It’s time to dive into Infrastructure as Code.

Infrastructure as code, or IaC, enables DevOps teams to test applications in production-like environments early in the development cycle. These DevOps teams are expected to provision multiple server instances, test environments or entire cluster groups of servers reliably and to be on-demand. Infrastructure represented as code can also be validated and tested to prevent common deployment issues. At the same time, the cloud dynamically provisions and tears down environments based on IaC definitions.

DevOps teams who implement IaC can deliver stable environments rapidly and at scale. Teams avoid manual configuration of environments and enforce consistency by representing the desired state of their environments via code. Infrastructure deployments with IaC are repeatable and prevent runtime issues caused by configuration drift or missing dependencies.

Before IaC, you might need to change some resources so you would SSH into a host and manually issue a series of commands to perform the change. Still, with infrastructure as code, you would instead change code files or configuration files that can be used alongside automation tools. You would then commit those files to source control to perform those changes, treating them like code. You would then use an automation tool to enact the developments defined in those code or configuration files and make the changes in your live environments. With infrastructure as code provisioning, new resources and changing existing resources are things that are done through automation.

Here is some more scope on why you should use IaC. First, you get consistency, and the creation and management of your infrastructure resources automation have the benefit that it will run the same way every time. Furthermore, it will do the same things every time as humans tend to do something a little differently each time. This gives you a lot of consistency in terms of how things are done within your virtual environments.

Secondly, you get usability when you use infrastructure as code. You can take that code from those code files or configuration files you used to make or change. You can use it to make the exact same change consistently across multiple hosts, or if you need to make the exact change again in the future, you can use that same code again to do that, which means you also get scalability.

Suppose we needed to create a new instance to add to a load-balanced cluster. You could provide the existing instances using infrastructure as code. You can take that code and use it to implement a new instance configured exactly the same way as the current instances in a concise time, which could be only a few seconds or even minutes in the worst case, allowing you to scale up or scale down quickly.

Another excellent benefit of infrastructure as code is that it makes the infrastructure self-documenting, and any changes made to it are documented. These changes made to the code and configuration files are used to store the infrastructure and execute changes.

For example, the way to a particular server is configured can be viewed in source control. In contrast, with the traditional model, understanding how the server is configured involves knowing who logged into it and what commands they executed.

Finally, infrastructure as code helps you simplify the complexity of the infrastructure. Complex infrastructures can be stood up quickly once they are defined as code. As an example, if you have a complex infrastructure involving several database servers, it may be an authentication server. Your organisation requests a new test environment. You could take the code used to configure the authentication infrastructure and immediately spit up an identical infrastructure, possibly even temporarily, to satisfy your organisation's needs.

To sum it up, DevOps teams can work together with a unified set of practices and tools to deliver applications and their supporting infrastructure rapidly, reliably, and at scale. It makes it much easier to manage that complexity when you use infrastructure as code.