Q&A on Infrastructure as Code

WITH RYAN KANE, LEIDOS SENIOR CLOUD ARCHITECT

Q: ***What's your 10-second elevator pitch on Infrastructure as Code?***

Infrastructure as Code (IaC), at its core, allows software engineers to define our infrastructure and its configuration using the same development, configuration management, and test processes we use for developing our applications.

Q: ***As an engineer, what are the most significant benefits derived from instituting Infrastructure as Code?***

Because IaC allows me as a software engineer to define our infrastructure and its configuration using the same development, configuration management, and test processes we use for developing our applications, I have the ability to develop, test and deploy my application the same way in multiple environments, significantly lowering the risk of bugs reaching production. IaC enables me to evaluate previous versions of my application exactly as they were deployed, allowing me to troubleshoot and compare behavior between versions. Deployment time errors caused by environmental differences and human error are eliminated as the infrastructure deployment and configuration is fully automated and repeatable. Leveraging the automation made possible by Infrastructure as Code enables more advanced capabilities, such as zero downtime blue/green deployments.

IaC also enables greater innovation and experimentation at a lower cost. I can deploy an exact replica of my production environment in minutes; I can try something out and then terminate the environment.

Another benefit of Infrastructure as Code, when combined with SecDevOps approaches and cloud computing, is its effect on the handling of IT failures. IT systems fail at unexpected times with or without warning. Before IaC, a server failure could mean hours or days of downtime. Using the automation available by leveraging IaC, we can accept that failure will happen, plan for it, and recover automatically. Instead of focusing on preventing failure, we focus on improving Mean Time to Recovery. Our infrastructure becomes disposable in nature; it is easier to replace than fix. Instead of hours or days to recover, the system replaces broken components in minutes and the broken components can be evaluated to understand the root cause of the failure without impact to end users.

Q: ***What business benefits have your customers shared with you about Infrastructure as Code?***

Using IaC, our customers have achieved significant cost savings through reduced system administration staff requirements and more repeatable deployments. Our customers are also deploying into production on a more regular basis, as production deployments are low risk and have minimal impact to end users. This results in faster delivery of features to their end users, with more reliability in the product, for less money.

One customer at the Chief Information Security Officer (CISO) level sees tremendous value in IaC from a security perspective, as every server running is automatically configured and hardened, with the appropriate security and monitoring tools installed and configured correctly — every time. One-off misconfigurations that can result in security incidents or data breaches are no longer a problem.

Q: ***What are some key lessons learned from Infrastructure as Code implementations you've been involved with?***

First, it is important to allocate the appropriate time and resources to develop and test the code behind IaC. It can be tempting to revert to a manual deployment process to overcome short-term schedule challenges, but this will always hurt you in the long run.

Second, IaC comes with a higher initial startup cost because you have to develop the code for your infrastructure. The higher startup cost is an investment in the future of your product and will begin to pay off the first time you make a production update, if not before.

And finally, implementing IaC and automation after the fact is challenging at best. Automate from day one and never stop. Set success criteria for development activities that include fully automated deployments and configuration management controlled infrastructure code.

Q: ***How do you see Infrastructure as Code evolving over the next two years?***

As with anything SecDevOps or cloud related, IaC is constantly evolving. Containerized deployments managed by tools like Kubernetes have become mainstream, greatly simplifying the IaC process and reducing infrastructure costs. I believe that we'll see advances towards server-less computing, continuing the simplification of infrastructure management and allowing developers to focus exclusively on their code.