



redhat.

Road to DevOps: From SysAdmin to Cloud

Automatización 101

Alex Callejas
Technical Account Manager | CEE México
Abril, 2018

About me

Alex Callejas

Technical Account Manager en Red Hat



[@dark_axl](https://twitter.com/dark_axl)



[/rootzilopochtli](https://www.facebook.com/rootzilopochtli)



www.rootzilopochtli.com

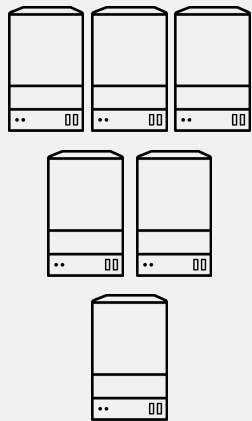


Geek by nature, Linux by choice, Fedora of course!

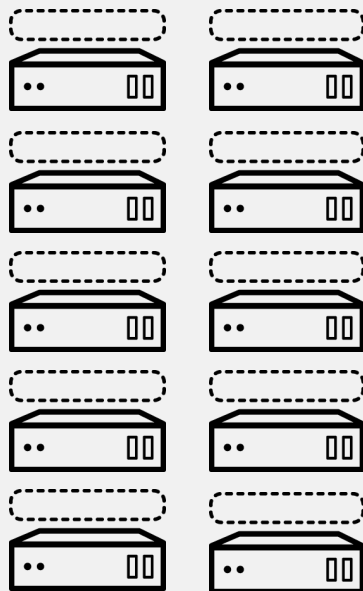
SYSADMIN WARS

Los tiempos han cambiado

+15 years ago



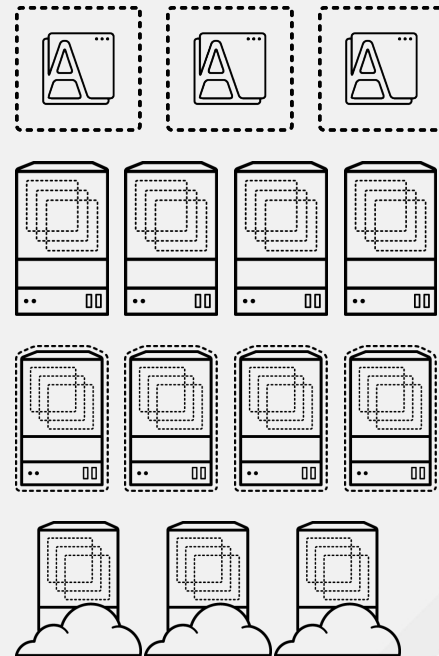
10 years ago



5 years ago



Today



Who you gonna call?



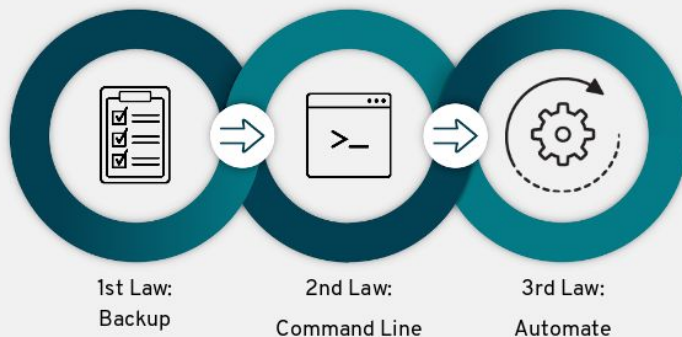
Los SysAdmins nos reinventamos constantemente, siempre estamos investigando y aprendiendo

Las Tres Leyes del SysAdmin



Basadas en las Leyes de la Robótica de Asimov, las Tres Leyes del SysAdmin son:

- **1a Ley:** Un SysAdmin debe hacer un respaldo de todo el sistema y validarlo regularmente.
- **2a Ley:** Un SysAdmin debe dominar la línea de comandos y evitar los gráficos, a menos que entre en conflicto con la 1a Ley.
- **3a Ley:** Un SysAdmin debe automatizar al máximo, y volverse perezoso, en la medida en que este tiempo libre no entre en conflicto con la 1a o 2a Ley.

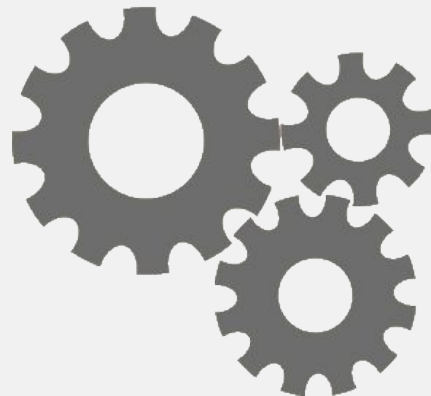


Fuente: <http://www.rootzilopochtli.com/2010/07/3-leyes-sysadmin/>

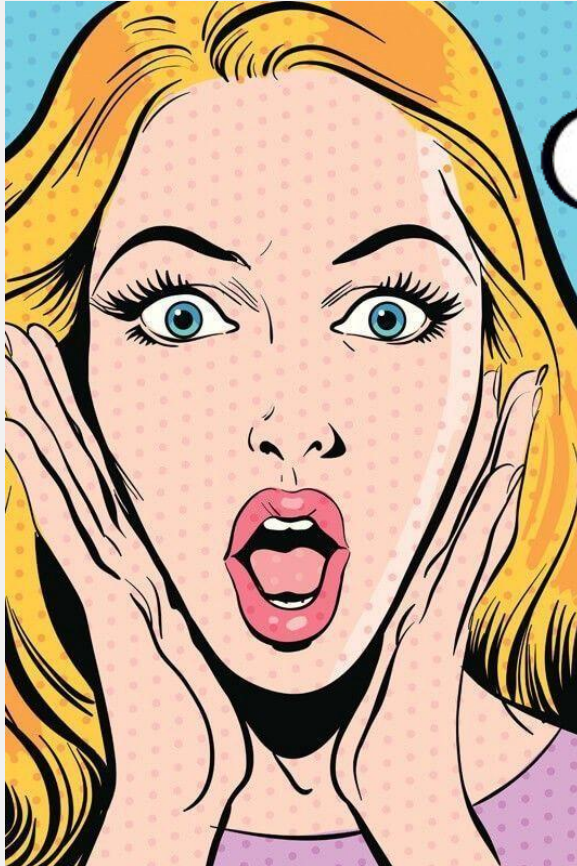
Tareas del día-a-día del SysAdmin



- Monitoreo
 - Disponibilidad (sla)
 - logs
 - performance
 - % filesystem use
- Backup
- Tuning
- Tareas específicas
 - Virtualización
 - Provisionamiento
 - Web y App Server
 - DataBase Server
 - Services Server
 - DNS, NTP, LDAP, Load Balancing, etc



Y muchas, muchas más...



...y ahora, nos vamos a la nube

Tranquilo, tú sabes el truco...

practice



Una nueva era



“Dear SysAdmin friend,

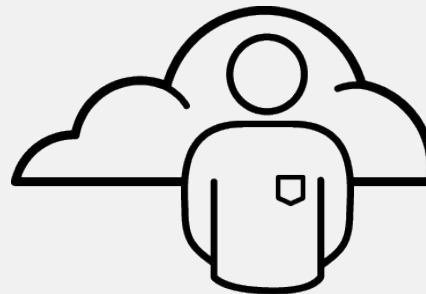
If you are using SSH to manage your cloud ...

you are doing it wrong.

Respectfully,

Future”

-- [Francisco Moctezuma Villa](#) (@yazpik)



La simplicidad es la máxima sofisticación

Leonardo da Vinci



Ansible está diseñado como la herramienta de implementación más simple:

- No es un lenguaje de programación completo
 - Templates YAML que definen tareas y las listan para realizar el trabajo

La mayoría ve a Ansible como una versión mejorada de "SSH en un 'for'"

- Pero realmente Ansible se trata de **tareas**, no de SSH



ANSIBLE

Ansible puede operar en dos capas separadas:

- Control plane (on cloud)
 - Todo lo que no se ejecuta en el SO dentro de la instancia
- On-instance resources
 - Iniciar y detener servicios, crear templates de archivos de configuración, instalar paquetes y todo lo relacionado con el SO que se pueda hacer a través de SSH

Fuente: <https://opensource.com/article/17/8/ansible-serverless-applications>

Antes de empezar

Automatización y Orquestación 101

No pienses en Ansible como una herramienta única

- Creamos una automatización, luego otra ... luego otra
- Cuando se unen, tenemos orquestación
- Cuando las automatizaciones/orquestaciones son portátiles y fáciles de entender, ocurre la eficiencia

Más importante:

ANSIBLE NO ESTÁ EN COMPETENCIA CONTRA OTRAS HERRAMIENTAS

Antes de empezar

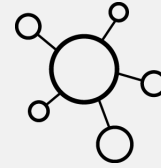
Casos de uso



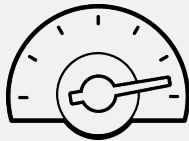
PROVISIONING



CONFIGURATION
MANAGEMENT



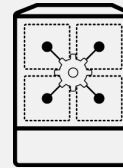
APPLICATION
DEPLOYMENT



CONTINUOUS
DELIVERY



SECURITY &
COMPLIANCE



ORCHESTRATION

Antes de empezar

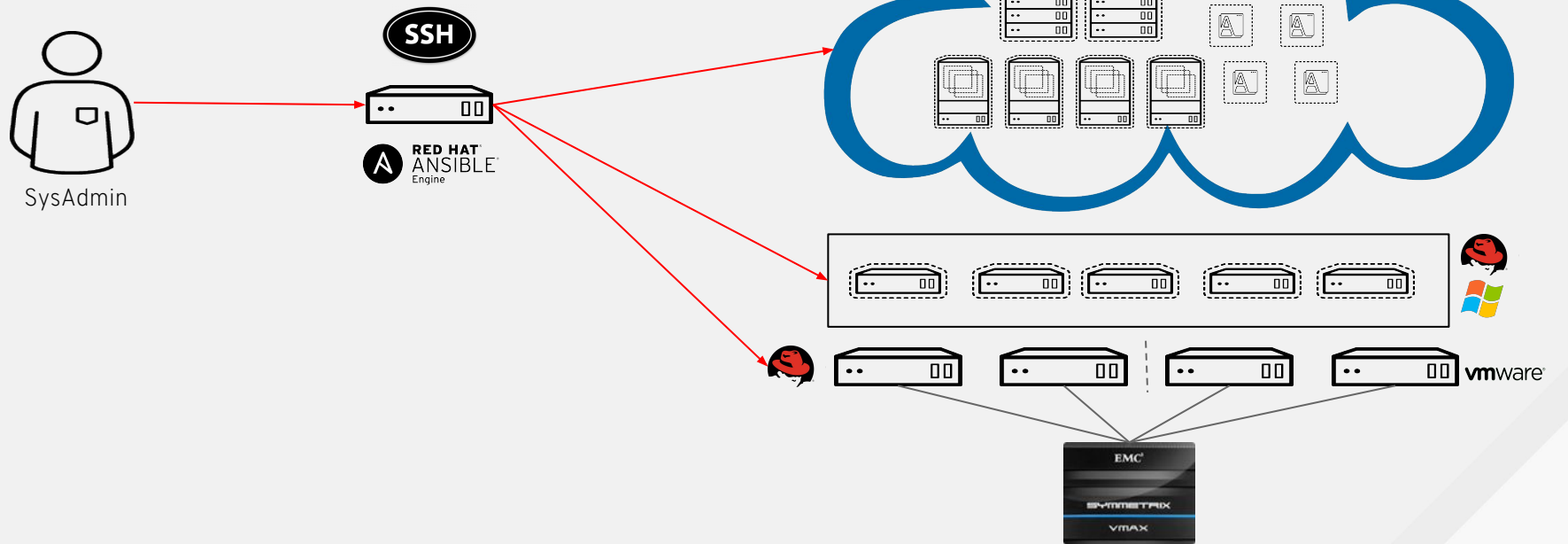
Automatización vs Orquestación

Enfoque general de orquestación:

- Administración de parches, autoservicio de infraestructura, implementación, administración de configuración, automatización de red
- Resuelves una cosa, después resuelves *la siguiente*
- Luego se *orquestan* esas cosas en general

Esto puede incluso significar automatizar cosas que tradicionalmente se han visto como rivales. El objetivo no es entrar en una competencia de herramientas, sino **automatizar los procesos**.

Un ejemplo de la vida real



Tareas del día-a-día del SysAdmin

Cómo iniciar?



Conceptos:

- **Tasks:** una tarea es la unidad de trabajo más pequeña. Puede ser una acción como "Instalar base de datos", "Instalar servidor web" o "Copiar este archivo de configuración al servidor"

Tip: Tasks ← [Module Index](#)

- **Plays:** una jugada se compone de tareas. Por ejemplo, la jugada: "Preparar una base de datos para un servidor web" está compuesta de las tareas:
 - Instalar el paquete de la base de datos
 - Establecer contraseña para el administrador de la base de datos
 - Crea base de datos
 - Establecer acceso a la base de datos
- **Playbook:** Un playbook está compuesto de jugadas. Por ejemplo, en el playbook: "Preparar sitio web con un back-end de base de datos", las jugadas serían:
 - Configurar servidor de base de datos
 - Configurar servidor web

Tareas del día-a-día del SysAdmin

Cómo iniciar?



Lo básico: Control Node

Instalar Ansible:

```
# yum install ansible
```

← [Basics / What Will Be Installed](#)

Crear:

- Directorio de trabajo (WorkDir)
- Llave de SSH

```
$ ssh-keygen -t dsa [*rsa] -b 1024 -f id_dsa_ansible
```

Tareas del día-a-día del SysAdmin

Cómo iniciar?



Lo básico: Control Node

Crear archivo de configuración

```
$ vi ansible.cfg
[defaults]
inventory = inventory
```

Listar equipos vs Inventario

```
$ vi server_list
192.168.122.101
192.168.122.102
192.168.122.103
192.168.122.104
192.168.122.105
192.168.122.106
192.168.122.107
192.168.122.108
192.168.122.109
192.168.122.110
```

```
$ vi inventory
[servers]
192.168.122.[101:110]

[servers:vars]
ansible_ssh_private_key_file=/root/wkdir/id_dsa_ansible
```


Tareas del día-a-día del SysAdmin

Cómo iniciar?



Exportar la key host:

```
# for i in $(cat server_list); do ssh-keyscan -t rsa $i >> ~/.ssh/known_hosts; done
```

Tip: `.vimrc`

```
set ts=2
set et
set ai
```

Crear playbook para enviar la llave a los servidores:

```
authorized_keys.yml
- name: Set authorized key      ← play:jugada
  hosts: all                    ← task:tarea
  tasks:
Module → - authorized_key: user=root state=present key="{{ lookup('file', 'id_dsa_ansible.pub') }}"
```

```
# ansible-playbook authorized_keys.yml -i server_list -k --check
```

Tareas del día-a-día del SysAdmin

Cómo iniciar?



Revisar la comunicación:

```
# ansible servers -m ping
```

```
# ansible servers -m command -a "hostname" ← ad-hoc command
```

Facts:

```
# ansible localhost -m setup ← setup module
```

```
# ansible localhost -m setup -a 'filter=*hostname*'
```

```
# ansible localhost -m setup --tree localhost_facts
```

*Facts: variables que Ansible detecta automáticamente en un host administrado

Tareas del día-a-día del SysAdmin

Estandarización



Modificamos el hostname:

```
pool_hostname.yml
- name: Set hostname
  hosts: servers
  vars:
    hostname: vm_web-*.flisol.mx

  tasks:
    - name: set the friendly hostname
      hostname: name={{ hostname.replace('*', ansible_all_ipv4_addresses[0].split('.')[3]) }}
```

```
# ansible-playbook pool_hostname.yml
```

Creamos alias y sus variables:

```
# vi inventory
[webservers]
vm_web-101 ansible_host=192.168.122.101
...
[webservers:vars]
ansible_ssh_private_key_file=/root/wkdir/id_dsa_ansible
```

Fuente: [Meaningful hostnames with Ansible](#)

Tareas del día-a-día del SysAdmin

Actualización de paquetes



Revisamos versión de kernel y release:

```
# ansible webserver -m command -a "uname -a"
# ansible webserver -m command -a "cat /etc/redhat-release"
```

Creamos directorio y archivo de variables para contraseña:

```
# vi vars/variables.yml
password: [contraseña suscripción]
```

Encriptamos el archivo de variables:

```
# ansible-vault encrypt vars/variables.yml
New Vault password:[contraseña archivo]
Confirm New Vault password:[contraseña archivo]
Encryption successful
```

```
# ansible-vault view vars/variables.yml
```

Tareas del día-a-día del SysAdmin

Actualización de paquetes



Creamos playbook para el registro, instalación y actualización de paquetes:

```
pool_update.yml
---
- name: Register and update OS packages
  hosts: webservers
  remote_user: root

  tasks:
    - name: Include the variables file
      include_vars: vars/variables.yml

    - name: Register pool machine
      redhat_subscription:
        state: present
        username: rhn-support-acalleja
        password: "{{ password }}"
        pool: "Employee SKU"

    - name: Disable all repos
      shell: "subscription-manager repos --disable=*"

    - name: Enable RHEL7 and RHSCl repos
      shell: "subscription-manager repos --enable rhel-7-server-rpms --enable rhel-server-rhsc1-7-rpms"
```


Tareas del día-a-día del SysAdmin

Actualización de paquetes



```
- name: Install the required packages
  yum:
    name: "{{ item }}"
    state: latest
  with_items:
    - sysstat
    - sos
    - cockpit

- name: Upgrade all packages
  yum:
    name: "*"
    state: latest
  register: systemupgrade

- set_fact:
  real_ansible_host: "{{ ansible_host }}" ← seteamos variable

- name: Restart pool machine
  shell: sleep 3 && shutdown -r +1 "Ansible packages upgrade triggered"
  async: 1
  poll: 0
  ignore_errors: true
```

Tareas del día-a-día del SysAdmin

Actualización de paquetes



```
- name: Wait for hosts to go down
local_action:
wait_for host={{ real_ansible_host }}
port=22 state=absent delay=1 timeout=120
become: false

- name: Wait for hosts to come back up
local_action:
wait_for host={{ real_ansible_host }}
port=22 state=started delay=30 timeout=120
become: false

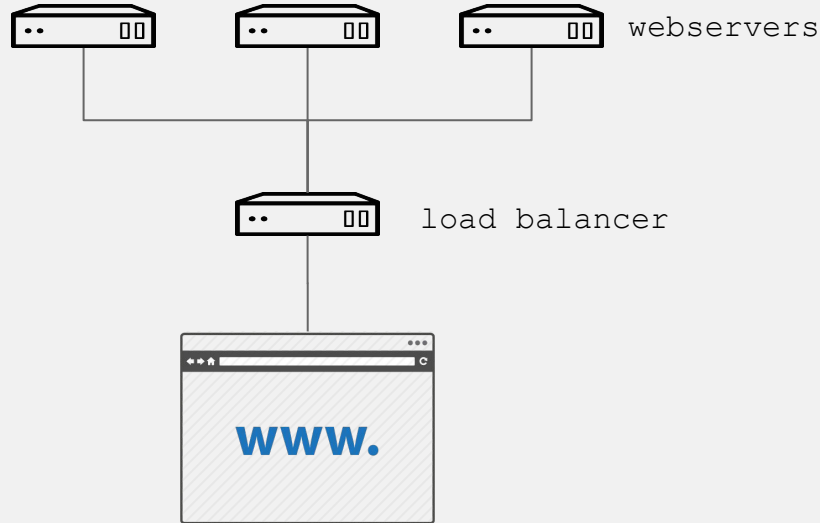
- name: Show host
debug:
msg: "{{ ansible_host }}"
```

Ejecutamos el playbook:

```
# ansible-playbook pool_update.yml --ask-vault-pass --syntax-check
```

Tareas del día-a-día del SysAdmin

Instalar aplicación web



Tareas del día-a-día del SysAdmin

Instalar aplicación web



Conceptos:

- **Roles:** Una colección redistribuible y reutilizable de:
 - tasks
 - files
 - scripts
 - templates
 - variables
- **Ansible Galaxy:** es un sitio gratuito para buscar, descargar, calificar y revisar todo tipo de roles Ansible desarrollados por la comunidad



<https://galaxy.ansible.com/>

Tareas del día-a-día del SysAdmin

Instalar aplicación web



Creamos la estructura de directorios para el rol:

```
# ansible-galaxy init --offline myapache
```

Copiamos los templates:

```
# mv apache_* myapache/templates/
```

Creamos el *handler* que reiniciará el servicio **httpd**:

```
# vi myapache/handlers/main.yml
---
# handlers file for myapache

- name: restart apache
  service:
    name: httpd
    state: restarted
```


Tareas del día-a-día del SysAdmin

Instalar aplicación web



Creamos las *tasks* para la instalación de `httpd`:

```
# vi myapache/tasks/main.yml
---
# tasks file for myapache

- name: install apache package
  yum:
    name: httpd
    state: latest

- name: install firewalld package
  yum:
    name: firewalld
    state: latest

- name: template out apache configuration file
  template:
    src: apache_httpdconf.j2
    dest: /etc/httpd/conf/httpd.conf
    owner: root
    group: root
    mode: 0444
  notify:
    - restart apache
  when: apache_enable
```

Tareas del día-a-día del SysAdmin

Instalar aplicación web



```
- name: template out apache configuration file
  template:
    src: apache_indexhtml.j2
    dest: /var/www/html/index.html
    owner: root
    group: root
    mode: 0444
  when: apache_enable

- name: start and enable apache daemon
  service:
    name: httpd
    state: started
    enabled: true
  when: apache_enable
```

```
- name: start and enable firewalld daemon
  service:
    name: firewalld
    state: started
    enabled: true
  when: apache_enable

- name: open http firewalld port
  firewalld:
    port: 80/tcp
    immediate: true
    permanent: true
    state: enabled
  when: apache_enable
```

Tareas del día-a-día del SysAdmin

Instalar aplicación web



Seteamos las variables de inicio:

```
# vi myapache/defaults/main.yml
---
# defaults file for myapache

apache_enable: true
```

Creamos el playbook:

```
# vi apache.yml
---
- name: setup apache on webservers
  hosts: webservers
  roles:
    - myapache
```

Ejecutamos el playbook:

```
# ansible-playbook apache.yml
```

Tareas del día-a-día del SysAdmin

Instalar aplicación web



Creamos el rol lbserver:

```
# ansible-galaxy init --offline lbserver
```

handler:

```
# vi lbserver/handlers/main.yml
---
# handlers file for lbserver
- name: restart haproxy
  service: name=haproxy state=restarted

- name: reload haproxy
  service: name=haproxy state=reloaded
```

templates:

```
# mv haproxy.cfg.j2 lbserver/templates/
```

Tareas del día-a-día del SysAdmin

Instalar aplicación web



tasks:

```
# vi lbserver/tasks/main.yml
---
# tasks file for lbserver
- name: install haproxy
  yum: name=haproxy state=latest
  tags: haproxy

- name: install firewalld package
  yum: name=firewalld state=latest
  tags: haproxy

- name: configure haproxy
  template: src=haproxy.cfg.j2 dest=/etc/haproxy/haproxy.cfg
  notify: reload haproxy
  tags: haproxy

- name: start haproxy
  service: name=haproxy state=started
  tags: haproxy
```

Tareas del día-a-día del SysAdmin

Instalar aplicación web



tasks:

```
- name: start and enable firewalld daemon
  service: name=firewalld state=started enabled=true
  tags: haproxy

- name: open haproxy firewalld port
  firewalld:
    port: 80/tcp
    immediate: true
    permanent: true
    state: enabled
  tags: haproxy
```

Ejecutamos el playbook:

```
# ansible-playbook lb.yml
```

We're ready!!



Tareas del día-a-día del SysAdmin

Primeros pasos en la nube



Google Cloud Platform (GCP):

- Cuenta de GMail
- [Google Cloud Platform Free Tier](#)
 - \$300 USD de crédito
 - 12 meses



Instalamos pre-requisitos para **Google Compute Engine (gce)**:

```
# yum install python27-python-pip
```

```
# scl enable python27 bash
```

Todos los módulos de GCE requieren el módulo `apache-libcloud` que puede instalar desde pip::

```
# pip install apache-libcloud
```

Tareas del día-a-día del SysAdmin

Primeros pasos en la nube



Generar credenciales JSON:

- [Crear cuenta de servicio](#)
- [Descargar credenciales JSON](#)



Creamos una instancia de prueba mediante un playbook:

```
# vi gce-test.yml
---
- name: Playbook test to create gce instance
  hosts: localhost
  connection: local
  gather_facts: no

  vars:
    service_account_email: rootzilopochtli@flisol2018-201620.iam.gserviceaccount.com
    credentials_file: /root/wkdir/FLISoL2018-51f71b5912a8.json
    project_id: flisol2018-201620
    machine_type: n1-standard-1
    image: centos-7
```

Tareas del día-a-día del SysAdmin

Primeros pasos en la nube



```
tasks:
  - name: Launch instances
    gce:
      instance_names: dev
      machine_type: "{{ machine_type }}"
      image: "{{ image }}"
      service_account_email: "{{ service_account_email }}"
      credentials_file: "{{ credentials_file }}"
      project_id: "{{ project_id }}"
```

Ejecutamos el playbook:

```
# ansible-playbook gce-test.yml
```

Tareas del día-a-día del SysAdmin

Primeros pasos en la nube



Google Cloud Platform FLISol2018

Compute Engine **Instanc... de VM** [+ CREAR INSTANCIA](#) [↓ IMPORTAR VM](#) [↻ ACTUALIZAR](#) [▶ INICIAR](#)

Instancias de VM

Grupos de instancias

Plantillas de instancia

Discos

Instantáneas

Imágenes

TPU

Descuentos por uso compro...

Metadatos

Filtrar instancias de VM

<input type="checkbox"/>	Nombre ^	Zona	Recomendación	IP interna	IP externa	Conectar
<input type="checkbox"/>	<input checked="" type="checkbox"/> dev	us-central1-a		10.128.0.2	35.192.210.251	SSH ▾ ⋮

Tareas del día-a-día del SysAdmin

Primeros pasos en la nube



Subamos nuestra aplicación web a la nube, con el playbook:

```
# vi gce-apache.yml
---
- name: Create gce webserver instance
  hosts: localhost
  connection: local
  gather_facts: True

vars:
  service_account_email: rootzilopochtli@flisol2018-201620.iam.gserviceaccount.com
  credentials_file: /root/wkdir/FLISoL2018-51f71b5912a8.json
  project_id: flisol2018-201620
  instance_names: web1,web2,web3
  machine_type: n1-standard-1
  image: centos-7
```

Tareas del día-a-día del SysAdmin

Primeros pasos en la nube



```
tasks:
- name: Create an Instance based on image {{ image }}
  gce:
    instance_names: "{{ instance_names }}"
    machine_type: "{{ machine_type }}"
    image: "{{ image }}"
    state: present
    preemptible: true
    tags: http-server,https-server
    service_account_email: "{{ service_account_email }}"
    credentials_file: "{{ credentials_file }}"
    project_id: "{{ project_id }}"
    metadata: '{"sshKeys": "user:ssh-key"}'
    register: gce

- name: Save host data within a Group
  add_host:
    hostname: "{{ item.public_ip }}"
    groupname: gce_instances_temp
    with_items: "{{ gce.instance_data }}"
```

Tareas del día-a-día del SysAdmin



Primeros pasos en la nube

```
- name: Wait for SSH to come up
  wait_for: host={{ item.public_ip }} port=22 delay=10 timeout=60
  with_items: "{{ gce.instance_data }}"

- name: setting fact
  set_fact: host={{ item.public_ip }}
  with_items: "{{ gce.instance_data }}"

- name: Configure instance post-creation
  hosts: gce_instances_temp
  gather_facts: True
  remote_user: dark_axl
  become: yes
  become_method: sudo

  roles:

    - myapache
```

Ejecutamos el playbook:

```
# ansible-playbook -e 'host_key_checking=False' gce-apache.yml --key-file id_dsa_ansible
```

Tareas del día-a-día del SysAdmin

Primeros pasos en la nube



Google Cloud Platform FLISoL2018

Compute Engine

Instanc... de VM [+ CREAR INSTANCIA](#) [IMPORTAR VM](#) [ACTUALIZAR](#)

Instancias de VM

Grupos de instancias

Plantillas de instancia

Discos

Instantáneas

Imágenes

Filtrar instancias de VM

<input type="checkbox"/>	Nombre ^	Zona	Recomendación	IP interna	IP externa	Conectar
<input type="checkbox"/>	✓ web1	us-central1-a		10.128.0.2	35.193.196.114 ↗	SSH ▾ ⋮
<input type="checkbox"/>	✓ web2	us-central1-a		10.128.0.3	35.194.11.121 ↗	SSH ▾ ⋮
<input type="checkbox"/>	✓ web3	us-central1-a		10.128.0.4	35.184.129.7 ↗	SSH ▾ ⋮

Tareas del día-a-día del SysAdmin

Primeros pasos en la nube



Creamos una instancia de Load Balancing, mediante el playbook:

```
# vi gce-lb.yml
---
- name: Playbook to create gce load balancing instance
  hosts: localhost
  connection: local
  gather_facts: no

  vars:
    service_account_email: rootzilopochtli@flisol2018-201620.iam.gserviceaccount.com
    credentials_file: /root/wkdir/FLISoL2018-51f71b5912a8.json
    project_id: flisol2018-201620
    machine_type: n1-standard-1
    image: centos-7
```

Tareas del día-a-día del SysAdmin

Primeros pasos en la nube



```
tasks:
  - name: Create gce load balancer
    local_action:
      module: gce_lb
      name: lbserver
      region: us-central1
      members: ['us-central1-a/web1', 'us-central1-a/web2', 'us-central1-a/web3']
      httphealthcheck_name: hc
      httphealthcheck_port: 80
      httphealthcheck_path: "/"
      service_account_email: "{{ service_account_email }}"
      credentials_file: "{{ credentials_file }}"
      project_id: "{{ project_id }}"
```

Ejecutamos el playbook:

```
# ansible-playbook gce-lb.yml -v
```

Tareas del día-a-día del SysAdmin

Primeros pasos en la nube



Google Cloud Platform FLISoL2018

Servicios de red

- Balanceo de cargas
- Cloud DNS
- Cloud CDN

Balanceo de cargas

+ CREAR BALANCEADOR DE CARGAS ACTUALIZAR

Balancedores de cargas Back-ends Front-ends

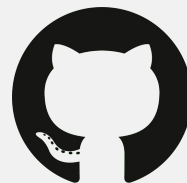
Filtrar por nombre o protocolo

<input type="checkbox"/> Nombre	Protocolo ^	Back-ends
<input type="checkbox"/> lserver-tp	TCP	1 grupo de destino (3 instancias) ⋮

Para editar los recursos de balanceo de cargas, como las reglas de reenvío y los proxies de destino, ve al [menú avanzado](#).

Siguientes pasos

Crear repositorio y subirlo a GitHub



Inicializamos un directorio para nuestro proyecto:

```
# git init ansible_flisol
```

Copiamos nuestros archivos en el directorio creado:

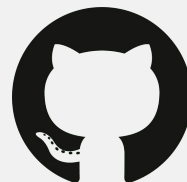
```
# ls ansible_flisol  
ansible.cfg  playbooks/  roles/  templates/
```

Creamos archivo README.md:

```
# vi README.md  
# ansible_flisol  
  
Scripts y playbooks usados en la platica del FLISoL 2018
```

Siguientes pasos

Crear repositorio y subirlo a GitHub



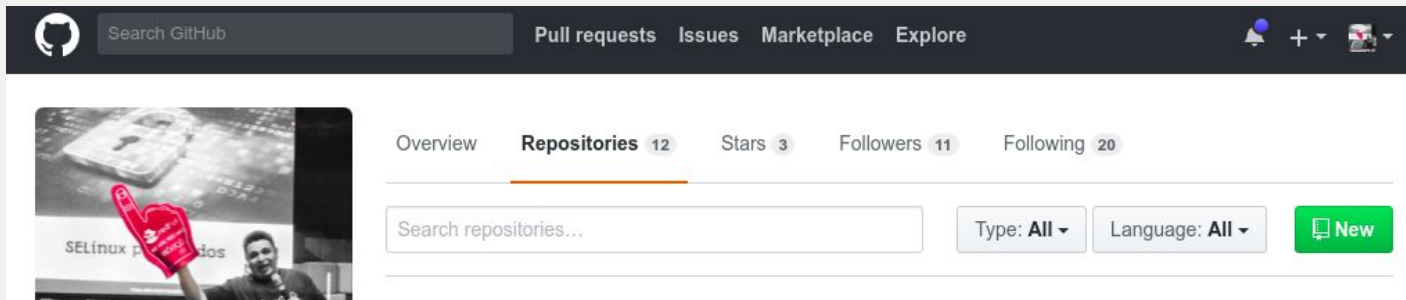
Agregamos los archivos al proyecto:

```
# git add .
```

Hacemos *commit*:

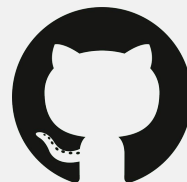
```
# git commit -m "Archivos agregados al proyecto"
```

Creamos el repositorio en github.com:



Siguientes pasos

Crear repositorio y subirlo a GitHub



Conectamos nuestro repositorio remoto:

```
# git remote add origin https://github.com/darkaxl/ansible_flisol.git
```

Empujamos el contenido local al repositorio remoto:

```
# git push origin master
```

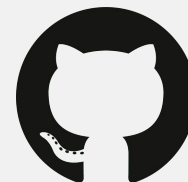
README.md

ansible_flisol

Scripts y playbooks usados en la platica "Road to DevOps: From SysAdmin to Cloud. Automatizacion 101"

Siguientes pasos

Clona el repositorio y prueba



The screenshot shows the GitHub interface for the repository 'darkaxl/ansible_flisol'. At the top, there are navigation links for 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. Below the repository name, there are statistics: 'Unwatch 1', 'Star 0', and 'Fork 0'. The main content area shows the repository description: 'Scripts y playbooks usados en la platica del FLISoL 2018'. Below this, there are statistics for '6 commits', '1 branch', '0 releases', and '1 contributor'. A list of files is displayed, including 'playbooks', 'roles', 'templates', 'README.md', 'ansible.cfg', and 'sample.c'. The 'README.md' file is selected, and its content is visible at the bottom of the screenshot, showing the repository name 'ansible_flisol' and a brief description: 'Scripts y playbooks usados en la platica "Road to DevOps: From SysAdmin to Cloud. Automatizacion 101"'. The repository is currently on the 'master' branch.

```
# git clone https://github.com/darkaxl/ansible_flisol.git
```

Siguientes pasos

Una reflexión..



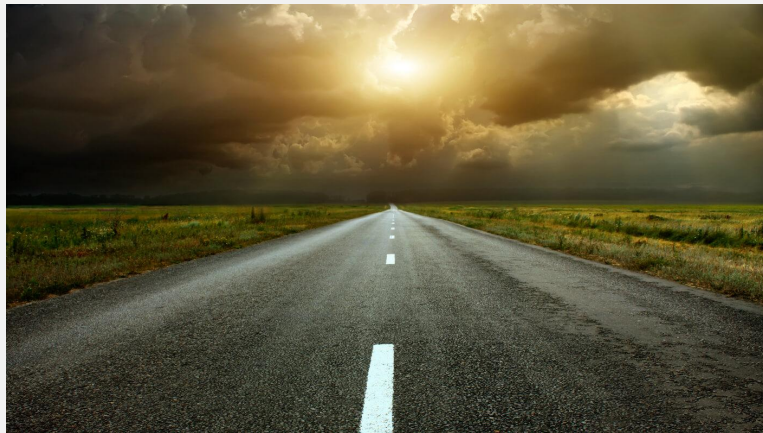
La automatización no es solo una herramienta

Es una estrategia, es un viaje

No es necesario abandonar o redefinir las operaciones

Construyamos con Ansible puentes entre cargas legacy y modernas

Fomentemos y aprovechemos el conocimiento comunitario, luego convirtamoslo en automatizaciones para compartir



Siguientes pasos



Join Us!

Sólo necesitas una cuenta de correo: developers.redhat.com



Try it

Obtén la prueba del servicio : cloud.google.com



Share

Comparte tus dudas, scripts, playbooks, github, trucos, etc



Gracias!



plus.google.com/+RedHat



facebook.com/redhatinc



linkedin.com/company/red-hat



twitter.com/RedHatNews



youtube.com/user/RedHatVideos

Red Hat