Lecture Notes: *Configuration Management*

Provisioning and Configuration Management

These two terms are often used interchangeably, but have different meanings

Provisioning: is the installation of those things which are necessary for infrastructure to operate; typically happens at the beginning.

Configuration Management: is an ongoing process which seeks to standardise configurations across a number of different environments.

Provisioning

This is a broad term which encompasses several different contexts:

- server
- network provisioning
- user provisioning
- service

Configuration Management

Originated in the 1950's to track, often, physical infrastructure configuration in the US Department of Defence.

- Originally, the focus was on *documentation* of systems over time
- Has evolved to defining a desired state which can be automatically enforced.

This evolution has been enabled by the progressive adoption of *software* over *hardware*.

Relationship to DevOps

Why is Configuration Management is a core concept of DevOps?

Answers

- it underpins the paradigms of Continuous Integration and Continuous Deployment
- it's a step on the way to Immutable Infrastructure

Immutable Infrastructure

Mutable Infrastructure is infrastructure which changes in place.

- Mutable Infrastructure is *artisanal infrastructure*: infrastructure "made in a traditional or non-mechanized way" by "someone skilled in a craft, especially a craft that involves making things manually."
- Mutable Infrastructure scales to high-volume production just as well as other artisanal manufacturing techniques.

Immutable Infrastructure is infrastructure that does not change *once it is deployed*.

- Immutable Infrastructure is _industrialised infrastructure+: standardised and capable of mass production.
- Immutable Infrastructure scales to high-volume production just as well as other mechanised/automated manufacturing techniques.

Short reference:
https://www.oreilly.com/radar/an-introduction-to-immutable-infrastructure/

Contemporary Technologies

There are three main technologies in use at present:

- Ansible https://www.ansible.com/
- Puppet https://puppet.com/
- Chef https://www.chef.io/

But, despite the claims on their respective websites, these are horses which are best suited to particular courses.

Puppet

Puppet is an agent-based automation tool, running a client-server architecture.

Pros

- Mature solution
- Good GUI
- Support for all major OS's
- Easy install

Cons

- Slow-ish to respond and address customer concerns
- Ruby-based, performance questionable compared to Python-based CM tools
- Soon all customers must learn the Puppet DSL

From: https://www.simplilearn.com/ansible-vs-puppet-the-key-differences-to-know-article

Chef

Chef is similar to Puppet, in that it is an agent-based tool.

Pros

- Manages large enterprises
- Known for reliability
- GUI is good
- Documentation is strong

Cons

- A Ruby programmer is needed to manage the Chef tool
- High maintenance is required
- communication process is very slow

From: https://www.educba.com/ansible-vs-puppet-vs-chef/

Ansible

Ansible is an *agentless* automation tool for provisioning and configuration management.

- Based on Python and OpenSSH
- Control node connects to remote machines temporarily
- Copies modules to remote, deleting them when the tasks have been completed.
- YAML used to describe configurations
- Simpler to install and use than Puppet or Chef

Pros

- Excellent performance, agentless install and deploy
- Low overhead, playbook based
- Based on ubiquitous Python language

- CLI accepts commands in almost any language

Cons

- Still very new; not yet tried and tested by many
- No support for Windows
- GUI a work in progress

Playbooks and Idempotency

A Playbook is, in its simplest form, a list of tasks to execute to define and enforce a specific configuration.

- The "Playbook" term appears to be a reference to American Football
- Ansible's Idempotency is the

Agentless vs. Agent-based Architectures

Simplicity & low resource demands vs. greater features and native power.

But, what about these remote servers/machines? How can we experiment?

VirtualBox

VirtualBox https://www.virtualbox.org/

> VirtualBox is a powerful x86 and AMD64/Intel64 virtualization product for enterprise as well as home use.

Documentation: https://www.virtualbox.org/wiki/Documentation

Vagrant

Vagrant https://www.vagrantup.com/ is a product from HashiCorp

> Vagrant is a tool for building and managing virtual machine environments in a single workflow. https://www.vagrantup.com/intro/index

- Vagrant manages VirtualBox boxes
- Automates many aspects of configuration
    - SSH keys
    - IP addresses
    - Provisioning
