

Infrastructure As Code – Is It REALLY Enough For DevOps?

July 12, 2021 | Posted by: Team Cloudify | Category: infrastructure as code

 Cloudify IAC for DevOps

The new age of modern enterprise architecture is defined by Infrastructure as Code (IAC). By virtualizing entire data ecosystems, organizations gain the ability to design, provision and manage them completely in software, ushering in a new era of infrastructure flexibility and cost-efficiency.

While this may seem far-fetched, and there are still some limitations, recent developments in DevOps can help organizations transform into a fully functional, data-driven entity. This particularly comes into play when tailoring IAC architectures to modern DevOps workflows.

According to [Carlos Schults](#), .NET software developer at Stackify, IAC has emerged in response to the need for highly flexible, highly dynamic infrastructure demands of modern applications and services. As the enterprise business model shifts from providing products to providing digital services, the static legacy approach to infrastructure, in which hardware and software were integrated on a fundamental level, is proving to be too restrictive and costly. These days, successful businesses require broad scalability and substantial flexibility when it comes to deploying, [infrastructure provisioning](#) and orchestrating disparate data resources.

What Is Infrastructure As Code?

Infrastructure as Code is the management of networks, virtual machines, load balancers, and connection topology in a descriptive model. It uses the same versioning as a DevOps team uses for a source code. Like a key that only opens a single door, an IaC model generates the same environment every time it is applied. It is essential to the DevOps process and is used simultaneously with continuous delivery.

An example of how enterprises can use IaC is for software development. Programmers use IaC to create and launch sandbox applications. At the same time, QA professionals can run tests with perfect copies of product environments to test for errors. And then, when it's time for deployment, you can push both infrastructure and code to production in one swift stroke.

Using IaC for cloud infrastructure management allows all actions that can be automated to be automated. IaC allows system components to be configured and provisioned with Terraform and Kubernetes, which can save time, money, and effort. IaC allows tasks from database backups to new feature releases to be done faster, better, and cheaper.

IaC best practices can be implemented when using cloud monitoring, logging, and altering tools to run in some environments and to deliver new system components. Solutions like FluentD, ELK stack, SumoLogic, Datadog, and Prometheus + Grafana can be configured to your project using some of the best practices below.

What Does It Come To Solve?

An Infrastructure as Code approach turns the manual, time consuming process of managing IT infrastructure and adapts it for the modern age. Without IaC, you are forced to hire professionals for each step of the process, from network engineers to hardware maintenance technicians. In addition, data centers would need to be built, which can increase costs exponentially. Manual configurations, due to their limited nature, can struggle with speed and spikes in access. This forces enterprises to set up backup servers or even data centers to prevent the application from potentially being unavailable for a long time.

Using Infrastructure as Code allows enterprises to perform unit testing, functional testing, and integration testing to significantly reduce the frequency of bugs in the product lifecycle. In addition, it allows for mistakes to be caught early in the process, eliminating the possibility of costly (reworks, bad word) late in the process.

The need for written documentation is also all but eliminated. The code itself documents the state of the machine, meaning that infrastructure documentation will always be up to date. This is important because I need to ask why, probably just costs less.

There is the decade old saying "works fine on my computer". It refers to when code that worked in testing doesn't work in production. Infrastructure as Code ensures continuity, because all environments are created and configured automatically. This leaves no room for human error, which greatly speeds up and simplifies the software development and infrastructure operations.

Infrastructure as Code goes beyond infrastructure automation. It involves incorporating DevOps practices into automation scripts to make sure they don't have any errors, ensure that they are able to be deployed on multiple servers, can be recalled in case of problems, and can be engaged by both development and operations teams. This is called DevOps infrastructure automation. Ansible and Puppet are designed to make IaC environments accessible to even novices with a basic understanding of modern coding techniques and structures.

Automated Infrastructure

In a hardware-centric environment, these functions must be done manually, requiring immense manipulation of server, storage and networking elements. This process is the antithesis of today's digital environment. Today's digital environment demands highly customized data environments that can be altered, added to, and decommissioned immediately. The Infrastructure as Code approach allows the enterprise to gain the flexibility to streamline infrastructure management for human operators and to extend full orchestration and automation capabilities to intelligent, autonomous applications and services to create their own virtualized data environments at will.

IAC makes all of this possible, and is key to the integration of DevOps business practices. DevOps allows for a fully collaborative approach to software development, replacing the clunky, monolithic approach of the past. Different teams are able to oversee design, testing, configuration, and deployment all at once.

The advantage is the near immediate deployment of new features and products on a continual basis instead of the once-per-year (if that) approach of traditional development cycles. This Continual Integration/Continual Development (CI/CD) model is strengthened by IAC because it automates all of the complicated, time consuming resource-provisioning and management operations at the speed that modern applications require.

With IAC, in other words, the era of on-demand infrastructure has arrived.

Not even a full-stack IAC environment is worry-free. The need for increasingly complex, high speed development is evolving and workloads are scaling. This forces the enterprise to weigh the pros and cons of the various tools they use to implement virtual ecosystems. As Infostretch Labs' Sanil Pillai explained on SD Times recently, some tools are better for certain jobs than others.

Optimize Existing Automation

Learn more about how Cloudify can help leverage your current automation tools, and help take your automation to the next level with Environment as a Service.

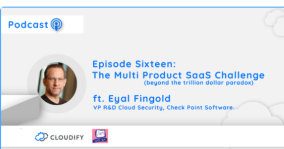
Orchestrate the Orchestrators!

Recent Posts



Cloudify News Roundup: August 2021

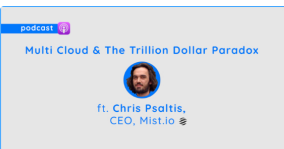
Cloudify News Roundup: 'Trillion dollar paradox' response; Podcast #16, agile-first approach to 5G network slicing, & 4 essential open-source tools for cloud management.



Podcast | Episode Sixteen: The Multi Product SaaS Challenge

ft. Eyal Fingold
VP R&D Cloud Security, Check Point Software.

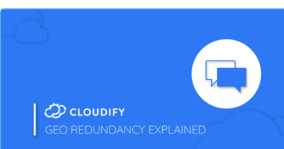
Podcast | Episode Sixteen- the Multi Product SaaS Challenge ft. Eyal Fingold @ Check Point Software



Podcast | Episode Fifteen- Multi Cloud and the Trillion Dollar Paradox

ft. Chris Psaltis, CEO, Mist.io

Podcast | Episode Fifteen- Multi Cloud and the Trillion Dollar Paradox ft Chris Psaltis, CEO at Mist.io



Geo Redundancy Explained | Cloudify

Taking Jenkins CI from Automation to Orchestration – A Continuous Integration A/B Testing Use Case

Infrastructure as Code: The Foundation for 'Environment as a Service'

Infrastructure as Code (IAC) is seen by some as the ultimate expression of the IT work environment. After all, once the entire stack can be...

Read more

An example of certain tools being better for certain jobs than others is when a high level OS abstraction and simplified software installation and maintenance is needed. Here, combining HashiCorp's Vagrant VM management software with Docker containers and the Ansible IT automation stack would accomplish this, however at the expense of a lack of Mac OS support and a high learning curve for inexperienced coders. This may improve efficiency for JDK 8, IDE, and Apache Tomcat projects, but not for some Apple applications. Similarly, combining the Terraform CLI management tool with Jenkins orchestration and AWS allows for easy creation, scale and tear-down of services, but may not provide the necessary support to develop cloud-based GUIs.

Techs Are Still In Charge

Another pitfall surrounding IAC that enterprise executives would be wise to avoid is thinking that they no longer have to worry about things like networking principals, says [Rollout.com's Mark Robinson](#). No matter how resources are managed, operators still need to have a thorough understanding of traffic routing, network architectures and configuration processes. Without that base of knowledge, there is no way to ensure that software-driven automation tasks are delivering optimal results.

Robinson also explains how it is important to distinguish between operations and development. While DevOps may group them together, they must remain distinct functions. Rather than simply writing network configuration scripts as part of the DevOps process, IAC allows for the complete automation of scripts, allowing fully scalable, on-demand infrastructure to be configured completely in code.

All of these developments – IAC, DevOps, CI/CD – push the enterprise towards a full service-based operational environment that is fittingly known as Enterprise as a Service (EaaS). Without these developments, enterprises are forced to deliver infrastructure as a set of discrete services like how enterprise applications deliver CRM and ERP today. These developments open the door for IT to be delivered at a scale and composition as needed. This allows organizations to define their working environments according to the demands of the business model, not the other way around.

IAC represents that first significant break from the confines of legacy infrastructure, but it should not be viewed as the end-game of IT evolution.

IAC DevOps Best Practices

Don't automate everything from the start

- If you are a startup, application or platform that is not fully developed yet, don't try to automate everything from the onset. This is because there can be a high pace of changes. Once your platform becomes more or less stable, you will be able to begin automating it's provisioning and maintenance.

Test and monitor your configurations

- Infrastructure as Code is still code, so it can and should be tested. You should implement testing and monitoring tools for IaC to check for errors and inconsistencies in your servers prior to deployment.

Exercise caution when extending IAC tools to novices

- Working with IaC is like operating a dam. In the hands of a capable operator, you can power an entire town or spin a hundred instances in a few minutes. However, in the hands of a novice, they could open up the dam gates too fast, or mistype some code, and cause massive damage quickly. Ensure that users have supervision and guidance, particularly when trying something new.

The stricter the better

- Specify everything in the environment you want to create as specifically as possible. Involve the developers in writing the IaC specifications for the infrastructure elements and runtime environments. Eliminating any holes in the code can help ensure smooth running.

[devops](#) [multi cloud](#)

Share



comments

leave a reply

Your email address will not be published. Required fields are marked *

[post a comment](#)

Tags

[ARIA](#)[Automation](#)[AWS](#)[Azure](#)[Chef](#)[Cloud](#)[Cloud Automation](#)[Cloud Computing](#)[Cloudify](#)[Cloud Management](#)[Cloud Native](#)[Cloud Orchestration](#)[Container Orchestration](#)[Deployment Automation](#)[DevOps](#)[Docker](#)[Edge Computing](#)[Hybrid Cloud](#)[Kubernetes](#)[Monitoring](#)[Multi Cloud](#)[Network Automation](#)[Networking](#)[Network Orchestration](#)[NFV](#)[NFV Orchestration](#)[ONAP](#)[Open Source](#)[OpenStack](#)[OpenStack Heat](#)[OpenStack Orchestration](#)[OpenStack Summit](#)[Orchestration](#)[PaaS](#)[Plugins](#)[Podcast](#)[Programming](#)[Python](#)[TOSCA](#)[TOSCA Orchestration](#)[vCPE](#)[VMware](#)[VNF](#)[VNF Management](#)[Workflow Automation](#)

Archives

Select Month

[To Multi-Cloud or Not To Multi-Cloud? Download the Whitepaper](#)[Read more](#)



Founded in 2012, Cloudify has robust financial backing from Intel Capital, VMware, BRM Group, Claridge and other leading strategic and financial investors. Cloudify has headquarters in Herzliya, Israel, and holds offices across the US and Europe. Cloudify sees the bigger picture of network service orchestration and specializes in network automation, providing service orchestration tools to claim the title of ultimate 'orchestrator of orchestrators'



Uses Cases

Infrastructure Automation
EDGE Networking
Multi Cloud Networking

Technologies

TOSCA
NFV
Jenkins
Kubernetes
Azure ARM
VMware
Terraform
Service Now

Company

About us
News
Careers
Contact us

Resources

Github
Podcast
Documentation
Whitepapers
Case Studies
Blog
Guide: Hybrid Cloud
Guide: Cloud Native
Transformation and Multi Cluster Management

