GORILLA GUIDE TRAIL MAP

Infrastructure as Code

Infrastructure as Code? (IaC) How can that be? All hardware has code under the hood, but isn't code for coders?

In 2011, Marc Andreessen (Internet and VC legend) wrote an essay, "Why Software Is Eating The World." The ability of software to disrupt just about every aspect of running a business was his call to arms. Fast-forward to now, and "there's an app for that" or "there's an X-as-a-Service for that" is nearly a guarantee.

As impossible as it seems on the surface, orchestrating infrastructure just like any other piece of code will blow your old ideas about infrastructure into millions of tiny pieces.

Follow the map to find out what the journey looks like!

Key Terms

Infrastructure as Code (IaC):

- Ability to automatically manage tech stacks through software, rather than manually

3 Key Objectives of IT Orgs:

- Efficiency (rapid deploys w/few humans); consistency (the same code gets the same results every time—no more misconfigs!); and compliance (IaC manifests are explicit docs for each config)

Continuous Integration (CI):

- A dev practice requiring developers to check-in code into a shared repository. Each check-in is then verified by an automated build and passed to auto or manual testing, making it easy to detect problems early and often

Software Versioning:

- The ability to store different versions of code in a repository, including rollback, rollforward, and self-documented audit review-ready code and logs

The Journey to the Cloud Was Step One

(LEAP TO CHAPTER 1)

- The beauty of "someone else's servers" was shedding cost and operations hassles
- Getting rid of manual configuration of remote infrastructure is step two
- Managing infrastructure is simple when it's a handful of resources. But automation throughout DevOps means hundreds, thousands... #sendrobotmonkeys!
- Don't monkey around with manual work, when automation, the core of IaC, can do the work of a thousand monkeys typing on laptops
- Cloud vendors run on IaC—it's a major reason they can crush their datacenter costs. Isn't it time you had that power too?

The Advantages of IaC over traditional Ops

(LEAP TO CHAPTER 2)

- Responsibility—DevOps has a greater role to play in Ops, instead of "pure" Dev. No more "flinging code over the wall" to Ops #monkeydo #monkeyown
- Standard Deploys—no more ad-hoc, manual scripts; it's centralized and automated deploy scripts FTW!
- Smaller, daily builds with automated testing = shorter, less complicated, and less expensive troubleshooting compared to month+ sized builds

DevOps Monkeys Love Being More Efficient

(LEAP TO CHAPTER 2)

- Managing infrastructure takes FAR less time, with much more consistent results
- Code repositories and a single source of truth are the rallying cry of monkey devs and ape ops #teamwork
- Using IaC to define environments makes hybrid and multi-cloud deployments much easier to abstract out and head off vendor lock-in

IaC Implementations—dance of the manifests, the resources, and the central server

(LEAP TO CHAPTER 3)

- Manifests are text descriptions that spec the infrastructure to deploy
- Manifests are WHAT the infrastructure should be—details like operating systems, networks and user accounts, provisioning to bare metal, VMs, or containers
- Central server reads the manifests, and issues low-level commands to deploy or update the architecture to configure it to spec

Servicing—SLAs, SLOs and SLIs in the age of IaC is... easy?

(LEAP TO CHAPTER 4)

- These are all formal documents—why not turn them into code too?
- Opportunity for a self-correcting system w/monitoring that solves many service issues automagically by morphing the infrastructure
- Version control gives your entire IaC environment the ability to rollback WITHOUT TOUCHING THE HARDWARE #mindblown 🤯

Download the Full Gorilla Guide!

The Gorilla Guide will help you understand the shift from managing relatively fixed, physical infrastructure to managing dynamic infrastructures by specifying virtual machines, containers, and other resources as code.

Highlights include:

- Details on how to maintain infrastructure as code, ensure operational efficiency, and realize an agile, adaptive infrastructure
- An in-depth look at many of the best practices developed for software engineering that can be used with infrastructure as code
- Use cases ranging from a simple task like deploying a web server in Microsoft Azure to managing large-scale clusters in AWS

GET YOUR COPY!

DEVELOPED BY ActualTech Media

www.gorilla.guide