1 scalable automated deployment of cluster VM environment

* 1. background

Due to the limited speed of hardware development, currently the power of an individual server is far away from supporting a large-scale application. With such a background, cloud solutions, a multiple-node, scalable distributed system, is necessary for programs development. To meet various needs from different applications and the need of setup and configuration of a large amount of VMs in cloud service, many professional automation management tools are developed.

* 1. Requirement

Besides scalability and automated deployment, we also have other requirements about an automation tool. Firstly, good compatibility is important evaluation metric for automation tools. They are expected to be compatible with different cloud platform, hardware, OS, environments, tools and software. Secondly, a successful automation tool need a good solution to avoid any security problems during operation. Thirdly, as the users and potential users of cloud service may come from different fields and the usability and cost should be considered.

* 1. automated cloud provisioning

As the management tools base on the API offered by cloud platform, although there are third-party program, the tools recommended by the platform are usually first considered. For Nectar, a private Openstack cloud platform, Boto is a good to tool to manage Nectar.

* 1. types of automated tools
     1. automated tools at system level

Compared with container technology, many famous automated tools working at system level. They can be categorized by the methods they work on remote machines. Some are agent based which means an agent should be firstly installed on remote OS and the commands work base on the support of agent. There are also other tools working by pull some small program or a short code working like a software to achieve remote control. Many well-developed tools in this area, like Chef, Fabric and ansible.

* + 1. container technology

Container technology is an idea different with visualization, which works within OS. It builds virtual environment according to the software and application user need and isolates resources above system level for each container. The automated deployment is a derivative function during this concept formation. There is a trend that container is become popular. In automated deployment it has some advantages which is that the usage of image simplifies the difficulty of configuration of some common but complex software. However, in contrast, the principle of virtual environment makes the deployment relevant to OS more complex. It also need more professional specialists, which means worse usability. Docker are up-rising solution in this area.

* 1. introduction of Boto and Ansible

In this assignment, boto used as a library in python, which is user-friendly with the widespread of python language. It directly accesses the API offered by Nectar with access token. Besides, Ansible is used for deployment tool. It is a free software and works under Linux/Unix based systems. There are a lot benefits using Ansible. Firstly, it can work on various operating system as remote system, including windows, different kinds of Linux/Unix based system etc. Secondly, it works without any pre-configuration on remote machines. Thirdly, it uses SSH as connection, which solves security problems in connection process. Fourthly, the language, YAML, is simple enough for users with different level of computing skills.

* 1. cloud provisioning

In this assignment, boto in python is used for creating or terminating instances in Nectar. The script accepts three parameters including ‘-c’ which specifies the user’s need to create or terminate instances, ‘-n’ to indicate how many instances user want to create and ‘-v’ to indicate the volume size to be attached for each instance.

In boto script, a connection to nectar is first established and method ‘run\_instances’is called for creating instances while ‘terminate\_instances’ is used for terminating instances.

In addition to these methods, the concept in nectar is showed in boto as object. An ‘instance’ object contains nearly all parameters of a instance on nectar as attributes, including private\_ip\_address, status etc. By access these attributes, user can use program language to operate nectar, by which user can achieve more powerful and efficient operation than GUI.

1.7 configuration setup

Ansible comprises several components. An inventory file is used to store information of candidate machines. A playbook script to call Ansible modules to operate remote machines.

1. inventory

The default inventory in ansible is hosts file located at /etc/ansible. In inventory level, ansible offers several function. Firstly is group function, which means the machines in the inventory files can belong to different group. In our work,a machine is grouped into five groups by its role in the system. The five groups are common, couchdb, harvestor, web, spark\_slave. Some configuration can also be set in inventory level. For example, the path of private key file is specified in inventory as our group member have different keys for different instances. This not only save the time to input the file path as parameters and avoid complex shell command but also lower the chance to expose the path of private key.

2. roles in playbook

To improve the structure, reuse and readability of code, the roles in playbook is used for VMs with different functions. Each role has its own main.yml in corresponding tasks folder to list the commands it need to do. Besides, files folder stores files which are used during deployment. Templates folder is used with the template module in ansible, which allows retrieve contents of file after templating with Jinja2. There are also some other folders supporting other functions, like handlers and vars, but they are not used in this assignment.

In root folder, a father yml file is used for

site.yml

webservers.yml

fooservers.yml

roles/

common/

tasks/

handlers/

files/

templates/

vars/

defaults/

meta/

webservers/

tasks/

defaults/

meta/