

This script is used to manage infrastructure resources using Terraform, specifically creating or deleting resources, based on the given arguments. Here's a detailed explanation of its functionality:

## Overview

The script helps you manage the lifecycle of Terraform-managed infrastructure resources with options for creating (**create**) or deleting (**delete**). It accepts a set of arguments to configure the specific settings needed for the Terraform configuration.

## Key Features Explained

### 1. Command Argument Handling (**create** or **delete**):

- **create**: Deploy the infrastructure using Terraform.
- **delete**: Tear down the infrastructure that was previously deployed.
- If neither **create** nor **delete** is supplied, the script exits with an error and displays the correct usage.

### 2. Argument Parsing:

- Parses multiple required arguments, including:
  - **--profile**: The AWS SSO profile name.
  - **--confluent-api-key** & **--confluent-api-secret**: The API key and secret for connecting to Confluent (likely for Kafka integration).
  - **--snowflake-warehouse**: The Snowflake warehouse name.
  - **--service-account-user**: The service account username.
  - **--day-count**: Number of days for some specific configuration.
  - **--auto-offset-reset**: Kafka offset reset behavior (**earliest** or **latest**).
  - **--number-of-api-keys-to-retain**: Number of Confluent API keys to retain.
  - **--admin-service-user-secrets-root-path**: The root path in AWS Secrets Manager for admin service user secrets.

### 3. Validation Checks:

- Validates that all required arguments are supplied.
- If any argument is missing, the script provides an appropriate error message and terminates.

### 4. AWS SSO Login:

- Logs in using the specified AWS SSO profile to get temporary AWS credentials.
- Uses **aws2-wrap** to export AWS credentials (**AWS\_ACCESS\_KEY\_ID**, **AWS\_SECRET\_ACCESS\_KEY**, **AWS\_SESSION\_TOKEN**, **AWS\_REGION**, **AWS\_ACCOUNT\_ID**) for further use by Terraform.

### 5. Creating the **terraform.tfvars** File:

- A **terraform.tfvars** file is generated dynamically using the supplied arguments.
- The file contains all the necessary variables to deploy infrastructure using Terraform, such as AWS credentials, Confluent API information, Snowflake warehouse details, and other configuration settings.

## 6. Terraform Actions:

- **Initialize:** Runs `terraform init` to initialize the working directory.
- **Create:**
  - Uses `terraform plan` to preview changes and `terraform apply` to create/update resources.
- **Delete:**
  - Uses `terraform destroy` to delete all resources managed by the configuration.
  - Deletes associated AWS Secrets Manager secrets used by Confluent and Snowflake.

## Usage Example

The script should be run with the following syntax:

```
./deploy-terraform.sh <create | delete> --profile=<SSO_PROFILE_NAME>
--confluent-api-key=
<CONFLUENT_API_KEY>
--confluent-api-secret=
<CONFLUENT_API_SECRET>
--snowflake-warehouse=
<SNOWFLAKE_WAREHOUSE>
--service-account-user=
<SERVICE_ACCOUNT_USER>
--day-count=<DAY_COUNT>
--auto-offset-reset=<earliest |
latest>
--number-of-api-keys-to-retain=
<NUMBER_OF_API_KEYS_TO_RETAIN>
--admin-service-user-secrets-root-
path=<ADMIN_SERVICE_USER_SECRETS_ROOT_PATH>
```

- **create:** Deploy infrastructure using Terraform.
- **delete:** Remove infrastructure managed by Terraform.
- `--profile=<SSO_PROFILE_NAME>`: The AWS SSO profile to use.
- `--confluent-api-key` & `--confluent-api-secret`: Credentials for Confluent API.
- `--snowflake-warehouse`: The Snowflake warehouse to use.
- `--service-account-user`: The service account username.
- `--day-count`: Number of days for some retention or lifecycle policy.
- `--auto-offset-reset=<earliest | latest>`: Specify Kafka offset reset strategy.
- `--number-of-api-keys-to-retain`: Specify the number of API keys to retain.
- `--admin-service-user-secrets-root-path`: The root path in AWS Secrets Manager for admin service user secrets.

## Summary

- **Lifecycle Management:** The script allows you to create and delete infrastructure with Terraform.
- **AWS Integration:** Uses AWS SSO for secure access and manages credentials through environment variables.

- **Confluent and Snowflake Configuration:** Integrates with Confluent (Kafka) and Snowflake, and dynamically generates the necessary configuration files for Terraform.
- **Secrets Management:** Deletes associated secrets in AWS Secrets Manager when tearing down the infrastructure.

This script provides a streamlined way to manage infrastructure resources with Terraform, ensuring all configurations and credentials are correctly handled throughout the infrastructure lifecycle.