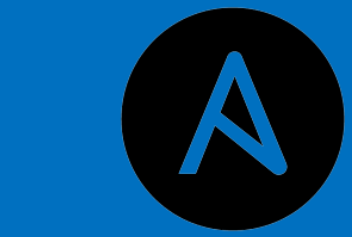


2장

Ad-hoc 명령 및 Playbook 개념

전체 내용

Ad-hoc 명령어 사용하기

Module 사용하기

Playbook 개념

쉬어가는 코너

Ad-hoc 명령어 사용하기

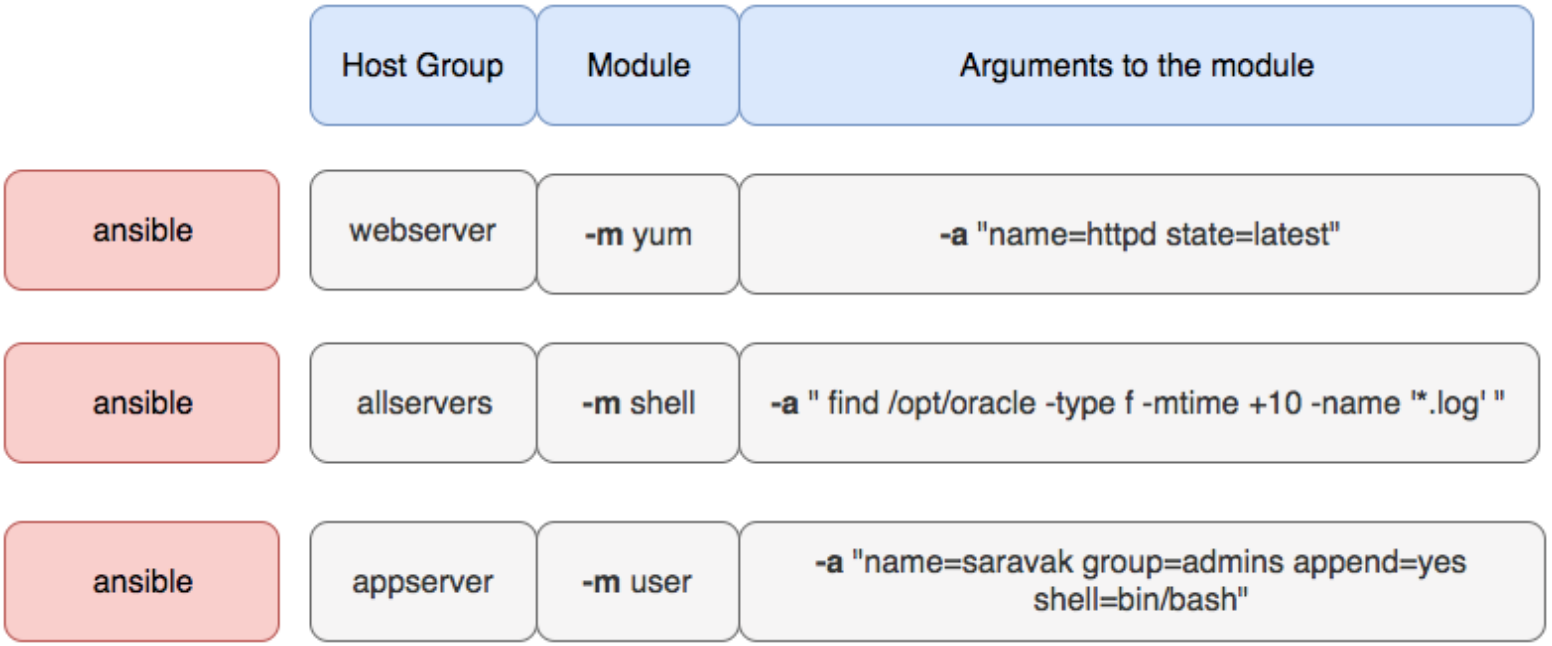
- Ansible의 명령어 종류
- Ansible Ad-hoc 실행

Ad-hoc 명령어 사용하기

- Ansible의 명령어 종류
  - **ansible**
    - Ad-Hoc 실행
  - **ansible-playbook**
    - Playbook 실행
  - **ansible-doc**
    - ansible 모듈 Documentation 조회
  - **ansible-galaxy** (=https://galaxy.ansible.com/)
    - **Ansible 자동화 소스의 GitHub**
    - Ansible Galaxy 커뮤니티 소스의 생성/설치/조회/삭제
    - Popular 지수 확인 및 다운로드 활용 가능
  - **ansible-vault**
    - SECRET 정보 암/복호화ansible에 ansible 패키지 설치하기

Ad-hoc 명령어 사용하기

- Ansible Ad-hoc 실행
  - Ad-hoc 명령어 사용에 대한 소개
    - https://docs.ansible.com/ansible/latest/user_guide/intro_adhoc.html
  - 명령 구문
    - **ansible** 호스트그룹명 **-m** 모듈명 **-a "module args"**



Ad-hoc 명령어 사용하기

- Ansible Ad-hoc 실행
  - AD HOC Command Examples – **Ansible Cheat Sheet**
    - https://www.middlewareinventory.com/blog/ansible-ad-hoc-commands/ (##강추)
  - ansible 버전 확인
    - **ansible --version**
  - ansible의 관리 대상(managed node) 확인하기
    - **cat /etc/ansible/hosts**
    - **ansible all --list-hosts**
    - **ansible all -i /etc/ansible/docker.host --list-hosts**
      - ## /etc/ansible/docker.host에 있는 모든 호스트

Ad-hoc 명령어 사용하기

- Ansible Ad-hoc 실행
  - 관리 대상에게 작업하기
    - **ansible all -m ping -k (##-k: 암호 입력하기)**
    - **ansible centos -m shell -a 'systemctl is-active sshd' -u adminuser -k (## -u는 remote user로 실행하기)**
      - ## -u는 원격 컴퓨터의 사용자이다. -u를 사용할 때는 꼭 -k를 사용하여 암호를 입력하도록 한다
      - ## -u를 사용하면서 -k를 사용하지 않으면 -u의 public key, private key로 인증 처리하는 셈이 된다**(##주의 요망)**
    - **ansible centos -m shell -a 'mkdir /lab/remote' -u adminuser -k**
      - ## remote에 있는 adminuser는 /lab 디렉터리에 새로운 디렉터리를 생성할 권한(permission)이 없어서 실패하게 된다
    - **ansible centos -m shell -a 'mkdir /lab/remote' -k**
      - ## 이렇게 하면 root로 실행하므로 /lab/에 remote라는 디렉터리가 생김

Ad-hoc 명령어 사용하기

- Ansible Ad-hoc 실행
  - 관리 대상에게 작업하기
    - 디렉터리와 파일 생성하기
      - **ansible all -m shell -a "mkdir /root/test ; touch /root/test/welcome.txt ; ls -l /root/test/"**
    - **ansible all -i /etc/ansible/docker.host --list-hosts**
    - docker 설치하고 실행하기
      - **ansible all -i /etc/ansible/docker.host -m shell -a "curl -sSL http://get.docker.com | sh"**
      - **ansible all -i /etc/ansible/docker.host -m shell -a "systemctl start docker ; docker version"**
    - nginx 서비스 설치하기
      - **ansible nginx -m yum -a "name=nginx"**
      - **ansible nginx -m service -a "name=nginx state=started"**
      - **ansible nginx -m shell -a "systemctl is-active nginx"**

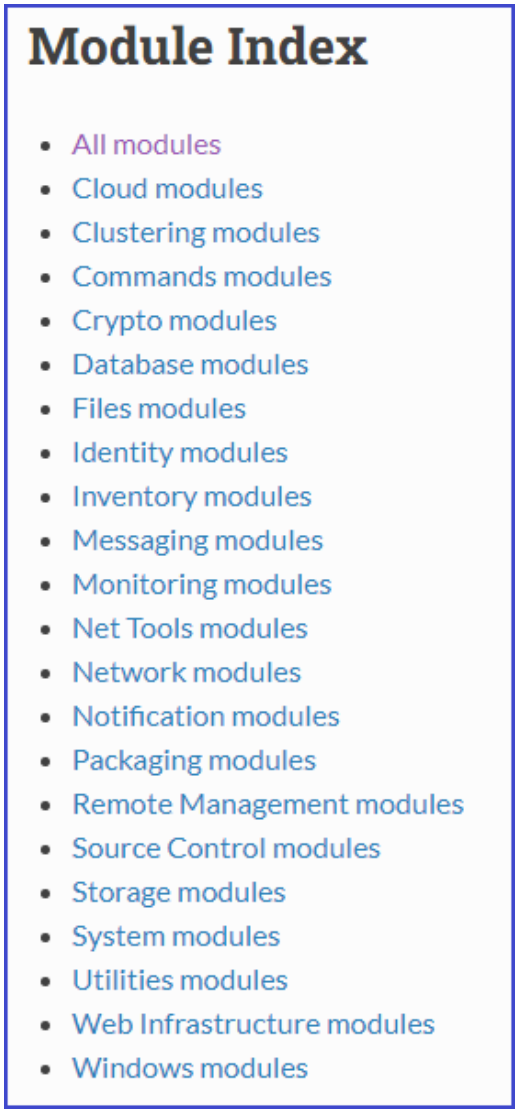
Module 사용하기

Module 이해

Module 종류별 사용하기

Module 사용하기

- Module 이해
  - Ansible Module
    - Ansible은 자동화 기능을 실행하는 다양하고 풍부한 모듈을 제공하며, 각각의 모듈 자체는 하나의 완전한 단위 기능을 제공한다
  - 모듈을 사용하는 Ansible 강점
    - 오픈소스 자동화 모듈을 빠르게 재활용하고 개발기간 단축
  - 다양한 종류의 Module에 대한 정보
    - https://docs.ansible.com/ansible/2.9/modules/modules_by_category.html **(##강추)**
  - 문서 보기
    - **ansible-doc yum**
    - **ansible-doc yum | grep -A 100 EXAM (강추)**
    - **ansible-doc setup | grep -A 100 EXAM**

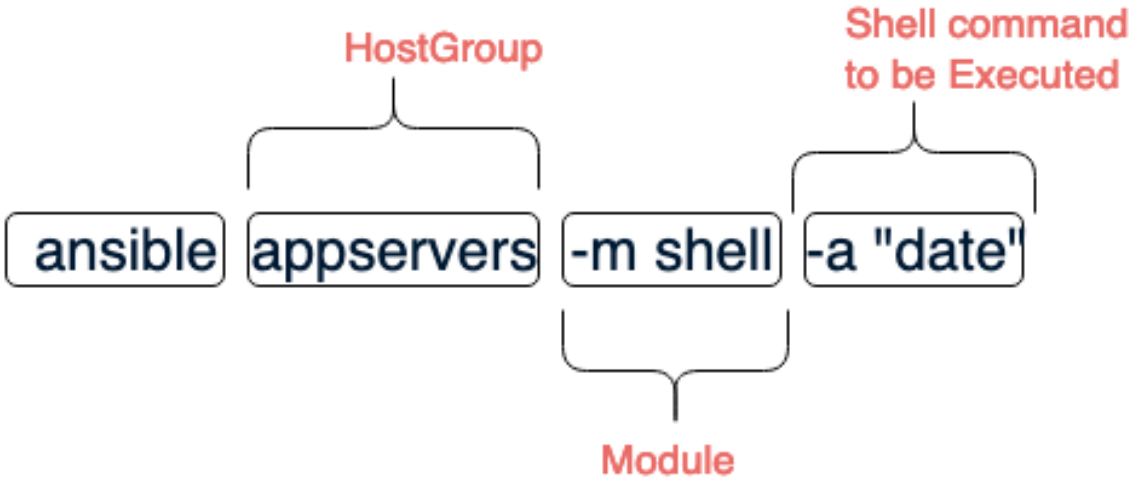


Module Index

- All modules
- Cloud modules
- Clustering modules
- Commands modules
- Crypto modules
- Database modules
- Files modules
- Identity modules
- Inventory modules
- Messaging modules
- Monitoring modules
- Net Tools modules
- Network modules
- Notification modules
- Packaging modules
- Remote Management modules
- Source Control modules
- Storage modules
- System modules
- Utilities modules
- Web Infrastructure modules
- Windows modules

Module 사용하기

- Module 이해
  - 모듈 사용 문법



Module 사용하기

- Module 종류별 사용하기
  - **ping 모듈**
    - 문서 보기
      - **ansible-doc ping**
      - ansible-doc ping | **grep -A 100 EXAMPLES (##예제)**
    - ansible로 서버들을 관리하기 전에 먼저 **-m ping**을 사용하여 SUCCESS가 나오는 것을 확인해야 한다
    - **ansible all -m ping**

Module 사용하기

- Module 종류별 사용하기
  - **shell 모듈**
    - 문서 보기
      - **ansible-doc** shell
      - **ansible-doc** shell | grep -A 100 EXAMPLES
    - 리눅스의 shell 명령어를 그대로 사용할 수 있고, 가장 많이 사용하는 modul이다. $HOME와 같은 변수, 파이프라인, Redirection 및 연산자(<, >, &, &&, ||)도 사용할 수 있다 (##command 모듈은 불가능)
    - **ansible all -m shell -a "date"**
    - **ansible all -m shell -a "timedatectl set-timezone Asia/Seoul"**
    - **ansible all -m shell -a "timedatectl | grep zone"**
    - **ansible all -m shell -a "mkdir /lab/testdir;touch /lab/testdir/hi.txt;ls -l /lab/testdir/"**
    - **ansible ubuntu -a "/sbin/reboot"** **(##-m shell 생략 가능)**
      - ## ansible ubuntu -m shell -a "reboot"와 동일

Module 사용하기

- Module 종류별 사용하기
  - **file 모듈**
    - 문서 보기
      - **ansible-doc file**
      - ansible-doc file | **grep -A 100 EXAMPLES (##예제)**
    - 파일/디렉터리에 대한 **상세정보를 보거나 설정한다**
      - Size, path, group, owner, mode(644)
    - 디렉터리 정보 상세 보기
      - **ansible node1 -m file -a 'path=/lab/'**

```
[root@ansible ~]# ansible node1 -m file -a 'path=/lab/'
node1 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "gid": 0,
    "group": "root",
    "mode": "0755",
    "owner": "root",
    "path": "/lab/",
    "size": 41,
    "state": "directory",
    "uid": 0
}
```

    - 디렉터리 생성하기
      - **ansible node1 -m file -a 'path=/lab/ansible/ state=directory mode=0700 owner=sudoadmin'**
      - **ansible node1 -m shell -a 'ls -l /lab/'**

```
[root@ansible ~]# ansible node1 -m shell -a 'ls -l /lab/'
node1 | CHANGED | rc=0 >>
total 0
drwx------ 2 sudoadmin root  6 Feb 22 16:20 ansible
```

Module 사용하기

- Module 종류별 사용하기
  - **file 모듈**
    - 파일/디렉터리를 만들거나 삭제할 때는 -m shell보다 **-m file**을 사용을 권장한다
    - **ansible all -m shell -a "mkdir -p /lab/ansible/"**

```
[root@ansible ~]# ansible all -m shell -a "mkdir -p /lab/ansible/"
[WARNING]: Consider using the file module with state=directory rather than running 'mkdir'.  If you need to
use command because file is insufficient you can add 'warn: false' to this command task or set
'command_warnings=False' in ansible.cfg to get rid of this message.
```

    - **ansible all -m shell -a "rm -rf /lab/ansible/"**
    - **ansible all -m file -a "dest=/lab/test/ state=directory"**
    - **ansible all -m file -a "dest=/lab/test/ state=absent"**
      - ## 디렉터리 생성(state=directory) 및 삭제(state=absent)
    - **ansible all -m shell -a "ls -l /lab/"**

Module 사용하기

- Module 종류별 사용하기
  - **copy 모듈**
    - 문서 보기
      - **ansible-doc copy**
      - ansible-doc copy | **grep -A 100 EXAMPLES (##예제)**
    - 파일을 복사할 때는 **-m copy**를 사용하고 src와 dest를 사용한다
    - **touch welcome.file**
    - **ansible all -m copy -a "src=/root/welcome.file dest=/lab/"**
    - **ansible all -m shell -a "ls -l /lab/"**

Module 사용하기

- Module 종류별 사용하기
  - **yum 및 service 모듈**
    - 문서 보기
      - **ansible-doc** yum
      - **ansible-doc** yum | grep -A 100 EXAMPLES
      - **ansible-doc** service
      - **ansible-doc** service | grep -A 100 EXAMPLES
    - -m yum은 **서비스를 설치/삭제하고**, -m service는 **서비스를 시작/중지한다**
    - **이 모듈들은 항상 쌍으로 사용한다**
    - **ansible all -m yum -a "name=wget state=present"**
    - **ansible all -m shell -a "wget http://down.cloudshell.kr/docker/Dockerfile"**
    - **ansible all -m shell -a "ls -l"**

Module 사용하기

- Module 종류별 사용하기
  - **yum 및 service 모듈**
    - **ansible all -m yum -a "name=vsftpd state=present"**
      - ## state가 present 대신 installed도 사용 가능
      - **## -m yum의 state: latest, present, absent, build-dep, fixed**
    - **ansible all -m service -a "name=vsftpd state=started enabled=yes"**
      - ## enabled=yes는 컴퓨터가 재부팅하더라도 자동으로 서비스 시작
      - **## -m service의 state: started, stopped, reloaded, restarted**
    - **ansible all -m shell -a "systemctl is-active vsftpd"**

    - **ansible all -m service -a "name=vsftpd state=stopped"**
    - **ansible all -m shell -a "systemctl is-active vsftpd"**
    - **ansible all -m yum -a "name=vsftpd state=absent"**
      - ## state가 absent 대신 removed도 사용 가능

Module 사용하기

- Module 종류별 사용하기
  - **apt 모듈**
    - **ansible ubuntu -m apt -a update_cache=true --become --ask-become-pass**
      - ## ubuntu는 yum이 아니고 apt를 사용한다
      - -a "update_cache=true"는 apt update와 같은 뜻이다
        - ubuntu에서는 필수로 실행할 명령
      - --become: sudo와 같다. 즉, 관리자 권한으로 실행
      - --ask-become-pass: 관리자 권한으로 실행할 때 사용하는 암호 입력

Module 사용하기

- Module 종류별 사용하기
  - **user 모듈**
    - 문서 보기
      - **ansible-doc** user | grep -A 100 EXAMPLES
    - 사용자를 생성/삭제하기 위해서 -m user를 사용한다
    - **ansible centos -m user -a "name=testuser1"**
    - **ansible centos -m shell -a "echo -e '1\n1\n' | passwd testuser1"**
      - ## 암호를 '1'로 한꺼번에 변경하는 작업
    - **ansible centos -m shell -a "tail -n 1 /etc/passwd"** (##생성한 사용자 확인)
    - **ansible centos -m user -a "name=testuser2"**
      - **ansible centos -a "tail -n 1 /etc/passwd"** (##user가 생성됨)
    - **ansible centos -m user -a "name=testuser2 state=absent"** **(##삭제)**
      - **ansible centos -a "tail -n 1 /etc/passwd"** (##user가 삭제됨)

Module 사용하기

- Module 종류별 사용하기
  - **user 모듈**
    - **ansible centos -m user -a "name=sudoadmin group=wheel"**
      - **ansible centos -a "id sudoadmin"**
    - **ansible ubuntu -m user -a "name=sudoadmin group=sudo"**
      - ##centos와 ubuntu에 sudo 명령을 사용할 수 있도록 sudoadmin을 관리자 권한으로 부여함
    - **ansible centos -m shell -a "echo -e '1\n1\n' | passwd sudoadmin"**
    - **ssh node4** (##ubuntu는 직접 암호 수정)
    - **passwd sudoamin**
    - **1**
    - **1**
    - **exit**
    - **ansible all -m shell -a "id sudoadmin"**

```
[root@ansible ~]# ansible all -m shell -a "id sudoadmin"
node4 | CHANGED | rc=0 >>
uid=1003(sudoadmin) gid=27(sudo) groups=27(sudo)
node1 | CHANGED | rc=0 >>
uid=1003(sudoadmin) gid=10(wheel) groups=10(wheel)
node3 | CHANGED | rc=0 >>
uid=1003(sudoadmin) gid=10(wheel) groups=10(wheel)
node2 | CHANGED | rc=0 >>
uid=1003(sudoadmin) gid=10(wheel) groups=10(wheel)
```

Module 사용하기

- Module 종류별 사용하기
  - **user 모듈**
    - **whoami**
      - ##현재는 root 사용자이다
    - 원격 컴퓨터에 있는 사용자에게 **관리자 권한을 부여하여, 그 사용자로** 관리 작업하기 위해서 **"--become-user 사용자"**를 사용한다
      - **ansible centos -m user -a "name=testuser100" --become-user sudoadmin**
      - **ansible centos -m user -a "name=ysleeuser200" --become-user testuser1**
      - **ansible centos -a "tail -n 2 /etc/passwd"**

Module 사용하기

- Module 종류별 사용하기
  - **setup 모듈 (=gather_facts)**
    - Ad-hoc에서 많이 사용하는 모듈중 하나는 **remote host에 대한 ansible fact를 수집**하는 setup 모듈이다
      - ansible fact에는 remote host 에 대한 정보로서 배포판 종류, 버전, IP Address, file system 정보 등이 있다
    - **ansible all -m setup**
    - 출력되는 양(각종 다양한 변수)이 너무 많으므로 **setup 모듈이 지원하는 필터 기능을 사용해서 필요한 데이터만 출력**한다
    - **ansible all -m setup -a "filter=ansible_distribution*"**
      - ansible_dist라는 문자열이 있는 fact 목록을 출력하며 **배포판 종류, 버전, 릴리스 코드명** 등을 출력
    - **ansible localhost -m setup | grep distribution**
      - ## grep을 사용해도 좋다

Module 사용하기

- Module 종류별 사용하기
  - **setup 모듈 (=gather_facts)**
    - **ansible localhost -m gather_facts**
    - **ansible localhost -m gather_facts | grep distribution**
      - 여기에 나오는 것들이 각종 사용가능한 **변수**들이다
      - 이 변수들을 이용하여 playbook을 사용할 때 유용하게 활용할 수 있다
      - **OS별로 패키지를 설치**하거나 특정한 IP에 대하여 작업을 할 수도 있다
      - when: **ansible_distribution** == "CentOS"
      - vi when.yml
      - **ansible-playbook** when.yml **-k**
      - **ansible centos -a "host www.google.com"**

```
[root@ansible ~]# ansible centos -a "host www.google.com"
node2 | CHANGED | rc=0 >>
www.google.com has address 142.250.76.132
www.google.com has IPv6 address 2404:6800:400a:80e::2004
node1 | CHANGED | rc=0 >>
www.google.com has address 142.250.76.132
www.google.com has IPv6 address 2404:6800:400a:805::2004
node3 | CHANGED | rc=0 >>
www.google.com has address 142.250.207.100
www.google.com has IPv6 address 2404:6800:400a:804::2004
```

```
---
- hosts: all
  tasks:
    - name: Install bind-utils
      yum:
        name: bind-utils
      when: ansible_distribution == "CentOS"


    - name: Update Repository Index
      apt:
        update_cache: yes
      when: ansible_distribution in ["Debian", "Ubuntu"]


    - name: Install apache2 package
      apt:
        name: apache2
        state: latest
      when: ansible_distribution == "Ubuntu"


    - name: Add php support for apache
      apt:
        name: libapache2-mod-php
        state: latest
      when: ansible_distribution == "Ubuntu"
```

      - **lynx** http://node4

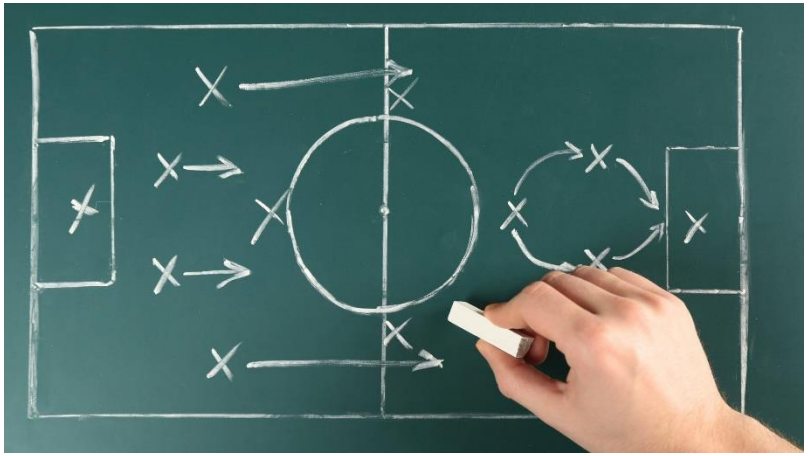
Playbook 사용하기

Ansible의 Playbook이란?

ansible-playbook 실행

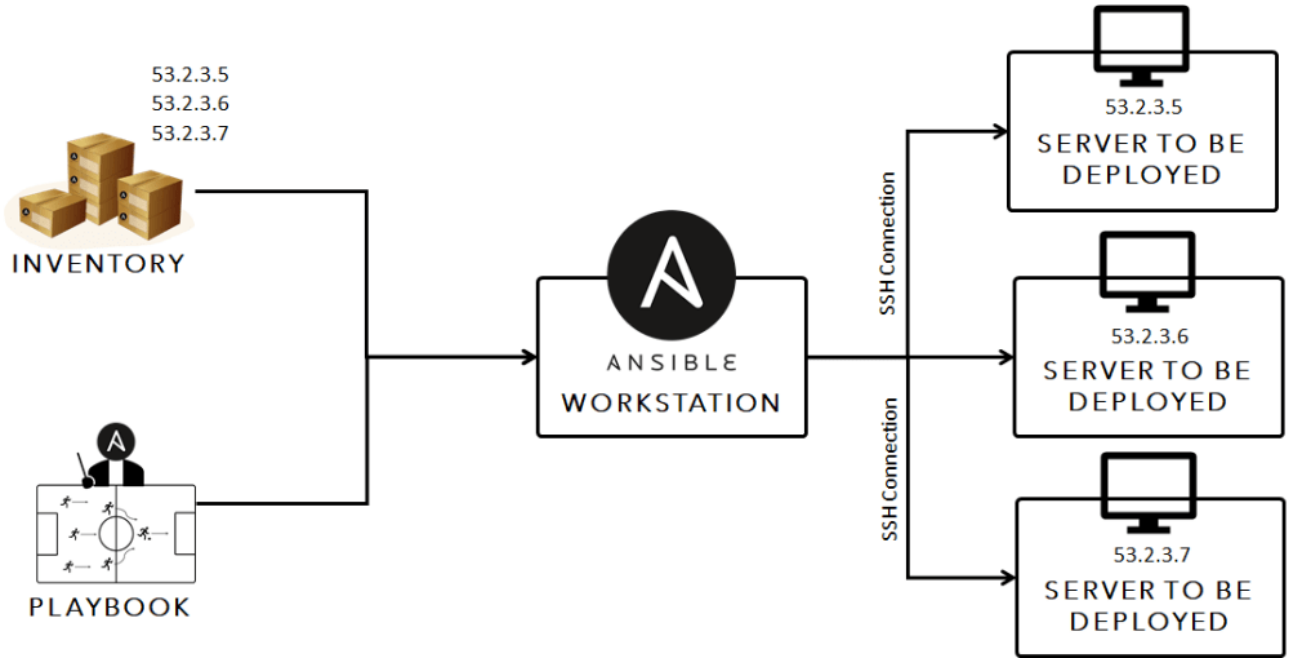
Playbook 사용하기

- Ansible의 Playbook이란?
  - Playbook이란?
    - 연극 대본, (스포츠)전술 계획표, 작전 수립도
      - 축구에서 상대방의 수비(또는 공격) 형태에 따라 우리 팀의 공격(또는 수비) 전술을 선택할 수 있는 공략집
      - 농구에서는 속공, 지공, 올코트 프레싱, 지역방어 등과 같은 전술을 수록해둔 작전 모음집



Playbook 사용하기

- Ansible의 Playbook이란?
  - **작업할 내용을 상세하게 정의한** Yaml 파일을 수행한다
  - **재사용**이 가능하고 **멱등성**을 제공한다
  - 이것을 사용하는 것을 **Infrastructure as Code(IaC)**라고 한다



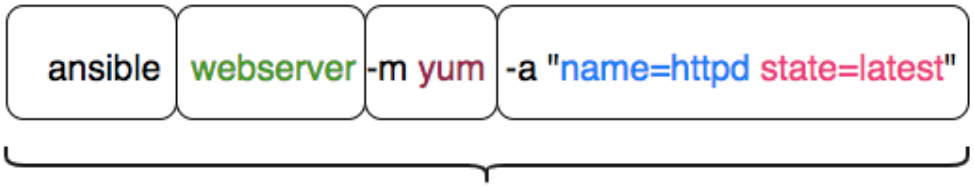
Playbook 사용하기

- ansible-playbook 실행
  - Playbook 명령 구문
    - **ansible-playbook 파일이름 --syntax-check** (##Yaml 구문 검사만)
    - **ansible-playbook 파일이름(.yaml 파일)**
  - Yaml 파일 특징
    - 들여쓰기는 Tab으로 하지 않고 Space로 한다
    - Array 또는 List 요소를 여러 줄에 사용할 때는 dash를 사용한다
    - 주석은 #으로 하고 한 줄 끝날 때가지만 유효
    - Key: Value를 할 때는 콜론을 사용한다
    - 하나의 스트림에 있는 여러 개의 문서는 dash 3개를 사용하여 나누고, period 3개로 스트림의 끝을 표시한다

Playbook 사용하기

- ansible-playbook 실행
  - Ad-hoc과 Playbook 비교하기

AD HOC command



Ansible Playbook

```
---
- name: playbook name
  hosts: webserver
  tasks:
    - name: name of the task
      yum:
        name: httpd
        state: latest
```

```
---
- name: Playbook          1   Name of Playbook
  hosts: webservers       2   HostGroup Name
3 become: yes                 Sudo (or) run as different user setting
  become_user: root
4 tasks:
    - name: ensure apache is at the latest version
      yum:
        name: httpd
        state: latest
    - name: ensure apache is running
      service:
        name: httpd
        state: started
```

Tasks

Playbook 사용하기

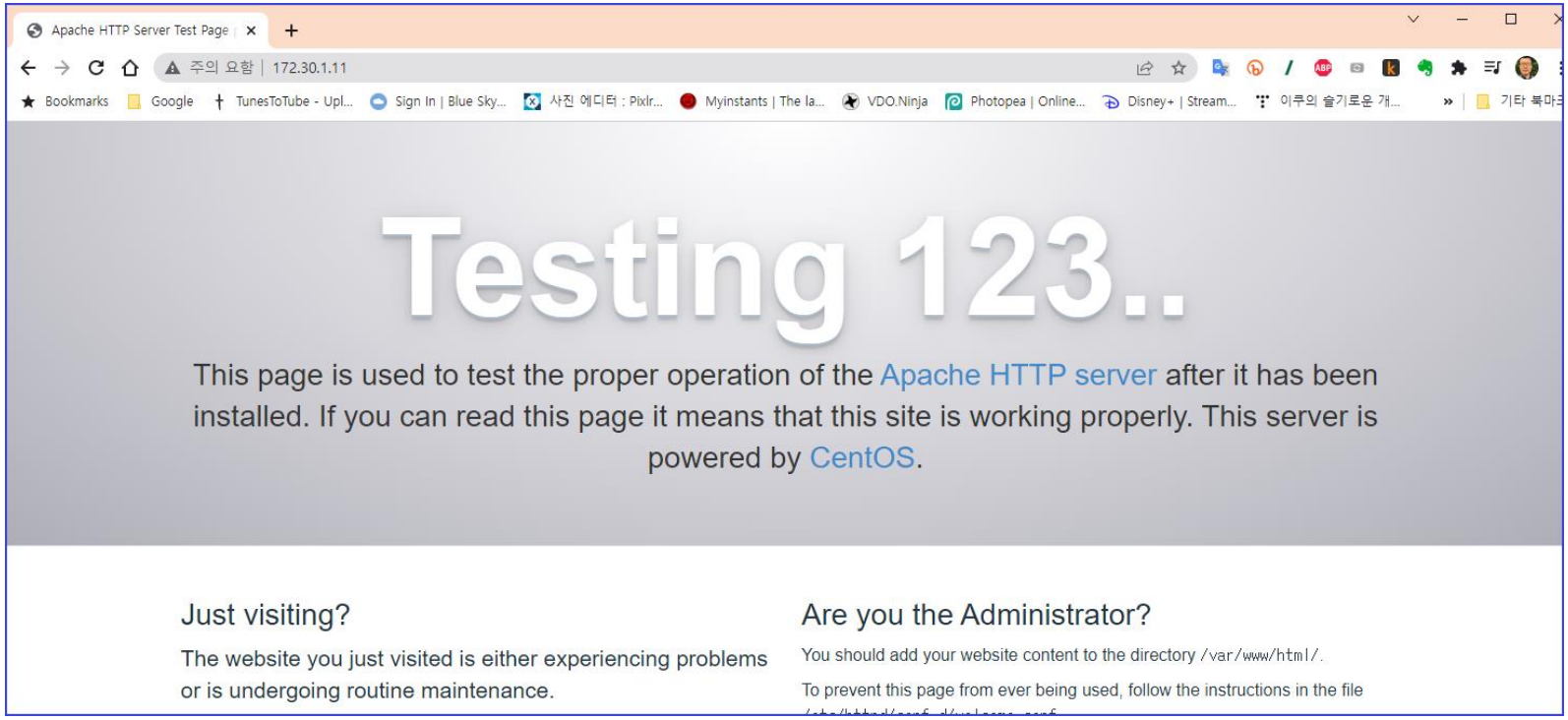
- ansible-playbook 실행
  - httpd를 설치하기 위한 yaml 파일 생성하기
    - mkdir -p /lab/playbook
    - vi !$/httpd.yaml

```
---
 - name: Apache server
   hosts: httpd
   become: yes
   become_user: root
   tasks:
     - name: Make sure that apache is at the
       yum:
         name: httpd
         state: latest
     - name: Ensure that apache is running
       service:
         name: httpd
         enabled: yes
         state: started
```

  - playbook 실행하기
    - **ansible-playbook** /lab/playbook/httpd.yaml **--syntax-check**
      - ## ymal 파일을 실행하기 전에 문법을 먼저 점검한다
    - **ansible-playbook** /lab/playbook/httpd.yaml

Playbook 사용하기

- ansible-playbook 실행
  - Apache Web Server가 실행중인지 확인하기
    - **curl node3**



Playbook 사용하기

- ansible-playbook 실행
  - Ansible에서 멱등성이란?
    - **동일한 기능을 여러 번 수행하지 않으므로** 성능이 좋고 빠르게 수행된다
    - 여러 번 적용해도 **바뀌는 부분이 있으면 그 부분만 반영**
    - **shell, command, file 모듈**은 **멱등성 보장 안됨**
    - 멱등성이 적용된 것은 **Green 색으로 표시**
    - 멱등성이 적용되지 않은 것은 **Yellow 색으로 표시**

  - 멱등성이 **안되는** 예제: **echo 명령어**
    - echo "node1" >> /etc/ansible/docker.host
    - echo "node3" >> /etc/ansible/docker.host
    - cat /etc/ansible/docker.host
      - 같은 값이 추가적으로 입력됨
    - 새롭게 추가한 것들을 삭제한다

```
cat /etc/ansible/docker.host
node1
node3
node1
node3
```

Playbook 사용하기

- ansible-playbook 실행
  - 멱등성이 되는 예제: **lineinfile 모듈**
    - **ansible localhost -c local -m lineinfile** -a "path=/etc/ansible/docker.host **line=node1**"

```
[root@ansible playbook]# ansible localhost -c local -m lineinfile
"
localhost | SUCCESS => {
    "backup": "",
    "changed": false,
    "msg": ""
}
[root@ansible playbook]# cat /etc/ansible/docker.host
node1
node3
[root@ansible playbook]#
```

Green 색이므로 멱등성이 적용이 된 것임

node1이 추가되지 않음

  - ansible에는 멱등성이 적용되는 모듈이 많이 있어서 성능이 좋다

Playbook 사용하기

- ansible-playbook 실행
  - 멱등성이 되는 예제: **httpd 그룹의 홈페이 파일 변경**
    - https://www.nginx.com 홈페이지의 index.html 파일 다운로드하기
      - **cd /lab/playbook/**
      - **curl -o index.html https://www.nginx.com**
      - **ls -l**
    - Playbook 파일인 httpd.yaml 파일 수정하기
      - **vi httpd.yaml**
      - 우측과 같이 새로운 내용을 입력
      - index.html 파일을 복사함

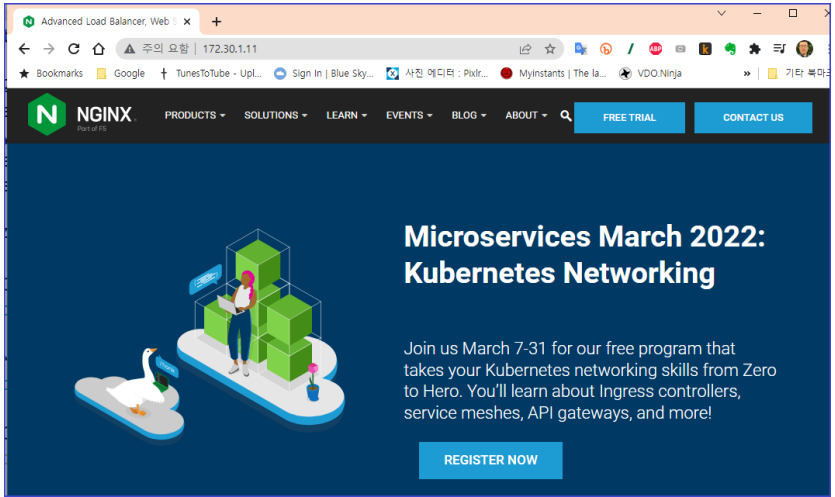
```
---
  - name: Apache server
    hosts: httpd
    become: yes
    become_user: root
    tasks:
      - name: Make sure that apache is at the latest
        yum:
          name: httpd
          state: latest
      - name: Copy index.html file downloaded from ht
        copy:
          src: index.html
          dest: /usr/share/httpd/noindex/index.html
          mode: 0644
      - name: Ensure that apache is running
        service:
          name: httpd
          enabled: yes
          state: started
```

Playbook 사용하기

- ansible-playbook 실행
  - 멱등성이 되는 예제: **httpd 그룹의 홈페이지 파일 변경**
    - playbook 실행하기
      - **ansible-playbook** httpd.yaml

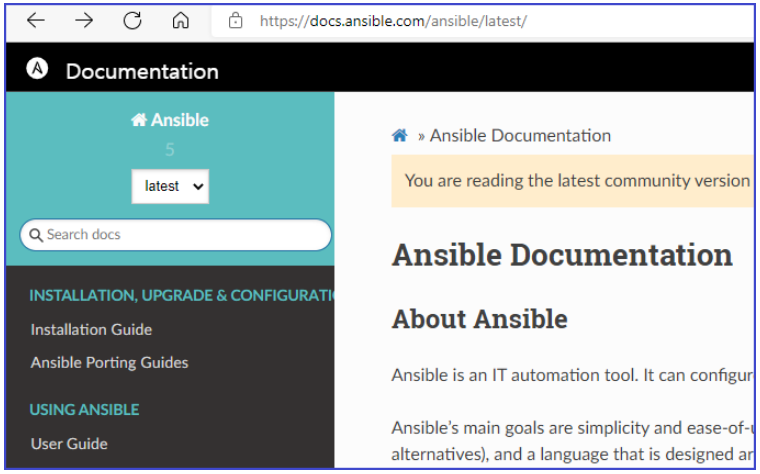
```
TASK [Copy index.html file downloaded from https://www.nginx.com to node3] *
changed: [node3]

TASK [Ensure that apache is running] ************************************
ok: [node3]

PLAY RECAP **************************************************************
node3                      : ok=4    changed=1    unreachable=0    failed=0
```

      - 추가된 항목만 적용되고, 기존의 것은 적용되지 않아서 멱등성의 혜택
    - 홈페이지 변경 여부 확인하기
      - Apache 서버인데 내용은 nginx



Playbook 사용하기

- Ansible-doc 실행
  - ansible 문서 웹싸이트
    - **https://docs.ansible.com/**
    - https://docs.ansible.com/**ansible/latest/**



  - Ansible Document 조회하기
    - **ansible-doc** yum
    - **ansible-doc -s** yum (##-s: snippet)

Playbook 사용하기

- Ansible-doc 실행
  - 명령 구문
    - **ansible-doc 모듈이름**
    - **ansible-doc -type <plugin type> -l**
  - ansible 문서 도움말
    - **ansible-doc --help**
  - Ansible Document 조회하기
    - **ansible-doc --version**
    - **ansible-doc** copy
    - **ansible-doc** yum
    - **ansible-doc -s** yum (##-s: snippet)
    - **ansible-doc -l** (##모든 플러그인 목록 확인)
    - **ansible-doc** -t connection -l (##-l: list / connection과 관련된 plugin)
    - **ansible-doc** -t connection **-s ssh**

쉬어가는 코너

"ansible all -m ping -k" 실행시 오류 해결

ubuntu-18.04에서 ansible 명령 실행시 오류 문제

ubuntu-18.04에 ansible 설치하여 관리하기

Command 모듈 vs Shell 모듈

Script 실행하기

Managed node 지정하는 방법

일반 User로 Ansible 관리하기

"ansible all -m ping -k" 실행시 오류 해결

- **ansible all -m ping -k" 실행시 오류 해결**
  - ansible을 설치한 후 호스트 그룹에게 ping을 할 때 아래와 같은 오류가 생기는 경우가 있다
    - Using a SSH password instead of a key is not possible **because Host Key checking is enabled** and sshpass does not support this. Please add this host's fingerprint to your known_hosts file to manage this host.
  - **/etc/ansible/ansible.cfg** 파일에 다음 내용을 입력하여 환경 변수를 추가하면 해결할 수 있다
    - [defaults]
      host_key_checking = False

ubuntu-18.04에서 ansible 명령 실행시 오류 문제

- **Python3을 사용하고 있는 ubuntu-18.04의 문제**
  - ansible all **-m yum -a "name=tree state=present"**
    - ubuntu-18.04는 python3 폴더가 아니라 python 폴더를 기본 python경로로 사용하고 있기 때문에 다음과 같은 오류 발생

```
[DEPRECATION WARNING]: Distribution Ubuntu 18.04 on host node4 should use /usr/bin/python3, but is using
/usr/bin/python for backward compatibility with prior Ansible releases. A future Ansible release will default
 to using the discovered platform python for this host. See
```

  - Managed Node에 Default Python이 3이든 2든 상관없이 Python 3만 설치되어 있으면 아래와 같이 하면 문제 해결
    - Inventory에서 파라미터 변수로 python3폴더의 경로를 지정한다
    - vi /etc/ansible/hosts

```
[ubuntu]
node4
[ubuntu:vars]
ansible_python_interpreter=/usr/bin/python3
```

      - **[ubuntu:vars]**
        ansible_python_interpreter=/usr/bin/python3
    - ansible all **-m yum -a "name=tree state=present"**
      - ## 더 이상 오류 메시지 발생하지 않음

Ubuntu-18.04에 Ansible 설치하여 관리하기

- **Ubuntu에 Ansible 설치하기**
  - Control(Management) node를 Ubuntu로 구성하기
    - **Control(Management) node**: Ansible을 설치하여 서버를 관리
    - **Managed node(host):** Ansible로 관리되는 대상 서버
    - 필요한 패키지 설치하기
      - **apt update**
      - **apt install -y ansible**
    - Ansible 설치 확인하기
      - **ansible --version**

Command 모듈 vs Shell 모듈

- **Command 모듈 vs Shell 모듈**
  - 가장 많이 사용하는 모듈이 shell 모듈이다
  - 그런데 command 모듈과 shell 모듈이 거의 비슷한데, 이 둘의 차이가 무엇인가?
  - **단순 명령**의 예제는 결과가 동일
    - **ansible centos -m command -a "hostname"**
      - **= ansible centos -a "hostname"** (##-m command를 생략한 것임)
      - ansible에서 module을 지정하지 않으면 default가 -m command이다
    - **ansible centos -m shell -a "hostname"**
  - pipeline과 redirection은 shell 모듈에서만 사용 가능
    - **ansible centos -a "hostname > /tmp/t.txt"** (##실패)
    - **ansible centos -m shell -a "hostname > /tmp/t.txt"** (##성공)
    - **ansible centos -a "cat /etc/passwd | grep root"** (##실패)
    - **ansible centos -m shell -a "cat /etc/passwd | grep root"** (##성공)

script 실행하기

- **Script 실행하기**
  - Ansible을 사용하여 원격 컴퓨터에 Script를 실행하기 위한 전제 조건은 원격 컴퓨터에 실행할 script 파일이 저장되어 있어야 한다는 것이다
  - script 생성하기
    - **vi test.sh**

```
#!/bin/bash
touch /lab/myfile.txt
```

    - **chmod +x test.sh**
  - 원격 컴퓨터에 script 복사하기
    - **ansible centos -m copy -a "src=/root/test.sh dest=/root/test.sh"**
  - Ansible로 script 실행하기
    - **ansible centos -m shell -a "sh /root/test.sh"**
    - **ansible centos -m shell -a "ls -l /lab/"**

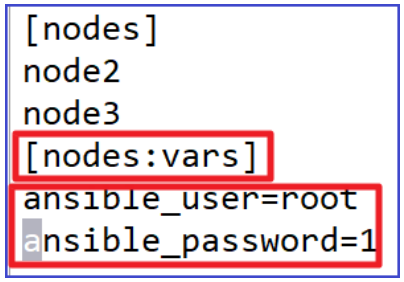
Managed node 지정하는 방법

- **Managed node 지정하는 방법**
  - Ansible 명령어가 실행되는 컴퓨터에 대한 정보를 가지고 있는 파일은 2개가 있다
    - **/etc/ansible/hosts** 파일
    - 스스로 만든 inventory 파일 사용 **(/etc/ansible/docker.host)**
  - Managed node 지정하기
    - **-i 옵션(--inventory-file)을 사용하지 않으면** default로는 /etc/ansible/hosts 파일의 내용을 참고한다
    - **-i 옵션을 사용하면 우선적으로** 해당 파일의 내용을 참고한다
    - 해당 파일에서는 **[ ]**로 **그룹을 지정**할 수 있다
    - **ansible all -m ping** (##etc/ansible/hosts 파일 사용)
    - **ansible centos -m ping** (##etc/ansible/hosts 파일 사용)

    - **ansible all -i /etc/ansible/docker.host -m ping**
    - **ansible docker -i /etc/ansible/docker.host -m ping**

항상 root 계정으로 실행하기

- **항상 root 계정으로 ansible 실행하기**
  - ansible 명령어를 실행할 때 어떤 계정으로 실행되는가?
    - 먼저 **whoami**를 실행했을 때 **그 계정으로 ansible 문장이 실행된다**
    - 다른 계정으로 실행하고자 하면 **-u remoteuser**를 사용하면 된다
  - 그런데 관리자이든 일반 사용자이든 무조건 root 계정으로 실행하고자 하면 어떻게 하면 되는가?
    - **/etc/ansible/hosts** 및 **/etc/ansible/inventory.file**에 변수를 입력하면 된다
    - vi /etc/ansible/hosts
      [nodes:vars]
      ansible_user=root
      ansible_password=1



    - 이것을 사용하면 -u remoteuser가 적용되지 않으니 주의해야 한다

항상 root 계정으로 실행하기

- **항상 root 계정으로 ansible 실행하기**
  - /etc/ansible/hosts에 **ansible_user=root가 적용된 경우**
    - **whoami**
    - **ansible nodes -m shell -a "touch ~/good.file"**
    - **ansible nodes -m shell -a "touch ~/good1.file" -k**
    - **ansible nodes -m shell -a "touch ~/bad1.file" -u heart**
    - **ansible nodes -m shell -a "touch ~/bad2.file" -u heart -k**
    - **ansible nodes -m shell -a "ls -l /root/"**

```
[root@node1 ~]# ansible nodes -m shell -a "ls -l /root/"
node3 | CHANGED | rc=0 >>
total 0
-rw-r--r-- 1 root root 0 Feb 23 12:49 bad1.file
-rw-r--r-- 1 root root 0 Feb 23 12:49 bad2.file
-rw-r--r-- 1 root root 0 Feb 23 12:48 good1.file
-rw-r--r-- 1 root root 0 Feb 23 12:48 good.file
```

    - ## ansible_user=root로 설정되어 있으면 -user heart가 적용되지 않고 모두 root 계정으로 실행된 것을 파일 생성 결과로 알 수 있다
    - ## root가 아닌 adminuser인 경우도 이와 동일한 결과가 나온다

항상 root 계정으로 실행하기

- **항상 root 계정으로 ansible 실행하기**
  - /etc/ansible/hosts에 **ansible_user=root가 없는 경우**
    - vi /etc/ansible/hosts
      [nodes:vars] 이하 모두 삭제할 것
    - **ansible nodes -m shell -a "rm -rf /root/*" -k** (##암호를 꼭 입력)
    - **ansible nodes -m shell -a "ls -l /root/" -k** (## 파일이 모두 삭제됨)

    - **ansible nodes -m shell -a "touch ~/good.file"** (##실패)
    - **ansible nodes -m shell -a "touch ~/good1.file" -k**
    - **ansible nodes -m shell -a "touch ~/bad1.file" -u heart** (##실패)
    - **ansible nodes -m shell -a "touch ~/bad2.file" -u heart -k**

항상 root 계정으로 실행하기

- **항상 root 계정으로 ansible 실행하기**
  - /etc/ansible/hosts에 **ansible_user=root가 없는 경우**
    - **ansible nodes -m shell -a "ls -l /root/" -k**

```
[root@node1 ~]# ansible nodes -m shell -a "ls -l /root/" -k
SSH password:
node3 | CHANGED | rc=0 >>
total 0
-rw-r--r-- 1 root root 0 Feb 23 13:06 good1.file
```

    - **ansible nodes -m shell -a "ls -l /home/heart" -k**

```
[root@node1 ~]# ansible nodes -m shell -a "ls -l /home/heart" -k
SSH password:
node2 | CHANGED | rc=0 >>
total 0
-rw-rw-r-- 1 heart heart 0 Feb 23 13:08 bad2.file
```

    - ## 각각 1개의 파일만 생성되었고, -u heart -k를 했을 때는 heart 사용자의 홈디렉터리에 파일이 생성된 것을 알 수 있다