Validating Upgrades and Backing Up the Cluster

This lesson discusses the validations needed for upgrades and the strategies that we should follow when backing up the cluster.

We'll cover the following

- Validation for upgrades
- Strategies for upgrading Jenkins X
- Backing up your cluster

Validation for upgrades

Before we jump into different upgrade options, I must make an important statement. Do not trust anyone or anything blindly. We (the community behind Jenkins X) are doing our best to make it stable and backward compatible. Upgrading **should work**, but that does not mean that it will **always work**. No matter how much attention we put into making the project stable, there is almost an infinite number of combinations, and you should make an extra effort to test upgrades before applying them to production, just as you're hopefully validating your applications.

> 🔍 **Do NOT trust anyone or anything.** Validate upgrades of all applications, no matter whether you wrote them or if they come from third-parties.

Testing your applications and validating system-level third-party applications is not easy. You are not in full control of third-party applications, especially when they are not fully open-source.

Strategies for upgrading Jenkins X

Excluding the option of upgrading Jenkins X blindly, the two most commonly used strategies are:

1. Run a test instance in parallel with production (e.g., in separate Namespaces).
2. Have a test cluster.

I prefer the second option when we have the ability to create and destroy clusters on demand. In such a case, we can create a new cluster, install the same Jenkins X version we're running in production, upgrade it, test it, and, if everything works as expected, upgrade production as well. If we do not have a test cluster, our best bet is to install Jenkins X in different namespaces and follow the same validation process we'd follow if it would be running in the separate cluster. The major problem with using different namespaces is the probability that a mistake would affect production. It should work well if we're careful and experienced with Kubernetes, but there's no denying that there is a higher chance of messing with production.

Backing up your cluster

No matter whether you test upgrades or how well it does, one thing is for sure; you should have a backup of your cluster. If you do, you should be able to manage the worst-case scenario. You will be able to restore your cluster to the last known working state.

Given that Kubernetes backups are not directly related to Jenkins X and that there is a myriad of options at our disposal, I will not go in depth on evaluating backup solutions nor will I provide detailed instructions. The only thing I will state is that my favorite tool is **Velero**. If you do not have periodic and on-demand backups in place, feel free to check it out and decide whether it is an option that fits your use-case.

All in all, I will assume that you are testing upgrades before you apply them to production, that you are backing up your cluster, and that you can restore the last known good version if everything else fails.

We are about to upgrade our Jenkins X cluster in the next lesson, and you've been warned that the commands that follow do not excuse you from testing and backing up. Off we go...