CURRICULUM WEBSERVICE RUNBOOK

SHORT DESCRIPTION

Spin up a webserver that will host the Curriculum website, along with instances for the bastion host and database. Ensure security groups allow internet access from specified locations and configure load balancer to use both webservers.

REQUIRED SOFTWARE

PHYSICAL MACHINE

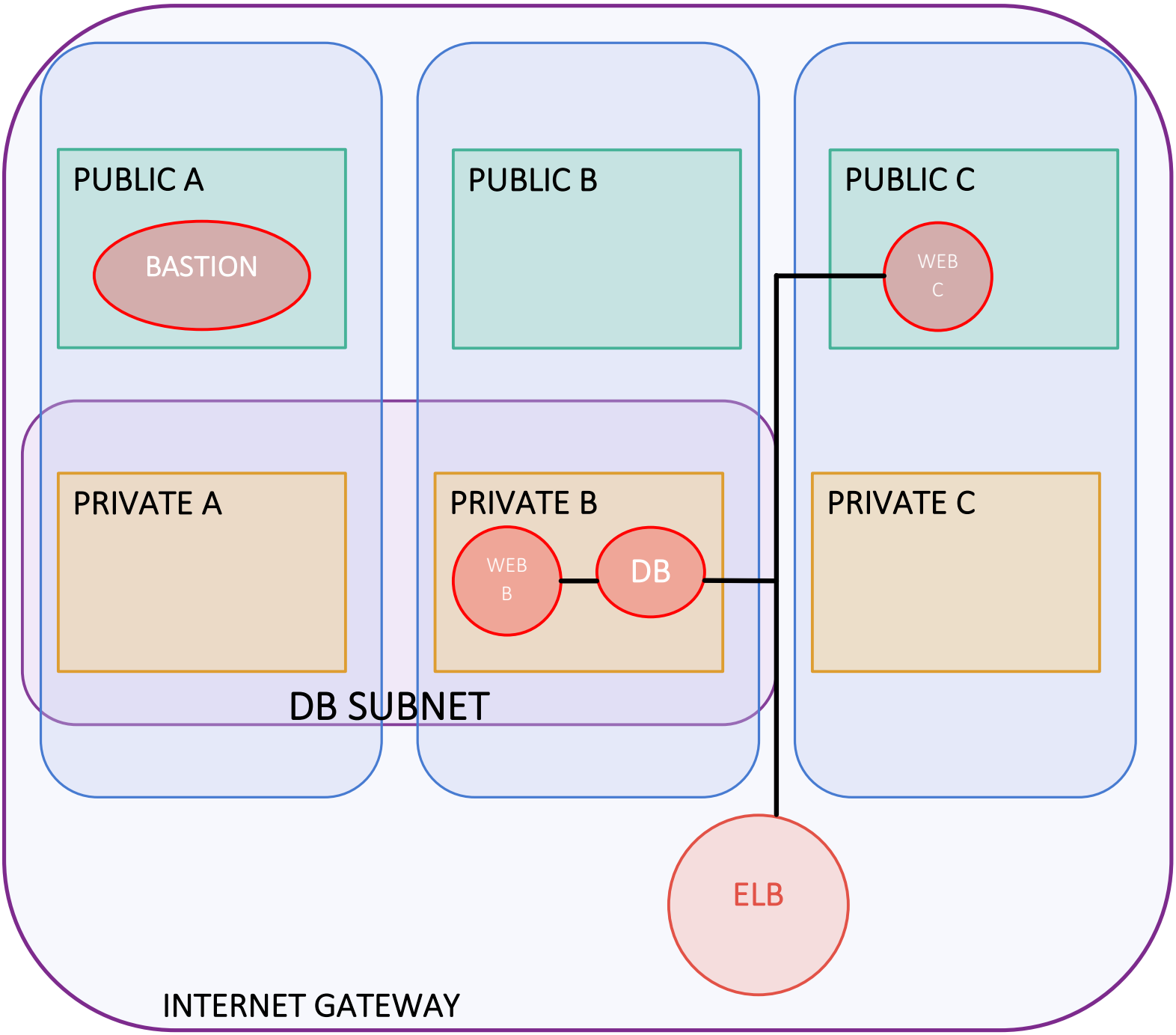
- Git
- Ansible
- Terraform
- SSH
    - Mac/Linux: Terminal
    - Windows: MobaXterm, Putty, etc
- AWS CLI

AWS Account

- Bastion Host
    - Git
    - Ansible
    - Maria DB
- Webserver (x2)
    - Nginx
    - PHP & modules
    - Composer

Note: Maria DB as well as requirements for webservers will be installed once Ansible scripts are run.

ARCHITECTURE DIAGRAM



DEPLOYMENT

First, you must set up the environment on your machine. You must install Git so that you may clone the Terraform code, and Terraform to run it. You may install Ansible if you would like to build on and test the Ansible code provided in the Git repository. The Ansible instructions will be included in the Bastion host installation instructions.

Git

1) Go to https://git-scm.com/downloads.
2) Select your Operating System and version of your machine.
3) Clone the repository.

   $ git clone https://github.com/ashleyvargas/cit-360

4) Go to terminal and set Git config.

   $ git config --global user.name "FirstName LastName"

   $ git config --global user.email youremail@domain.com

Terraform

1) Go to https://www.terraform.io/downloads.html.
2) Select appropriate download for your Operating System.
3) You should have downloaded a zipped file. Extract it where you want terraform installed.
4) Add Terraform to your PATH.

   $ PATH=/usr/local/terraform/bin:/home/account-name/terraform:$PATH

5) You should receive a manual page if you type $ terraform in the terminal.

AWS CLI

In order for Terraform to access your AWS account, you must install the client where you will store the access key and secret access key.

1) Log into AWS account and under IAM, find your user and Create Access key under the Security tab. Show and download User Security Credentials.

   https://illbecreativelater.signin.aws.amazon.com/console

2) Configure AWS access keys on your machine.

   $ aws configure

3) Enter AWS Access Key ID and AWS Secret Access key when prompted.

Run Terraform

1) Change your directory to where the Git repository was cloned.
2) Run Terraform command and enter database password when prompted.

```
$ terraform apply
```

3) When the code is complete, you should see the following:

```
Apply complete! Resources: 20 added, 2 changed, 0 destroyed.

The state of your infrastructure has been saved to the path
below. This state is required to modify and destroy your
infrastructure, so keep it safe. To inspect the complete state
use the `terraform show` command.

State path: terraform.tfstate
```

Bastion Instance

1) SSH into Bastion Instance

```
$ ssh –i path-to-pem-key ec2-user@ec2-public-dns.us-west-2.compute.amazonaws.com
```

2) Install Ansible

```
$ sudo yum update

$ sudo yum install epel-release

$ sudo pip install ansible
```

3) Install Git and clone the Git repository

```
$ sudo yum install git

$ git clone https://github.com/ashleyvargas/cit-360
```

4) Copy over the private key pair from your local to your Bastion instance.
5) Go to AWS console and find IP addresses of webserver Instances. Edit hosts file and add them under web.

```
$ sudo vi PATH/cit-360/Ansible/hosts.ini
```

```
[web]
10.0.6.124 ansible_ssh_private_key_file=~/.ssh/cit360.pem
10.0.10.204 ansible_ssh_private_key_file=~/.ssh/cit360.pem
```

6) Replace RDS endpoint (find it under RDS in AWS console under appropriate database) in web.yml with localhost in db_host: localhost and db.yml in command: ./make_databases.sh {{ db_password }} localhost chdir=~/db.

```
app_env: test
db_host: tf-20161214092742765480951hse.crlf1byyazaj.us-west-2.rds.amazonaws.com:3306
db_database: mydb
```

```
- name: Run the script to create a db for the website
  become: yes
  command: ./make_databases.sh "{{ db_password }}" tf-20161214092742765480951hse.crlf1byyazaj.us-west-2.rds.amazonaws.com:3306 chdir=~/db
  ignore_errors: True
```

7) Run Ansible. Enter password originally made when secrets.yml was created. Ask creator if unsure.

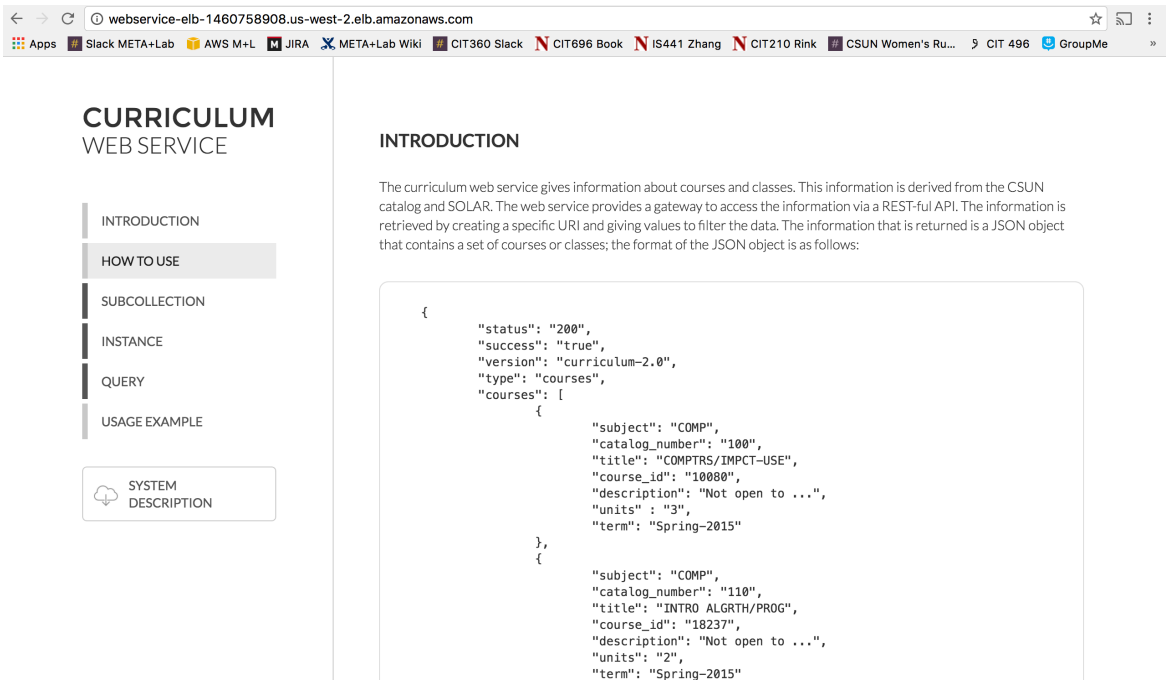
```
$ ansible-playbook -i hosts.ini web.yml db.yml --ask-vault-pass
```

8) Enter password to decrypt the secrets.yml file and let Ansible run.

```
PLAY RECAP *********************************************************************
10.0.10.204                : ok=15   changed=4    unreachable=0    failed=0
10.0.6.124                 : ok=15   changed=4    unreachable=0    failed=0
localhost                  : ok=6    changed=3    unreachable=0    failed=0
```

Website

Our website should now be up and running! Return to the AWS console and navigate to the EC2 dashboard for the public DNS of the ELB. Copy and paste it into a web browser.



ISSUES

Instance / Host Unreachable

```
fatal: [10.0.4.94]: UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: no such identity: /home/ec2-user/cit360.pem: No such file or directory\r\nPermission denied (publickey).\r\n", "unreachable": true}
```

Description: Private Key not found; can't connect to host.

Remediation Steps: Make sure you copied .pem file over to Bastion Host. If you did, private key may be in a different file specified in hosts.ini file. Make sure they match, and try again.

Cannot Connect to Yum Server

```
Cannot find a valid baseurl for repo: amzn-main/latest
```

Description: Instance is having trouble contacting Yum server. Internet access is not allowed outbound.

Remediation Steps: Return to security groups and ensure egress, or outbound, internet access is open to accept from all sources, accepting all traffic.

Website Returning 500 Error

`500 Internal Server Error`

Description: Website not showing; instead a 500 error returns. This means a service has gone down and must be restarted so that the website will work again.

Remediation Steps: Restart Nginx service using the command:

```
$ sudo service nginx restart
```