

DevOps in a Large Sports Organization

Ensuring compliance and implementing Infrastructure as Code (IaC) in an enterprise environment



Industry
Sports and Tournaments

year founded
Late 1910s

location
USA

The client is a major sports league in the U.S. They have about 150 in-house business apps and a multi-channel content management system (CMS) with a few dozen public-facing frontends.

The software is spread across several on-prem data centers and some AWS cloud servers.

Challenge

Before the client started working with our DevOps team, they had an on-premises infrastructure where software delivery was a manual process. Software delivery consists of two phases: preparing the infrastructure and actually delivering the software to the user. The weakest link in our client’s process was the first part: *infrastructure preparation/configuration*. It was slow and time-consuming because of the lack of automation.

A decision was made to extend/reinforce the DevOps team because:

- There was a need to provide an isolated security environment for a group of apps with PII (personally-identifiable information).
- There was a need to migrate one of the client’s in-house datacenters to a new location without disrupting their services or operations.
- There was a need for a mechanism that would allow the client to migrate any datacenter anywhere quickly and easily in the future

Solution

Problem I

PII (personally-identifiable information) is subject to laws and regulations that require it to be well-secured and auditable. Our client wanted to isolate apps with PII into a separate database/network.

Ensuring compliance

To insulate apps with sensitive data (and thus reduce potential attack surface), it was decided to move them to AWS cloud servers.

The rationale behind the decision was that it would take us an indefinite amount of time to build a high-security infrastructure on premises. Another reason we chose AWS was that it uses a pay-as-you go model, so there was little financial risk or commitment involved. AWS also audits its system on a regular basis and makes the results of those audits public. This would help us demonstrate compliance with applicable laws and regulations, if necessary.

Tooling-wise, we used the Terraform + Ansible combination for encoding and then migrating the infrastructure to AWS. The reason we did NOT go with AWS's native CloudFormation app was that we wanted a vendor-agnostic, universal solution that would let us switch vendors or move back on-premises at any time.

Problem II

Another pressing project was the migration of an in-house data center to a new location. To painlessly move the apps to a new home, our DevOps team needed to automate the client's app provisioning and delivery.

IaC and moving to new data center

IaC ([Infrastructure as Code](#)) is a “cookie cutter” approach that allows a DevOps engineer to “take measurements” off of a particular infrastructure, put it down in code, and recreate it **any** number of times across **any** number of servers, as needed.

So the first step was to describe the client's infrastructure **in code**. We introduced *Terraform* to codify the server creation process and upgraded *Puppet* (the configuration management tool) to the latest version. We also added Foreman on top of Puppet, making Puppet a single point of access to the latest system patches and updates on the Internet. Adding *Foreman* also enabled us to get visual statistics for Puppet configuration status changes over time. Our client's IT staff also installed *VMware* - a server virtualization technology - to enable our DevOps to “copy and paste” the infrastructure code onto the new servers.

Once we introduced automation and made it easy to replicate the client's infrastructure in a new environment, it became possible to start the migration to the new data center.

Problem III

By solving the above two problems, we ended up with an automation system that now permits our DevOps team to easily migrate the existing configuration elsewhere and quickly replicate it where needed.

Results

There is now a significant amount of DevOps automation in place in the client’s system. It used to take a full day to set up **one** server/VM (virtual machine). We can now set up **any number** of them in just hours. The next goal is to introduce an even greater degree of automation, probably through Kubernetes, once the migration to the new datacenter is complete. That would allow us to bring down the setup time from hours to minutes.

Technology stack

- AWS
- Ansible
- Docker
- Terraform
- Puppet
- Foreman
- Katello
- VMware

Time Span and Resources

Duration: N/A – not disclosed

Effort: N/A – not disclosed

<https://www.objectstyle.com/case-studies/devops>