWS Cloudshell to communicate with AWS EKS Cluster.
* You have successfully deleted all the resources.

**End Lab**

1. Sign out of AWS Account.
2. You have successfully completed the lab.
3. Once you have completed the steps, click on from your whizlabs dashboard.