Ansible

Ansible

- 15 Top Open-Source DevOps Tools in 2024
  - ✓ https://itechcraft.com/blog/top-open-source-tools-and-technologies/

Ansible

- 15 Top Open-Source DevOps Tools in 2024
  - ✓ 종류
    - Kubernetes – Docker Swarm
    - Jenkins – GitHub Actions
    - GitHub Actions – TeamCity
    - GitLab – GitHub
    - ArgoCD – Flux
    - Terraform – Puppet
    - Progress Chef – Ansible
    - Puppet – Progress Chef
    - Ansible – Terraform
    - Prometheus – Nagios
    - Grafana – Datadog
    - Bamboo – Jenkins
    - OpenStack – AWS
    - AWS – Google Cloud
    - Azure – IBM Cloud

Ansible

- ❖ Infracstructure as Code
  - ✓ 개요
    - 시스템, 하드웨어 또는 인터페이스의 구성 정보를 파일(스크립트)을 통해 관리 및 프로비저닝
    - IT 인프라스트럭처, 베어 메탈 서버 등의 물리 장비 및 가상 머신과 관련된 구성 리소스를 관리
    - 버전 관리를 통한 리소스 관리
  - ✓ 종류

Ansible

- Infracstructure as Code
  - ✓ 도구 비교

항목	Terraform	Ansible
개발 회사	HashiCorp	Red Hat
주요 사용 용도	클라우드 인프라 관리	서버 구성 관리, 배포
도구 종류	IaC 도구	IT 자동화 도구
작업 정의 언어	HCL	YAML
에이전트 필요성	없음	없음 (에이전트리스)
모듈/플러그인	프로바이더	모듈
클라우드 지원	다양한 클라우드 프로바이더	일부 클라우드 프로바이더

Ansible

- Ansible
  - ✓ 개요
    - 여러 개의 서버를 효율적으로 관리할 수 있게 해주는 파이썬으로 만들어진 환경 구성 자동화 도구
    - Configuration Management, Deployment & Orchestration tool
    - IT infrastructure 자동화
  - ✓ 작업 방식
    - 모든 호스트에 대해 병렬로 작업을 수행
    - 다음 작업으로 넘어가기 전에 모든 호스트에서 작업이 완료될 때까지 대기
    - 작업을 지정한 순서대로 수행

Ansible

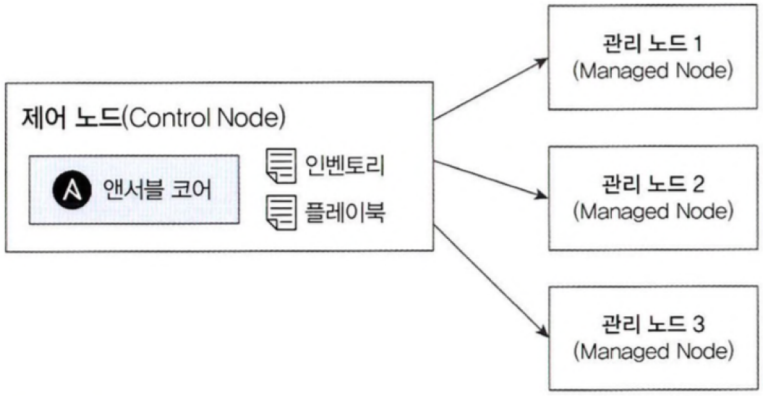
- Ansible
  - ✓ 장점
    - Agentless
      - 기존의 IaC(Puppet, Chef)에서는 클라이언트에 Agent를 설치해서 작업을 수행했는데 Ansible은 Python만 설치되어 있으면 연결해서 작업 수행 가능
      - 데몬 형식의 에이전트에 기반한 자동화 도구는 관리를 위한 복잡한 추가 작업이나 운영체제 버전에 따라 추가 패키지나 모듈을 설치하는 등의 작업이 발생
      - 앤서블은 에이전트 설치 없이 SSH로 접속하여 쉽게 대상 서버들을 관리
      - 앤서블의 가장 큰 특징이자 장점
    - Idempotency(멱등성): 동일한 연산을 여러 번 적용하더라도 결과가 달라지지 않는 성질
    - Simple
      - Ansible은 YAML 파일 형식과 Jinja2 템플릿을 사용하며 선택 가능
      - 파일 복사와 같은 일반 시스템 관리 모듈부터 다양한 환경의 퍼블릭 클라우드 관련 모듈 및 컬렉션까지 제공하므로 쉽게 플레이북 예제를 찾아보고 자동화를 수행할 수 있음
    - Push 기반 서비스

Ansible

- Ansible
  - ✓ 수행 가능한 작업
    - 소프트웨어 설치: apt-get, yum, homebrew
    - 파일 및 스크립트 배포: copy
    - 다운로드: get_url
    - 실행: shell, task
    - Version Control: git
    - Secrets Management
    - Infrastructure: AWS, Azure, Google Cloud, VMWare, OpenStack
    - Feedback
    - Containers: Docker, kubernetes
    - Build & Test: junit, maven, gradle
    - Operating System: linux, windows
  - ✓ 결과
    - ok
    - failed
    - changed
    - unreachable

Ansible

- Ansible
  - ✓ 사용 가능 환경
    - 앤서블은 리눅스, MacOS, BSD 계열 유닉스, WSL을 지원하는 윈도우에 파이썬과 앤서블 코어만 설치하면 어디에서나 플레이북(YAML 형식의 작업들을 순서대로 작성해 놓은 파일)을 작성하고 이를 실행시킬 수 있음
    - 앤서블은 이렇게 앤서블 코어가 설치되고 플레이북을 작성하여 실행할 수 있는 제어 노드와 플레이북이 실행되어 애플리케이션 설치나 클라우드 시스템의 가상 서버 생성과 같은 작업이 수행되는 관리 노드로 구성됨
    - 앤서블은 제어 노드에만 설치되고 관리 노드에는 설치되지 않음
    - 제어 노드에는 앤서블 코어가 설치되며 사용자에 의해 정의된 플레이북과 관리 노드를 정의해놓은 인벤토리 파일에 의해 SSH 프로토콜을 사용하여 다양한 환경의 관리 노드 업무 자동화를 수행



Ansible

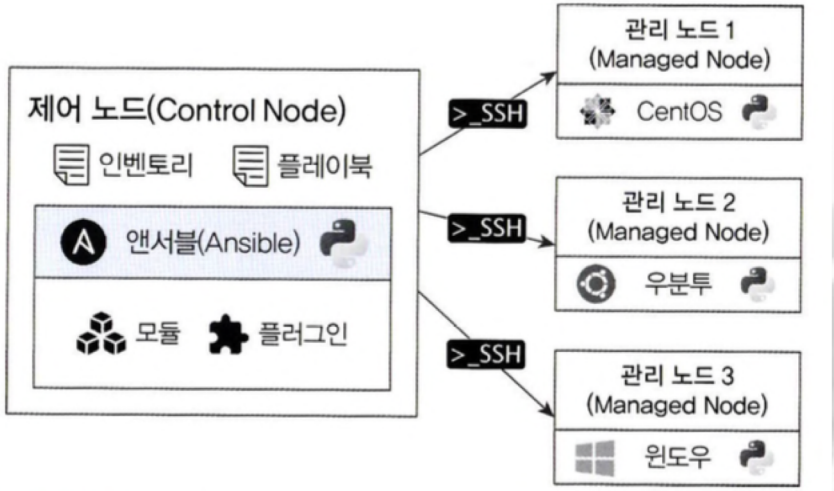
- Ansible
  - ✓ 관련 자료
    - 공식 git hub: https://github.com/ansible
    - 공식 문서: https://docs.ansible.com/ansible/latest/index.html
    - 공식 블로그: https://www.redhat.com/ko/engage/security-automation-ebook?sc_cid=RHCTA0250000454613&gad_source=1&gad_campaignid=22743724974&gbraid=0AAAADsbVMTrhf5ebKk38UNZOt9ccK_oP&gclid=CjwKCAjw7rbEBhB5EiwA1V49nbYGhnTS6XnMIx39aI9GWZnA6C9dLJGwIShbbCqLoGYBkt4M5tHngRoC_vkQAvD_BwE

Ansible

- ❖ Ansible Architecture
  - ✓ 유형
    - Community Ansible
    - Red Hat Ansible Automation Platform

Ansible

- ❖ Ansible Architecture
  - ✓ Community Ansible
    - 앤서블 아키텍처는 제어 노드와 관리 노드라는 두 가지 유형의 시스템으로 구성됨
    - 앤서블은 제어 노드에 설치되어 실행되며 앤서블이 실행되기 위해서는 파이썬이 기본적으로 설치되어 있어야 함
    - 앤서블 안에는 다양한 모듈과 플러그인이 함께 설치되어 있으며 앤서블이 관리하는 노드 정보를 저장하고 있는 인벤토리와 관리 노드에서 수행될 작업 절차가 작성되어 있는 플레이북이 존재
    - Community Ansible Architecture



Ansible

- Ansible Architecture
  - ✓ Community Ansible
    - 제어 노드
      - 제어 노드는 앤서블이 설치되는 노드로 운영체제가 리눅스라면 제어 노드가 될 수 있음
      - 앤서블은 파이썬 모듈을 이용하므로 앤서블을 설치하고 실행하려면 파이썬이 함께 설치되어 있어야 함
    - 관리 노드
      - 관리 노드는 앤서블이 제어하는 원격 시스템 또는 호스트를 의미
      - 관리 노드는 리눅스가 설치된 노드일 수도 있고 윈도우가 설치된 노드일 수도 있음
      - 퍼블릭 클라우드나 프라이빗 클라우드 시스템에서 생성한 가상 서버가 될 수도 있음
      - 앤서블은 별도의 에이전트를 설치하지 않으므로 관리 노드는 제어 노드와 SSH 통신이 가능해야 하며 파이썬이 설치되어 있어야 함

Ansible

- ❖ Ansible Architecture
  - ✓ Community Ansible
    - ● 인벤토리
      - ◆ 인벤토리는 제어 노드가 제어하는 관리 노드의 목록을 나열해 놓은 파일
      - ◆ 앤서블은 인벤토리에 사전에 정의되어 있는 관리 노드에만 접근할 수 있음
      - ◆ 인벤토리 목록은 관리 노드의 성격 별로 그룹핑 할 수 있음
      - ◆ $ vi inventory

        192.168.10.101

        [Webserver]

        web1.example.com

        web2.example.com

        [DBServer]

        db1.example.com

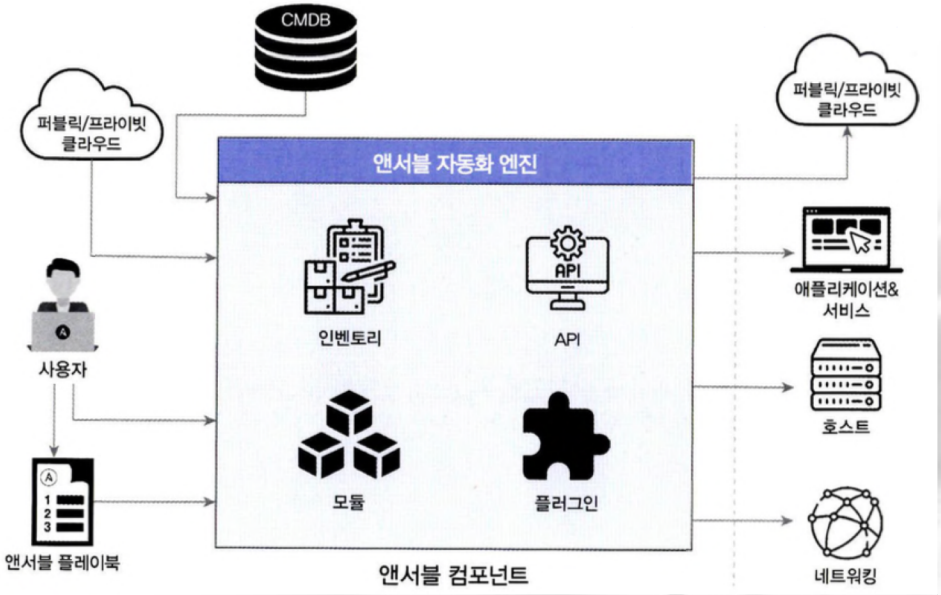
        db2.example.com

Ansible

- ❖ Ansible Architecture
  - ✓ Community Ansible
    - 모듈
      - ◆ 앤서블은 관리 노드의 작업을 수행할 때 SSH를 통해 연결한 후 앤서블 모듈 Rules 라는 스크립트를 푸시하여 작동
      - ◆ 대부분의 모듈은 원하는 시스템 상태를 설명하는 매개 변수를 허용하며 모듈 실행이 완료되면 제거됨
    - 플러그인
      - ◆ 모듈이 대상 시스템에서 별도의 프로세스로 실행되는 동안 플러그인은 제어 노드에서 실행
      - ◆ 플러그인은 앤서블의 핵심 기능(데이터 변환, 로그 출력, 인벤토리 연결 등)에 대한 옵션 및 확장 기능을 제공
    - 플레이북
      - ◆ 관리 노드에서 수행할 작업들을 YAML 문법을 이용해 순서대로 작성해놓은 파일
      - ◆ 플레이북을 활용하여 관리 노드에 SSH로 접근해 작업을 수행
      - ◆ 플레이북은 자동화를 완성하는 가장 중요한 파일이며 사용자가 직접 작성

Ansible

- ❖ Ansible Architecture
  - ✓ Ansible Automation Platform(AAP)
    - 커뮤니티 앤서블이나 레드햇의 앤서블 오토메이션 플랫폼의 주요 컴포넌트는 동일
    - 앤서블 오토메이션 플랫폼에는 커뮤니티 앤서블과 다르게 인벤토리, 제어 노드에 대한 인증 정보, 실행 환경 등을 관리하는 CMDB(Configuration Management DataBase)가 존재
    - 자원들을 관리하는 관리 웹 UI가 존재하며 REST API를 제공

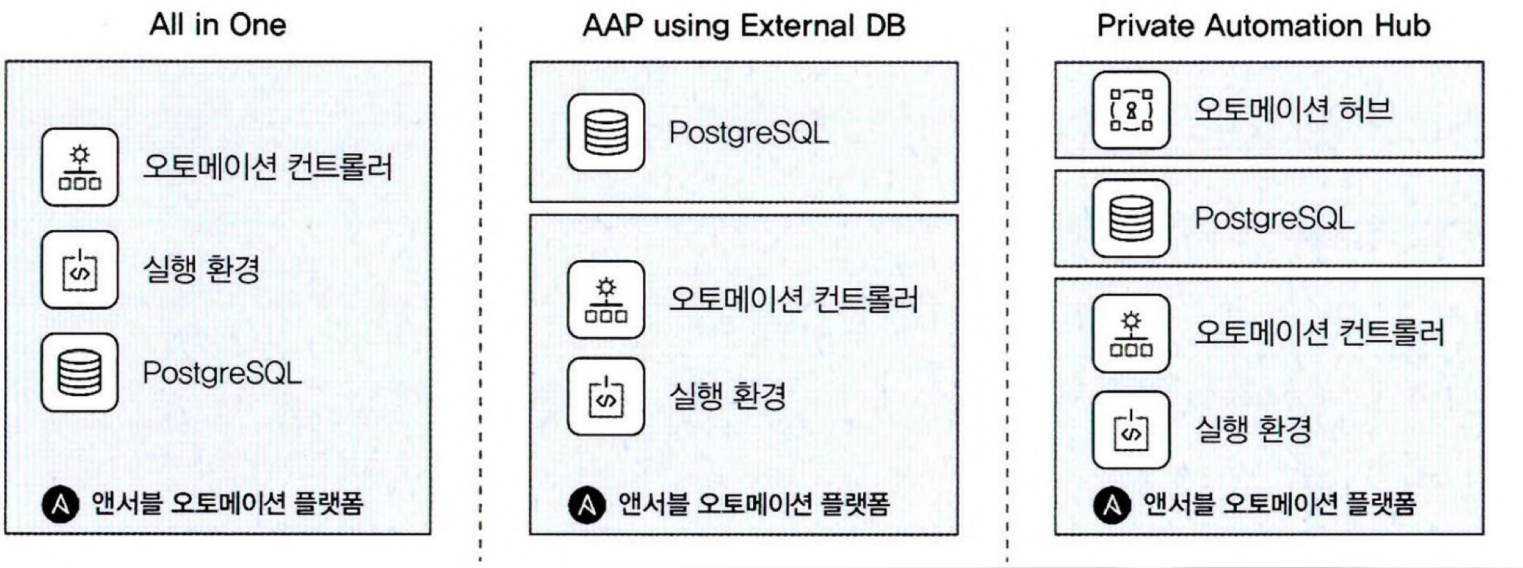


Ansible

- Ansible Architecture
  - ✓ Ansible Automation Platform(AAP)
    - 앤서블 플레이북은 사용자가 외부에서 작성한 내용을 앤서블 관리 웹 이를 통해 가져올 수 있음
    - 호스트, 애플리케이션 및 서비스, 퍼블릭 및 프라이빗 클라우드, 네트워킹과 같은 다양한 환경에 접속하여 플레이북에 작성된 작업을 수행
    - 레드햇 앤서블 오토메이션 플랫폼은 다음과 같이 크게 3가지 필수 컴포넌트와 1가지 옵셔널 컴포넌트로 구성
      - 오토메이션 컨트롤러: 앤서블 타워로 불리던 앤서블 관리 웹 UI
      - 실행 환경: 앤서블 모듈과 플러그인이 존재하는 컨테이너 기반의 실행 환경
      - PostgreSQL: 인벤토리, 인증 정보, 실행 환경 등의 메타데이터를 관리하는 CMDB
      - 오토메이션 허브(옵션): 레드햇에서 기술지원을 받을 수 있는 컬렉션 제공 서비스

Ansible

- ❖ Ansible Architecture
  - ✓ Ansible Automation Platform(AAP)
    - Architecture



Ansible

- ❖ Ansible Architecture
  - ✓ Ansible Automation Platform(AAP)
    - ● Architecture
      - ◆ All in One: 가장 기본적인 구성 아키텍처로 오토메이션 컨트롤러, 실행 환경, PostgreSQL이 모두 한 노드에 구성
      - ◆ Ansible Automation Platform Using External DB: PostgreSQL이 별도의 노드로 오토메이션 컨트롤러와 실행 환경이 같은 노드로 구성되며 오토메이션 컨트롤러는 별도로 구성된 PostgreSQL과 연동되어 앤서블 관리 웹 UI 서비스를 제공
      - ◆ Private Automation Hub: AAP using External DB와 같은 아키텍처에 오토메이션 허브를 내부에서 사용할 수 있도록 별도의 노드로 구성하는데 주로 인터넷이 되지 않는 경우에 구성하는 방식

Ansible

- ❖ 설치
  - ✓ 리눅스 설정
    - 컴퓨터의 이름 변경: sudo hostnamectl set-hostname [새로운-호스트이름]
    - IP 변경
      - ◆ 인터페이스 확인: ip addr
      - ◆ IP 설정: sudo vi /etc/netplan/01-netcfg.yaml

```
network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s8:
      dhcp4: no
      addresses:
        - 10.0.2.15/24
      routes:
        - to: default
          via: 10.0.2.1
      nameservers:
        addresses: [8.8.8.8, 8.8.4.4]
```

Ansible

- 설치
  - ✓ ansible 제어 노드로 사용할 컴퓨터에 ansible 패키지 설치
    - CentOS
      - dnf install epel-release
      - dnf install ansible
    - Ubuntu
      - sudo apt update
      - sudo add-apt-repository --yes --update ppa:ansible/ansible
      - sudo apt install -y ansible

Ansible

- ❖ 설치
  - ✓ ansible 제어 노드로 사용할 컴퓨터에 ansible 패키지 설치
    - ● 버전 확인: ansible --version

```
ansible [core 2.16.3]
  config file = None
  configured module search path = ['/home/adam/.ansible/plugins/modules',
    '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location =
    /home/adam/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.12.3 (main, Jan 22 2026, 20:57:42) [GCC 13.3.0]
    (/usr/bin/python3)
  jinja version = 3.1.2
  libyaml = True
```

Ansible

- ❖ 설치
  - ✓ ansible 제어 노드로 사용할 컴퓨터에 ansible 패키지 설치
    - ● 동작 확인: ansible localhost -m pingansible [core 2.16.3]

```
[WARNING]: No inventory was parsed, only implicit localhost is available
localhost | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
```

Ansible

- SSH 접속 설정
  - ✓ 관리 노드에 비밀번호 입력없이 SSH 접속 가능하도록 설정
    - 유저 생성 및 설정
      - 신규 유저 생성: sudo adduser [유저명]
      - 유저 비밀번호 생성: sudo passwd [유저명]
      - 유저 권한 변경
        - sudo chmod u+w /etc/sudoers
        - sudo vi /etc/sudoers: 파일 하단에 코드 추가: 유저명 ALL=(ALL:ALL) NOPASSWD:ALL
    - ssh 재시작: sudo service ssh(sshd) restart
    - 외부에서 접속: ssh [유저명]@IP
    - 비밀번호 없이 접속
      - 접속하고자 하는 컴퓨터에서 ssh-keygen 수행
      - ssh-copy-id [원격서버계정ID]@[원격서버IP]

Ansible

- SSH 접속 설정
  - ✓ EC2의 경우
    - ssh -i pem파일경로 유저명@접속할IP
    - EC2로 파일 복사: sudo scp -i pem파일경로 전송할파일경로 계정@전송할EC2퍼블릭 DNS:복사될경로

Ansible

- ❖ 설치
  - ✓ Inventory(관리할 컴퓨터) 설정
    - ● IP를 이용한 설정
      - ◆ inventory.ini 파일을 만들고 관리할 컴퓨터를 설정

```
[itstudy]
10.0.2.200
10.0.2.201
10.0.2.202
```

Ansible

- ❖ 설치
  - ✓ Inventory(관리할 컴퓨터) 설정
    - ● IP를 이용한 설정
      - ◆ 확인: ansible-inventory -i inventory.ini --list

```
{
    "_meta": {
        "hostvars": {}
    },
    "all": {
        "children": [
            "ungrouped",
            "itstudy"
        ]
    },
    "itstudy": {
        "hosts": [
            "10.0.2.200",
            "10.0.2.201",
            "10.0.2.202"
        ]
    }
}
```

Ansible

- ❖ 설치
  - ✓ Inventory(관리할 컴퓨터) 설정
    - ● IP를 이용한 설정
      - ◆ 확인: ansible all -i inventory.ini -m ping

```
10.0.2.200 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
10.0.2.201 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

Ansible

- ❖ 설치
  - ✓ Inventory(관리할 컴퓨터) 설정
    - ● host를 이용한 설정
      - ◆ 자동화 대상 호스트 등록: /etc/hosts

        127.0.1.1 controlnode

        10.0.2.200 master
        10.0.2.201 worker1
        10.0.2.202 worker2
      - ◆ inventory.ini 파일에 관리할 컴퓨터를 설정

        [itstudy]
        master
        worker1
        worker2

Ansible

- ❖ 설치
  - ✓ Inventory(관리할 컴퓨터) 설정
    - ● host를 이용한 설정
      - ◆ 확인: ANSIBLE_HOST_KEY_CHECKING=False ansible all -i inventory.ini -m ping

```
master | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
worker1 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

Ansible

- ❖ 설치
  - ✓ Inventory(관리할 컴퓨터) 설정
    - ● EC2의 경우에는 아래처럼 설정
      - ◆ [컴퓨터이름] ansible_host=[IP] ansible_user=[계정₩ ansible_ssh_private_key_file=[pem 파일 경로]
      - ◆ Permission 에러가 발생하는 경우 chmod 400 [pem 파일 경로]

```
[itstudy]
worker1 ansible_host=172.31.6.9 ansible_user=ubuntu
ansible_ssh_private_key_file=/home/ubuntu/itstudy.pem
worker2 ansible_host=172.31.2.171 ansible_user=ubuntu
ansible_ssh_private_key_file=/home/ubuntu/itstudy.pem
```

Ansible

- 설치
  - ✓ Inventory(관리할 컴퓨터) 설정
    - 여러 호스트 그룹 설정

```
[webservers]
web1.example.com
web2.example.com
192.0.2.42

[db-servers]
db01.example.com
db02.example.com
```

    - 여러 그룹에 설정 가능

```
[east-datacenter]
web1.example.com
db01.example.com

[west-datacenter]
web2.example.com
db02.example.com

[production]
web1.example.com
web2.example.com
db01.example.com
db02.example.com
```

Ansible

- 설치
  - ✓ Inventory(관리할 컴퓨터) 설정
    - 중첩 그룹 설정 가능

```
[webservers]
web1.example.com
web2.example.com

[db-servers]
db01.example.com
db02.example.com

[datacenter:children]
webservers
dbservers
```

Ansible

- ❖ 설치
  - ✓ Inventory(관리할 컴퓨터) 설정
    - ● 범위 설정 가능

```
[webservers]
web[1:2].example.com

[db-servers]
db[01:02].example.com

# IP 범위 설정 - 192.168.4.0 ~ 192.168.4.255 사이의 IP 범위를 표현
[defaults]
192.168.4.[0:255]

# 호스트명 범위 설정 - com01.example.com ~ com20.example.com의 범위를 표현
[compute]
com[01:20].example.com

# DNS 범위 설정 - a.dns.example.com, b.dns.example.com, c.dns.example.com을 의미함
[dns]
[a:c].dns.example.com

# IPv6 범위 설정 - 2001:db08::a ~ 2001:db08::f 사이의 IPv6 범위를 표현
[ipv6]
2001:db8::[a:f]
```

Ansible

- 설치
  - ✓ Inventory(관리할 컴퓨터) 설정
    - 현재 프로젝트 디렉토리에 ansible.cfg 파일을 만들면 –i 옵션을 사용하지 않아도 ansible.cfg 설정 파일에 정의된 인벤토리의 호스트 정보를 확인할 수 있음
      - ansible.cfg 파일을 생성하고 작성

        [defaults]

        inventory = ./inventory.ini
      - ANSIBLE_HOST_KEY_CHECKING=False ansible all -m ping

        master | SUCCESS => {

        "ansible_facts": {

        "discovered_interpreter_python": "/usr/bin/python3"

        },

        "changed": false,

        "ping": "pong"

        }

Ansible

- 기본 명령어
  - ✓ 실행 옵션
    - -i (--inventory-file): 적용 될 호스트들에 대한 파일 경로(생략하면 /etc/ansible/hosts)
    - -m (--module-name): 사용할 모듈 선택
    - -k (--ask-pass): 관리자 암호 요청
    - -K (--ask-become-pass): 관리자 권한 상승
    - --list-hosts: 적용되는 호스트 목록
  - ✓ Ansible은 동일한 명령을 여러 번 수행하면 한 번만 수행

Ansible

- ansible module: https://docs.ansible.com/ansible/2.9/modules/list_of_all_modules.html#
  - ✓ 메모리 확인

    ansible all -m shell -a "free -h"

```
worker1 | CHANGED | rc=0 >>
              total        used        free      shared  buff/cache   available
Mem:           957Mi       343Mi       407Mi       892Ki       358Mi       614Mi
Swap:            0B          0B          0B
worker2 | CHANGED | rc=0 >>
              total        used        free      shared  buff/cache   available
Mem:           957Mi       328Mi       422Mi       880Ki       358Mi       628Mi
Swap:            0B          0B          0B
ubuntu@ip-172-31-0-99:~$
```

Ansible

- ansible module: https://docs.ansible.com/ansible/2.9/modules/list_of_all_modules.html#
  - ✓ 파일 복사

    touch test.txt

    ansible all -m copy -a "src=./test.txt dest=/tmp"

```
ubuntu@ip-172-31-6-9:~$ ls /tmp
snap-private-tmp
systemd-private-8f26a814f0d84183beb22b19afa81870-ModemManager.service-2fvtPi
systemd-private-8f26a814f0d84183beb22b19afa81870-chrony.service-SHqC8M
systemd-private-8f26a814f0d84183beb22b19afa81870-fwupd.service-a60Pku
systemd-private-8f26a814f0d84183beb22b19afa81870-polkit.service-WtymjK
systemd-private-8f26a814f0d84183beb22b19afa81870-systemd-logind.service-xrKFBq
systemd-private-8f26a814f0d84183beb22b19afa81870-systemd-resolved.service-WMaHdf
test.txt
```

Ansible

- ansible module: https://docs.ansible.com/ansible/2.9/modules/list_of_all_modules.html#
  - ✓ nginx 설치

    ansible all -m apt -a "name=nginx state=present" –b

```
ubuntu@ip-172-31-6-9:~$ apt list --installed | grep nginx

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

nginx-common/noble-updates,noble-security,now 1.24.0-2ubuntu7.1 all [installed,automatic]
nginx/noble-updates,noble-security,now 1.24.0-2ubuntu7.1 amd64 [installed]
```

  - ✓ nginx 삭제

    ansible all -m apt -a "name=nginx* state=absent purge=yes autoremove=yes" -b