Infraestructura I
Profesor: David Pignalberi

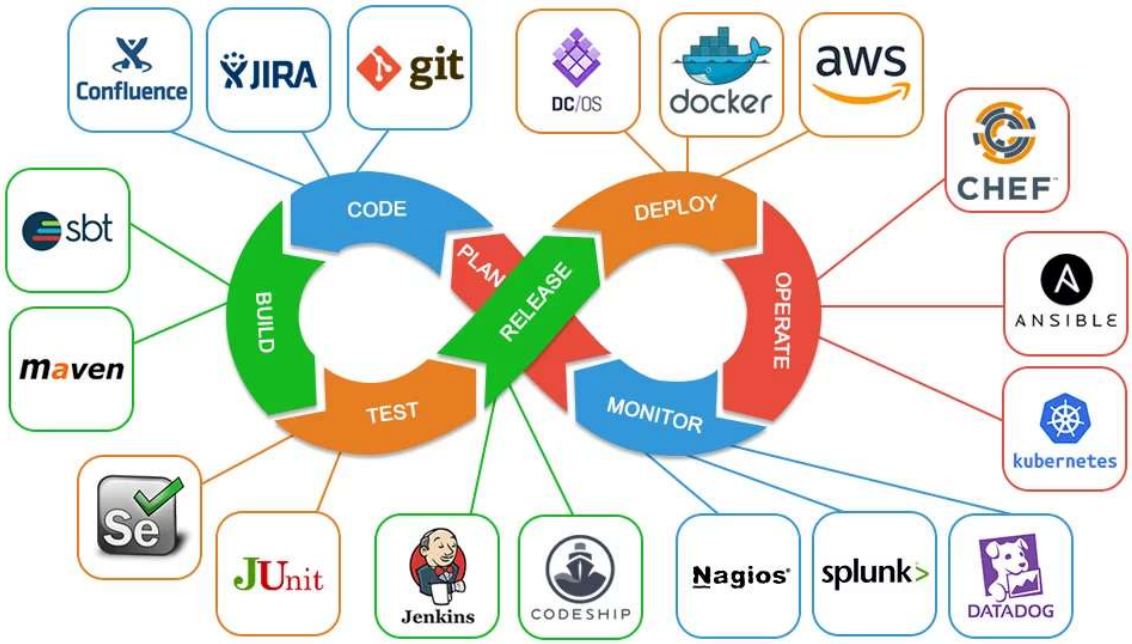
Módulo 2: Automatización

Sesión 2: Automatización
Marzo 3 de 2022

Automatización de la Infraestructura
Consiste en el uso de sistemas de software para crear instrucciones y procesos repetibles a fin de reemplazar o reducir la interacción humana con los sistemas de IT.

Las 5 tareas más comunes en TI para automatizar:
- Aprovisionamiento de las áreas de TI.
- Gestión de Configuración.
- Seguridad y cumplimiento.
- Organización de la nube
- Implementar aplicaciones

Configuración y Mantenimiento del Sistema
DevOps Engineer: Encargado de automatizar los procesos del área de Development y Operations.



Infraestructura y Servicios:
- Servidores legay o cloud providers.

- Cloud Providers: AWS, Google Cloud, Azure
- Servicios: load balancing, backup, cluster, security, etc.

Manejo del código:
- El Gestor de versionado e integrador de código más utilizado es Git.
- Los alojamientos de repositorios más conocidos son GitLab, Github, Bitbucket, Gitea, etc.
- Automatizar el deploy del código con CI/CD: Jenkins, GitActions, JetBrains.

Contenedores:
- Permiten la virtualización de ambientes de trabajo compatibles transportables, totalmente configurados para que el código funcione en todos los equipos. El más popular es Docker.
- Cuando se manejan muchos contenedores, se requiere migrar a un cluster de contenedores, dirigido por un orquestador de contenedores. Ej: Kubernetes, Docker Swarm.

Ambientes de Trabajo:
- Ambientes: Desarrollo, Producción, Testing
- Los ambientes más populares son: Ansible, Chef, Puppet, Terraform

Monitores de Red:
- Supervisión de los dispositios de la red y de los servicios, y programación de alertas en caso de que suceda algún cambio. Ej: Nagios, Prometheus, Icinga2, DataDog.

Lenguajes de Scripting:
- Según el S.O.: Bash (Linux), PowerShell (Windows)
- Lenguajes independientes del S.O.: Python, Ruby, Go

Virtualización
La virtualización permite mejorar la agilidad, la flexibilidad y la escalabilidad de la infraestructura de IT, al mismo tiempo que proporciona un importante ahorro de costos.

Ventajas:
- Mayor movilidad de las cargas de trabajo.
- Aumento del rendimiento.
- Menos esfuerzo en los upgrades/updates del sistema.
- Mejor disponibilidad de los recursos o la automatización de las operaciones.

Componentes de Virtualización:

- Máquinas virtuales

- Administrador de máquinas virtuales.

Administra todos los recursos físicos y virtuales de los guest que son las instancias virtuales que se crean para usos específicos. Desde el mismo management se puede establecer clustering con otras virtual machines manager para tener alta disponibilidad y tolerancia a fallos. Además, administra todos los recursos virtuales de las virtual machines.

- Sistemas Operativos base.
S.O. encargado de administrar los dispositivos físicos (hardware) y proveer una capa de abstracción a los entornos virtuales.

- Hardware (servidores físicos)

Virtual Box:
Una plataforma de virtualización utiliza las características necesarias de hardware y software para permitirnos ejecutar múltiples máquinas virtuales en una misma computadora física. Virtual Box: plataforma de virtualización multiplataforma.

Vagrant:
Utilidad desarrollada por HashiCorp que permite controlar el ciclo de vida de las máquinas virtuales, automatizar su configuration, generar imágenes con software preinstalado y gestionar ambientes de desarrollo de manera sencilla.

Sincrónico

Ejericio 1. Debian

Ejercicio Debian:

Máquina virtual:
superusuario: maosuper

Nombre Usuario: Mauricio Pineda Angel
usuario: mauricio
contraseña: maouserdebian

Para pasar de un usuario a superusuario:
su -

1. Paso:
2. Instalar servidor web
3, Crear servidor SSH. LogIn remoto a ese servidor

Para ver la IP:
ip a
ip address

Usar la segunda que aparece
ip: 192.168.1.61

….

Para cerrar la sesión en ssh:
logout

Si la conexión es por wifi, cambiar el tipo de adaptador y activar el modo promiscuo. (No fue necesario).

Ejercicio 2

Box de vagrant:
app.vagrantup

En esta página, se encuentran los boxes para diferentes hipervisores para máquinas virtuales:
https://app.vagrantup.com/boxes/search

Para descarga el box de Debian:
- Crear una carpeta

- En Powershell, ir a la ruta del carpeta creada. El siguiente comando descarga el box:
- **vagrant box add debian/buster64**

Con el comando **vagrant up** se lee el archivo vagrantfile y se crea la máquina virtual automáticamente.

Para conectarse al servidor:
vagrant ssh server

Cuando solicita interfaz de red, especificar la No. 1

which inbterface should the network bridge to? 1

Sesión 4 (No hubo sesión 3). Shell Scripting – Parte 1
Marzo 3 de 2022

La consola de Linux

La **interfaz de línea de comandos** (CLI) es un método de comunicación entre usuario y máquina que acepta instrucciones del usuario a través de líneas de texto.
La herramienta que posibilita la función de interfaz de usuario se denomina **shell**.

Tipos de Shell

Bash, Bourne Shell (sh), Korn Shell (ksh), C Shell (csh) y su evolución TC Shell (tcsh).

Bourne-Again Shell (BASH)

Versión actualizada de la Bourne Shell original. Utilizada ampliamente en la comunidad de código abierto.

Para elevar privilegios, se usa la palabra sudo. Ej:
sudo service cron start

Comandos útiles en la terminal de Linux

ls: Muestra los archivos y carpetas en un directorio, omitiendo los ocultos.
ls -a: Muestra los archivos y carpetas en un directorio, incluso los ocultos.
ls -l: Muestra los archivos y carpetas en un directorio, en forma de lista, con información más detallada

mkdir: Permite crear un directorio con un nombr ey una ruta especificada
rmdir: Permite eliminar un directorio vacío
rm: Permite borrar archivos y directorios que no estén vacíos
rm -r: Permite borrar un directorio de forma recursiva (elimina todo su contenido)

cp: Permite copiar archivos y directorios. Al mismo tiempo se pueden renombrar las copias
mv: Permite mover o renombrar archivos

cat:
cat >dir1/archivo1.txt
Abre el archivo1.txt, permitiendo editarlo. Con la combinación CTRL+D termina la edición y se guarda el contenido.

Invocando el comando sin el símbolo “>”, se muestra por pantalla su contenido. Se puede usar con el modificador -n, para numerar las líneas y con el -b, con el propósito de no mostrar las líneas en blanco.

more
more /var/log/dpkg.log
Útil para imprimir por pantalla el contenido de un archivo de texto. Adecuado para leer archivos largos.

nano
Editor de textos para la terminal.

grep
Permite buscar un patrón definido en un archivo de texto. Acepta comodines como *, y puede recorrer recursivamente con el modificador -r.

Ej:
grep "Digital House" * -r
Busca la cadena “Digital House” en todos los archivos, de manera recursiva

tee
Lee una entrada estándar y la escribe en la salida estándar y en uno o más archivos.

Ej:
ls -l | tee listado.txt
Muestra el directorio y lo guarda en un archivo.

ls -l | tee -a listado.txt
Se agrega contenido al archivo, sin pisar lo anterior.

Repasar otros comandos

Sincrónico

vagrant no corre:
- Instalar Microsoft C++ Redistributable
- Desinstalar vagran t e instalar una versión anterior 2,2,18

En Linux, la consola inicia siempre en nivel 1.
Cada nivel permite realizar diferentes acciones.

En Windows, se aplicaba F8 para entrar en símbolo del sistema.

Niveles de ejecución en Linux:

Nivel de ejecución	Nombre o denominación	Descripción
0	Alto	Alto o cierre del sistema (Apagado).
1	Modo de usuario único (Monousuario)	No configura la interfaz de red o los demonios de inicio, ni permite que ingresen otros usuarios que no sean el usuario root, sin contraseña. Este nivel de ejecución permite reparar problemas, o hacer pruebas en el sistema.
2	Multiusuario	Multiusuario sin soporte de red.
3	Multiusuario con soporte de red.	Inicia el sistema normalmente sin GUI.
4	Multiusuario con soporte de red.	No es igual que el 3.
5	Multiusuario gráfico (X11)	Similar al *nivel de ejecución* 3 + display manager.
6	Reinicio	Se reinicia el sistema.

sudo init 0: Hace referencia al nivel 0 y apaga la máquina.

sudo: Eleva los permisos del usuario a administrador.

vagrant ssh server: Entra a la máquina virtual (revisar)

con Ctr*C se sale del comando cat

vi: permite ver un archivo

cat no sirve para editar el archivo. Se abre con nano o vi.

En vi:
- Con i se habilita la edición
- con Esc se sale de la edición
- Con Ctr+ZZ se guarda y se sale del archivo1

vim = vi

clear: Limpia la pantalla

cat –help: Muestra los usos del comando cat.

Curl www.google.com

curl cheat.sh

cheat.sh/vim

chmod: Cambia los permisos de escritura y lectura en los archivos

curl cheat.sh/chmod: Muestra todas las opciones del comando. Se pueden utilizar representaciones numéricas

chmod 777 nombre_archivo: Cambia los permisos de nombre_archivo

Administrar servicios:
systemctl status apache2: Muestra el estado del servicio

man systemctl: Revisar el uso

Para consultas en general, ver reddit

systemctl start stop apache2: Apaga el servicio

systemctl start stop apache2: Inicia el servicio

netstat: Obsoleto

ss -plnt: Muestra todos los puertos que están siendo utilizados. No se requiere instalar nada.

Cambiar el puerto para un servicio determinado (ej: Apache): Se requiere modificar la configuración de apache (buscar archivo de configuración)

Para los scripts, se puede usar gitbash en lugar de la máquina virtual.

Sesión 5. Shell Scripting - Parte 2
Marzo 9 de 2022

Bash es una interfaz que interpreta las órdenes que el usuario le hace al sistema ejecutado en una consola de Unix.
Un archivo Script permite hacer persistentes una lista de tareas a realizar por el sistema operativo.
Bash es un lenguaje de scripting

Tipos de variables

Globales o de entorno
Se escriben con mayúsculas.

Para ver las variables de entorno que se están usando y que están cargadas en la sesión.
env, printenv

Pata declarar una variable global:
export NOMBREVARIABLE=valor

Para acceder a la variable:
$NOMBREVARIABLE

Variables de usuario o locales
Puede ser accedida solo por el usuario y la sesión en la que fueron creadas.

Para declarar una variable local:
nombrevariable=valor

Para acceder a la variable:
$nombrevariable

Estructuras de control

Sentencia if-then

// Ejemplo de condicional:

```
#!/bin/bash

if whoami; then
        echo "It works"
fi
```

Sentencia if-else

```
#!/bin/bash

ping -c 1 8.8.8.8
if [ $? -ne 0]; then
echo “No está en la red”
else
echo “Sí está en la red”
fi
```

Comparaciones numéricas

Para comprobar si number 1 es igual a number 2:
number1 -eq number2

Para comprobar si number 1 es mayor o igual que number 2:
number1 -ge number2

Para comprobar si number 1 es mayor que number 2:
number1 -gt number2

Para comprobar si number 1 es menor o igual que number 2:
number1 -le number2

Para comprobar si number 1 es menor que number 2:
number1 -lt number2

Para comprobar si number 1 es diferente a number 2:
number1 -ne number2

Ejemplo:

```
#!/bin/bash
num=11
if [ $num -gt 10]; then
        echo "$num is bigger than 10"
else
        echo "$num is less than 10"
fi
```

Comparaciones de cadenas

Para comprobar si string1 es idéntico a string2:
string1 = string2

Para comprobar si string1 no es idéntico a string2:

string1 != string2

Para comprobar si string1 es menor que string2:
string1 < string2

Para comprobar si string1 es mayor que string2:
string1 > string2

Para comprobar si string1 es mayor que cero:
-n string1

Para comprobar si string1 tiene una longitud de cero:
-z string1

Cálculos matemáticos:

Ejemplo:

```
#!/bin/bash
var1 = $(( 5 + 5 ))
echo $var1
var2 = $(( $var1 * 2 ))
echo $var2
```

Sincrónico

Si se crea el archivo sh con sudo, se requieren privilegios de sudo para cambia rlos permisos

sudo nano script.sh

La indentación en el .sh es opcional.

./script.sh // Permiso denegado

Se requiere chmod
chmod +x script.sh // Permiso dengado, ya que el archivo se creó con sudo

sudo chmod +x script.sh

jq:
Permite consultar JSON

Sesión 6.
Marzo 10 de 2022

let: Evalúa expresiones aritméticas.

Consultar el pelao nerd (youtuber)

Sesión 7. Powershell
Marzo 14 de 2022

Falta

Sincrónico

Duda sobre vagrant:
Hola, buenas noches, solicito ayuda acerca del siguiente problema:
No he podido reproducir el ejercicio de la clase 2, en el que se utilizaba vagrant para crear una máquina virtual y se instalaba apache2, ya que el comando vagrant box add debian/buster64 no anda.
Por recomendación del profe David Pignalberi, actualicé la versión de Visual C++ Redistributable, y reemplacé la versión 2.2.19 de vagrant que tenía instalada por una versión anterior, específicamente la 2.2.18, pero sigue sin funcionar.
Estoy usando Windows 10 y Virtual Box 6.1.22. Este es el mensaje de error que aparece:

Se puede forzar la descarga del box incluyendo en el comando la flag --insecure:

sin embargo, al tratar de ejecutar el comando vagrant up, vuelve a aparecer el mismo error.

Examen: Jueves 24 de marzo. Clase 12.
5 multiple choice. + 5 a desarrollar.

En Powershell, hay un cmdlet que hace las veces de curl. Se llama invoke.

Los cmdlets tienen en general el mismo formato: palabra-palabra.

En general, PowerShell puede considerarse m+as potente que Bash.

Módulos: Paquetes que permiten ampliar la funcionalidad de la cli. Hay una gran librería de módulos, que se expande continuamente.

Documentación de PowerShell
https://docs.microsoft.com/es-es/powershell/

Al ejecutar el script en Powershell, no se reconocen los caracteres especiales. Con Powershell Core no se tiene este error.

Sesión 8. Python
Marzo 16 de 2022

Python
Lenguaje de programación versátil, multiplataforma y multiparadigma que se destaca por su código legible y limpio

- Lenguaje open source
- Interpretado
- Multiparadigma
- Multiplataforma
- Al instalar Python se añade le comando python y se instala el intérprete de python correspondiente

Historia
- 1991: Nace python
- 2000: python 2.0
- 2001: Se funda la Python Software Foundation
- Python 3.4.0

Aplicaciones:
- IA
- Aplicaciones web
- Big Data
- Scraping
- GUI

Principales Industrias que usan Python:
- Inteligencia Artificial
- Big Data
- Data Science
- Frameworks de pruebas
- Desarrollo Web

Variables en Python
En algunos lenguajes de programación, las variables se pueden entender como "cajas" en las que se guardan los datos, pero en Python las variables son "etiquetas" que permiten hacer referencia a los datos (que se guardan en unas "cajas" llamadas objetos).
Python es un lenguaje de programación orientado a objetos y su modelo de datos también está basado en objetos. Para cada dato que aparece en un programa, Python crea un objeto que lo contiene.

Cada objeto tiene:
- Un identificador
- Un tipo de datos
- Un valor

Las variables en Python se crean cuando se definen por primera vez, es decir, cuando se les asigna un valor por primera vez.

Estructuras de control en Python

Condicionales

```
if (condicion1):
    codigo_si_condicion1
elif (condicion2):
    codigo_si_condicion2
else:
    codigo_en_otro_caso
```

Condiciones en Python

```
a == b
a > b
a < b
not
and
or
```

While

```
while vuelta < 10:
    print("vuelta " + str(vuelta))
    vuelta = vuelta + 1
```

For

```
for vuelta in range(1,10):
    print("vuelta" + str(vuelta))

for item in range(5):
        print(item) // Imprime los números del 0 al 4
```

La función range devuelve un objeto de rango que representa la secuencia 0, 1, 2, 3, 4.

```
coches = ('Ferrari', 'Tesla', 'BMW', 'Audi')
for coche in coches: // Funciona para listas, tuplas y cadenas
    print(coche)

coches = ('Ferrari', 'Tesla', 'BMW', 'Audi')
for i, coche in enumerte(coches):
    print(str(i) + " - " + coche)
```

Listas

```
listaCuadrados = []
for x in range(1,11):
    listaCuadrados.append(x*x)
```

Es equivalente a:

```
listaCuadrados = [x*x for x in range(1,11)] // Comprensión de lista
```

Agregando una condición:

```
listaCuadrados = [x*x for x in range(1,11) iif x%2 != 0] // Agrega solamente los cuadrados de los números impares
```

Ejemplo:
[letra.upper() for letra in 'estructuras' if letra not in 'aeiou'] // Retorna ['S', 'T', 'R', 'C', 'T', 'R', 'S']

Sincrónico

Para tomar captura del código en VSCode: Polacode (Extensión de vsc).

Sesión 9.
Marzo 17 de 2022.

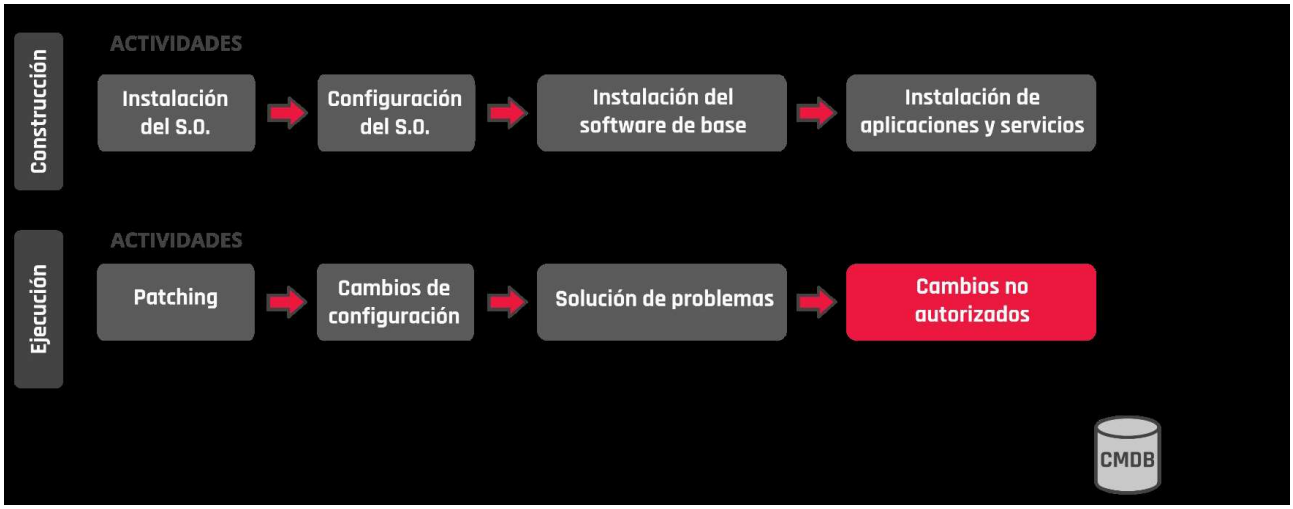
Sesión 10. Configuration Management
Marzo 21 de 2022

ITIL: Information Technology Infrastructure Library
Describe la gestión de cambios como el proceso de controlar y gestionar un cambio a lo largo de todo su ciclo de vida con el objetivo de minimizar el riesgo.

Cambio
Es la modificación o eliminación de cualquier cosa que pueda afectar directamente o indirectamente los servicios.
Ej: Reemplazo de hardware, instalación de software en un servidor o modificación de una configuración de un sistema.

Configuration Management
Es el proceso que permite gestionar los cambios de configuración de nuestros servicios informáticos -ya sean de hardware o de software-, permitiendo a la organización mantener un registro histórico y a su vez aplicar controles. Cada uno de los activos informáticos en el contexto de este proceso se los conoce como **Configuration Items** (CI) y se almacenan en lo que es llamado CMDB (**Configuration Management Database**).

Ciclo de vida de un servidor



Etapa de construcción
Conjunto de pasos que pueden ser automatizados, documentados o sencillamente conocimiento común del área de IT, que permiten la puesta en marcha de un servidor y registrarlo en la CMDB como un nuevo CI.

Etapa de Ejecución
Todo lo que suceda con relación al activo y cambios que puedan suceder en él.

Etapa de construcción de un servidor:
- Instalación del S.O.
- Configuración del S.O.
- Instalación del software de base
- Instalación de aplicación y servicios

Etapa de ejecución de un servidor
- Patching: Instalar actualizaciones del S.O.
- Cambios de configuración
- Solución de problemas
- Cambios no autorizados. Cualquier cambio o actividad realizado en el servidor y que no queda documentado en un tocket, ya sea como un cambio o un incidente.

Configuration as Code
- Gestionar el parque tecnológico 'como ganado' no sería posible sin la utilización de un sistema de configuration management.
- Cualquier cosa definida como código puede ser automatizada.
- El código puede ser sometido a pruebas.
- CaC es un paso anterior a habilitar Self-Healing y Self-Remediation.
- Es compatible con el proceso de Change Management de ITIL, ya que las modificacione son suceden en los activos, sino en un repositorio que soporta versionado. Los cambios pueden ser testeados y desplegados en ambientes bajos para luego aplicarlos en producción.
- No reemplaza el proceso de Configuration Management.

Procesos Tradicionales:
- Imperativos

Proceso Modernos:
- Declarativos. Los cambios pueden ser guardados en un sistema de control de versiones (GitOps).

¿Cómo llega la CaC a un sistema?
- Monitorear el repositorio Git.
- Pipeline

Pipeline
Uutilizados en el proceso de despliegue de software. Tiene dos etapas:

Build: Testaer y compilar la aplicación y construir el artefacto que será distribuido e instalado.
Release: Enviar al sistema de Configuration Management el resultado del proceso de Build.

Herramientas de Configuration Management:
- Chef
- Puppet
- Powershell DSC
- Ansible

Sincrónico

AXELOS: Buenas prácticas en ITIL. Certificación.

Configuration Management permite guardar un registro de los cambios de configuración.

Rangus: Antes de hacer un cambio en un servidor ? Consultar.

Hoy en día, en Infraestructura, más que Servidores, se administran Servicios.

Modo Pull / Modo Push. Revisar. Ansible funciona en modo Push.

RedHat: Ansible. Ansible no crea infraestructura directamente en la nuble. Con Terraform sí.
HashiCorp: Vagrant, TerraForm

Sesión 11. Configuration Management – Ansible
Marzo 23 de 2022

Ansible

Es un software de gestión de la configuración automática y remota, que permite centralizar la configuración de numerosos servidores, dispositivos de red y cloud providers de una forma sencilla y automatizada

- Herramienta open-source de configuration management y de aprovisionamiento, similar a Chef, Puppet o Salt.
- Usa SSH para conectarse a los servidores y ejecutar las tareas de configuración. Permite controlar y configurar nodos desde un servidor central.
- Fue comprado por Red Hat en 2015.

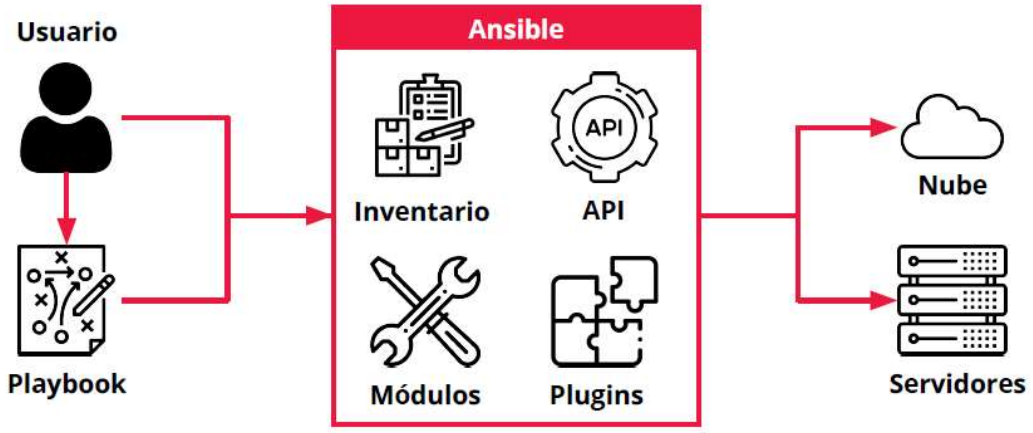
Ventajas

- No usa agentes
- Idempotente. Solo se hacen configuraciones si son necesarias y se pueden aplicar d emanera repetible sin provocar efectos secundarios.
- Es declarativo. A diferencia de un script, en donde se debe escribir la lógica necesaria para efectuar una configuración, Ansible permite escribir una descripción del estado deseado para un servidor o conjunto de servidores. Ansible se encarga de aplicar dicha descripción.
- Fácil de aprender.

Idempotencia

Propiedad para realizar una acción determinada varias veces y aun así conseguir el mismo resultado que se obtendría si se realizase una sola vez.

Arquitectura



Inventario:

- Es una lista de los nodos que pueden ser accedidos por Ansible. Por defecto, el inventario está soportado por un archivo de configuración, cuya ubicación es /etc/ansible/hosts. Los nodos pueden estar listados por nombre o IP.
- Cada nodo es asignado a un grupo, como pueden ser "web servers", "db servers", entre otros.
- El inventario debe estar escrito en uno de muchos formatos: YAML, INI, etc. YAML es el formato más utilizado en la industria.

Ejemplo de un inventario (YAML)

```
mail.example.com

[webservers]
foo.example.com
bar.example.com

[dbservers]
one.example.com
two.example.com
three.example.com
```

Playbooks
Son archivos escritos en YAML. Contienen la descripción del estado deseado de los sistemas que se van a configurar. Ansible hace el trabajo requerido para llevar los servidores al estado que se haya especificado sin importar el estado en el que se encuentren cuando la configuración se aplique. Los playbooks hacen que las nuevas instalaciones, actualizaciones y la administración del día a día sea repetible, predecible y confiable.

- Los playbooks son simples de escribir y mantener; se escriben en un lenguaje natural.
- Los Playbooks contienen Plays (jugadas), las jugadas contienen tareas (tasks) y las tareas invocan módulos.

Ejemplo:
Este playbook instala la versión más reciente de Apache y se asegura que este corriendo en aquellos servidores que estén bajo el grupo "webservers" en el inventario:

```
---
- hosts: webservers
remote_user: root

tasks:
- name: Asegurarse que la ultima version de Apache esté instalada
  yum:
```

```
    name: httpd
    state: latest

- name: Asegurarse que Apache este corriendo
  service:
    name: httpd
    state: started
    enabled: yes
```

Módulos

- Hay más de 1000 módulos incluidos con Ansible para automatizar las diferentes partes de un ambiente. Realizan el trabajo real de configuración.
- Cada módulo es independiente y se lo puede escribir en diferentes lenguajes de scripting: Python, Perl, Ruby, Bash, etc. Uno de los principios de diseño de los módulos es la idempotencia.
- Módulos populares: Service, file, copy, iptables.

Ejemplos de invocación de módulos (Bash):

```
ansible 127.0.0.1 -m service -a "name=httpd state=started"
ansible localhost -m ping
```

// El primero reproduce una de las tareas vistas que asegura que el servicio de Apache está corriendo. El segundo invoca el módulo 'ping' para hacer un 'ping' localmente contra 'localhost':

Ansible está desarrollado en Python y, en consecuencia, hereda y/o implementa algunos aspectos del lenguaje. Ver:
- El lenguaje de templating (jinja2)
- Operador ternario
- Errores.

Casos de uso de Ansible
- Aprovisionamiento
- Configuration management
- App deployment
- Continous delivery
- Seguridad y compliance
- Orchestration

Configuration Management con Ansible
Ansible es la herramientas más simple para implementar una estrategia de configuration management. Está diseñado para ser minimalista, consistente, seguro y altamente confiable.

Falta?

Ejercicio Ansible

Descargar rmáquinas virtuales:
https://descargarmaquinasvirtuales.com/

Sincrónico

No se pudieron obtener algunos archivos, ¿quizá deba ejecutar «apt-get update» o deba intentarlo de nuevo con –fix-missing?

Para resolver este error, correr el comando:
apt-get update –allow-releaseinfo-change

Para escribir el inventario: => revisar la IP de cada máquinas

Documentación Ansible:
https://docs.ansible.com/ansible/latest/user_guide/playbooks_intro.html

Repaso Parcial

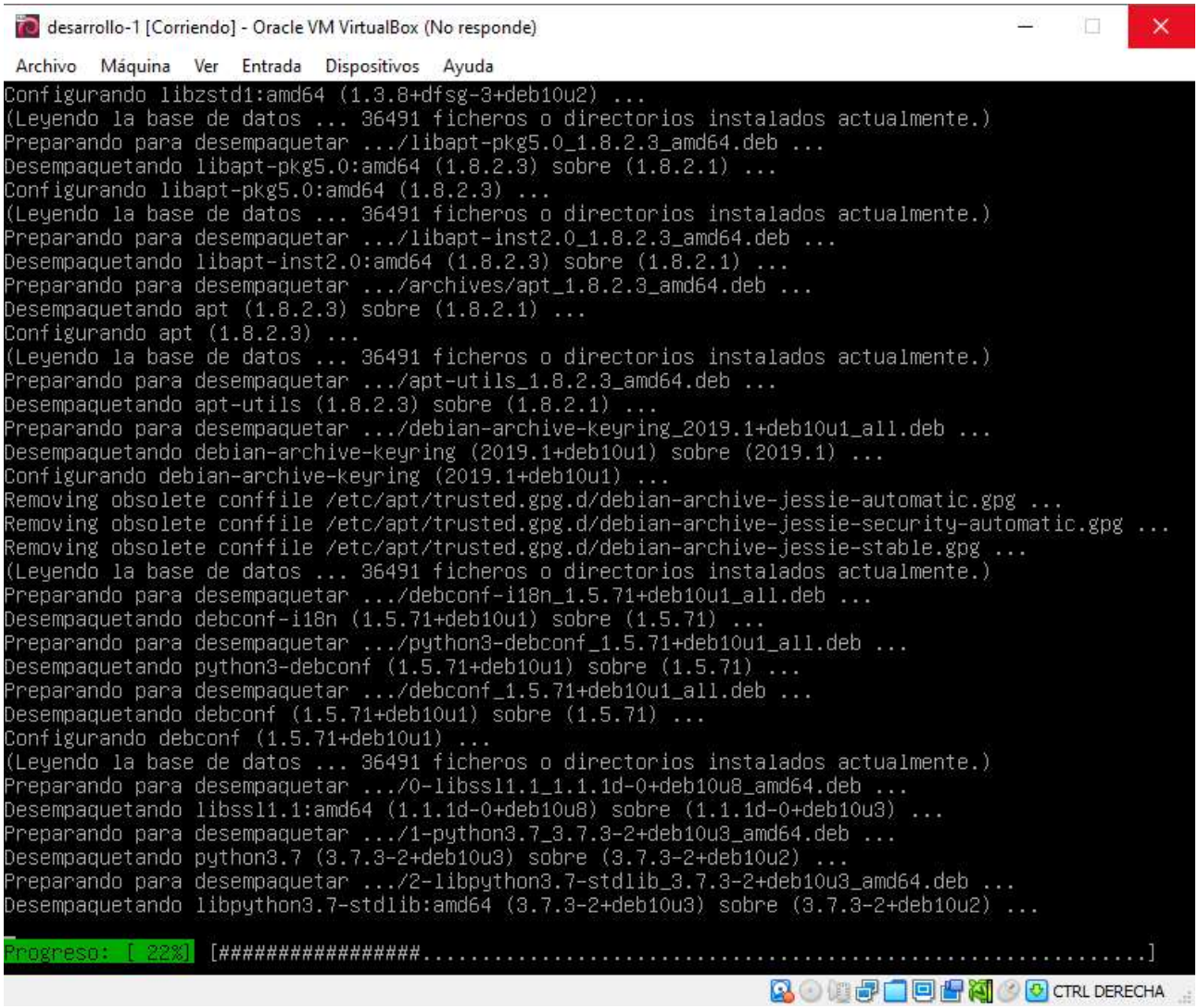
15 preguntas multiple chice
5 preguntas para desarrollar

sudo vs su

Niveles de ejecución de Linux: La consola inicia en el nivel 1.

systemctl restart ssh // Reinicia el servicio cuando cambian las configuraciones

g



Olivia
Juan Pablo
Manuel
Ariana Novo
Charly
Matías Erramuspe

1.

¿Para qué sirve el comando 'tcpdump'? 5 punti

- (●) Analizar trafico de red
- ( ) Analizar conectividad de red
- ( ) Abrir un puerto en el firewall

Cancella selezione

En Bash, el comando 'rm' sirve para... 5 punti

- (●) Borrar Archivos
- ( ) Copiar directorios
- ( ) Renombrar Archivos

Cancella selezione

2.

¿Para que sirve el siguiente comando de Unix : "su"? 5 punti

- ○ Para hacer peticiones a un recurso web
- ○ Para elevar privilegios
- ◉ Para cambiar de usuario

Cancella selezione

¿Cual/es es/son un software/servicio de publicación web? 5 punti

- ☑ Apache
- ☐ Vagrant
- ☑ NGINX

mas.

PowerShell se puede usar para... 5 punti

- ○ Automatización de procesos
- ○ Configuration Managment
- ○ Automatización de tareas
- ◉ Todas las anteriores

Cancella selezione

¿ Cuáles de los siguientes elementos forman parte del sistema operativo ? 5 punti

- ☐ Aplicacion
- ☐ Core
- ☑ Nucleo
- ☑ Interfaz Grafica

Más

¿Cuáles de las siguientes afirmaciones sobre el modelo cliente-servidor son correctas? 5 punti

- [x] El servidor se inicia al recibir el mensaje del cliente.
- [ ] El clientes es el que recibe datos.
- [ ] El servidor es el que recibe datos.
- [x] El servidor siempre esta en escucha

Las tolerancias a fallos de un S.O. son: 5 punti

- [ ] La posibilidad de que se reinicie automáticamente ante fallos
- [x] La posibilidad de seguir operando ante los fallos
- [ ] Las posibilidad de capturar los errores en caso de fallos
- [x] La posibilidad de que otro servidor tome nuestro rol en caso de caída

más

3.

Explique CON SUS PALABRAS la arquitectura CLIENTE - SERVIDOR 10 punti

La arquitectura Cliente-Servidor permite procesar la información de forma disitribuida asignando tareas diferentes a un servidor y a uno o varios clientes . De esta forma, el servidor permanece inactivo hasta que recibe un request de uno de lo clientes. Al recibir el request, el servidor se activa y prepara una response para el cliente. Una vez enviada, el servidor vuelve a su estado inactivo, hasta que vuelve a ser requerido por otro cliente. Esta arquitectura permite optimizar los recursos y facilita la escalabilidad.

Explicar CON SUS PALABRAS alguna de las tareas más comunes para automatizar en TI. Ejemplifique 10 punti

Implementar aplicaciones. Una de las tareas más comunes en Automatización de IT consiste en la instalación y configuración de algunas aplicaciones. Esta tarea puede ser realizada de forma automática, minimizando la ejecución manual.

Explique CON SUS PALABRAS porque es importante gestionar a la infraestructura como GANADO y no como MASCOTAS. 10 punti

4.

Explique CON SUS PALABRAS porque es importante gestionar a la infraestructura como GANADO y no como MASCOTAS. 10 punti

En los procesos modernos se requiere trabajar con una gran cantidad de recursos de infraestructura, y por lo tanto, se requiere administrarlos de manera conjunta, con un enfoque industrial, e implementar soluciones rápidas que se puedan reproducir rápidamente y ser aplicados a múltiples componentes. De esta forma, los recursos se tratan masivamente, como ganado, y no individualmente como si se tratara de mascotas.

.

```
# -*- mode: ruby -*-
# vi: set ft=ruby :
# All Vagrant configuration is done below. The "2" in Vagrant.configure
# configures the configuration version (we support older styles for
# backwards compatibility). Please don't change it unless you know what
# you're doing.
Vagrant.configure("2") do |config|
  config.vm.define "miservidor" do |server|
  config.vm.box = "debian/buster64"
  server.vm.hostname = "miservidor"
  server.vm.network "public_network"
  end
  config.vm.provider "virtualbox" do |vb|
    vb.memory = "4096"
    vb.gui = false
  end
end
```

https://github.com/repoinfradh/deploy_app/blob/main/VagrantfileTwo