

Assignment 5 - CI/CD - GitOps

- Sangram Jagtap
- Omkar Nagarkar
- Purvil Patel
- Atharva Jadhav

GitOps is a paradigm or a set of practices that emphasizes the use of Git as the source of truth for declarative infrastructure and applications. With Git at the center of the delivery pipelines, every change is submitted as a commit, which then goes through a version control system for approval before being applied to the infrastructure. It combines the version control system for code with the automation to apply that code to the infrastructure. This approach allows for improved collaboration, increased deployment speed and frequency, enhanced reliability and stability, and better compliance and auditing.

Benefits that organizations can realize by adopting GitOps:

Increased Deployment Speed and Frequency:

Automation and integrated collaboration mechanisms greatly increase the speed and frequency of deployments.

Increased Reliability: With Git's revert/rollback capabilities, it's straightforward to roll back to the last known good configuration, reducing recovery time.

Improved Stability: Audit logs are automatically created for all changes, ensuring greater system stability.

Consistency and Standardization: Environments are defined as code, allowing for consistent deployment across different environments.

Improved Collaboration:

GitOps keeps the entire state of your environment under source control, making it easier to collaborate on infrastructure changes.

Ease of Adoption: Git is widely used and understood, making it easier for developers to adopt GitOps practices.

Greater Efficiency: Teams don't have to switch tools to manage updates, increasing efficiency.

More Robust Security: Using Git for infrastructure definitions enhances security and makes all changes auditable.

Case Study:

Improving Developer Experience at a Financial Services Company with GitOps

Organization:

A large, heavily regulated financial services company.

Challenge:

The company needed to improve its developer experience to deliver value more efficiently while maintaining compliance and oversight.

Solution:

The company implemented GitOps to manage their infrastructure and application deployment processes.

Implementation:

1. They stored the state of their infrastructure in Git, ensuring every change was tracked and had an audit trail.
2. All infrastructure changes were made through automated Git commits.
3. They used a tool that automatically deployed whatever was in the Git repository, streamlining the deployment process.

Benefits Realized:

1. Increased Visibility:
Keeping infrastructure configuration in Git allowed new starters or colleagues from other teams to quickly become acquainted with the infrastructure.
2. Improved DORA Metrics:
GitOps helped improve key DevOps Research and Assessment (DORA) metrics, which are crucial indicators of a team's performance in software system development.
 - a. Deployment Frequency: More frequent and successful releases to production.
 - b. Lead Time for Changes: Reduced time from commit to production.
 - c. Change Failure Rate: Lower percentage of deployments causing a failure in production.
 - d. Time to Restore Service: Quicker recovery from failures in production.
3. Enhanced Developer Experience:
The GitOps framework provided a repeatable pattern to automate tasks, thereby improving the overall developer experience.
4. Disaster Recovery:
In the event of a complete cluster failure, pointing a rebuilt or new cluster to the Git repository allowed for quick restoration of resources, improving the Time to Restore Service metric.

Key Principles:

1. The company focused on key principles rather than specific tooling, emphasizing the importance of Git as the only required tool.
2. They chose the best tools for the job, preferring flexibility over alignment with a specific vendor.

Task 2:

1. Integrated with GitHub: GitHub Actions is natively integrated with GitHub, eliminating the need for third-party integrations or additional sign-ups. This makes setting up CI/CD pipelines for repositories hosted on GitHub straightforward.
2. Easy Setup: With GitHub Actions, you can set up CI/CD workflows directly within your repository using YAML files. There's no need to set up and maintain separate servers or infrastructure as with Jenkins.
3. Built-in Secret Management: GitHub provides a secure way to store and use encrypted secrets in your workflows, ensuring sensitive data like API keys or credentials are kept safe.
4. Granular Permissions: With GitHub Actions, you can set permissions at the workflow, job, or step level, providing fine-grained control over who can execute or modify specific parts of the CI/CD process.
5. Cost-effective: For public repositories, GitHub Actions provides a generous amount of free build minutes. Even for private repositories, the pricing is competitive, especially considering the integrated nature of the service.