Local Environment Setup Guide

This document provides instructions for setting up your local machine to manage the AWS infrastructure defined in this Terraform project.

Dependencies

You will need the following command-line tools installed:

1. **Terraform**: For managing infrastructure as code.
2. **AWS CLI**: For interacting with AWS services.
3. **SSH Client**: For connecting to the EC2 instances. (This is typically pre-installed on macOS and Linux).

1. Install Terraform

If you don't have Terraform installed, download it from the official website.

- **Website**: https://developer.hashicorp.com/terraform/downloads

Follow the instructions for your operating system to install it and ensure the `terraform` binary is available in your system's PATH.

2. Install the AWS CLI

The AWS Command Line Interface (CLI) is the primary tool for managing AWS resources from your terminal.

macOS Using Homebrew (recommended):

```
brew install awscli
```

Or, using the official installer:

```
curl "[https://awscli.amazonaws.com/AWSCLIV2.pkg](https://awscli.amazonaws.com/AWSCLIV2.pkg)" -o "AWSCLIV2.pkg
sudo installer -pkg AWSCLIV2.pkg -target /
```

Linux (x86_64)

```
curl "[https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip](https://awscli.amazonaws.com/awscli-exe-linux
unzip awscliv2.zip
sudo ./aws/install
```

Windows Download and run the official MSI installer from the AWS documentation: - **Link**: Installing the AWS CLI version 2 on Windows

---

AWS Authentication and Configuration

This project is designed to be managed using AWS CLI profiles and environment variables for authentication and region selection.

1. Configure AWS CLI Profiles

The AWS CLI stores credentials in a file located at `~/.aws/credentials`. You can configure multiple profiles for different accounts or roles.

To configure a new profile, run the `aws configure` command with the `--profile` flag. The Terraform configuration for this project specifies the profile name `epcvip-asg`.

```
aws configure --profile epcvip-asg
```

You will be prompted to enter your AWS Access Key ID and Secret Access Key.

```
AWS Access Key ID [None]: YOUR_ACCESS_KEY_HERE
AWS Secret Access Key [None]: YOUR_SECRET_KEY_HERE
Default region name [None]: us-east-1
Default output format [None]: json
```

2. Using Environment Variables

For the helper scripts (`terraform_*.sh`, `asg_stress_test.sh`) to work correctly, you should export the following environment variables in your terminal session.

- `AWS_PROFILE`: Specifies which credentials profile to use.
- `AWS_REGION`: Specifies the AWS region to send API requests to.

Example:

```
export AWS_PROFILE=epcvip-asg
export AWS_REGION=us-east-1
```

By setting these variables, all subsequent `aws` and `terraform` commands in that terminal session will automatically use the correct profile and target the correct region, ensuring consistency with the project's configuration. You can add these lines to your shell's startup file (e.g., `~/.bash_profile`, `~/.zshrc`) to make them permanent.

3. SSH Access

This project has been configured to use SSH for instance access instead of SSM Session Manager.

- **Key Generation**: When you run `terraform apply`, a new 4096-bit RSA key pair is generated. The public key is uploaded to AWS as an EC2 Key Pair, and the private key is saved to your local machine.
- **Private Key Location**: The private key (`.pem` file) is automatically saved to your home directory's `.ssh` folder. The exact path is an output of the Terraform apply command, typically something like `~/.ssh/dp-webapp-dev-key.pem`.
- **Connecting to an Instance**: To connect to an instance, you will need its public IP address and the private key.