Terraform Multi-Cloud Auto-Deploy with AWS & GCP

Abstract

This project demonstrates the design and implementation of a multi-cloud infrastructure using Terraform to provision resources on both Amazon Web Services (AWS) and Google Cloud Platform (GCP).

The solution focuses on Infrastructure as Code (IaC) to automate the deployment of compute resources and networking configurations across multiple cloud providers. The project outputs public IP addresses from both environments, allowing seamless integration of cloud services.

This highlights Terraform's power in enabling consistent, repeatable, and automated deployments in heterogeneous cloud environments.

Introduction

With enterprises increasingly adopting hybrid and multi-cloud strategies, managing infrastructure across multiple cloud providers has become a complex challenge. Traditional manual setups are error-prone and time-consuming.

This project introduces a Terraform-based multi-cloud deployment that provisions:

- EC2 instances on AWS
- Compute Engine instances on GCP
- Networking resources such as Security Groups and Firewall Rules

By maintaining declarative infrastructure code, this project enables repeatable deployments, version control, and automation — reducing operational overhead and ensuring consistent cloud provisioning.

Tools Used

- **Terraform** – Infrastructure as Code tool for AWS & GCP provisioning
- **AWS EC2** – Virtual machines on Amazon Web Services
- **AWS Security Group** – Manages inbound/outbound traffic for EC2
- **GCP Compute Engine** – Virtual machines on Google Cloud Platform
- **GCP Firewall Rules** – Defines and controls network traffic for GCP resources
- **GitHub** – Version control and collaborative management of Terraform code

Steps Involved in Building the Project

1. Environment Setup

- Installed Terraform
- Configured AWS CLI & GCP SDK

- Created project directory structure

2. **Provider Configuration**
    - Defined AWS and GCP providers with credentials and regions
3. **Resource Provisioning**
    - AWS: EC2 instance + Security Group (HTTP/HTTPS)
    - GCP: Compute Engine instance + Firewall Rule (HTTP/HTTPS)
4. **Outputs**
    - Exported AWS IP, GCP IP, and combined multi-cloud IPs
5. **Testing & Validation**
    - Applied Terraform config
    - Retrieved and validated public IPs
    - Confirmed instance accessibility
    - Resolved conflicts via resource import

Final Outcome

- A multi-cloud Terraform configuration capable of provisioning and managing AWS & GCP resources from a single codebase
- Automated deployment of compute instances and network rules
- Centralized visibility of outputs for seamless integration across environments
- Reduced manual errors and achieved consistency across platforms

Conclusion

The project successfully implements a multi-cloud deployment model using Terraform. It demonstrates how infrastructure can be provisioned consistently across AWS and GCP, enabling hybrid and multi-cloud strategies. The use of IaC reduces human intervention, improves scalability, and allows infrastructure to be version-controlled. This approach provides a foundation for future extensions, such as integrating CI/CD pipelines, monitoring, or Kubernetes deployments across multiple cloud providers.

Author
Vaishnavi Tiwari

https://github.com/VaishnaviT501/terraform-multicloud-autodeploy.git