Auto Scaling Group Stress Test

This document describes the usage and functionality of the `asg_stress_test.sh` script.

Purpose

The `asg_stress_test.sh` script is designed to validate the auto-scaling functionality of the deployed infrastructure. It simulates a high CPU load event across all running EC2 instances within a target environment's Auto Scaling Group (ASG) by connecting to them via **SSH**. This allows you to observe and confirm that the CPU-based scaling policies trigger correctly, both for scaling out (adding instances) and scaling in (removing instances).

Prerequisites

Before running the script, ensure the following requirements are met:

1. **AWS CLI Installed and Configured**: You must have the AWS CLI installed and configured with credentials that have permission to manage EC2 and Auto Scaling.
2. **Terraform Initialized and Applied**: The Terraform infrastructure for the target environment must be successfully deployed (`terraform apply`). The script relies on Terraform to fetch the ASG name.
3. **SSH Client Installed**: A standard SSH client must be available in your system's PATH.
4. **SSH Private Key**: The private key (`.pem`) file generated by Terraform must be available on your local machine. The script is pre-configured to look for it at `~/.ssh/dp-webapp-dev-key.pem`, but this path can be changed inside the script.
5. **Scripts are Executable**: The script must have execute permissions.

   ```
   chmod +x asg_stress_test.sh
   ```

Usage

To run the script, execute it from your terminal, passing the target environment (`dev`, `uat`, or `prd`) as the single argument.

Example for the `dev` environment:

```
./asg_stress_test.sh dev
```

Script Workflow

The script performs the following actions automatically:

1. **Retrieves ASG Name**: It uses the `terraform output` command to read the name of the Auto Scaling Group from your environment's Terraform state file.
2. **Identifies Target Instances**: It queries the AWS API to get the **public IP addresses** of all `InService` instances currently running in the ASG.
3. **Initiates Stress Test via SSH**: It connects to each instance using SSH and executes a shell script remotely. This remote script:
   - Installs the `stress` utility using `yum`.
   - Determines the number of CPU cores on the instance.
   - Runs `stress` to generate 100% CPU utilization on all cores for a predefined duration (default is 6 minutes).
4. **Waits for Scaling Activity**: The script pauses for two periods:
   - It waits for the `stress` command's duration to allow the scale-out alarm to trigger.
   - It then waits for a "cooldown" period (default is 10 minutes) to allow the CPU utilization to drop and the scale-in alarm to trigger.
5. **Displays Activity History**: After the cooldown, the script fetches and displays the activity history for the Auto Scaling Group, showing a chronological log of the scaling events that occurred during the test.

Expected Output

After the test and cooldown periods, you will see a table listing the recent activities of your Auto Scaling Group. You should expect to see:

- Events indicating that new instances were launched in response to the high CPU alarm.
- Later events indicating that instances were terminated in response to the low CPU alarm after the test concluded.

This confirms that your scaling policies are configured and working as expected.