ANSIBLE PARA DEV+OPS

DÍA 4

LO QUE VEREMOS HOY

- Testing: ¿que es? ¿Porqué es importante en la filosofía DevOps?
- Unit testing
- Integración de técnicas de test con Ansible
- Test Driven Development
- Labs: Testing

TESTING

- Todo código debe ser probable de forma independiente y sin afectar a entornos en uso
- Todo código debe ser validado en entornos no productivos antes de ser desplegado en un entorno productivo
- Ansible aunque no lo parezca, es código

TESTING Y DEVOPS

- Mantra DevOps -> Filosofía LEAN

Sanjeev Sharma, DevOps SME, Agile evangelist and author at IBM, mentions continuous testing as an integral part of DevOps, in his blog on understanding DevOps.

> Continuous Integration and delivery are both (almost) meaningless without continuous test. Not having motoring and hence, not knowing how the application is performing in production makes the whole process of DevOps moot.

UNIT TESTING

- Unit testing es el proceso de implementar un test para cada función independiente del código
- Unit testing implica probar solo el código implementado, no las integraciones con librerías de terceros, ni las integraciones en despliegue
  - Para ello se usan mocks que simulan APIs remotas, bases de datos…
- Es por ello que en Ansible unit testing no tiene la menor importancia, dado que por su forma de ejecutarse todo depende de librerías de terceros o de módulos que ya han sido testeados por Ansible
- En el mundo de Ansible unit testing solo tiene sentido para:
  - Custom Modules
  - Dynamic Inventories
  - Filtros custom de Jinja

INTEGRACIÓN DE TESTING CON ANSIBLE

- En Ansible se pueden implementar métodos de testing de integración
  - Una vez ejecutado el código de Ansible validan que los resultados son los esperados
  - Generan un error si no es así
- Existen varios métodos / lenguajes :
  - Rspec
  - ServerSpec
  - InSpec
- Estos métodos pueden servir para validar que los servidores cumplen ciertas condiciones (compliance)

INTEGRACIÓN DE TESTING CON ANSIBLE

- Para el laboratorio, usaremos el software Test Kitchen con la integración de Ansible y Docker, que nos permitirá:
  - Usar el lenguaje InSpec para definir tests
  - Probar la ejecución de nuestros playbooks en un docker desechable
  - Validar que el resultado es el adecuado
  - Automatizar el proceso de testing mediante herramientas de CI como Jenkins, Travis…

TEST DRIVEN DEVELOPMENT

- Metodología de desarrollo que se basa en escribir el test antes de escribir el código
- El test se escribe en un lenguaje más de alto nivel (puede ser escrito por alguien que no sea el que escribirá el código)
- La ejecución del test nos sirve para validar si el desarrollo está completado

TDD EJEMPLO: INSTALAR Y CONFIGURAR NGINX

- Para instalar y configurar nginx las acciones son las siguientes:
  - Instalar la paquetería de nginx
  - Adaptar los ficheros de configuración
  - Arrancar nginx
  - Asegurarse que nginx arranca tras un reinicio

TDD EJEMPLO: INSTALAR Y CONFIGURAR NGINX

- Formato del test en lenguaje InSpec:

```
describe package ‘nginx’ do
        it { should be installed }
done
describe file ‘/etc/nginx/nginx.conf’ do
        its(‘content’) { should match /listen=0.0.0.0/ }
done
describe service ‘nginx’ do
        it { should be_installed }
        it { should be_running }
end
```

TDD EJEMPLO: INSTALAR Y CONFIGURAR NGINX

- Formato de la implementación en ansible

```
- name:“install nginx”
  yum:
    name: nginx
    state: present
- name: “configure nginx”
  template:
    src: nginx.conf
    dest: /etc/nginx/nginx.conf
- name:“start nginx”
    service:
      name: nginx
      state: started
      enabled: yes
```

UN POCO DE DEBATE ANTES DEL LAB

- ¿No tiene Ansible sus propias herramientas de test?
  - No, la postura oficial de Ansible es que puedes crear un Playbook de test para validar que se ha implementado todo como debe
    - Ejemplo: https://github.com/blueboxgroup/ursula/tree/master/playbooks/tests/tasks
    - Ejemplo: https://github.com/geerlingguy/ansible-role-mysql

UN POCO DE DEBATE ANTES DEL LAB

- ¿Puedo probar lo que hará un Playbook antes de ejecutarlo?
  - Si, pero no completamente. Se lanza el comando ansible-playbook con el parámetro "--check"
  - También podemos usar el parámetro "--diff" (con check o sin check) para comparar el antes y el después de un run
- Pensaba que eso era testing…
  - Check no comprueba el resultado final, solo mira que podría cambiar
  - Algunos módulos no soportan "check" (especialmente los de infraestructura)

UN POCO DE DEBATE ANTES DEL LAB

- ¿qué es el lint?
  - Lint es una técnica por la cual se pasa un código por un validador que asegura que la sintaxis sea correcta y que se respeten ciertas reglas de código
  - Casi todos los flujos de CI pasan un proceso de lint antes de realizar los tests
- ¿Ansible tiene herramienta de lint?
  - No, hay un script de terceros que lo hace (https://github.com/willthames/ansible-lint/blob/master/README.md)
  - Lo que si integra es un test de sintaxis mediante el parámetro syntax check

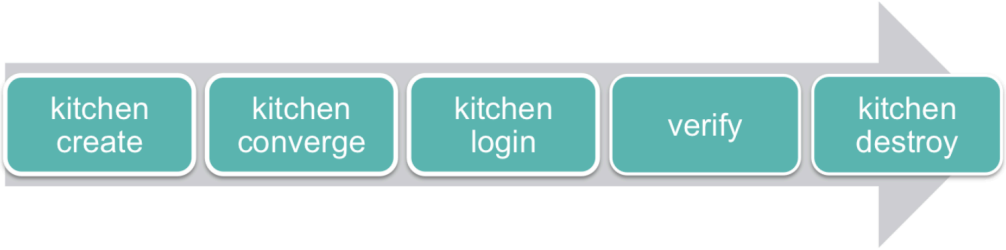
LABS

- Configurar Test Kitchen
- Integrar Test Kitchen con Ansible y Docker
- Crear usando TDD un rol que instala nginx y muestra una página web con la palabra “Hello World”

No subestimes este lab ;)

LABS HELPER

Ciclo de vida de Test Kitchen



kitchen create	In this step, Test Kitchen creates an instance of your virtual environment, for example, a CentOS virtual machine.
kitchen converge	In this step, Test Kitchen applies your cookbook to the virtual environment.
kitchen login	In this step, Test Kitchen creates an SSH session into your virtual environment.
verify	In this step, you manually verify that your virtual environment is configured as you expect.
kitchen destroy	In this step, Test Kitchen shuts down and destroys your virtual environment.