Curriculum Runbook

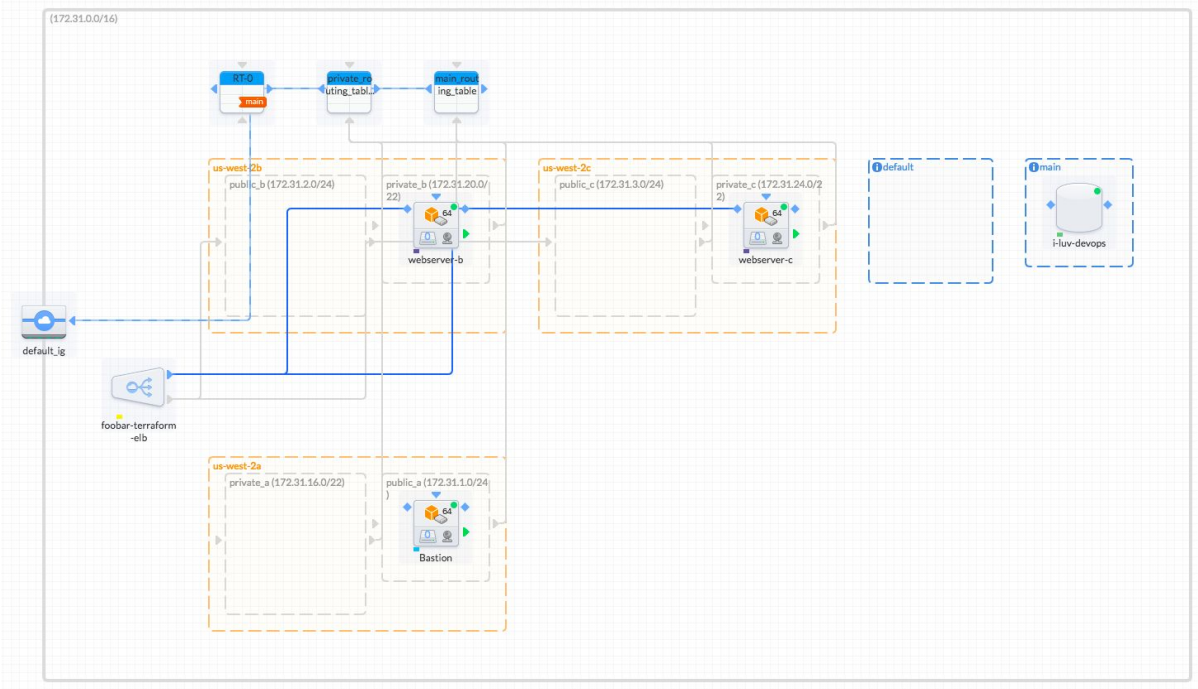
Short Description

The Curriculum Runbook displays the classes available at CSUN. We do this by dumping a database and web services onto dedicated servers.

Required Software

The software required are Amazon Web Services, Ansible, Terraform, AWS CLI, Git, OpenSSH, MariaDB (community-fork of the MYSQL relational database software). For the web servers, you'd need nginx, composer, php54, php54-fpm, php54-ldap, php54-mbstring, php54-mcrypt, php54-mysql, php54-phpunit-PHPUnit

Architecture Diagram



Deployment

After creating the virtual infrastructure via Terraform, it is now time to SSH into our Bastion host. For convenience, it is recommended to create an "ssh_config" or "config" file in our .ssh folder. For reference, it should be as followed:

Host bastion
    Identityfile ~/.ssh/cit360.pem
    user ec2-user
    *Hostname *insert AWS EC2 instance**

PasswordAuthentication no

We can reuse and modify this file as needed in order to SSH into webservers-b/c from inside the bastion host. Then we SSH into our Bastion host to access our webservers and run ansible off of. We then run `$ ssh bastion` *It will ask for permission to add to known hosts, then type yes. We also need to import the pem file we created with cit360 public key using secure copy on mac. To run it, we use $* `scp -i ~/path/to/pem/file ~/path/to/pem/file ec2-user@public-dns-hostname:~/path/to/.ssh` *because we'll reuse the key to connect into our webservers. Then we installed ansible on our Bastion host . We can install our ansible folders one of two ways but for simplicity we'll use secure copy with a directly flag: $* `scp -i ~/path/to/pem/file -r  ~/path/ansible ec2-user@public-dns-hostname:~/` *When everything is finished uploading, installing ansible is done through sudo pip install ansible. Then we run the web.yml and db.yml playbooks. But before that is done, we need to update the hosts.ini file.*

Issues

Title: *Permission Denied SSH Into Instance*
Description: *Unable to SSH into instance with the permission denied error.*
Remediation Steps:
Your private key file must be protected from read and write operations from any other users. If your private key can be read or written to by anyone but you, then SSH ignores your key and you see the following warning message below.

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@         WARNING: UNPROTECTED PRIVATE KEY FILE!          @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
Permissions 0777 for '.ssh/my_private_key.pem' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
bad permissions: ignore key: .ssh/my_private_key.pem
Permission denied (publickey).
```

If you see a similar message when you try to log in to your instance, examine the first line of the error message to verify that you are using the correct public key for your instance. The above example uses the private key .ssh/my_private_key.pem with file permissions of 0777, which allow anyone to read or write to this file. This permission level is very insecure, and so SSH ignores this key. To fix the error, execute the following command, substituting the path for your private key file.

$ **chmod 0400** ***.ssh/my_private_key.pem***