**Học viện chuyên đào tạo Phần cứng máy tính và Mạng**

**FPT- Jetking**

-----🖳-----

****

**BÁO CÁO ĐỒ ÁN**

**Tên đề tài:**

**NGHIÊN CỨU GIẢI PHÁP TỰ ĐỘNG HÓA ĐỂ QUẢN LÝ HỆ THỐNG BẰNG CÔNG CỤ ANSIBLE**

**Giảng viên hướng dẫn:** Lê Văn Thuận

**Nhóm:**

|  |
| --- |
| **Phạm Phú Trung Dũng** |
| **Hán Hoàng Lâm** |
| **Nguyễn Công Quốc Chung** |

Lớp: J1902E

Hà Nội, ngày 30 tháng 10 năm 2020

**MỤC LỤC**

1. LỜI MỞ ĐẦU3
2. Lý do chọn đề tài 3
3. Tổng quan vấn đề nghiên cứu 3
4. Mục tiêu nghiên cứu của đề tài 4
5. Đối tượng và phạm vi nghiên cứu của đề tài 4
6. Phương pháp nghiên cứu của đề tài 4
7. NGHIÊN CỨU GIẢI PHÁP TỰ ĐỘNG HÓA ĐỂ QUẢN LÝ HỆ THỐNG BẰNG ANSIBLE 5
8. Ansible là gì?6
9. Tại sao lại là ansible 6
10. Ứng dụng7
11. Một số thuật ngữ 7
12. Kiến trúc Ansible 8
13. Cách thức hoạt động 9
14. Cài đặt Ansible10
15. Sử dụng một số lệnh kiểm tra cơ bản14
16. Tìm hiểu về playbook của Ansible25
17. Role26
18. Demo27
19. KẾT LUẬN27
20. TÀI LIỆU THAM KHẢO28
21. **LỜI MỞ ĐẦU**

## Lý do chọn đề tài

Điện toán đám mây không còn là một khái niệm mới ở Việt Nam cũng như  
trên thế giới. Sự phát triển nhanh chóng và mạnh mẽ của điện toán đám mây đã đặt ra nhiều thách thức đối với các nhà cung cấp dịch vụ (cloud providers) cũng như các doanh nghiệp, tổ chức sử dụng nó. Áp lực đó không chỉ là cơ sở hạ tầng (hệ thống máy chủ, hệ thống mạng, trung tâm dữ liệu .v.v.) mà quan trọng hơn đó là yếu tố con người.

Đối với doanh nghiệp, điện toán đám mây giúp giảm thiểu thời gian và  
chi phí đưa sản phẩm tới người dùng. Đối với quản trị hệ thống, điện toán  
đám mây mang lại khả năng mở rộng hệ thống một cách nhanh chóng, nhưng  
đồng thời nó cũng đặt ra thách thức trong việc quản lý hạ tầng máy chủ. Với  
số lượng hàng trăm server thì việc phải quản lý thủ công theo cách truyền  
thống là không hiệu quả và đem lại nguy cơ cao cho hệ thống. Từ yêu cầu  
thực tế đó, các framework cho việc tự động hóa hệ thống ra đời. Chúng cho  
phép người quản trị hệ thống có thể quản lý cấu hình của các máy chủ, đồng  
bộ hóa chúng với nhau, đồng thời đảm bảo việc nó không bị thay đổi sai bởi  
sự vô tình của con người

Từ nhu cầu về tự động hóa hệ thống của các tổ chức, doanh nghiệp, nhóm chúng em xin chọn đề tài nghiên cứu “NGHIÊN CỨU GIẢI PHÁP TỰ ĐỘNG HÓA ĐỂ QUẢN LÝ HỆ THỐNG BẰNG CÔNG CỤ ANSIBLE”

## Tổng quan vấn đề nghiên cứu

Sự gia tăng của ảo hóa cùng với sức mạnh ngày càng tăng của các máy chủ chuẩn công nghiệp và sự sẵn có của điện toán đám mây đã dẫn đến một sự gia tăng đáng kể số lượng các máy chủ cần phải được quản lý trong và ngoài các doanh nghiệp, tổ chức. Trước kia, chúng ta có thể làm việc với hàng loạt các máy chủ vật lý hay truy cập trực tiếp vào các trung tâm dữ liệu nhưng giờ đây, chúng ta phải quản lý nhiều hơn rất nhiều các máy chủ và chúng lại còn được đặt ở rất nhiều các trung tập dữ liệu trên toàn thế giới.

Những trung tâm dữ liệu - đó chính là nơi các công cụ tổ chức và quản lý cấu hình vào cuộc. Trong nhiều trường hợp, chúng ta quản lý nhiều nhóm máy chủ giống hệt nhau, chạy các ứng dụng và dịch vụ giống hệt nhau. Chúng được triển khai trên các nền tảng ảo hóa trong các doanh nghiệp, tổ chức, hay ở trên hệ thống điện toán đám mây hoặc một máy chủ nào đó tại một trung tâm dữ liệu ở xa. Điều đó dẫn đến cần có một công cụ tự động hóa để có thể quản lý các cơ sở hạ tầng lớn và đang phát triển không ngừng này

Việc áp dụng giải pháp tự động hóa để quản lý nhằm:

* Đảm bảo việc quản lý cấu hình của thiết bị, ứng dụng một cách hiệu quả, tức là quản lý đơn giản, kiểm soát được các cấu hình đúng và đủ hay chưa, chúng chạy có chính xác hay không.
* Tiết kiệm được công sức & thời gian khi phải triển khai đi triển khai lại.
* Tái sử dụng được các bước triển khai trước đó (các bước lặp đi lặp lại khi cài đặt, cấu hình máy chủ, cấu hình ứng dụng)

## Mục tiêu nghiên cứu của đề tài

Mục tiêu nghiên cứu của đồ án là tìm hiểu các yêu cầu, giải pháp tự động hóa để quản lý hệ thống bằng Ánible nhằm mục đích triển khai áp dụng trong thực tế.

## Đối tượng và phạm vi nghiên cứu của đề tài

Đối tượng và phạm vi nghiên cứu của đồ án là các giải pháp tự động hóa cho việc quản lý và quản trị hệ thống bằng Ansible.

## Phương pháp nghiên cứu của đề tài.

Thu thập, khảo sát, tìm hiểu các tài liệu liên quan đến giải pháp tự động hóa để quản lý hệ thống bằng Ansible.

1. **NGHIÊN CỨU GIẢI PHÁP TỰ ĐỘNG HÓA ĐỂ QUẢN LÝ HỆ THỐNG BẰNG ANSIBLE**

## Giới thiệu

Ansible là một giải pháp đơn giản được sử dụng trong việc tự động hóa hàng loạt các công việc liên quan đến hạ tầng CNTT. Trong mô hình của Ansible, hạ tầng CNTT được nhìn góc độ là một kiến trúc tổng thể gồm các thành phần có liên quan tới nhau thay vì chỉ là một hệ thống riêng lẻ.

Ansible là một công cụ dùng để tự động hóa việc cấu hình trên nhiều server. So với các công cụ khác với tính năng tương đương thì Ansible dễ học và dễ tiếp cận hơn rất nhiều. Cộng đồng người dùng cũng nhiều hơn so với các công cụ khác.

So sánh một vài thông số về **Ansible, SaltStack, Chef, Puppet** trên **GitHub (11/2018)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Ansible** | **SaltStack** | **Chef** | **Puppet** |
| **Số sao trên GitHub** | 33,500 | 9,341 | 5,543 | 5,125 |
| **Số lượt fork** | 13,338 | 4,366 | 2,268 | 2,049 |
| **Ngôn ngữ lâp trình** | Python | Python | Ruby | Ruby |
| **Ngôn ngữ cấu hình** | YAML | YAML | Ruby DSL | Puppet DSL |

Có thể thấy Ansible là công cụ tự động hóa phổ biến nhất trên GitHub với số sao được người dùng bình chọn cho project này là 33,500 sao. Ansible cũng là tool dễ tiếp cận và làm quen do được build bằng Python và sử dụng file cấu hình theo dạng YAML (**Y**AML **A**in’t **M**arkup **L**anguage) dễ đọc và dễ hiểu.

1. Ansible là gi?

Ansible là một công cụ mã nguồn mở, sử dụng ngôn ngữ lập trình là python. Ansible rất đơn giản để sử dụng vì nó sử dụng giao thức ssh để liên lạc giữa các máy với nhau. Ansible sử dụng playbook để định nghĩa ra các công việc mà ta muốn triển khai một cách tự động và playbook được viết bằng định dạng yaml.

Ansible cung cấp độ tin cậy, tính nhất quán và khả năng mở rộng cho cơ sở hạ tầng CNTT của bạn.

## Tại sao lại là ansible?

* Việc duy trì và quản lý từ 1 đến 2 server khá là dễ dàng nhưng khi chúng ta có một cơ sở hạ tầng IT phức tạp với rất là nhiều sever trong khoảng từ hàng chục đến hàng trăm thì việc quản lý sẽ càng trở nên khó khăn hơn.
* Tức là những công việc lặp đi lặp lại như update, backup, tạo user, chỉ định nhóm, phân quyền, vv… đều phải làm bằng tay và bằng cách vào từng con server một để cấu hình cho chúng. Khiến chúng ta mất rất nhiều t/g, công sức và thậm chí là gây nên lỗi hệ thống or những sai lầm và từ đó chúng ta lại bắt đầu mọi thứ từ đầu.
* Đó là lúc chúng ta cần đến Ansible khiến cho công việc trở nên hiệu quả hơn và tiết kiệm t/g
  + Thực thi các công việc của tất cả server cùng 1 lúc trên ngay chính server chủ mà ko cần phải truy cập vào từng con server phụ nữa
  + Các bước Config/Install/Deploy chỉ cần đúng một file YAML thay vì làm thủ công hoặc kết hợp shell script một cách thủ công
  + Có thể tái sử dụng cùng một file để thực hiện cùng một chuỗi task nhiều lần nếu cần or trên các môi trường khác nhau.
  + Con người thì hay mắc lỗi hoặc quên j đó nhất là khi thực thi các chuỗi công việc IT phức tạp nên việc tự động hóa bằng Ansible khiến nó trở nên đáng tin cậy và ít mắc lỗi hơn
* Việc Ansible hỗ trỡ hầu hết các cơ sở hạ tầng cũng như là hệ điều hành hoặc nhà cung cấp dịch vụ điện toán đám mây khiến cho nó rất được ưa thích.
* Ansible là một agentless và đó là một lợi thế độc nhất tức là chúng ta không cần phải cài đặt agent lên các server mục tiêu mà chỉ cần cài đặt trên đúng một máy đó là main server. Qua đó quản lý và diều khiển toàn bộ hạm đội sever qua giao thức từ xa là ssh.

## Ứng dụng

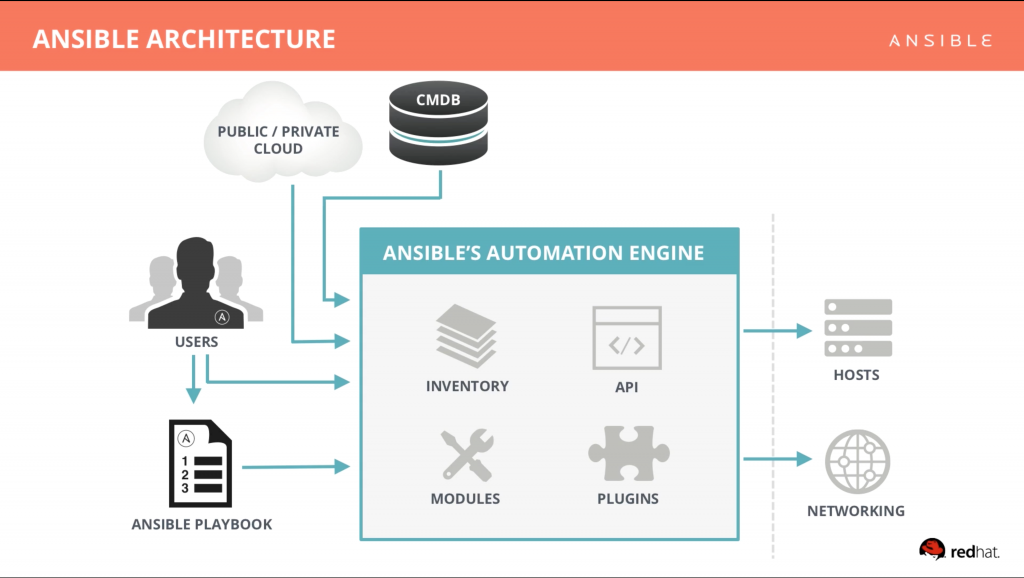
Ansible có rất nhiều ứng dụng trong triển khai phần mềm và quản trị hệ thống.

* **Provisioning:** Khởi tạo VM, container hàng loạt trong môi trường cloud dựa trên API (OpenStack, AWS, Google Cloud, Azure…)
* **Configuration Management:**Quản lý cấu hình tập trung các dịch vụ tập trung, không cần phải tốn công chỉnh sửa cấu hình trên từng server.
* **Application Deployment:** Deploy ứng dụng hàng loạt, quản lý hiệu quả vòng đời của ứng dụng từ giai đoạn dev cho tới production.
* **Security & Compliance:** Quản lý các chính sách về an toàn thông tinmột cách đồng bộ trên nhiều môi trường và sản phẩm khác nhau (deploy policy, cấu hình firewall hàng loạt trên nhiều server…).

## Một số thuật ngữ

* **Controller Machine**: Là máy cài Ansible, chịu trách nhiệm quản lý, điều khiển và gởi task tới các máy con cần quản lý.
* **Inventory**: Là file chứa thông tin các server cần quản lý. File này thường nằm tại đường dẫn /etc/ansible/hosts.
* **Playbook**: Là file chứa các task của Ansible được ghi dưới định dạng YAML. Máy controller sẽ đọc các task trong Playbook và đẩy các lệnh thực thi tương ứng bằng Python xuống các máy con.
* **Task**: Một block ghi tác vụ cần thực hiện trong playbook và các thông số liên quan. Ví dụ 1 playbook có thể chứa 2 task là: yum update và yum install vim.
* **Module**: Ansible có rất nhiều module, ví dụ như moduel yum là module dùng để cài đặt các gói phần mềm qua yum. Ansible hiện có hơn ….2000 module để thực hiện nhiều tác vụ khác nhau, bạn cũng có thể tự viết thêm các module của mình nếu muốn.
* **Role**: Là một tập playbook được định nghĩa sẵn để thực thi 1 tác vụ nhất định (ví dụ cài đặt LAMP stack).
* **Play**: là quá trình thực thi của 1 playbook
* **Facts**: Thông tin của những máy được Ansible điều khiển, cụ thể là thông tin về OS, network, system…
* **Handlers**: Dùng để kích hoạt các thay đổi của dịch vụ như start, stop service.

## Kiến trúc Ansible



Inventory là danh sách các node hoặc host có địa chỉ IP, cơ sở dữ liệu, máy chủ, v.v. cần được quản lý.

API Ansible hoạt động như một phương tiện vận chuyển cho các dịch vụ cloud private/public.

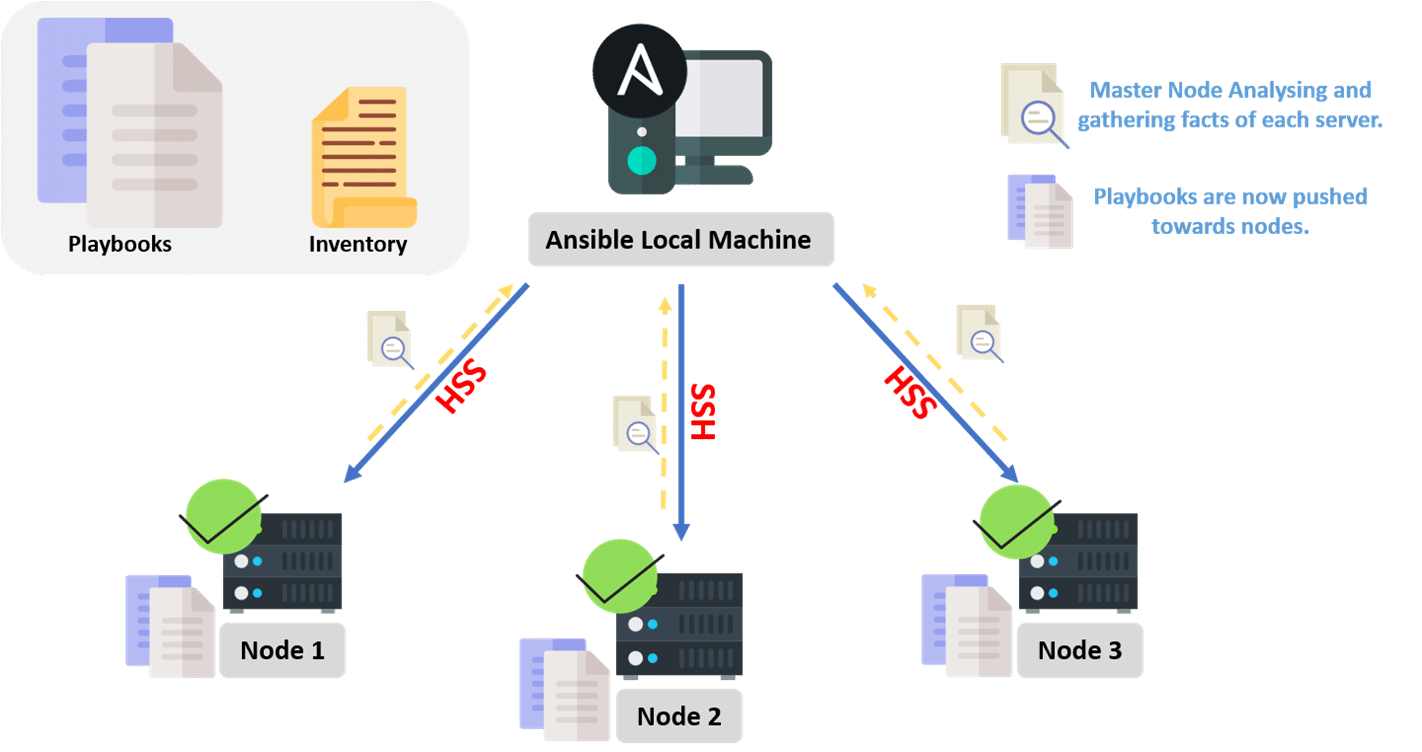
Modules là các đơn vị thực thi mã Ansible. Mỗi mô-đun có một mục đích sử dụng cụ thể, từ quản lý người dùng trên một loại cơ sở dữ liệu cụ thể đến quản lý các giao diện VLAN trên một loại thiết bị mạng cụ thể. Bạn có thể gọi một mô-đun duy nhất với một nhiệm vụ hoặc gọi một số mô-đun khác nhau trong một playbook.

Plugin là một đoạn mã sử dụng chức năng cốt lõi của Ansible. Có rất nhiều plugin hữu ích và bạn cũng có thể tự viết plugins cho chính mình.

CMDB là một loại kho lưu trữ hoạt động như một kho dữ liệu cho các cài đặt IT của bạn.

Cloud là một mạng lưới các máy chủ từ xa mà bạn có thể lưu trữ, quản lý và xử lý dữ liệu. Các máy chủ này được lưu trữ trên internet, chúng lưu trữ dữ liệu từ xa thay vì lưu trữ ở máy chủ cục bộ. Nó chỉ khởi chạy các tài nguyên và phiên bản trên cloud, kết nối chúng tới máy chủ và bạn có thể vận hành các tác vụ của mình từ xa.

## Cách thức hoạt động



Giống như đa phần các phần mềm quản lý cấu hình tập trung khác. Ansible có 2 loại server là control machine và node. Control machine là máy có trách nhiệm quản lý các node con trong hệ thống. Đây cũng là máy lưu trữ các thông tin về các node, playbook và các script cần dùng để deploy trên các node khác qua giao thức SSH.

Để quản lý các node, Ansible sẽ gởi các module lệnh tới các node con qua SSH. Các module lệnh này sẽ được lưu trữ tạm thời trên các node con và giao tiếp với máy chủ Ansible bằng JSON. Khi đã thực thi xong tác vụ trên các máy này, các module đó sẽ được xóa đi. Các module này thường được lưu ở folder /root/.ansible hoặc /home/<user>/.ansible, tùy theo user mà Ansible dùng để quản lý các node con.

Khi Ansible ở chế độ rảnh, ko có task để thực hiện máy chủ Ansible sẽ không chiếm dụng tài nguyên do Ansible không sử dụng trình daemon hoặc program chạy ở chế độ background. Chỉ khi nào thực thi lệnh thì Ansible mới sử dụng tài nguyên của hệ thống.

## Cài đặt Ansible

Thiết lập hostname, IP cho các node

|  |  |  |
| --- | --- | --- |
| **Hostname** | **OS** | **IP** |
| AnsibleServer | CentOS7 | 10.88.88.10 |
| Slave1 | CentOS7 | 10.88.88.20 |
| Slave2 | CentOS7 | 10.88.88.21 |
| Slave3 | Ubuntu 18.04 | 10.88.88.30 |
| Slave4 | Centos 6 | 10.88.88.40 |

*Cài đặt Ansible trên node Ansible Server*

Việc cài đặt ansible khá đơn giản, chỉ thông qua lệnh bên dưới là có thể cài xong Ansible.

Lưu ý: Bước này thực hiện trên node AnsibleServer node 10.88.88.10

Login với tài khoản root và thực hiện các lệnh dưới (lưu ý lúc nếu bạn có IP Planning khác, nên đặt IP tĩnh theo phân hoạch của bạn trước).

yum install -y epel-release

yum update -y

yum install -y ansible

Kiểm tra lại phiên bản ansible sau khi cài bằng lệnh anssible --version. Ta có kết quả phiên bản ở dưới.

[root@Ansible ~]# ansible --version

ansible 2.9.13

config file = /etc/ansible/ansible.cfg

configured module search path = [u'/root/.ansible/plugins/modules', u'/usr/share/ansible/plugins/modules']

ansible python module location = /usr/lib/python2.7/site-packages/ansible

executable location = /usr/bin/ansible

python version = 2.7.5 (default, Apr 2 2020, 13:16:51) [GCC 4.8.5 20150623 (Red Hat 4.8.5-39)]

## Cấu hình SSH Key

Ansible hoạt động theo cơ chế agentless, có nghĩa là không cần cài agent vào các máy client để điều khiển, thay vào đó ansible sẽ sử dụng việc điều khiển các client thông qua SSH.

Sử dụng ssh keypair. Có nghĩa là ta sẽ tạo ra private key và public key trên node AnisbleServer và copy chúng sang các node client (hay còn gọi là các host).

Đứng tại node AnsibleServer tạo SSH Key, sau đó copy các sang node còn lại. Mục tiêu là sử dụng keypair để không phải nhập password mỗi khi đăng nhập vào các client.

Đứng tại user root của node AnsibleServer và thực hiện bước tạo key

ssh-keygen

Thực hiện các thao tác Enter và để mặc định các tùy chọn khi lab.

Generating public/private rsa key pair.

Enter file in which to save the key (/root/.ssh/id\_rsa):

Enter passphrase (empty for no passphrase):

Enter same passphrase again:

Your identification has been saved in /root/.ssh/id\_rsa.

Your public key has been saved in /root/.ssh/id\_rsa.pub.

The key fingerprint is:

SHA256:ialESG10iMjC8ppgMWh46DaB34s7iuFwbUCB6a0FDP4 root@ansibleserver

The key's randomart image is:

+---[RSA 2048]----+

|X=o+... |

|%@oo+. |

|\*=Oo. |

|.B++. o . |

|+o=E..o S |

|o..+.. |

|o ..+ |

|+oo. |

|oo . |

+----[SHA256]-----+

Thực hiện copy file key sang các node còn lại

ssh-copy-id root@10.88.88.20

Ấn Yes và nhập mật khẩu root của node Slave1

[root@Ansible ~]# ssh-copy-id root@10.88.88.20

/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id\_rsa.pub"

/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed

/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys

root@10.88.88.20's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'root@10.88.88.20'" and check to make sure that only the key(s) you wanted were added.

Sau đó đứng từ node AnsibleServer, thử ssh sang các node Slave1, nếu không bị hỏi mật khẩu thì ta đã sử dụng ssh key thành công.

[root@Ansible ~]# ssh root@10.88.88.20

Last login: Fri Oct 30 21:54:54 2020

[root@Slave1Centos7 ~]#

Lưu ý nhớ dùng lệnh exit sau khi đăng nhập ở từng node.Tiếp tục chuyển sang bước khai báo file inventory để chứa danh sách các host cần thiết.

## Khai báo file inventory

Thực hiện việc này trên node AnsibleServer

Mặc định thì danh sách các host mà AnsibleServer điều khiển sẽ nằm ở file /etc/ansible/hosts. File mặc định này sẽ chứa các khai báo mẫu, ta sẽ thực hiện sao lưu lại và khai báo file theo bài LAB này.

mv /etc/ansible/hosts /etc/ansible/hosts.org

Tạo file mới để khai báo các host

vi /etc/ansible/hosts

Nội dung của file sẽ có dạng như sau

[centos7]

10.88.88.20

10.88.88.21

[ubuntu]

10.88.88.30

[centos6]

10.88.88.40

Sau khi thêm các dòng trên, ta lưu file trên lại. Chú ý rằng ta có thể sử dụng cặp thẻ [ ] để khai báo các group. Các group này sẽ do ta quy hoạch sao cho phù hợp với hệ thống, với ứng dụng của chúng ta. Thực hiện kiểm tra danh sách host đã đươc khai báo trong file inventory ở trên bằng lệnh ansible all --list-hosts. Trong đó all là một tùy chọn của lệnh trên, mục tiêu là liệt kê tất cả các hosts nằm trong file inventory, bất kể các host đó nằm ở group nào.

ansible all --list-hosts

Kết quả:

[root@Ansible ~]# ansible all --list-hosts

hosts (4):

10.88.88.20

10.88.88.21

10.88.88.30

10.88.88.40

Nếu chúng ta chỉ muốn kiểm tra các host trong group [centos7] ta thực hiện với tùy chọn lệnh như sau ansible centos7 --list-host, kết quả sẽ như bên dưới.

[root@Ansible ~]# ansible centos7 --list-host

hosts (2):

10.88.88.20

10.88.88.21

Tới bước này chúng ta mới dừng lại việc khai báo danh sách các host. Trong thực tế các ta cần khai báo thêm các tùy chọn về mật khẩu, về port thậm chí cả về user mà AnsibleServer được phép sử dụng để điều khiển các host.

## Sử dụng một số lệnh kiểm tra cơ bản

Để kiểm tra xem việc khai báo này đã đúng và bắt đầu tiếp các bài lab tiếp theo hay chưa, hãy sử dụng tùy chọn -m trong lệnh dưới, đây là dấu hiệu nhận biết về việc sẽ sử dụng module có tên là ping (-m là viết tắt của module). Trong bài này sẽ giới thiệu thêm một số moudle khác ngoài module ping này.

ansible all -m ping

Như ở trên đã nói hoặc trong các tài liệu khác có đề cập, ansible có khái niệm sử dụng các module được cung cấp sẵn để người quản trị làm các công việc cấu hình, sử dụng ansible cho hợp lý. Việc sử dụng module trong ansible rất linh hoạt và có rất nhiều module, ta cần đọc hoặc tra cứu từ trang chủ về các module mà ansible hỗ trợ.

Kết quả của lệnh trên như bên dưới

[root@Ansible ~]# ansible all -m ping

10.88.88.20 | SUCCESS => {

"ansible\_facts": {

"discovered\_interpreter\_python": "/usr/bin/python"

},

"changed": false,

"ping": "pong"

}

10.88.88.21 | SUCCESS => {

"ansible\_facts": {

"discovered\_interpreter\_python": "/usr/bin/python"

},

"changed": false,

"ping": "pong"

}

10.88.88.40 | UNREACHABLE! => {

"changed": false,

"msg": "Failed to connect to the host via ssh: ssh: connect to host 10.88.88.40 port 22: Connection timed out",

"unreachable": true

}

10.88.88.30 | UNREACHABLE! => {

"changed": false,

"msg": "Failed to connect to the host via ssh: ssh: connect to host 10.88.88.30 port 22: Connection timed out",

"unreachable": true

}

Kết quả trên cho thấy slave1 và slave2 phản hồi về trạng thái màu xanh. Có nghĩa là đứng từ AnsibleServer ta đã ping thành công tới slave1 và slave 2, còn màu đỏ chỉ ra rằng AnsibleServer không tìm thấy slave3 và slave4

Sở dĩ có kết quả như trên là vì trong phần này của bài LAB, tôi đã khai báo đủ 04 node client nhưng mới có 02 node slave1 & slave2 là được bật khi tôi thực hiện lab.

Ta có thể lựa chọn các group để thực hiện tương tác thông qua Ansible, ví dụ tôi sẽ thực hiện module ping với group là ubuntu đã khai bào ở trên bằng lệnh ansible ubuntu -m ping. Kết quả là

[root@Ansible ~]# ansible ubuntu -m ping

10.88.88.30 | SUCCESS => {

"ansible\_facts": {

"discovered\_interpreter\_python": "/usr/bin/python3"

},

"changed": false,

"ping": "pong"

}

* Các module cơ bản của ansilbe trong chế độ tương tác

Chế độ tương tác của ansible tạm hiểu là chế độ dùng các lệnh đơn, ngay sau khi thực hiện lệnh thì sẽ có output để quan sát.

Trong thực tế với mức độ sử dụng như này thì bạn mới ở mức con số 0 về Ansible, nó chỉ có ý nghĩa cho bạn vọc vạch ansible với trình độ khởi đầu và dùng để thao tác nhanh trong ngữ cảnh cần kiểm tra hoặc xác nhận lại các config trước đó mà thôi.

Để phát huy sức mạnh của Ansible, bạn phải luyện tới các trình độ cao cấp hơn, ví dụ như sử dụng được các playbooks, tự viết được playbooks hoặc hiểu các khái niệm về task, vars …

### Sử dụng module command

Như đã nói ở trên, việc sử dụng module ở chế độ tương tác của ansible chưa thể phát huy được hiệu quả của Ansible, nhưng bạn vẫn cần có các bước đi khởi đầu này để hiểu và phân biệt được cách sử dụng ansible cho hiệu quả trong tương lai.

Sử dụng một trong các câu lệnh dưới để thực hành moulde command, ta sẽ kiểm tra uptime của toàn bộ các node hoặc một số node thuộc group nào đó bằng lệnh "ansible all -m command -a "ten\_cau\_lenh\_module\_command\_ho\_tro". Ví dụ:

Lưu ý thêm có tùy chọn -a để truyền thêm các đầu vào cho module command. Tùy chọn này cũng được dùng trong các module khác nhé.

[root@Ansible ~]# ansible centos7 -m command -a "uptime"

10.88.88.20 | CHANGED | rc=0 >>

22:41:49 up 47 min, 2 users, load average: 0.00, 0.01, 0.03

10.88.88.21 | CHANGED | rc=0 >>

17:20:43 up 58 min, 2 users, load average: 0.08, 0.03, 0.05

Thực hành thêm với lệnh ansible all -m command -a "uname -rms" và xem kết quả trả về phiên bản kernel của các host

[root@Ansible ~]# ansible all -m command -a "uname -rms"

10.88.88.20 | CHANGED | rc=0 >>

Linux 3.10.0-693.el7.x86\_64 x86\_64

Ta quan sát thấy trong các output trên là thông tin trả về khi module command thực hiện lệnh uptime trong linux.

Thực hành thêm với lệnh ansible all -m command -a "uname -rms" và xem kết quả trả về phiên bản kernel của các host nhé.

[root@Ansible ~]# ansible all -m command -a "uname -rms"

10.88.88.21 | CHANGED | rc=0 >>

Linux 3.10.0-957.el7.x86\_64 x86\_64

10.88.88.20 | CHANGED | rc=0 >>

Linux 3.10.0-693.el7.x86\_64 x86\_64

10.88.88.30 | CHANGED | rc=0 >>

Linux 4.15.0-121-generic x86\_64

10.88.88.40 | CHANGED | rc=0 >>

Linux 2.6.32-754.33.1.el6.x86\_64 x86\_64

### Tới đây tạm hiểu tính năng module command của ansible là cho phép ta dùng ansible để thực thi các lệnh của Linux từ xa, đa số là các lệnh đơn thôi nhé.

### Module setup trong ansible

Ta có thể sử dụng module setup để kiểm tra các thông tin tổng quát về hệ điều hành của các node, ví dụ kiểm tra phiên bản, kiểm tra thông tin card mạng, tên host, thông số về phần cứng ….

Ví dụ ta có thể sử dụng lệnh sau để kiểm tra xem distro của các host là gì ansible all -m setup -a 'filter=ansible\_distribution'. Kết quả ta sẽ nhìn thấy như sau:

### [root@Ansible ~]# ansible all -m setup -a 'filter=ansible\_distribution'

### 10.88.88.20 | SUCCESS => {

### "ansible\_facts": {

### "ansible\_distribution": "CentOS",

### "discovered\_interpreter\_python": "/usr/bin/python"

### },

### "changed": false

### }

### 10.88.88.21 | SUCCESS => {

### "ansible\_facts": {

### "ansible\_distribution": "CentOS",

### "discovered\_interpreter\_python": "/usr/bin/python"

### },

### "changed": false

### }

### 10.88.88.40 | SUCCESS => {

### "ansible\_facts": {

### "ansible\_distribution": "CentOS",

### "discovered\_interpreter\_python": "/usr/bin/python"

### },

### "changed": false

### }

### 10.88.88.30 | SUCCESS => {

### "ansible\_facts": {

### "ansible\_distribution": "Ubuntu",

### "discovered\_interpreter\_python": "/usr/bin/python3"

### },

### "changed": false

### }

### Hoặc ta có thể sử dụng lệnh ansible all -m setup -a 'filter=ansible\_default\_ipv4' để liệt kê ra các địa chỉ IPv4 trên các node. Kết quả sẽ trả về

[root@Ansible ~]# ansible all -m setup -a 'filter=ansible\_default\_ipv4'

10.88.88.20 | SUCCESS => {

"ansible\_facts": {

"ansible\_default\_ipv4": {

"address": "10.88.88.20",

"alias": "ens33",

"broadcast": "10.88.88.255",

"gateway": "10.88.88.2",

"interface": "ens33",

"macaddress": "00:0c:29:98:41:99",

"mtu": 1500,

"netmask": "255.255.255.0",

"network": "10.88.88.0",

"type": "ether"

},

"discovered\_interpreter\_python": "/usr/bin/python"

},

"changed": false

}

10.88.88.21 | SUCCESS => {

"ansible\_facts": {

"ansible\_default\_ipv4": {

"address": "10.88.88.21",

"alias": "ens33",

"broadcast": "10.88.88.255",

"gateway": "10.88.88.2",

"interface": "ens33",

"macaddress": "00:0c:29:35:6d:56",

"mtu": 1500,

"netmask": "255.255.255.0",

"network": "10.88.88.0",

"type": "ether"

},

"discovered\_interpreter\_python": "/usr/bin/python"

},

"changed": false

}

10.88.88.40 | SUCCESS => {

"ansible\_facts": {

"ansible\_default\_ipv4": {

"address": "10.88.88.40",

"alias": "eth0",

"broadcast": "10.88.88.255",

"gateway": "10.88.88.2",

"interface": "eth0",

"macaddress": "00:0c:29:dd:c7:25",

"mtu": 1500,

"netmask": "255.255.255.0",

"network": "10.88.88.0",

"type": "ether"

},

"discovered\_interpreter\_python": "/usr/bin/python"

},

"changed": false

}

10.88.88.30 | SUCCESS => {

"ansible\_facts": {

"ansible\_default\_ipv4": {

"address": "10.88.88.30",

"alias": "ens33",

"broadcast": "10.88.88.255",

"gateway": "10.88.88.2",

"interface": "ens33",

"macaddress": "00:0c:29:6d:9d:b4",

"mtu": 1500,

"netmask": "255.255.255.0",

"network": "10.88.88.0",

"type": "ether"

},

"discovered\_interpreter\_python": "/usr/bin/python3"

},

"changed": false

}

### Dùng ad-hoc command trong ansible

Với chế độ ad-hoc command mà ansible cung cấp, chúng ta sẽ thực hiện các task đơn (các yêu cầu thực hiện độc lập) để ra lệnh cho hạ tầng (các client được quản lý bởi ansible) thực hiện các yêu cầu rời rạc khi quản lý các client.

Khi vận hành hệ thống, bạn thường xuyên phải lặp đi các thao tác rất cơ bản ví dụ như:

* Cần phải khởi động lại OS
* Cần phải khởi động lại mysql hoặc đơn giản là cần phải update các gói phần mềm phổ thông (VIM, WGET ….) trên hàng loạt các máy chủ khác nhau (CentOS, Ubuntu …).
* Hoặc phải kiểm tra trạng thái của các services chẳng hạn.
* Hoặc phải thực hiện thao tác copy một file nào đó.
* Hoặc xem nội dung của file log nào đó.

Nếu ta có 1 vài máy chủ thì các thao tác này rất đơn giản, nhưng giả sử ta cần thực hiện trên hàng chục máy chủ một lúc thì chế độ ad-hoc command của ansible bắt đầu phát huy tác dụng ở đây. Nó sẽ giúp tiết kiệm được thời gian và công sức khi thực hiện các task này trên hàng loạt các máy trong hạ tầng của chúng ta.

Trong phần này chỉ giới thiệu một số cách thực thi ansible theo dạng ad-hoc command cho một số trường hợp cơ bản, trong thực tế thì bạn có thể vận dụng thêm các lệnh khác. Các lệnh này sẽ được tra cứu trên trang của của ansible.

Cú pháp chung của ad-hoc command trong ansible là

ansible [pattern] -m [module] -a "[module options]"

#### Thực hiện kiểm tra dung lượng RAM của client

Cú pháp

ansible all -a "free -m"

Ta sẽ có kết kết của RAM trên tất cả các client

[root@Ansible ~]# ansible all -a "free -m"

10.88.88.21 | CHANGED | rc=0 >>

total used free shared buff/cache available

Mem: 972 145 681 7 146 663

Swap: 952 0 952

10.88.88.20 | CHANGED | rc=0 >>

total used free shared buff/cache available

Mem: 976 130 692 6 153 680

Swap: 4095 0 4095

10.88.88.40 | CHANGED | rc=0 >>

total used free shared buffers cached

Mem: 995 182 812 0 18 45

-/+ buffers/cache: 118 877

Swap: 2047 0 2047

10.88.88.30 | CHANGED | rc=0 >>

total used free shared buff/cache available

Mem: 962 195 171 1 594 603

Swap: 1923 3 1920

Ta có thể thay từ all bằng từ centos6 hoặc ubuntu trong file inventory của ansible đã được khai báo ở phần trước. Kết quả như sau:

[root@Ansible ~]# ansible ubuntu -a "free -m"

10.88.88.30 | CHANGED | rc=0 >>

total used free shared buff/cache available

Mem: 962 196 171 1 595 602

Swap: 1923 3 1920

## Tìm hiểu về playbook của Ansible

## Cấu trúc của Playbook

Ansible Playbook được viết bằng YAML (Yet Another Markup Language).  
– Tệp tin YAML (YML) bắt đầu bằng dấu 3 gạch ngang (—)  
– YAML sử dụng cặp key:value đơn giản để trình bày dữ liệu.  
– Cặp dictionary thì được trình bày trong cặp key: value (để ý có khoảng trắng giữa key và value)  
– Mỗi phần tử của list sẽ được viết trên dòng mới, với thụt đầu dòng, bắt đầu với gạch ngang (-) và dấu cách.  
– list có thể được sử dụng bên trong dictionary  
– list của dictionary  
Mỗi play chứa một danh sách (list) các task. Các task thực hiện theo tuần tự. Tất nhiên, các host trong một play đều nhận cùng chỉ thị task.  
Thứ tự chạy playbook là từ trên xuống dưới. Nếu task nào thực hiện lỗi thì quá trình thực hiện playbook bị dừng tại đấy và kết thúc luôn quá trình chạy.  
Kết quả của mỗi task là thực hiện một module của ansible, với các tham số chỉ định. Các biến có thể được sử dụng trong các đối số của module.  
Mỗi task sẽ có một “name”. Name này chính là phần sẽ diễn giải cho task sẽ thực hiện, mà thông tin sẽ được xuất ra khi chạy playbook.

## Các thành phần trong playbook

Một số tags trong playbook:

* name: Xác định tên của ansible playbook
* hosts: Chỉ định hosts hoặc host groups mà ansible sẽ điều khiển và thực thi các task.
* vars: định nghĩa các biến mà có thể sử dụng trong playbook.
* tasks: là danh sách các hành động cần được thực hiện
* handlers: khai báo các task notify

Một ví dụ về playbook :

---

- hosts: dbservers

gather\_facts: no

vars:

who: World

tasks:

- name: say hello

debug: msg="Hello {{ who }}"

- name: retrieve the uptime

command: uptime

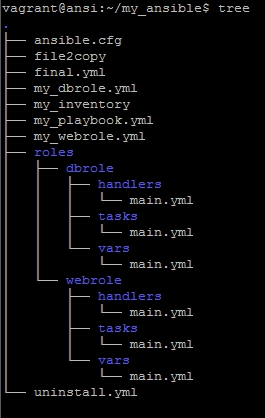
## Role

Trong Ansible, Role là một cơ chế để tách 1 playbook ra thành nhiều file. Việc này nhằm đơn giản hoá việc viết các playbook phức tạp và có thể tái sử dụng lại nhiều lần .

**Role không phải là playbook**. Role là **một bộ khung (framework)**để chia nhỏ playbook thành nhiều files khác nhau. Mỗi role là một thành phần độc lập, bao gồm nhiều ***variables, tasks, files, templates, và modules***bên dưới.

Việc tổ chức playbook theo role cũng giúp người dùng dễ chia sẻ và tái sử dụng lại playbook với người khác. Đặc biệt trong môi trường doanh nghiệp khi có từ vài trăm tới vài ngàn playbook thì Role chính là cách quản lý các playbook này.

Cấu trúc một role cơ bản bao gồm các thư mục sau:



Ở đây gồm các roles: dbrole và webrole. Trong mỗi role đó thì gồm các thông tin cấu hình về tasks, var, handlers hoặc templates của riêng mỗi role đó.

## Demo

Link Github: https://github.com/namlun9115/PRJ3-Ansible

1. **KẾT LUẬN**

**Các kết quả đạt được của luận văn:**

Với mục tiêu nghiên cứu giải pháp tự động hóa để quản lý hệ thống bằng Ansible.

- Hiểu rõ được các tính năng,đặc điểm,cách hoạt động của Ansible

- Ansible giúp chúng ta quản lý hệ thống dễ dàng

**Hướng phát triển tiếp theo:**

Học viên sẽ tiếp tục nghiên cứu, hoàn thiện các giải pháp tự động hóa để quản lý hệ thống bằng Ansible để có thể áp dụng và triển khai một cách hiệu quả trong thực tế.

1. **TÀI LIỆU THAM KHẢO**

Gồm các trang web:

<https://viblo.asia/p/phan-1-tim-hieu-ve-ansible-4dbZNxv85YM>

<https://viblo.asia/p/phan-2-tim-hieu-ve-ansible-YWOZry8rKQ0>

<https://docs.ansible.com/ansible/2.9/index.html>

<https://cloudcraft.info/gioi-thieu-ve-ansible/>

<https://blog.vietnamlab.vn/phan-1-ansible-khai-niem-co-ban/>

<https://blog.vietnamlab.vn/phan-2-ansible-cai-dat/>

<https://vnsys.wordpress.com/2018/08/18/to-chuc-cac-role-ansible-theo-cau-truc-thu-muc/>

<https://vnsys.wordpress.com/2018/08/04/quan-ly-cau-hinh-cm-voi-ansible/>

<https://vnsys.wordpress.com/2018/08/05/ansible-playbook/>

<https://cloudcraft.info/ansible-huong-dan-su-dung-va-quan-ly-role-trong-ansible/>

<https://news.cloud365.vn/10-phut-ansible-co-ban-phan-1-ansile-dung-de-lam-gi/>

<https://news.cloud365.vn/10-phut-ansible-co-ban-phan-2-dung-lab-de-thuc-hanh-ansible/>

<https://news.cloud365.vn/10-phut-ansible-co-ban-phan-3-huong-dan-su-dung-ad-hoc-command-trong-ansible/>

<https://news.cloud365.vn/10-phut-ansible-co-ban-phan-4-viet-playbook-tren-ansible/>

<https://news.cloud365.vn/10-phut-ansible-co-ban-phan-5-viet-playbook-cai-dat-wordpress-tren-centos-7/>