Overview

Setup an AWS VPC and ELB.

Requirements

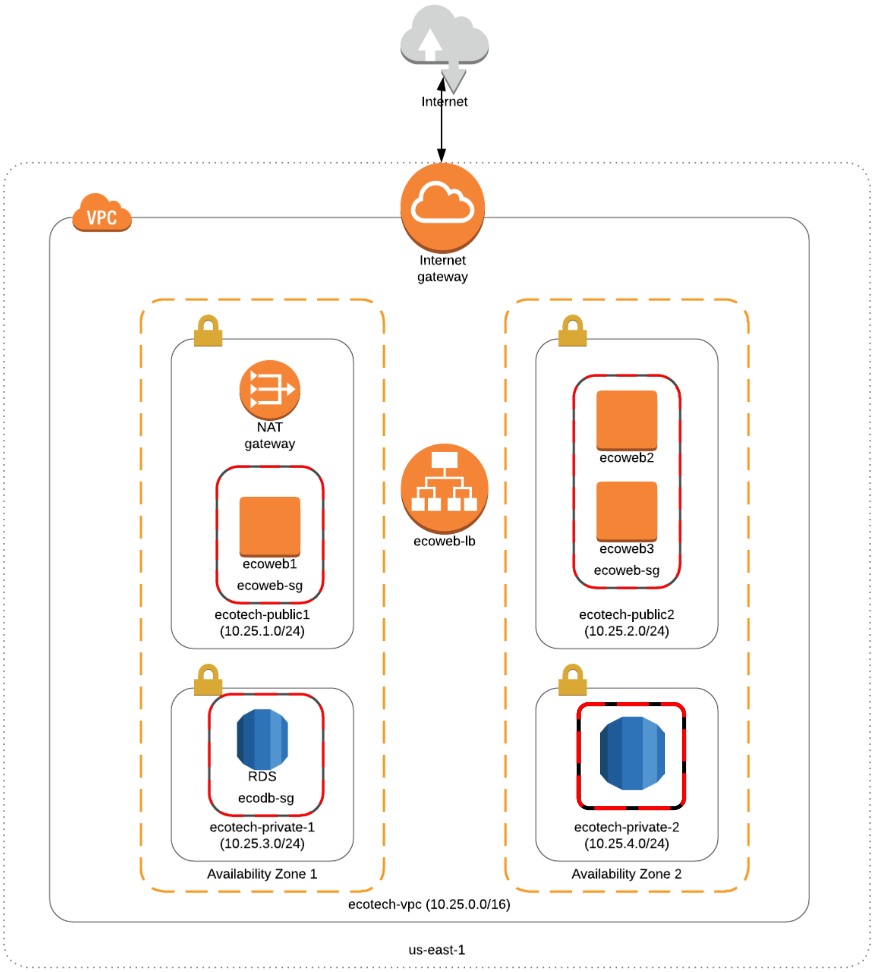
- AWS account.
- SSH terminal application.

The project

Let's get started!

Create a VPC

Create a VPC with the following resources and architecture:


Internet
VPC
Internet gateway
NAT gateway
ecoweb1
ecoweb-sg
ecotech-public1 (10.25.1.0/24)
ecoweb-lb
ecoweb2
ecoweb3
ecoweb-sg
ecotech-public2 (10.25.2.0/24)
RDS
ecodb-sg
ecotech-private-1 (10.25.3.0/24)
Availability Zone 1
ecotech-private-2 (10.25.4.0/24)
Availability Zone 2
ecotech-vpc (10.25.0.0/16)
us-east-1

The VPC (`ecotech-vpc`) exists in the us-east-1 region, uses a network CIDR block of 10.25.0.0/16 and is partitioned into 4 subnets:

- ecotech-public1 (10.25.1.0/24)
- ecotech-public2 (10.25.2.0/24)
- ecotech-private1 (10.25.3.0/24)
- ecotech-private2 (10.25.4.0/24)

Each availability zone contains a public subnet and a private subnet. The exact AZs don't make a difference, as long as the AZs are different.

Create an Internet Gateway for the VPC.

Create a NAT gateway in the `ecotech-public1` subnet. Configure the routing tables for the public subnets and private subnets. Do you remember which subnets route through the Internet Gateway and which subnets route through the NAT gateway?

Create EC2 instances

Create three EC2 instances with the following properties:

- t2.micro instance size
- AMI: ami-3ea13f29 (public community image)
- Instances are named `ecoweb1`, `ecoweb2`, and `ecoweb3`
- The three instances should launch into public subnets in separate AZs (two instances are in the same AZ).
- The instances should have public IP addresses.
- The instances are associated with a security group called `ecoweb-sg`. This security group should allow:
    - Incoming traffic on port 80 (http) from the Internet.
    - Incoming traffic on port 22 (ssh) from your workstation (use http://checkip.amazonaws.com to determine your IP address).

Create ELB

Create an Application Elastic Load Balancer with the following properties:

- Listen on port 80 (http)
- Perform a health check on port 80 to the URL endpoint: `/index.php`
- ELB is associated with the `ecoweb-sg` security group.
- Setup a target group called `ecotech–webservers` and associate webserver instances with the group.

Verify that you can access the webservers by going to the ELB endpoint address in your web browser. Refresh your browser to see the ELB distribute the request to a different instance each time.

Fail over instances

Simulate a failure of one of the instances by stopping the instance. Verify that status of the instance in the ELB target group. Access the ELB endpoint in the browser to verify that the ELB is no longer distributing requests to the failed instance.

Create RDS instance

RDS now requires a database instance to be in a DB subnet group, which is a collection of subnets (typically private) that you create for a VPC and that you then designate for your DB instances. Each DB subnet group must have at least one subnet in at least two Availability Zones in the region.

To create a DB subnet group

1. Open the Amazon RDS console at https://console.aws.amazon.com/rds/.
2. In the navigation pane, choose **Subnet groups**.
3. Choose **Create DB Subnet Group**.
4. For **Name**, type **ecotech-db-subnet-group**
5. For **Description**, type a description for your DB subnet group.
6. For **VPC**, choose **echotech-vpc**.
7. In the **Add subnets** section, click the **Add all the subnets related to this VPC** link.
8. You will see all four subnets are added to this group. **REMOVE ecotech-public1 and echotech-public2 from the group**.
9. Click **create**

Now you should have a DB subnet group called `ecotech-db-subnet-group` Launch a single-instance MySQL RDS instance called `ecotech-db1` into this DB subnet group. Create a security group for this instance called `ecodb-sg`. Setup the security group to allow traffic from the webservers in `ecoweb-sg` to the RDS instance over port 3306.

Verify that you can reach the database endpoint from one of the webservers. One way to verify this connectivity is to shell into an EC2 instance and run the command:

```
nc -vz <RDS instance endpoint> 3306
```

where you substitute the actual RDS endpoint address in the command string.

Create a Git repo

SSH into the `ecoweb1`, while you are on the `ecoweb1` server, create a directory called `assignment-5-vpc-<username>` (where <username> is your GitHub account name). Change to this directory and initialize a Git repository. Because you are creating a Git repository on a private network, Git might complain about your identity when you try to commit files. You can set your local Git identity by running these commands:

```
$ git config --global user.email "your@email.com"
$ git config --global user.name "your GitHub username"
```

When you create a Git repository and start working with files, the master branch of the repository is checked out by default. Go ahead and create a file in the directory called `README.md` and add the following contents:

```
*Assignment 5 Files*
```

Commit the file to the repository.

You just committed the file to the master branch of the repository. We generally try to avoid working directly on the master branch in our professional life. Go ahead and create a new branch in the Git repository called `develop` and check it out. This branch will have the README.md file in it because you just branched from the most recent commit in master.

Create a shell script

The next step is to create a shell script called `metadata.sh` inside the assignment directory. The shell script should meet the following requirements:

- If the user runs the script with the `-c` or `--create` arguments, the script will perform the following operations:
    - Create a file called `rds-message.txt` in the present working directory containing message returned by

      running the command:

      ```
      nc -vz <RDS instance endpoint> 3306
      ```

      where you substitute the actual RDS endpoint address in the command string.

    - Create a file called `ecoweb1-identity.json` in the present working directory containing the data returned by accessing the URL:

      ```
      http://169.254.169.254/latest/dynamic/instance-identity/document/
      ```

- If the user starts the script with a `-v` or `--version` argument, the script will return the value `0.1.0`
- If the user doesn't provide any arguments the script will provide basic usage information.
- You should encapsulate all of the features of the script in functions.

Once you have completed the shell script, make sure you run the script properly to create the required files. Commit all the files in the assignment directory to the develop branch in the repository. Next, you will need to push your local repository up to GitHub:

- Create a new repository on GiHub Classroom by clicking this link: https://classroom.github.com/a/ccBKkRtU
- Merge your `develop` branch back into the `master` branch on your local Git repository.
- Push your local master branch up to the GitHub repository.

Confirm that the three files are now in your new GitHub repository:

- metadata.sh
- ecoweb1-identity.json
- rds-message.txt

Terminate AWS resources

Remember to terminate your EC2 instances, RDS instance, ELB, and NAT gateway. Delete the VPC and all of the network components created.