

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:

- [1.0 Overview](#)
- [2.0 Key Features Explained](#)
- [3.0 Example](#)
- [4.0 Summary](#)

1.0 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.

2.0 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:
 - **--profile**: The AWS SSO profile name.
 - **--confluent-api-key** and **--confluent-api-secret**: The API key and secret for connecting to Confluent Kafka.
 - **--snowflake-warehouse**: The Snowflake warehouse name.
 - **--admin-user-secrets-root-path**: The root path for admin user secrets.
- Parses multiple optional arguments:
 - **--day-count**: (Default: **30**, when not included) How many day(s) should the API Key be rotated for.
 - **--debug**: (Default: **false**, when not included) Enable debug mode for more verbose output during the **terraform apply** process.

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. Debug Mode:

- If the **--debug** flag is set, the script enables debug mode on **terraform apply**, providing more verbose output for troubleshooting.

5. 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`) for further use by Terraform.

6. 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.

7. 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:**
 - Remove all `snowflake_execute` resources from the Terraform state to prevent the `terraform destroy` run from failing, ensuring a clean teardown process.
 - Uses `terraform destroy` to remove all resources managed by the configuration.
 - Deletes the AWS Glue Database and Tables created, as the `terraform destroy` command does not automatically remove these resources.
 - Deletes associated AWS Secrets Manager secrets used by Confluent and Snowflake because the `terraform destroy` command does not automatically remove these resources.

3.0 Example

The script should be executed using the following syntax:

```
./deploy.sh <create | delete> --profile=<SSO-PROFILE-NAME> \
--confluent-api-key=<CONFLUENT-API-KEY> \
--confluent-api-secret=<CONFLUENT-API-SECRET> \
--snowflake-warehouse=<SNOWFLAKE-WAREHOUSE> \
--admin-user-secrets-root-path=
<ADMIN_USER_SECRETS_ROOT_PATH> \
[--day-count=<DAY_COUNT>] \
[--debug]
```

- **create:** Deploy infrastructure using Terraform.
- **delete:** Remove infrastructure managed by Terraform.

Argument placeholder

Replace with

Argument placeholder	Replace with
<SSO_PROFILE_NAME>	Your AWS SSO profile name for your AWS infrastructue that host your AWS Secrets Manager.
<CONFLUENT_API_KEY>	Your organization's Confluent Cloud API Key (also referred as Cloud API ID).
<CONFLUENT_API_SECRET>	Your organization's Confluent Cloud API Secret.
<SNOWFLAKE_WAREHOUSE>	The Snowflake warehouse (or "virtual warehouse") you choose to run the resources in Snowflake.
<ADMIN_USER_SECRETS_ROOT_PATH>	The root path in AWS Secrets Manager where the admin user secrets are stored.
[<DAY_COUNT>]	(Default: 30, when not included) How many day(s) should the API Key be rotated for.

Flags:

- `[--debug]`: (Default: `false`, when not included) Enable debug mode for more verbose output during the `terraform apply` process.

4.0 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.