T6 - DevOps

T-DOP-600

DevOps - Ansible

Bootstrap

1.0

{EPITECH.}

DevOps - Ansible

- The totality of your source files, except all useless files (binary, temp files, obj files,...), must be included in your delivery.
- All the bonus files (including a potential specific Makefile) should be in a directory named *bonus*.
- Error messages have to be written on the error output, and the program should then exit with the 84 error code (0 if there is no error).

Automation - Ansible

+ Environment

During this project, you will need multiple virtual machines. You can run it locally or:

- On Digital Ocean ($50 credit, 30 days)

This is a referral link. Feel free to invite each other !

+ Setup time

Install Ansible.

Start 3 virtual machines (Debian Buster) and upload your public SSH key.

+ INVENTORY

Before describing your infrastructure, you will need to list targeted servers into an inventory file. We often create an inventory for each environment: development, staging, production… For easier manipulation, it is recommended to organize servers in groups.

Create your first inventory:

- group “web”: 2 hosts “web-1” and “web-2”
- group “redis”: 1 host “cache-1”

Now you can check if hosts are up and responding:

```
$ ansible -i my-inventory web -m ping
$ ansible -i my-inventory redis -m ping
$ ansible -i my-inventory all -m ping
```

1- Here, “ping” is an Ansible module. Can you find a command, that will execute the “uptime” linux command on all 3 hosts ?
2- Can you reboot those machines simultaneously ?
3- Using the “apt” Ansible module, can you install “htop” everywhere then “nginx” on web servers ?
4- We don’t need htop anymore, can you uninstall it, using the “apt” Ansible module ?
5- Can you ensure nginx will be started automatically at system start, using module “service” ?

+ PLAYBOOK, ROLES AND TASKS

Ansible will help you configure a lot of machines and ensure the expected state is reached. But executing tons of Ansible commands is not very funny! Starting now, you will persist your configurations in Yaml files.

In Ansible architecture, a Playbook calls multiple Roles, that call multiple Tasks. Playbooks must be as simple as possible, and call reusable Roles.

Roles must do one thing and do it well. For example: installing and configuring a database. If you need multiple databases on different machines, this Ansible role will use variables to configure different username and password.

1- Create your first playbook and a role “base”:

- this role must be executed on every hosts
- it updates the OS if necessary
- then installs some essential packages: apt-transport-https, ca-certificates, emacs-nox, git, curl, unzip, zsh…
- set the timezone: “Europe/Paris” (if your server is based in France)
- set zsh as the root shell

2- Create a “redis” role that installs …redis.
3- Create a “nodejs” role, responsible for installing NodeJS and npm
4- Create a “deploy” role that:

- uploads application provided in this bootstrap to both web servers
- unzip application
- installs dependencies

5- Right now, the NodeJS application is still not running. You have to write your own SystemD service and upload it to `/etc/systemd/system/my-awesome-app.service`.

None of those tasks need the “shell” module !

+ VARIABLES

In the last step, your NodeJS app has started. But as probably figured out, the application configuration file has been hard-coded.

1- Can you create a “template” of config.json, using Ansible and Jinja2, then upload it on server from the “deploy” role ?
2- The HTTP port and Redis hostname should be configured in group_vars/all.yml. This is the right place to write your variables with Ansible.
3- Ooops, sorry, the Redis hostname must be set dynamically, based on inventory.
4- The SystemD service must be restarted (only!) when application is updated. If you deploy the app many times without changing anything, the service should not restart.

+ KEEP SECRETS ... SECRETS

Ansible comes with a tool that encrypt passwords: Vault.

Could you add an encrypted variable “password” containing “usa-secrets” to the group_vars/all.yml file ?

Then print it into /tmp/snowden.txt on “redis” server.

+ A LONG TIME AGO IN A GALAXY FAR, FAR AWAY...

Red Hat, who leads the Ansible project, built a registry for roles: `https://galaxy.ansible.com/`.

On this platform, find a Role for installing both Docker and Docker-compose. Then add this role to your playbook.

Hint: geerlingguy is a very nice man ;)

+ DEBUGGING TIPS

The following tasks prints the value of `name` variable.

```
- name: Create variable
  set_fact:
    name: world

- name: Print debug
  debug:
    msg: "Hello {{ name }}"
```