Getting Started with GitOps

This lesson introduces us to the concept of GitOps, its benefits and the challenges faced in applying it.

We'll cover the following

- Git as the only source of truth
- What is GitOps?
- Why do we need GitOps?
- What are the challenges faced while applying GitOps?

Git as the only source of truth

Git is the de-facto code repository standard and hardly anyone argues against that statement today. Where we might disagree is whether Git is the only source of truth, or even what we consider in that.

When I speak with teams and ask them whether Git is their only source of truth, almost everyone answers yes. However, when I start digging, it usually turns out that's not true. Can you recreate everything using only the code in Git? By everything, I mean the entire cluster and everything running in it. Is your entire production system described in a single repository? If the answer to that question is yes, you are doing a great job, but we're not done with the questioning. Can any change to your system be applied by making a pull request, without pressing any buttons in Jenkins or any other tool? If your answer is still yes, you are most likely already applying GitOps principles.

What is GitOps?

GitOps is a way to do Continuous Delivery. It assumes that Git is a single source of truth and that both infrastructure and applications are defined using the declarative syntax (e.g., `YAML`). Changes to infrastructure or applications are made by pushing changes to Git, not by clicking buttons in Jenkins.

Why do we need GitOps?

Developers understood the need for having a single source of truth for their applications a while back. Nobody argues anymore whether everything an application needs must be stored in the repository of that application. That's where the code is, that's where the tests are, that's where build scripts are located, and that's where the pipeline of that application is defined.

The part that is not yet common is to apply the same principles to the infrastructure. We can think of an environment (e.g., production) as an application. As such, everything we need that is related to an environment must be stored in a single Git repository. We should be able to recreate the whole environment by executing a single process based only on information in that repository. We can also leverage the development principles we apply to applications. A rollback is done by reverting the code to one of the Git revisions. Accepting a change to an environment is a process that starts with a pull request.

What are the challenges faced while applying GitOps?

The major challenge in applying GitOps principles is to unify the steps specific to an application with those related to the creation and maintenance of whole environments. At some moment, a pipeline dedicated to our application needs to push a change to the repository that contains that environment. In turn, since every process is initiated through a Git webhook fired when there is a change, pushing something to an environment repo should launch another build of a pipeline.

Where many diverge from "Git as the only source of truth" is in the deploy phase. Teams often build a Docker image and use it to run containers inside a cluster without storing the information about the specific release to Git. Stating that the information about the release is stored in Jenkins breaks the principle of having a single source of truth. It prevents us from being able to recreate the entire production system through information from a single Git repository. Similarly, saying that the data about the release is stored as a Git tag breaks the principle of having everything stored in a declarative format that allows us to recreate the whole system from a single repository.

Many things might need to change for us to make the ideas behind GitOps a reality.

For the changes to be successful, we need to define a few rules that we'll use as must-follow commandments. Given that the easiest way to understand something is through vivid examples, I will argue that **the processes employed in Continuous Delivery and DevOps are similar to how Buckingham Palace operates and are very different from Hogwarts School of Witchcraft and Wizardry**. If that did not spark your imagination, nothing will. But, since humans like to justify their actions with rules and commandments, we'll define a few of those as well.

Let's get started and discuss these commandments in the next lesson.