

AWS Web Hosting Platform Runbook

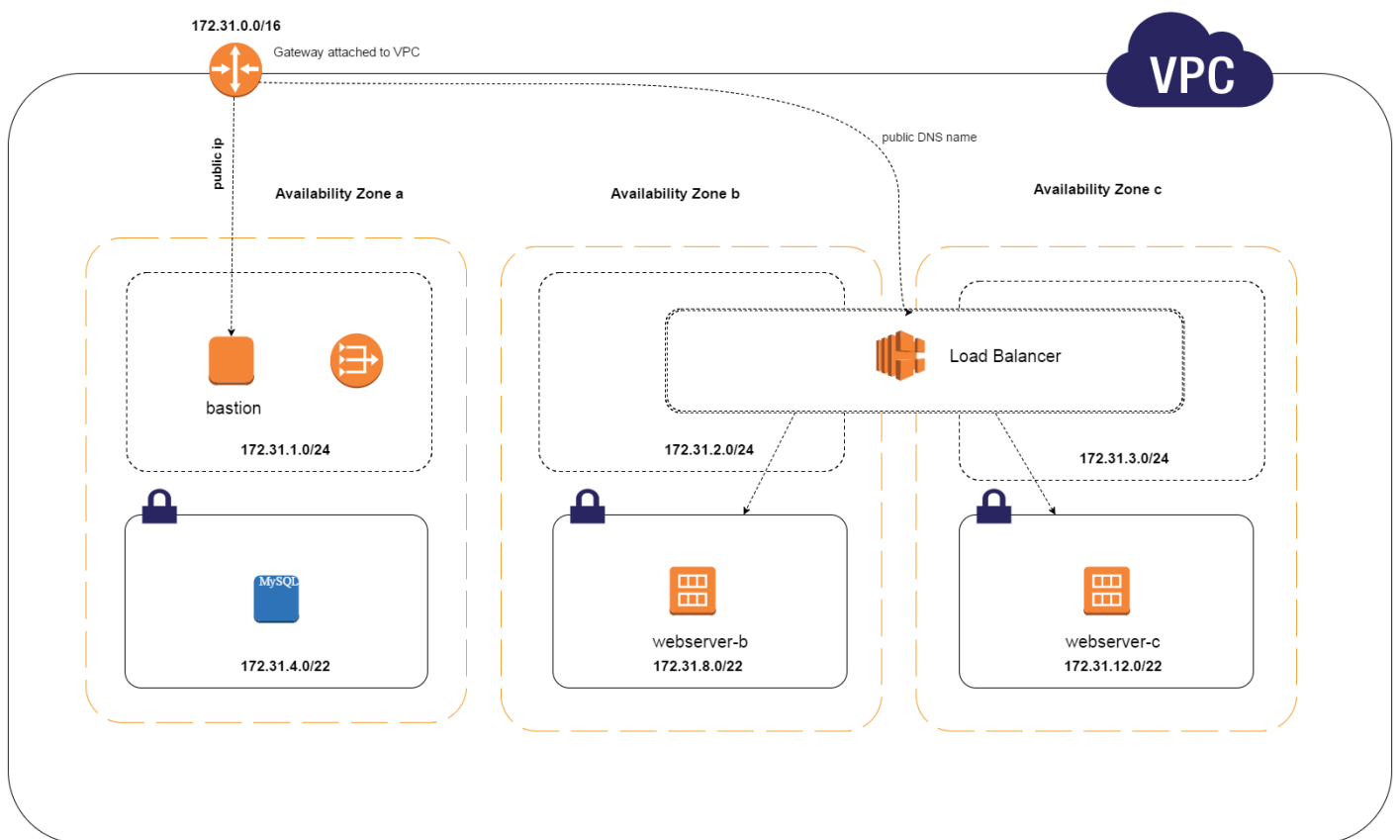
Short Description

Setup Amazon AWS Infrastructure using Terraform and use Ansible to setup/configure web servers.

Required Software

- Amazon AWS subscription
- Amazon EC2 AMI Linux X3
- Amazon RDS MariaDB
- Putty [optional]
- Terraform
- Ansible 2.1
- nginx.x86_64
- php54.x86_64, php-fpm.x86_64, php-ldap.x86_64, php-mbstring.x86_64, php-mcrypt.x86_64, php-mysql.x86_64

Architecture Diagram



Deployment

Before carrying out the deployment please follow the listed Pre-requisites:

- 1) Centos 7 installed in your computer or VirtualBox
- 2) Installed Terraform in the Centos 7
- 3) Installed git

Part 1) Create AWS Infrastructure

- 1) Using git clone the cit-360 folder [**will only need terraform folder for setting up AWS infrastructure*]
- 2) cd into terraform folder and open `terraform.tfvars` file
 - a) insert your variables
 - ✓ `vpc_id, aws_access_key, aws_secret_key, aws_key_name , aws_key_path` [AWS account info]
 - ✓ `db_username, db_password` [RDS master user/password]
- 3) Open AWS, navigate to EC2. Note the private IP address of webserver-b and webserver-c. Note the public IP of remaining instance.
- 4) Navigate to RDS and note down your endpoint.

Part 2) Deploy web services and database configurations

- 1) ssh into the public ip with the user ec2-user
 - a) *ssh -i ~/.ssh/cit360.pem ec2-user@bastion_public_ip*
- 2) Install git
- 3) Using git, clone cit-360 folder again
- 4) cd into ansible folder, open `hosts.ini` file.
 - a) Under `[web]`, insert your obtained 2 private ips.
 - b) Under `[db]`, insert your RDS endpoint and master db_username from Part 1.
- 5) Run ansible playbook by running the following commands:
 - a) *ansible-playbook -I hosts.ini db.yml --ask-vault-pass*
 - b) *ansible-playbook -I hosts.ini web.yml --ask-vault-pass*
- 6) Once both ansible playbooks have finished running successfully, navigate back to your AWS Console.
- 7) Go into EC2 and click Load Balancer from the left panel.
- 8) Click on the load balancer and copy down the DNS name.
- 9) Insert the DNS name into the browser and Curriculum website should be up and running.
- 10) Congratulation, now you have your web services perfectly setup inside AWS.

Issues

Title: db.yml failed

Description: localhost ssh not allowed

Remediation Steps:

1. ssh-keygen -t rsa
Press enter for each line
2. cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
3. chmod og-wx ~/.ssh/authorized_keys
- 4) localhost's public key is now added in authorized_keys

Title: Nginx server might be disabled

Description: Something might have stopped nginx service

Remediation Steps:

- 1) ssh into both webserver from bastion
- 2) Systemctl status nginx
- 3) If "running" is not shown start nginx by systemctl start nginx and systemctl enable nginx

Title: Error with nginx due to syntax problem

Description: nginx.conf file might got some syntax error

Remediation Steps:

- 1) ssh into both webserver from bastion
- 2) Nginx -t [in each server]
- 3) If the result show "test failed", there will be a file listed on which the test failed.
- 4) Open the listed file and check for improper syntax.

Title: failed to ssh into bastion instance

Description: blank screen when trying to login into bastion instance

Remediation Steps:

- 1) Login to AWS, navigate to EC2.
- 2) Check the public ip of bastion instance again.
- 3) If IP is changed, try to login in with the newly assigned IP.