RHCE 덤프 정리

21/10/25

시험치기전에 숙지해야할 사항들,

Rhce는 8버전부터 rhcsa의 시험과목에 ansible을 이용하여 host서버에서 배포하여 node서버들에게 적용시키는 원리입니다.

기본적으로 vim 편집기를 사용하여 yaml파일을 만들어서 배포하는데, 줄맞춤이 중요합니다.

예를 들어, 우리가 ubuntu bonding 세팅할 때, 줄이 틀리면 bonding이 안먹히는 것처럼 줄맞춤을 필수로 해줘야 합니다.

2칸, 4칸씩 줄맞춤을 해야하는데 자동 줄맞춤을 필수로 해야합니다.(실수를 줄이기 위해서)

# vi /home/username/.vimrc

autocmd FileType yaml setlocal ai ts=2 sw=2 et

username에는 해당 username을 넣으면 됩니다.

tab키를 누르면 2칸씩 띄어쓰기가 됩니다.

명령어 옵션 값에 헷갈리는 부분이 있으면 ansible-doc 모듈명으로 예시를 확인 할 수 있습니다.

Ansible 설치 및 구성

**<문제>**

\* 다음과 같이 ansible 및 ansible 제어 노드 control.labx.example.com을 설치하고 구성합니다.

\* 다음과 같이 /home/matthew/ansible/inventory라는 정적 인벤토리 파일을 생성합니다.

- node1은 dev 호스트 그룹의 구성원입니다.

- node2는 test 호스트 그룹의 구성원입니다.

- node3 및 node4는 prod 호스트 그룹의 구성원입니다.

- node5는 balancers 호스트 그룹의 구성원입니다.

- prod 그룹은 webservers 호스트 그룹의 구성원입니다.

\* 다음과 같이 /home/matthew/ansible/ansible.cfg라는 구성 파일을 만듭니다.

- 호스트인벤토리파일/home/matthew/ansible/inventory

- 플레이북에서역할의위치/home/matthew/ansible/roles

- 연습 환경에서 워크 스테이션의 학생 사용자 (시험 환경 의 remote\_user = matthew)로 로그인 합니다.

<정답>

(1) ansible 설치, 사용자 전환, role 디렉토리 생성

# yum install ansible  
# su - matthew  
# mkdir -pv /home/matthew/ansible/roles

ansible을 설치하기 위해 따로 repository를 잡는 등의 행위는 일절 하지 않았다.

(2) 정적 인벤토리 /home/student/ansible/inventory 생성

[matthew@workstation ansible]$ cat inventory

[dev]  
node1

[test]  
node2

[prod]  
node3  
node4

[balancers]  
node5

[webservers:children]  
prod

(3) /home/student/ansible/ansible.cfg 구성파일 생성

[matthew@workstation ansible]$ cat ansible.cfg

[defaults]  
remote\_user = matthew  
inventory = /home/matthew/ansible/inventory  
roles\_path = /home/matthew/ansible/roles:/usr/share/ansible/roles:/home/matthew/.ansible/roles  
ask\_pass = false

[privilege\_escalation]  
become = true  
become\_method = sudo  
become\_user = root  
become\_ask\_pass = false

- 문제를 잘 읽어보면, role 경로가 딱 /home/matthew/ansible/roles 만 하라는게 아니고, 다른 role 경로가 추가될 수 있는 뉘앙스이다. 따라서 위와 같이 2가지로 설정한다.

host 및 node1~5에 이미 ssh-key등은 다 설정되어있고, matthew유저도 다 클라이언트에 생성되어있다. 따라서 아래 (4)는 할 필요가 없지만, 연습을 위해 가상머신에 환경을 만들기 위해서는 필요하다.

(4) sudo 설정과 ssh-key 배포

**\* sudoers 설정**

- 관리 당하는 서버들, 즉 node1~5에서 수행한다.

- sudoers 파일에 예시가 그대로 있으므로 그것 그대로 사용하면 된다.

- /etc/sudoers.d/matthew 파일을 열어서 그 내용 그대로 matthew 계정에 대해 만들어서 저장하면 된다.

**\* student 유저로 키 생성 후 배포**

- 이 부분은 host에서 수행한다.

1. ssh key 생성  
[matthew@control ansible]$ ssh-keygen  
Generating public/private rsa key pair.  
Enter file in which to save the key (/home/student/.ssh/id\_rsa):  
Created directory '/home/matthew/.ssh'.  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /home/matthew/.ssh/id\_rsa.  
Your public key has been saved in /home/matthew/.ssh/id\_rsa.pub.  
The key fingerprint is:  
SHA256:ebV+PHBONsu+VYPED2ThZVsMhFeIWTiio0sRUeCJpAs matthew@control  
The key's randomart image is:  
+---[RSA 2048]----+  
| . o+. @=Bo|  
| o o.. . @ = +|  
|E . . o. . ..B . |  
| . . . o. ...+ |  
| . oS.. o.=o.|  
| o . . O oo|  
| . . . B .|  
| . o o |  
| o. |  
+----[SHA256]-----+  
[matthew@control ansible]$

2. ssh key 배포  
[matthew@control ~]$ ssh-copy-id devops@node1  
/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/matthew/.ssh/id\_rsa.pub"  
The authenticity of host 'servera (192.168.1.101)' can't be established.  
ECDSA key fingerprint is SHA256:NqwepoRPDx8HGKDtZIKIjHCf58+B00cb3GXRHQl4C/k.  
ECDSA key fingerprint is MD5:30:cd:9a:d9:89:c5:05:8f:61:11:01:72:a8:f9:26:5a.  
Are you sure you want to continue connecting (yes/no)? yes  
/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed  
/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys  
matthew@node1's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'matthew@node1'"  
and check to make sure that only the key(s) you wanted were added.

[matthew@control ~]$   
(이렇게 node1~5 까지 수행한다)

(5) ping 테스트

[matthew@control ansible]$ ansible all -m ping  
serverd | SUCCESS => {  
 "changed": false,  
 "ping": "pong"  
}  
servera | SUCCESS => {  
 "changed": false,  
 "ping": "pong"  
}  
serverb | SUCCESS => {  
 "changed": false,  
 "ping": "pong"  
}  
serverc | SUCCESS => {  
 "changed": false,  
 "ping": "pong"  
}

yum repository 생성하는 sh파일 만들기

**<문제>**

/home/matthew/ansible/adhoc.sh아래와 같이 각 관리 노드에 yum저장소를 만들기 위해 ansible ad-hoc 명령을 실행하는 라는 셸 스크립트를 만듭니다.

- 저장소 이름 (name) : exam\_rhel

- 저장소 설명 (description) : EX407 software

- 기본 URL (baseurl) :  <http://content.example.com/rhel7.5/x86_64/dvd>

- GPG 서명 확인 활성화

- GPG 키 URL :  <http://content.example.com/rhel7.5/x86_64/dvd/RPM-GPG-KEY-redhat-release>

- 저장소가 사용 설정됨

- 저장소 이름 (name) : exam\_rhel

- 저장소 설명 (description) : EX407 software

- 기본 URL (baseurl) :  <http://content.example.com/rhel7.5/x86_64/dvd>

- GPG 서명 확인 활성화

- GPG 키 URL :  <http://content.example.com/rhel7.5/x86_64/dvd/RPM-GPG-KEY-redhat-release>

- 저장소가 사용 설정됨

**<정답>**

**# vi adhoc.sh**

#!/bin/bash  
ansible all -m yum\_repository -a "name=exam\_rhel description='EX407 software' baseurl=http://content.example.com/rhel7.5/x86\_64/dvd gpgcheck=yes gpgkey=http://content.example.com/rhel7.5/x86\_64/dvd/RPM-GPG-KEY-redhat-release enabled=yes"

ansible all -m yum\_repository -a "name=exam\_rhel description='EX407 software' baseurl=http://content.example.com/rhel7.5/x86\_64/dvd gpgcheck=yes gpgkey=http://content.example.com/rhel7.5/x86\_64/dvd/RPM-GPG-KEY-redhat-release enabled=yes"

# chmod 0755 adhoc.sh

실행 : ./adhoc.sh

- yum\_repository 모듈의 name 속성은 repo 파일 및 대괄호로 둘러쌓인 제목이며, description 속성은 repo 파일의 name 속성과 대응한다.

- adhoc 명령의 -a 옵션 뒤 " " 안의 내용들은 콤마로 구분하지 않고 스페이스로 구분한다.

- 스크립트에 실행권한 755와 맨 위에 bash shebang (#!/bin/bash)을 꼭 넣을 것

- 같은 형식으로 2가지의 repository를 만든다. 즉 한 스크립트에 정답 부분이 2개 들어간다.

문자열은 “”안에 들어간다

소프트웨어 패키지 설치

**<문제>**

/home/matthew/ansible/packages.yml이라는 플레이 북을 생성합니다.

1. dev, test, prod 3개의 호스트 그룹에 소속된 호스트에 php 및 mariadb 패키지를 설치합니다.

2. dev 호스트 그룹의 호스트에 RPM development tools 그룹을 설치합니다.

3. dev 호스트 그룹의 호스트에서 모든 패키지를 최신 버전으로 업데이트합니다.

**<정답>**

- name : install packages  
 hosts: dev, test, prod  
 tasks:  
   
 - name: install mariadb, php  
 yum:  
 name: "{{ item }}"  
 state: present  
 loop:  
 - php  
 - mariadb

- name: install development tools only dev group.  
 yum:  
 name: "@RPM Development tools"  
 state: present  
 when: ansible\_hostname in groups['dev']

- name: update all packages only dev group  
 yum:  
 name: '\*'  
 state: latest  
 when: (‘dev’ in group\_names)

- hosts 부분에는 문제가 하라는 그대로 맞춰서 쓰면 된다.

# 참고 : 플레이북에서 팩트변수 참조하기

playbook에서 팩트변수를 참조할 때, ansible\_facts 까지는 이미 인식하고 있고, 그 이하부터 변수로 명시하면 된다. ansible\_facts['ansible\_hostname'] 이렇게 하지말고, ansible\_hostname 이렇게 하면 된다.

RHEL system role 사용하기

**<문제>**

redhat system role인 timesync role을 사용한다.

아래 설정을 가지는 /home/matthew/ansible/timesync.yml이라는 플레이 북을 만듭니다.

- 모든 관리 호스트에서 실행

- timesync role을 사용합니다.

- time server 172.24.254.254 를 사용하도록 role 구성

- iburst 매개 변수를 활성화하도록 역할 구성

**<정답>**

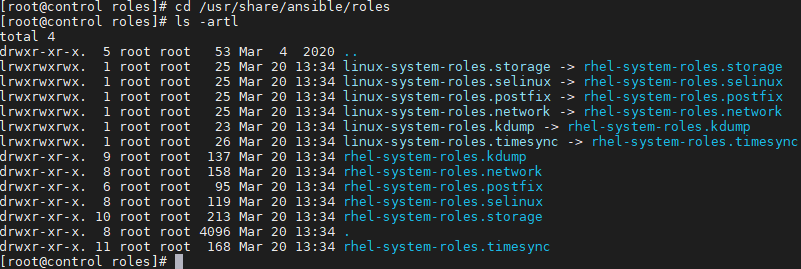
**\* ansible에서 제공하는 role을 설치**

1. role 설치하기

yum install rhel-system-roles

2. 설치된 role을 나의 roles 디렉토리로 이동

- rhel-system-roles는  /usr/share/ansible/roles 경로에 저장된다.



- 이 경로는 1번 문제의 인벤토리 생성할 때 roles\_path에 추가로 명시한 경로이다.

- 따라서 이렇게 설치만 하고 ansible-galaxy list를 수행하면 설치된 rhel role들이 보이게 된다.

**\* 스크립트 작성 및 실행**

1. 스크립트 파일 작성

- hosts: all  
 vars:  
 timesync\_ntp\_servers:  
 - hostname: 172.25.254.254  
 iburst: yes  
 roles:  
 - rhel-system-roles.timesync

- 이 스크립트 구문 형식은 외울 필요 없다. 설치된 role 디렉토리의 README.md 파일에 친절히 나와있다.

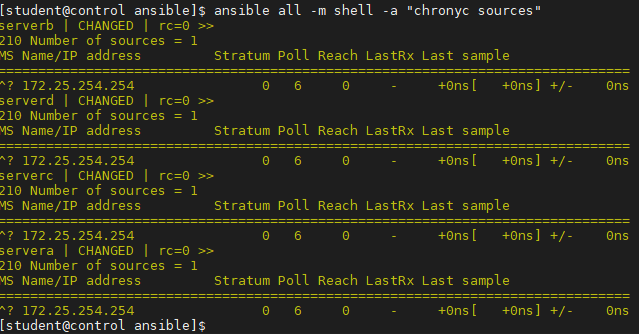
2. 실행

ansible-playbook timesync.yml

3. 결과확인

chrony sources 명령으로 결과를 확인한다. 시간 싱크가 잘 되는것으로 보이면 완료

ansible all -m shell -a 'chronyc sources'



- 문제를 자세히 읽어보면 애매하게 "timesync의 기본 설정된 세팅을 사용해라" 이런 말이 있는데, 기본 설정이라는 것은 아무 설정도 하지 않는것을 말한다. 즉 특별한 매개변수를 넣지 않고 그냥 쓰면 된다.

- 실행 전에, ansible all -m shell -a 'chronyc sources' 구문을 수행하여 현재 chronyc 설정이 어떻게 되어있는지 확인한다. 아무것도 설정이 안되어있을 줄 알았는데 의외로 각 노드에 설정이 다 되어있다. 따라서 작업 전과 작업 후를 구분하기가 어렵다. (심지어 타임서버 아이피도 동일했고, 단순히 iburst만 추가된 것으로 보인다)

Role 설치 및 사용하기

**<문제>**

2가지 role 파일을 다운받고, 설치를 하기 위한 파일은 /home/matthew/ansible/roles/requirememts.yml입니다. 이 파일을 사용하여 /home/matthew/ansible/roles에 다운로드 및 설치를 수행합니다.

- [http://](http://materials/haproxy.tar.gz)[ukdisk.asuscomm.com:8888/pdf\_wav/](http://ukdisk.asuscomm.com:8888/pdf_wav/incubateurpe-haproxy-0.0.6.tar.gz)[ansible-role-haproxy-master.tar.gz](http://ansible-role-haproxy-master.tar.gz/), role 이름은 balancer 로 만들어야 합니다.

- [http://](http://materials/phpinfo.tar.gz)[ukdisk.asuscomm.com:8888/pdf\_wav/](http://ukdisk.asuscomm.com:8888/pdf_wav/geerlingguy-php_roles-1.0.0.tar.gz)[ansible-role-php-master.tar.gz](http://ansible-role-php-master.tar.gz/), role 이름은 phpinfo 로 만들어야 합니다.

(ukdisk로 시작된 경로는 실제로 다운받을 수 있는 경로이다. 내가 임시로 만든 것)

**<정답>**

1. 설치 파일 생성

vi /home/matthew/ansible/roles/requirememts.yml

- src: <http://ukdisk.asuscomm.com:8888/pdf_wav/ansible-role-haproxy-master.tar.gz>  
 name: balancer

- src: <http://ukdisk.asuscomm.com:8888/pdf_wav/ansible-role-php-master.tar.gz>  
 name: phpinfo

- 여기서 name은 ansible-galaxy list에서 식별하는 이름이다. 또한 playbook에서 role을 식별하는 이름이기도 하다.

2. role 파일 설치 실행

ansible-galaxy install -r /home/matthew/ansible/roles/requirements.yml

- 이렇게 하면 inventory에 명시한 /home/matthew/ansible/roles에 설치된다.

Ansible-galaxy list

custom으로 role 생성 및 사용하기

**<문제>**

1. 다음 요구 사항에 따라 /home/matthew/ansible/role에 apache라는 role을 만듭니다.

- httpd 패키지를 설치하고 시작할 때 활성화 한 다음 시작합니다.

- 방화벽이 활성화되고 웹 서버에 대한 액세스를 허용하는 규칙으로 실행됩니다.

- /var/www/html/index.html 파일을 만드는 데 사용되는 index.html.j2 템플릿 파일이 있으며 출력은 다음과 같습니다.

  Welcome to {{ FQDN }} on {{ IPADDRESS }}

(실제 문제에서는 {{ HOSTNAME }} 으로 써있는데, 영어로 문제를 잘 보면 fully qualified domain name 을 입력하라고 한다. 즉 FQDN을 입력해야 함)

2. 해당 role을 실행하는 /home/matthew/ansible/newrole.yml 파일을 생성한다.

- 스크립트는 webservers 호스트 그룹의 호스트에서 실행됩니다.

**<정답>**

1. apache role의 뼈대 초기화하기

ansible-galaxy init apache

2. tasks 디렉토리의 main.yml 수정

---  
# tasks file for apache  
- name: Install httpd  
 yum:  
 name: httpd  
 state: present

- name: start httpd  
 service:  
 name: httpd  
 state: started  
 enabled: yes

- name: start firewalld  
 service:  
 name: firewalld  
 state: started  
 enabled: yes

- name: firewalld permits http service  
 firewalld:  
 service: http  
 state: enabled  
 permanent: yes  
 immediate: yes

- name: create /var/www/html/index.html  
 template:  
 src: index.html.j2  
 dest: /var/www/html/index.html  
 setype: httpd\_sys\_content\_t

3. templates 디렉토리의 index.html.j2 파일 생성

Welcome to {{ ansible\_fqdn }} on {{ ansible\_default\_ipv4['address'] }}

4. role을 실행하기 위한 newrole.yml 생성

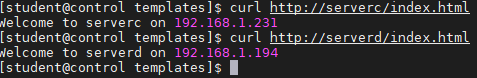
---  
- hosts: webservers  
 roles:  
 - role: apache

5. 실행

ansible-playbook newrole.yml

6. 결과 확인

curl <http://서버호스트명/index.html>



다운받은 role을 실행하기

**<문제>**

다운받았던 2개의 role (balancer, phpinfo) 을 설치하는 roles.yml 생성 및 아래와 같이 배포한다.

- 호스트 그룹 balancers에 balancer role 배포

- 호스트 그룹 webservers에 phpinfo role 배포

**<정답>**

1. playbook 생성하기

vi roles.yml

- hosts: webservers  
 roles:  
 - role: phpinfo

- hosts: balancers  
 roles:  
 - role: balancer

2. 실행

ansible-playbook roles.yml

3. 결과보기

- balancer : 웹브라우저에서 node5의 주소로 접속하고 여러번 새로고침 하면 node3이 나올때도 있고 node4가 나올때도 있으면 완료

- phpinfo : hello? welcome? 이런 단어가 포함된 내용이 나오고 그 아래에는 php의 정보가 나오면 정상.

- 꼭 phpinfo를 첫번째로 쓸 것. balancer를 먼저 하면 eth0를 못찾는다? 이런 식의 메시지가 발생하면서 에러가 난다.

- balacner란, node5에 설치하고, 사용자가 node3또는4에 접속하려는 목적으로 node5에 접속하면 node5가 알아서 로드밸런싱을 하여 node3 또는 node4로 접속하게 만드는 것이다.

- phpinfo는 그냥 php이다. 특별한 것은 없음.

- playbook 만들 때, 특별한 설정을 넣을 필요는 없다. 알아서 잘 된다.

- 영문,한글 꼭 전환해서 볼 것.

LV 생성 및 적용

**<문제>**

/home/matthew/ansible/ 에 lv.yml 파일을 만들고 모든 호스트에서 실행되도록합니다. 요구 사항은 다음과 같습니다.

1. research 볼륨 그룹에서 1500MiB의 논리 볼륨을 만듭니다.

2. ext4를 사용하여 이 논리 볼륨을 포맷합니다.

3. 호스트그룹 dev에 포함된 호스트만 시작시 / data 디렉토리에 자동으로 마운트합니다.

4. 이 논리 볼륨의 크기를 생성 할 수없는 경우 오류 메시지가 출력됩니다. Could not create logical volume of that size

5. 대신 800MiB 크기를 사용하여 논리 볼륨 생성

6. research 볼륨 그룹이 존재하지 않는 경우 오류 메시지가 출력됩니다. 이 오류 메시지도 정확하게 써야 한다.

7. 현재 research 볼륨 그룹은 존재하지 않습니다.

**<정답>**

1. research VG가 존재하지 않으므로 직접 생성한다. (시험에서는 vg가 있을 것임)

ansible -all -m lvg -a "vg=research pvs= /dev/sdb state=present"

2. 아래와 같이 스크립트를 생성한다.

- hosts: all  
 tasks:

- name: "1. check an existance of research VG"  
 shell: vgdisplay research  
 register: check\_vg  
 ignore\_errors: yes

- name: "2. output error mesasge if research the VG doesn't exist"  
 debug:  
 msg: "the research VG is not exist"  
 when: check\_vg is failed  
 failed\_when: check\_vg is failed

- name: "3. create 1500MiB logical volume"

block:  
 - name: "3.1 create lvx LV"  
 lvol:  
 vg: research  
 lv: lvx  
 size: 1500M

rescue:  
 - name: "3.2 if it couldn't create lvx LV, report this message"  
 debug:  
 msg: "3.2 Coundn't create lvx LV of that size"

- name: "4. create 800MiB size LV"  
 lvol:  
 lv: lvx  
 vg: research  
 size: 800M

- name: "5. format lvx LV to ext4"  
 filesystem:  
 dev: /dev/research/lvx  
 fstype: ext4

- name: "6. create mount point"  
 file:  
 path: /data  
 state: directory  
 mode: '0755'  
 ignore\_errors: yes

- name: "7. mount lvx LV to the /data"  
 mount:  
 src: /dev/research/lvx  
 path: /data  
 fstype: ext4  
 state: mounted # mounted로 하면, /etc/fstab에 정보를 등록하며, 마운트까지 수행한다.  
 when: ansible\_hostname in groups['dev']

3. 결과를 확인한다.

[matthew@control ansible]$ ansible all -m shell -a "df -h"

- 실제로 시험에서는 마운트를 하지 말라고 한다. 혹시 몰라서 덤프의 마운트 부분은 그대로 첨부한다.

host 파일 생성하기

**<문제>**

특정 경로에 있는hosts.j2를 /home/matthew/ansible에 다운로드한다. (wget을 사용한다)

/home/matthew/ansible/hosts.yml라는 플레이북을 만들고, 해당 템플릿을 이용하여 /etc/hosts 파일을 만들되 dev 호스트 그룹안에 있는 호스트에 /etc/myhosts로 저장한다.

만들어진 파일은 아래 내용과 동일해야 한다.

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4  
:: 1 localhost localhost.localdomain localhost6 localhost6.localdomain6

172.25.250.11 node1.lab.example.com node1  
172.25.250.10 control.lab.example.com servera  
172.25.250.12 node2.lab.example.com node2  
172.25.250.13 node3.lab.example.com node3  
172.25.250.14 node4.lab.example.com node4  
172.25.250.15 node5.lab.example.com node5

**<정답>**

1. wget으로 hosts.j2 다운로드

2. hosts.j2 템플릿 만들기

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4  
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6

{% for host in groups['all'] %}  
{{ hostvars[host]['ansible\_default\_ipv4']['address'] }} {{ hostvars[host]['ansible\_fqdn'] }} {{ hostvars[host]['ansible\_hostname'] }}  
{% endfor %}

3. 템플릿 생성 playbook 만들기

- name: create hosts  
 hosts: all  
 tasks:  
 - name: create hosts using tempalte  
 template:  
 src: hosts.j2  
 dest: /tmp/hosts  
 when: ansible\_hostname in groups['dev']

- 만들 때 동일해야하는 내용과 완전히 동일하도록 엔터값 공백까지 잘 신경쓸 것.

- /etc/myhosts의 파일은 순서대로 결과가 나오지 않을 수 있는데 그것은 무시해도 된다.

파일 내용 수정하기

**<문제>**

다음과 같이 /home/matthew/ansible/issue.yml이라는 플레이 북을 만듭니다.

- 스크립트는 모든 호스트에서 실행됩니다.

- 플레이 북은 아래와 같이 /etc/issue의 내용을 텍스트 줄로 대체합니다.

   - dev 호스트 그룹의 호스트에서 내용 : developer

   - test 호스트 그룹의 호스트에서 내용 : test

   - prod 호스트 그룹의 호스트에서 내용 : production

**<정답>**

- name: chnage text data  
 hosts: all  
 tasks:

- name: replace data1  
 copy:  
 content: developer  
 dest: /etc/issue  
 when: ansible\_hostname in groups['dev']

- name: replace data2  
 copy:  
 content: test  
 dest: /etc/issue  
 when: ansible\_hostname in groups['test']

- name: replace data3  
 copy:  
 content: production  
 dest: /etc/issue  
 when: ansible\_hostname in groups['prod']

- 여기서 copy 모듈의 content 지시자는, 대상의 내용을 다 없애버리고 content에 명시된 값을 넣는다. 즉, 바꿔치기한다.

웹 컨텐츠 디렉토리 만들기

<문제>

/home/matthew/ansible/webcontent.yml이라는 스크립트를 만듭니다. yml은 다음과 같습니다.

- 플레이 북은 dev 호스트 그룹의 관리 노드에서 실행됩니다.

- 다음 요구 사항에 따라 /webdev 디렉터리를 만듭니다.

     - Webdev 그룹 구성원

     - 권한 : owner = read + write + excute, group = read + write + excute, other = read + excute

     = 특별 권한 : 그룹 ID 설정

/webdev/index.html 파일을 만듭니다. 내용은 다음과 같습니다. Development

/webdev를 /var/www/html/webdev에 link (연결)

<정답>

- name: create web  
 hosts: dev  
 become: true  
 tasks:

- name: 1. install httpd  
 yum:  
 name: "{{ item }}"  
 state: present  
 loop:  
 - httpd  
 - firewalld

- name: 2.start httpd  
 service:  
 name: httpd  
 state: started  
 enabled: yes

- name: 3. start firewalld  
 service:  
 name: firewalld  
 state: started  
 enabled: yes

- name: 4. firweall permits http service  
 firewalld:  
 service: http  
 state: enabled  
 permanent: true  
 immediate: yes

- name: 5. create a group  
 group:  
 name: webdev  
 state: present

- name: 6. create a directory  
 file:  
 path: /webdev  
 state: directory  
 group: webdev  
 mode: '2775'  
 setype: httpd\_sys\_content\_t

- name: 7. create a link  
 file:  
 src: /webdev  
 dest: /var/www/html/webdev  
 state: link

- name: 8. copy content  
 copy:  
 content: "Development\n"  
 dest: /webdev/index.html  
 setype: httpd\_sys\_content\_t

curl <http://node1/webdev/index.html> 명령 또는 웹브라우저에서 확인하면 Development라는 값을 확인할 수 있다.

- 여기서는 다 설치했지만, 실제로는 http 설치 및 세팅, firewalld 설치 및 세팅 등은 하지 않았다. 문제에 나온대로만 하면 된다.

- link를 할 때, 디렉토리 /webdev를 생성하고, 링크를 하라고 했으므로 실제 디렉토리는 /webdev, 링크는 /var/www/html/webdev가 된다. 영문 한글 둘 다 봐도 문제가 좀 햇갈릴 수 있으니 참고할 것.

- setype줄 때는, /var/www/html/webdev는 html 디렉토리 밑에 생기므로 자동으로 httpd setype이 들어간다. 하지만 /webdev 디렉토리는 setype이 그냥 default값이며, 신경 안써도 된다.

- index.html 파일은 setype을 지정해 주자. 그렇지 않으면 웹브라우저에서 나오지 않는다.

하드웨어 보고서 생성

**<문제>**

/home/matthew/ansible/hwreport.yml이라는 스크립트를  다운로드 받고, 받은 파일을 수정하라.

hwreport.empty 파일의 내용은 다음과 비슷한 형태이다.

inventory\_name = hostname  
total\_mem = totalmem\_mb  
bios\_version = bios\_version  
vda\_size = vda.size  
vdb\_size = vdb.size

이 파일을 가지고 /root/hwreport.txt 라는 파일을 생성하고 아래 정보를 제공하라. (hwreport.empty 파일 형식과 동일) 또한 값이 없으면 기본값으로 NONE 을 넣는다.

- 인벤토리 호스트 이름

- 총 메모리 (MB)

- BIOS 버전

- 장치 vda 크기

- 장치 vdb 크기

**<정답>**

- name: hwreport2  
 hosts: all  
 tasks:  
   
 - name: download hwreport.empty  
 get\_url:  
 url: <http://example.com/path/hwreport.empty>  
 dest: /root/hwreport.txt  
 mode: '0644'

- lineinfile:  
 line: "inventory\_name = {{ ansible\_hostname | default('NONE') }}"  
 regexp: "^inventory\_name"  
 path: /root/hwreport.txt

- lineinfile:  
 line: "total\_mem = {{ ansible\_memtotal\_mb | default('NONE') }}"  
 regexp: "^total\_mem"  
 path: /root/hwreport.txt

- lineinfile:  
 line: "bios\_version = {{ ansible\_bios\_version | default('NONE') }}"  
 regexp: "^bios\_version"  
 path: /root/hwreport.txt

- lineinfile:  
 line: "sda\_size = {{ ansible\_devices['sda']['size'] | default('NONE') }}"  
 regexp: "^sda\_size"  
 path: /root/hwreport.txt

- lineinfile:  
 line: "sdd\_size = {{ ansible\_devices['sdd']['size'] | default('NONE') }}"  
 regexp: "^sdd\_size"  
 path: /root/hwreport.txt

- regexp에서^는 ^뒤 단어로 시작하는 형태를 찾아서 line의 내용으로 바꾸라는 의미이다. 즉 hwreport.empty에서 inventory\_name으로 시작하는 줄을, line 지시문으로 명시한 내용으로 변경하게 되는 것이다.

- mode: 0644는 따로 정해졌다기 보다는 그냥 일반적으로 touch로 파일을 만들면 기본적으로 적용되는 permission을 넣으면 된다. (0644가 맞는지 헷갈림..)

암호 라이브러리 만들기

**<문제>**

다음과 같이 Ansible 저장소 사용자 비밀번호를 생성합니다.

1. vault로 암호화된 암호리스트 yml 파일

2. vault를 암호화시킨 비밀번호 txt 파일

vault의 이름은 /home/matthew/ansible/locker.yml입니다. vault에는 다음 두 가지 변수가 있습니다.

   2.1 pw\_developer의 값은 Imadev입니다.

   2.2 pw\_manager의 값은 Imamgr입니다.

비밀번호는 /home/matthew/ansible/secret.txt 파일에 저장됩니다.

암호화 및 복호화를위한 비밀번호 : whenyouwishuponastar

**<정답>**

\* secret.txt 파일 생성

vi secret.txt  
whenyouwishuponastar

\* ansible vault로 암호화된 비밀번호 파일 생성

ansible-vault create --vault-password-file=secret.txt locker.yml  
# 위 명령을 실행하면 vi 에디터가 나온다. 아래를 입력한다.

pw\_developer: Imadev  
pw\_manager: Imamgr

- 이 locker.yml 파일은 암호화되었으며, secret.txt로 풀어야 한다.

\* ansible vault로 암호화된 비밀번호 파일 내용 보기

ansible-vault view --vault-password-file=secret.txt locker.yml

pw\_developer: Imadev  
pw\_manager: Imamgr

- 막히는 경우 -h옵션을 사용한다. ansible-vault -h, ansible-vault view -h 이런식으로.

\* 기존 vault 파일을 복호화하고, 다시 암호화하는데는 아래와 같이 실행한다.

ansible-vault decrypt --vault-password-file=secret.txt locker.yml  
ansible-vault encrypt --vault-password-file=secret.txt locker.yml

사용자 계정 만들기

**<문제>**

<http://rhgls.labx.example.com/materials/user_list.yml> 파일을 다운로드한다. 다운로드한 파일의 내용은 다음과 같다.

[student@workstation ansible]$ cat user\_list.yml   
---  
users:  
 - name: node1  
 job: developers  
 - name: node2  
 job: developers  
 - name: node3  
 job: manager

위 user\_list.yml 파일과 14번에서 만든 locker.yml 파일을 사용하여 아래와 같은 내용을 수행하는 users.yml 파일을 생성한다.

생성옵션1

- dev, test 호스트 그룹이어야 한다.

- 해당 유저의 암호는 pw\_developer 변수에서 받는다.

- devops 그룹은 이 유저의 보조 그룹 중 하나이다.

- job은 developers이어야 한다.

생성옵션2

- prod 호스트 그룹이어야 한다.

- 해당 유저의 암호는 pw\_manager 변수에서 받는다.

- opsmgr 그룹은 이 유저의 보조 그룹 중 하나이다.

- job은 manager이어야 한다.

비밀번호는 SHA512 해시 형식을 사용해야 한다. 또한 ansible-vault 암호 파일을 사용해야 한다.

**<정답>**

- name: createing users  
 hosts: all

vars\_files:  
 - locker.yml  
 - user\_list.yml

tasks:

- group:  
 name: devops  
 state: present  
 when: ansible\_hostname in groups['dev'] or ansible\_hostname in groups['test']  
   
 - group:  
 name: opsmgr  
 state: present  
 when: ansible\_hostname in groups['prod']

- user:  
 name: "{{ item.name }}"  
 password: "{{ pw\_developer | password\_hash('sha512') }}"  
 state: present  
 groups: devops  
 append: yes  
 loop: "{{ users }}"  
 when: (ansible\_hostname in groups['dev'] or ansible\_hostname in groups['test']) and item.job == "developers"

- user:  
 name: "{{ item.name }}"  
 password: "{{ pw\_manager | password\_hash('sha512') }}"  
 state: present  
 groups: opsmgr  
 append: yes  
 loop: "{{ users }}"  
 when: ansible\_hostname in groups['prod'] and item.job == "manager"

이후 아래 명령어들의 결과로 상태를 확인하면 된다.

ansible all -m shell -a 'id node1'

ansible all -m shell -a 'id node2'

ansible all -m shell -a 'id node3'

- 원래 user, firewalld같은 daemon, vg 등은 다 기본적으로 클라이언트에 설정되어있어서 문제에서 하라고 하지 않으면 내가 따로 만들거나 건드릴 필요가 없었다. 근데 여기서 그룹 devops, opsmgr 두개는 없었다. 그래서 직접 생성함.

- 그리고 영문으로 문제를 잘 읽어보면 devops, opsmgr은 보조 그룹으로 추가되는 개념으로 하라고 한다. 따라서 groups, append 값을 넣었다. 그렇게 하면 원래 그룹은 유지하고 명시한 그룹을 추가하게 된다.

- 또한 그룹 만들 때 모든 호스트 대상으로 만들지 않고 여기 나온 대상으로만 그룹을 만들기 위해 when 조건을 사용한다.

ansible vault rekey

<http://rhgls.labx.example.com/materials/salaries.yml을> 다운로드 하고 /home/matthew/ansible/salaries.yml로 저장합니다. (예시에서는 다운로드 할 수 없으므로 직접 만든다. 실제로 다운받을 때는 wget을 사용한다.)

해당 vault 파일을 새 비밀번호를 설정하고 새 비밀번호로 내용을 확인한다.

**<정답>**

1. vault 파일 생성

ansible-vault create salaries.yml

- 새 패스워드 입력

- 새 패스워드 확인 입력

이후 vi 에디터가 나오면, 아래와 같이 입력하고 저장한다.

RED HAT ANSIBLE 2.7 EXAM

GOOD LUCK

2. 새 비밀번호 생성

ansible-vault rekey salaries.yml

- 현재 패스워드 입력

- 새 패스워드 입력

- 새 패스워드 확인 입력

3. 새 암호를 사용하여 vault 파일 보기

ansible-vault view salaries.yml

- 패스워드 입력

RED HAT ANSIBLE 2.7 EXAM

GOOD LUCK

============================================================================

16, 17은 시험에는 나오지 않았지만 덤프에는 있었던 내용, 랜덤하게 바뀔 수 있으니 숙지

커널 업데이트(16)

<문제>

- 최신 버전의 커널을 설치합니다.

- 모든 호스트 커널이 업데이트 된 후 다시 시작합니다.

- 재시작을 기다린 후 커널 버전 정보를 /root/update.txt에 기록합니다.

<정답>

- name: "kernel update"  
 hosts: all  
 tasks:  
 - name: "1. update kernel"  
 yum:  
 name: kernel  
 state: latest  
 register: result\_kernel  
 ignore\_errors: yes

- name: "2. if kernel is update over"  
 debug:  
 msg: "kernel is update over"  
 when: result\_kernel is failed  
 failed\_when: result\_kernel is failed

- name: "3. reboot host"  
 shell: "sleep 1 && shutdown -r now"  
 async: 1  
 poll: 0  
 ignore\_errors: yes

- name: "4. wait host start"  
 wait\_for:  
 host: "{{ inventory\_hostname }}"  
 state: started  
 delay: 60  
 timeout: 300  
 port: 22  
 delegate\_to: localhost

- name: "5. write update info to file"  
 shell: uname -r > /root/update.txt

<설명>

- 4번에서 student 유저는 delegate\_to를 를 통해 hosts가 all이 아닌 localhost로 명령을 실행한다. 근데 localhost로 명령을 실행하면, ssh가 아니라 local로 실행을 하게 되는 것으로 추측된다. 따라서 ansible.cfg에 devops계정으로 하는게 아니라, 그냥 student 유저가 sudo로 명령을 실행하는것이다 .따라서 student 유저가 localhost에서 바로 sudo를 쓸 수 있도록 sudoers.d에 설정을 넣어줘야 한다.

- 2번을 왜 하는지 모르겠다. 1번 작업은 이미 최신 상태에서 수행해도 fail이 아닌 true를 출력하기 때문이다.

- 그냥 shutdown -r now를 바로 써버리면, 다음 명령어를 넣을 수 없어서 sleep과 함께 쓴다고 한다.

- async와 poll은 비동기로 작업을 날린 후 (비동기라는 것은 ssh 화면과 서버가 서로 동기화되는데, 이걸 동기화 안한다는 것) async 기간동안 poll 간격으로 task를 확인 후 결과를 리턴하는 것이다. ansible의 기본 동작은 작업이 완료될 때까지 ssh 접속을 유지하기 때문에, ssh가 끊기는 작업이나 ssh 타임아웃 시간보다 더 긴 작업을 하는 경우 ssh가 끊겨서 timeout 에러가 발생한다. 따라서, 이를 위해 asnyc+poll을 사용하는 것이다. 여기서 poll은 원래 명시한 수치만큼의 간격으로 확인을 하는데, 0으로  해버리면 확인을 안하겠다는 것이다. 즉 비동기 전송 후 확인하지 않는다. 확인을 할라고 작업이 끝날때까지 기다리지 않는다. 그냥 넘어간다.

- 원래 모든 task는 hosts에 명시된 리스트에서 실행되는데, delegate\_to 는 그 리스트를 바꾼다. 즉 delegate\_to에 명시된 클라이언트를 대상으로 해당 task를 실행한다. 인벤토리에 있는 대상 외에 다른 대상에서 작업이 필요한 경우 이 옵션을 사용하여 다른 호스트 서버에서 작업할 수 있다.

- 4번의 delay는 해당 시간까지 기다린 후 polling(검사)를 시작하는 것이며, timeout은 시작하고 나서 명시한 시간동안까지 기다리고 그 시간이 지나면 fail을 내는 것이다.

- wait\_for는 여러가지 대상을 wait 할수 있는데, 여기서는 port를 wait 하며, 22번 port가 started 상태가 될 때까지 기다리는 것이다.

- 이 4번에서는 원격 호스트들이 모두 종료되므로, delegate\_to를 사용하여 ansible명령을 친 localhost에서 기다리는 작업을 실행하고, 22번 포트가 살아 올라오면 서버가 리스타트 되었다는 것이므로 작업을 지속할 수 있는 것이다.

<결과>

[student@control ansible]$ ansible all -m shell -a "cat /root/update2.txt"  
serverc | CHANGED | rc=0 >>  
3.10.0-1160.21.1.el7.x86\_64  
serverd | CHANGED | rc=0 >>  
3.10.0-1160.21.1.el7.x86\_64  
serverb | CHANGED | rc=0 >>  
3.10.0-1160.21.1.el7.x86\_64  
servera | CHANGED | rc=0 >>  
3.10.0-1160.21.1.el7.x86\_64  
[student@control ansible]$

파티션 생성 및 적용(17)

**<문제>**

/home/student/ansible/partition.yml에 스크립트를 작성하고 모든 호스트에서 실행합니다. 요구 사항은 다음과 같습니다.

1. vdb에서 기본 파티션을 만듭니다. 숫자는 1, 크기는 1500MiB입니다.

2. ext4 파일 시스템으로 포맷하고 /newpart 에 멈춥니다.

3. 요청 된 파티션 크기를 생성 할 수없는 경우 "Unable to create a partition of this size"라는 오류 메시지가 표시되어야하며 800Mib 크기를 사용해야합니다.

4. 장치 vdb가 존재하지 않는 경우 디스크가 존재하지 않는다는 오류 메시지가 표시되어야합니다.

**<정답>**

[student@control ansible]$ vi partition.yml  
- hosts: all  
 tasks:

- name: "1. check the existance of /dev/sdb"  
 shell: ls /dev/sdb  
 register: check\_sdb  
 ignore\_errors: yes

- name: "2. if /dev/sdb isn't exist, output error message"  
 debug:  
 msg: "sdb is not exist"  
 when: check\_sdb is failed  
 failed\_when: check\_sdb is failed

- name: "3. create a partition that size of the 1500MiB"

block:  
 - name: "3-1. create a sdb1"  
 parted:  
 number: 1 # 작업할 파티션 넘버 또는 생성할 파티션의 넘버  
 device: /dev/sdb  
 part\_start: 1MiB  
 part\_end: 1500MiB  
 state: present

rescue:  
 - name: "3-2. if some error occurs, show error message"  
 debug:  
 msg: "couldn't create a partition of that size"

- name: "3-3. create a partition that the size of the 800MiB"  
 parted:  
 number: 1  
 device: /dev/sdb  
 part\_start: 1MiB  
 part\_end: 800MiB  
 state: present

- name: "4. create filesystem"  
 filesystem:  
 dev: /dev/sdb1  
 fstype: ext4

- name: "5. create mount point"  
 file:  
 path: /newpart  
 state: directory  
 mode: '0755' # 콤마로 꼭 씌워 줘야 한다

- name: "6. mount the filesystem"  
 mount:  
 src: /dev/sdb1  
 path: /newpart  
 fstype: ext4  
 state: mounted

확인 - ansible all -m shell -a "lsblk"

[student@control ansible]$ ansible all -m shell -a "lsblk"  
serverc | CHANGED | rc=0 >>  
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT  
sda 8:0 0 40G 0 disk  
├─sda1 8:1 0 200M 0 part /boot/efi  
├─sda2 8:2 0 1G 0 part /boot  
└─sda3 8:3 0 38.8G 0 part  
 ├─centos\_k8s--node1-root 253:0 0 36.8G 0 lvm /  
 └─centos\_k8s--node1-swap 253:1 0 2G 0 lvm [SWAP]  
sdb 8:16 0 1G 0 disk  
└─sdb1 8:17 0 799M 0 part /newpart  
sdc 8:32 0 1G 0 disk  
sr0 11:0 1 1024M 0 rom  
serverd | CHANGED | rc=0 >>  
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT  
sda 8:0 0 40G 0 disk  
├─sda1 8:1 0 200M 0 part /boot/efi  
├─sda2 8:2 0 1G 0 part /boot  
└─sda3 8:3 0 38.8G 0 part  
 ├─centos\_k8s--node1-root 253:0 0 36.8G 0 lvm /  
 └─centos\_k8s--node1-swap 253:1 0 2G 0 lvm [SWAP]  
sdb 8:16 0 1G 0 disk  
└─sdb1 8:17 0 799M 0 part /newpart  
sdc 8:32 0 1G 0 disk  
sr0 11:0 1 1024M 0 rom  
servera | CHANGED | rc=0 >>  
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT  
sda 8:0 0 40G 0 disk  
├─sda1 8:1 0 200M 0 part /boot/efi  
├─sda2 8:2 0 1G 0 part /boot  
└─sda3 8:3 0 38.8G 0 part  
 ├─centos\_k8s--node1-root 253:0 0 36.8G 0 lvm /  
 └─centos\_k8s--node1-swap 253:1 0 2G 0 lvm [SWAP]  
sdb 8:16 0 1G 0 disk  
└─sdb1 8:17 0 799M 0 part /newpart  
sdc 8:32 0 1G 0 disk  
sr0 11:0 1 1024M 0 rom  
serverb | CHANGED | rc=0 >>  
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT  
sda 8:0 0 40G 0 disk  
├─sda1 8:1 0 200M 0 part /boot/efi  
├─sda2 8:2 0 1G 0 part /boot  
└─sda3 8:3 0 38.8G 0 part  
 ├─centos\_k8s--node1-root 253:0 0 36.8G 0 lvm /  
 └─centos\_k8s--node1-swap 253:1 0 2G 0 lvm [SWAP]  
sdb 8:16 0 1G 0 disk  
└─sdb1 8:17 0 799M 0 part /newpart  
sdc 8:32 0 1G 0 disk  
sr0 11:0 1 1024M 0 rom