EL INVENTARIO DE ANSIBLE

Publicado el 12 de junio de 2020 por atareao 33.7K Deja un comentario

Este es **uno** de los capítulos del tutorial Automatización con Ansible. Una introducción.. Encontrarás los enlaces a todos los de capítulos, al final de este artículo.

Como te he ido contando en los capítulos anteriores de este tutorial sobre **Ansible**, esta herramienta a permitir gestionar uno o varios servidores o máquinas. Estos servidores o máquinas pueden ser desde una Raspberry Pi hasta varios cientos de VPS, o lo que tu consideres. Quiero insistir en el punto de que no es necesario que tengas que gestionar varios cientos de máquinas, con **una sola basta**. El objetivo de esto es conseguir ser mas eficiente en la gestión de tu máquina y conseguir reducir al mínimo los errores que puedas cometer. Es mas, y como ya te adelanté, la ventaja de utilizar **playbooks** es que puedes versionarlos, puedes someterlos al control de versiones. Dicho esto, la cuestión ahora es ¿como organizar las máquinas a gestionar?¿como controlar y organizar la máquina o máquinas que quieras gestionar con Ansible?. Para esto, existe el concepto de **inventario de Ansible**, que es exactamente lo que esperas con este nombre un conjunto de nodos, máquinas o *hosts*, como **tu lo quieras llamar**. En pocas palabras un **inventario de Ansible** no es mas que la **colección organizada** de todas las máquinas que quieres gestionar.

Sin embargo, este inventario te permite hacer algo mas que simplemente tener un listado de máquinas, y esto es, precisamente, lo que voy a contarte en esta nueva entrega del tutorial sobre **Ansible**.

INVENTARIO DE ANSIBLE

Como te comentaba en la introducción, en **inventario de Ansible**, no es mas que una colección organizada de máquinas, *hosts* o *nodos*. Pero, va mas allá de un simple listado de máquinas, puesto que te permite agruparlos y definir variables, para simplificar así su tratamiento y gestión.

Una vez ya tengas definido tu inventario o *inventarios*, en el caso de que decidas tener mas de uno, ya puedes comenzar esa gestión cómoda de todas tus máquinas, seleccionando aquella máquina o grupo de máquinas sobre la que quieres trabajar.

FORMATOS

Inicialmente para definir los inventarios puedes utilizar dos formatos. O bien un `INI` básico o bien un formato del tipo `YAML`. En mi caso, en este tutorial me voy a centrar en el segundo. Es una cuestión de manías. Al final, ya que tendrás que utilizar **YAML** para los *playbooks*, ¿porque no utilizarlo para todo?. Al menos, este es el razonamiento que me he *dado a mi mismo* para definir los inventarios con este formato.

GRUPOS

Todas tus máquinas están agrupadas, y además pertenecen a dos grupos. Un primer grupo llamado `all`, que recoge todas las máquinas. Si has definido un grupo al que pertenezca la máquina, ya lo tienes claro, cual es ese segundo grupo. En el caso de que no le hayas definido un grupo entonces, el segundo grupo al que pertenece es `ungrouped`. Esto lo debes tener siempre presente, porque si bien, puede ser que no estén declarados de forma explícita, lo cierto es que **están ahí**.

Otra **cuestión interesante** que debes tener presente, es que **una máquina puede pertenecer a varios grupos**. Esto es, sinceramente, una idea genial. Supongamos que quieres actualizar todas tus máquinas **Ubuntu** pero no tus máquinas **Red Hat**, si hiciste correctamente la agrupación, esta operación te resultará muy sencilla.

Así puedes agrupara tus diferentes máquinas en función de diferentes criterios, como podrían ser,

- **Quien** en referencia al sistema operativo que tengan
- **Que** para separar por el servicio que ofrecen, por ejemplo, si es una base de datos o un servidor web. Incluso puedes clasificarlo por la tipología de base de datos o servidor.
- **Donde** en el caso que quieras distinguir por su ubicación geográfica o la red en la que se encuentra, u otras especificaciones de ese estilo.
- **Cuando** para definir el estado en el que se encuentra. Por ejemplo máquinas para desarrollo o producción o incluso test.

Otra cuestión que deberías tener presente al definir los grupos, es que puedes crear grupos anidados, es decir, grupos dentro de otros grupos.

Otra característica interesante de los inventarios, y que te puede ahorrar mucho trabajo, es que puedes definir rangos de máquinas o *hosts*. Si normalmente creas las máquinas de forma secuencial, puedes agruparlas utilizando rangos. Por ejemplo, las bases de datos,

```
databases:
 hosts:
   db[1:20].center.es
```

De la misma manera que utilizas rangos numéricos, también es posible define rangos alfabéticos.

VARIABLES

Hasta el momento solo te he hablado de los nombres de cada una de las máquinas o *hosts* que componen tu inventario. Sin embargo, es seguro que vas a querer definir otra serie de variables, como puede ser el puerto al que quieres conectarte o incluso con que usuario lo quieres hacer, en el caso que utilices diferentes usuarios.

Así, por ejemplo, puedes definir las variables por *host*, como en el siguiente ejemplo,

```
rpiu:
 ansible_host: 192.168.1.141
 ansible_user: ansible
 ansible_private_key_file: /home/lorenzo/.ssh/ansible_id_rsa
```

Pero también puedes definir variables por grupo, por ejemplo,

```
rpis:
 hosts:
   rpi1:
     ansible_host: 192.168.1.10
   rpi2:
     ansible_host: 192.168.1.11
   rpi3:
     ansible_host: 192.168.1.12
 vars:
   ansible_user: ansible
   ansible_private_key_file: /home/lorenzo/.ssh/ansible_id_rsa
```

OTRA ORGANIZACIÓN ES POSIBLE

Hasta el momento, has organizado todo tu inventario en un solo archivo. En mi caso, tengo el inventario en un directorio, que he llamado `inv` y dentro del mismo, el archivo `inventory.yml`, pero a los efectos es lo mismo. En ese archivo tienes *mezclados hosts* y variables, todo en uno. Sin embargo, **otra organización es posible**. Puedes separar *hosts* por un lado y **variables** por otro lado, según tus necesidades. Es mas, puedes tener varios archivos de *host* y varios archivos de variables, lo que a ti te resulte más cómodo y práctico.

En este sentido y como ejemplo, la estructura podría ser algo como la que te muestro a continuación,

```
inv/
 grupo1.yml
 grupo2.yml
 group_vars/
   all.yml
   grupo1.yml
   grupo2.yml
```

VARIABLES

Las variables que definas son las que te van a permitir controlar como te conectas a las distintas máquinas de tu entorno. Así, algunas de estas variables son las siguientes,

- `ansible_host` en caso de que el nombre de la máquina a la que te conectas sea distinto del alias que tu le has asignado.
- `ansible_port` para indicar al puerto en que al que te conectas, en el caso de que sea distinto del `22`.
- `ansible_user` el nombre del usuario con el que te conectas
- `ansible_password` la contraseña del usuario con el que te conectas. Como sabes, yo no soy muy partidario de utilizar contraseña, sino mas bien, clave público privada.
- `ansible_private_key_file` se refiere a la clave público privada.
- `ansible_become` permite escalar privilegios
- `ansible_become_user` igual que el anterior, pero además para un usuario concreto.

- `ansible_python_interpreter` te permite indicar la ruta en la que se encuentra Python.

En este caso, indicarte que en uno de los VPS que tengo, me salía un mensaje de aviso como el que te muestro a continuación,

```
[DEPRECATION WARNING]: Distribution Ubuntu 18.04 on host co01 should use /usr/bin/python3, bu
```

La solución a este aviso, fue tan sencilla como añadir la siguiente variable al inventario `ansible_python_interpreter: /usr/bin/python3` y dejó de aparecer el aviso.

Esta no son mas que algunas de las variables que puedes definir para realizar tus conexiones. Puedes leer toda la documentación relativa a estas variables en la introducción al inventario de Ansible.

CONCLUSIÓN

Como ves esto del inventario da mucho de si, y realmente es algo importante, puesto que te va a permitir gestionar todas tus máquinas. Indicarte que esto del inventario todavía tiene mas recorrido, porque existe la posibilidad de crear inventarios dinámicos, y alguna que otra cosa mas. Sin embargo, en este primer tutorial de Ansible, creo que ya tienes mas que suficiente. Es mas, yo te diría que, inicialmente no te pelees con una estructura de archivos de inventario muy compleja, pero, definitivamente, esto dependerá casi en exclusiva de la cantidad y organización de máquinas que tengas en tu ecosistema.

- Imagen de portada de Glenn Carstens-Peters en Unsplash

Listado de capítulos

Estos son todos los capítulos del tutorial Automatización con Ansible. Una introducción.,

YAML el reemplazo de JSON
Comandos ad-hoc

YAML EL REEMPLAZO DE JSON

YAML es un lenguaje similar a JSON en cuanto a funcionalidad, pero que trata de ser mas simple. Es el lenguaje utilizado para los playbooks de Ansible

COMANDOS AD-HOC

Los comandos ad-hoc son una potente opción que te ofrece Ansible para realizar acciones no tan habituales sobre todas tus máquinas de forma sencilla.

Buscar

aKademy-es 2020

Del 20 al 22 de Noviembre

¡No te la pierdas!

YouTube

iVoox

Spotify

Apple podcast

Google podcast

Telegram

Twitter

Mastodon

GitHub

Facebook