ANSIBLE
by Red Hat®

ansible.com
+1 919.667.9958

WHITEPAPER

HOW TO DO DEVOPS WITHOUT LEAVING LEGACY BEHIND

How can IT organizations successfully apply DevOps and Agile methodologies to legacy environments?

FROM LEGACY TO DEVOPS

An increasingly apparent and large challenge in IT organizations is how teams can actively modernize software development and IT operations while still operating and maintaining their traditional applications and infrastructure. Often the approach is to merely draw a line in the sand, creating an arbitrary cut-off whereby new implementations make use of the much desired DevOps and agile methodology.

But what about the legacy environments?

First, let's be clear about something: Just because something is "legacy" doesn't automatically mean that it's outdated or no longer needed. There are plenty of organizations that generate huge percentages of revenues from "legacy" applications and systems. We're not using legacy in a negative context here. At all.

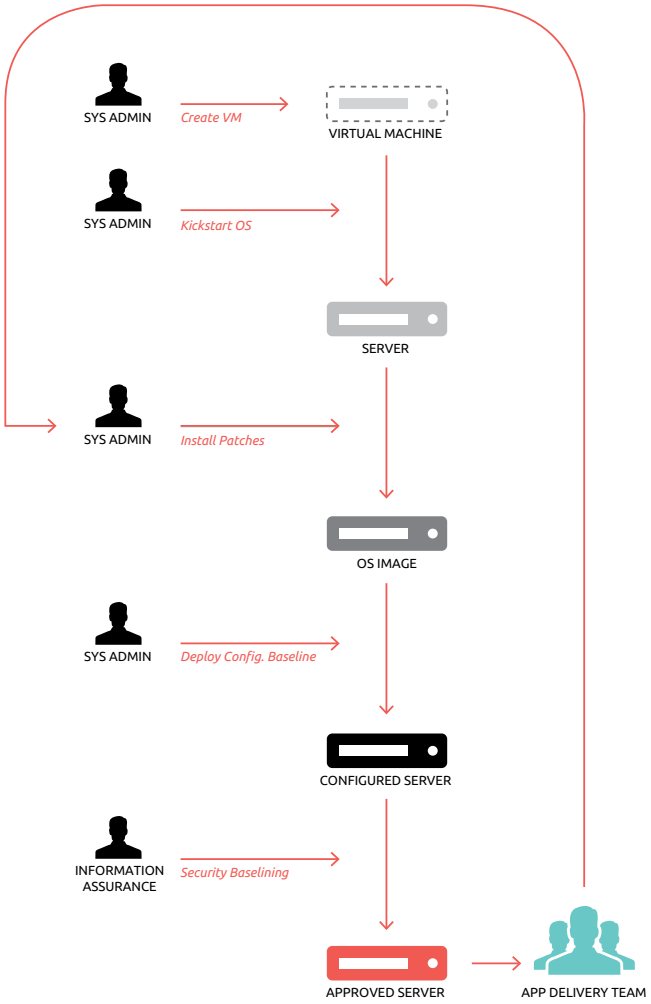
Also, many so-called legacy systems were deployed mere months ago—and on modern hardware, operating systems, and storage. For the sake of an agile organization, however, a legacy deployment or environment is anything that is not included in the new processes and approaches required for a DevOps-enabled organization. The question remains: how can IT organizations successfully apply DevOps and agile methodologies to existing traditional--or legacy--environments, and what are the benefits from doing this?

"Every change we make gets deployed in Ansible...except for the first time. The first time we figure out how to do it manually and then we'll make a playbook to do it"

MIKE REGAN
SR. CLOUD OPERATIONS ENGINEER, SPLUNK

START WITH THE INFRASTRUCTURE

Regardless of the type and variety of applications in an enterprise IT environment, there are likely many commonalities in the operating system and infrastructure components. Manual OS build processes typically require significant admin-hours to deliver a single build. Additionally, the reliability of the result is a totally dependent on an admin's experience, skill and ability to precisely follow a set of complicated directions. Then that admin needs to repeat this process over and over again for each of the systems in the environment. There are other teams involved with build processes as well. Before the application delivery team can do their job, the information assurance team needs to validate that the correct security baseline has been applied. The more teams that need to touch a system, the longer it will take to implement, and the more likely you are to encounter delays and errors. The system and server build process is thankfully well-understood, and easily automated. Regardless of your current OS build process (core build at provisioning, gold disk, etc.), there are likely many commonalities across your environment. Automating this build and configuration process will enable you to repeatedly deploy OS images on-demand, and, with the right tooling, manage those existing builds as easily as you create new ones so that systems will always look the same. Once the OS build and management process have been automated, making these automations available to other teams becomes relatively trivial. Typical consumers of OS builds, such as development, testing, and QA teams, can trust they're always working with the proper base OS configuration while building their applications. Merely building systems, however, ignores the much harder part of the problem: keeping them updated through their lifecycles. How can you ensure that these meet the current baseline requirements as well? This is again where your choice in automation tooling makes a difference. A key problem with the traditional virtual machine (VM) lifecycle approach is that in the past, it has required a separate process for maintenance of existing VMs... and many provisioning tools have a difficult time updating and making changes to existing systems in a live-running environment. Thankfully, it's relatively straightforward to create and apply a continuous deployment

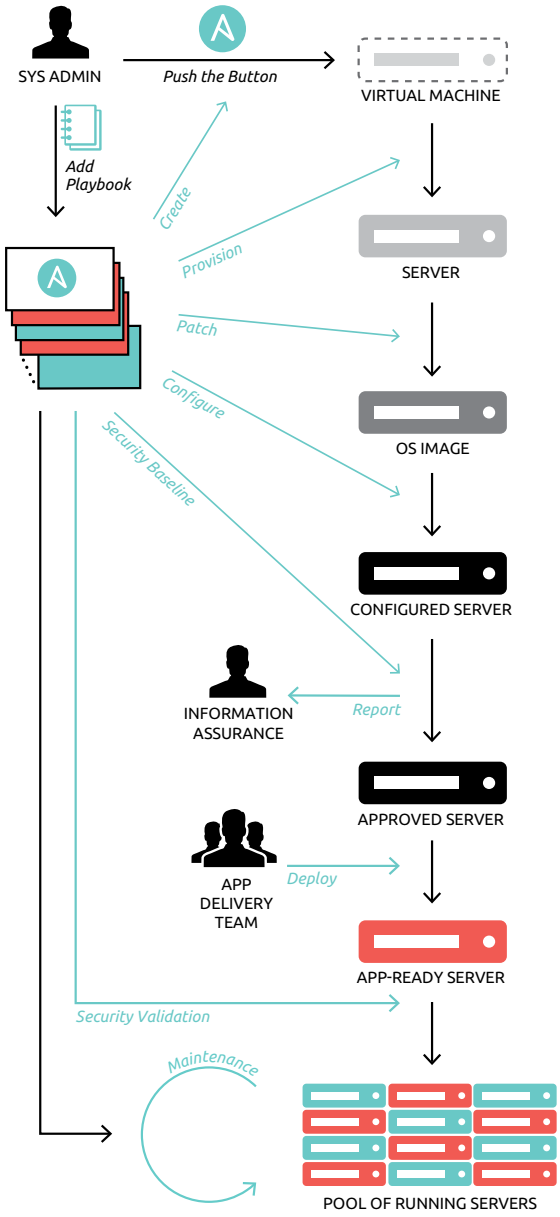


By using Ansible as this automation tool, the same playbooks used to build systems can be used to ensure that existing systems meet the current specification.

methodology to the OS environment. By automating the server build and maintenance process, it's possible to drastically decrease or totally eliminate manual steps in the server delivery process. An added benefit is also that by using Ansible as this automation tool, the same playbooks used to build systems can be used to ensure that existing systems meet the current specification. Even before addressing the application layer, automating the creation, delivery, and management of the OS and infrastructure layer will save considerable amounts of time, and provide other material benefits to an organization's extended teams.

ABOUT ANSIBLE

Ansible, an open source community project sponsored by Red Hat, is the simplest way to automate IT. Ansible is the only automation language that can be used across entire IT teams – from systems and network administrators to developers and managers. Ansible by Red Hat provides enterprise-ready solutions to automate your entire application lifecycle – from servers to clouds to containers and everything in between. Ansible Tower by Red Hat is a commercial offering that helps teams manage complex multi-tier deployments by adding control, knowledge, and delegation to Ansible-powered environments.



READY TO AUTOMATE? **info@ansible.com** • **+1 919.667.9958** • **ansible.com**

redhat.