T7 - Network & Sys Admin

T-NSA-700

devOps

Bootstrap

1.1

{EPITECH.}

Creation of a Linux template

The first step is to create a Linux machine template that can be used for all deployments. A Debian for example.

This template will allow you to quickly deploy machines as needed, and not have to re-install and reconfigure some basic settings.

Do not install GUI when creating your template, it will slow down your work.

For the template creation, you must :

- install a minimal Debian server
- configure the users and password you want
- add your own public SSH key to the virtual machine
- set up an IP address reserved for the creation of a new machine. It must always be free.

Next, prepare the machine to be turned off, then backed up or converted into a template. This often involves deleting temporary files and information specific to the current machine.

You can then turn off the machine.

For the next machines, you just have to clone that one, or, depending on your hypervisor, create a model of that one and redeploy it.

Ansible : Getting started

Ansible is an automation platform that can run on Linux in standlone, without any server to install and allows you to configure servers using *YAML* files.

If you do not know Ansible, refer to your peers, Google or JARVISS.

First, install the necessary elements to run Ansible on your computer.

Then, deploy a blank machine that will serve as a test.

Ansible uses SSH to do its deployments, this machine will have to be accessible from your computer, in SSH.

Create your Ansible inventory file, adding this blank machine.

Try to execute the `ping` role on this machine.

Continue by trying out the different basic Ansible roles : package installation, file configuration …

Ansible : First role

To create your roles and use them, the best way is to store them in different Git repositories. Then you'll be able to use them with Ansible.

Create a first **base** role that will perform the following tasks:

- install "emacs-nox" (lightweight emacs, distributed without GUI/X)
- put the machine name / date in the default terminal prompt
- create a user and assign them a public SSH key

Put your role in a git repository, and try to apply it with Ansible, using an inventory file and a `requirements.yml` file. You can now work on the other necessary roles for the project.

Ansible : Inventory, Variable, Vault & Environment

Thanks to the inventory file, you will be able to inform the various machines that you want to configure, and depending on their type and the groups you will make, apply appropriate roles and variables to them.

To be able to effectively use the roles you have created, it seems better to dynamize as many parameters as possible.

Some of these parameters could be quite sensitive. Take a look on *Ansible Vault*

Create a **MySQL** role, which installs and configures a MySQL instance with the password and the user passed as a parameter.

Then create an **Nginx role**, which installs an Nginx server, and create a file containing the MySQL connection information in `/root/`, and an `index.html` file in`/var/www/html`.
Your inventory file will contain four groups:

- Linux : Contains three machines
- Nginx : Contains two machines
- FPM : Contains one machine
- MySQL : Contains one machine

For the Linux group, the **base** role is applied, with the SSH key retrieved from the Vault.
For the MySQL group, the MySQL role is applied, with the user and the password retrieved from the Vault. Same to retrieve the login information on the Nginx server.

In your ansible folder, you will deposit two `index.html` files for your production and development environment. Using an `env` variable passed to Ansible's execution, the file deployed on the Nginx server must be the correct one.

Using Gitlab

In order to host the different Git repositories, you will be able to deploy and configure a Gitlab server. You have several ways to do this that are already documented in the official documentation.

One of the easiest ways is to deploy it using the *Omnibus* package. Install it and connect to the interface to activate the first connection.

Acknowledge that an IP address is mandatory to access an instance on a VM.

Configure your Gitlab server, according to your needs.

Gitlab integrates a lot of very useful tools that you will need for your projects! Therefore we ask you to deploy it on a dedicated instance, which you will keep in the long run.