Infrastructure_Ansible

Container

- Ansible을 이용한 도커 설치
  - ✓ Playbook 생성 – install_docker.yaml

```
---
- hosts: localhost
  become: yes
  tasks:
    - name: 1. GPG 키 디렉토리 생성
      ansible.builtin.file:
        path: /etc/apt/keyrings
        state: directory
        mode: '0755'

    - name: 2. 도커 GPG 키 직접 다운로드
      ansible.builtin.get_url:
        url: https://download.docker.com/linux/ubuntu/gpg
        dest: /etc/apt/keyrings/docker.asc
        mode: '0644'
        force: yes
```

Container

- Ansible을 이용한 도커 설치
  - ✓ Playbook 생성 – install_docker.yaml

```
- name: 3. 저장소 강제 주입 (Ubuntu 24.04 noble 전용)
    ansible.builtin.copy:
      dest: /etc/apt/sources.list.d/docker.list
      content: "deb [arch=amd64 signed-by=/etc/apt/keyrings/docker.asc]
https://download.docker.com/linux/ubuntu noble stable"
      mode: '0644'

  - name: 4. 시스템 패키지 목록 강제 업데이트
    ansible.builtin.apt:
      update_cache: yes

  - name: 5. 도커 엔진 설치 (버전 무관 강제 설치)
    ansible.builtin.apt:
      name:
        - docker-ce
        - docker-ce-cli
        - containerd.io
      state: present
      update_cache: yes
```

Container

- Ansible을 이용한 도커 설치
  - ✓ Playbook 생성 – install_docker.yaml

```
- name: 6. 도커 서비스 시작
  ansible.builtin.service:
    name: docker
    state: started
    enabled: yes
```

Container

- Ansible을 이용한 도커 설치
  - ✓ Playbook 실행: ansible-playbook install_docker.yml -K

```
BECOME password:

PLAY [localhost] ***************************************************************

TASK [Gathering Facts] *********************************************************
ok: [localhost]
TASK [필수 패키지 설치] ********************************************************
ok: [localhost]
TASK [GPG 키 전용 디렉토리 생성] ***********************************************
ok: [localhost]
TASK [도커 GPG 키 다운로드 (강제 덮어쓰기)]
*******************************************
changed: [localhost]
TASK [도커 저장소 설정 (정확한 경로 지정)] *************************************
changed: [localhost]
TASK [도커 설치] ***************************************************************
ok: [localhost]
TASK [Ensure Docker is started] ************************************************
ok: [localhost]
PLAY RECAP *********************************************************************
localhost                  : ok=7    changed=2    unreachable=0    failed=0    skipped=0
rescued=0    ignored=0
```

Container

❖ Ansible을 이용한 도커 설치

✓ 확인

- sudo docker --version

  Docker version 29.3.1, build c2be9cc

- sudo docker ps

  CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

Container

- Ansible을 이용한 도커 설치
  - ✓ 도커 컨테이너 배포: deploy-docker.yml

```
- name: 도커 컨테이너 배포
  hosts: localhost          # 대상 호스트 (보통은 remote_server 그룹)
  become: yes               # sudo 권한 필요
  vars:
    container_name: my-web-server
    image_name: nginx:latest
    host_port: 8080         # 호스트에서 접속할 포트
    container_port: 80      # 컨테이너 내부 포트

  tasks:
    - name: 1. 파이썬 도커 라이브러리 설치
      ansible.builtin.apt:
        name: python3-docker
        state: present
        update_cache: yes

    - name: 2. 도커 이미지 Pull
      community.general.docker_image:
        name: "{{ image_name }}"
        source: pull
```

Container

- Ansible을 이용한 도커 설치
  - ✓ 도커 컨테이너 배포: deploy-docker.yml

```
- name: 3. 도커 컨테이너 생성 및 실행
  community.general.docker_container:
    name: "{{ container_name }}"
    image: "{{ image_name }}"
    state: started
    restart_policy: always
    published_ports:
      - "{{ host_port }}:{{ container_port }}"
    env:
      MY_ENV_VAR: "hello-ansible"

- name: 4. 배포 결과 확인
  ansible.builtin.debug:
    msg: "컨테이너 {{ container_name }}가 {{ host_port }}번 포트에서 실행 중입니다."
```

Container

- Ansible을 이용한 도커 설치
  - ✓ 도커 컨테이너 배포: ansible-playbook deploy-docker.yml -K

```
PLAY [도커 컨테이너 배포] ****************************************************

TASK [Gathering Facts] ********************************************************
ok: [localhost]

TASK [1. 파이썬 도커 라이브러리 설치] ****************************************
ok: [localhost]

TASK [2. 도커 이미지 Pull] ***************************************************
changed: [localhost]

TASK [3. 도커 컨테이너 생성 및 실행] *****************************************
changed: [localhost]

TASK [4. 배포 결과 확인] *****************************************************
ok: [localhost] => {
    "msg": "컨테이너 my-web-server가 8080번 포트에서 실행 중입니다."
}

PLAY RECAP ********************************************************************
localhost                  : ok=5    changed=2    unreachable=0    failed=0    skipped=0
rescued=0    ignored=0
```

Container

- Ansible을 이용한 도커 설치
  - ✓ 도커 컨테이너 배포: sudo docker ps

```
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS
PORTS                    NAMES
ec4165c4ccf7   nginx:latest   "/docker-entrypoint.…"   18 minutes ago   Up 18 minutes
0.0.0.0:8080->80/tcp     my-web-server
```

Container

- Ansible을 이용한 도커 설치
  - ✓ 도커 컨테이너 삭제: remove-docker.yml

```
---
- name: 도커 컨테이너 삭제
  hosts: localhost
  become: yes
  vars:
    container_name: my-web-server  # 삭제할 컨테이너 이름

  tasks:
    - name: 1. 특정 이름의 컨테이너를 찾아 삭제하기
      community.general.docker_container:
        name: "{{ container_name }}"
        state: absent
        # 만약 컨테이너와 연결된 볼륨(데이터)까지 지우고 싶다면 아래 주석 해제
        # keep_volumes: no

    - name: 2. 결과 확인
      ansible.builtin.debug:
        msg: "컨테이너 {{ container_name }}가 성공적으로 제거되었습니다."
```

Container

- Ansible을 이용한 도커 설치
  - ✓ 도커 컨테이너 삭제: ansible-playbook remove-docker.yml -K

```
PLAY [도커 컨테이너 삭제]
**********************************************************************************

TASK [Gathering Facts]
**********************************************************************************
ok: [localhost]

TASK [1. 특정 이름의 컨테이너를 찾아 삭제하기]
*****************************************************************
changed: [localhost]

TASK [2. 결과 확인]
*************************************************************************************************
ok: [localhost] => {
    "msg": "컨테이너 my-web-server가 성공적으로 제거되었습니다."
}

PLAY RECAP
*************************************************************************************************
******
localhost                  : ok=3    changed=1    unreachable=0    failed=0    skipped=0
rescued=0    ignored=0
```

Container

- Ansible을 이용한 도커 설치
  - ✓ 도커 컨테이너 삭제: sudo docker ps -a

    CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

Container

- Ansible을 이용한 도커 설치
  - ✓ 도커 삭제: uninstall-docker.yml

```
---
- name: 도커 엔진 및 관련 데이터 완전 삭제
  hosts: localhost
  become: yes
  tasks:
    - name: 1. 도커 패키지 삭제 (Purge)
      ansible.builtin.apt:
        name:
          - docker-ce
          - docker-ce-cli
          - containerd.io
          - docker-buildx-plugin
          - docker-compose-plugin
          - docker.io
          - docker-doc
          - docker-compose
        state: absent
        purge: yes              # 설정 파일까지 모두 삭제
        update_cache: yes
```

Container

- Ansible을 이용한 도커 설치
  - ✓ 도커 삭제: uninstall-docker.yml

```
- name: 2. 사용되지 않는 의존성 패키지 정리 (Autoremove)
  ansible.builtin.apt:
    autoremove: yes
- name: 3. 도커 데이터 디렉토리 삭제 (이미지, 컨테이너, 볼륨)
  ansible.builtin.file:
    path: "{{ item }}"
    state: absent
  loop:
    - /var/lib/docker
    - /var/lib/containerd
    - /etc/docker
    - /etc/apt/keyrings/docker.gpg
    - /etc/apt/sources.list.d/docker.list

- name: 4. 도커 그룹 삭제 (선택 사항)
  ansible.builtin.group:
    name: docker
    state: absent

- name: 5. 삭제 확인 결과 출력
  ansible.builtin.debug:
    msg: "도커 엔진과 모든 데이터가 시스템에서 삭제되었습니다."
```

Container

- Ansible을 이용한 도커 설치
  - ✓ 도커 삭제: ansible-playbook uninstall-docker.yml -K

```
PLAY [도커 엔진 및 관련 데이터 완전 삭제]
**********************************************************************

TASK [Gathering Facts]
**********************************************************************************************
ok: [localhost]

TASK [1. 도커 패키지 삭제 (Purge)]
**************************************************************************
changed: [localhost]

TASK [2. 사용되지 않는 의존성 패키지 정리 (Autoremove)]
*****************************************************
changed: [localhost]

TASK [3. 도커 데이터 디렉토리 삭제 (이미지, 컨테이너, 볼륨)]
***********************************************
changed: [localhost] => (item=/var/lib/docker)
changed: [localhost] => (item=/var/lib/containerd)
ok: [localhost] => (item=/etc/docker)
changed: [localhost] => (item=/etc/apt/keyrings/docker.gpg)
changed: [localhost] => (item=/etc/apt/sources.list.d/docker.list)
```

Container

- Ansible을 이용한 도커 설치
  - ✓ 도커 삭제: ansible-playbook uninstall-docker.yml -K

```
TASK [4. 도커 그룹 삭제 (선택 사항)]
**********************************************************************
changed: [localhost]

TASK [5. 삭제 확인 결과 출력]
**********************************************************************************
ok: [localhost] => {
    "msg": "도커 엔진과 모든 데이터가 시스템에서 삭제되었습니다."
}

PLAY RECAP
**********************************************************************************************
******
localhost                  : ok=6    changed=4    unreachable=0    failed=0    skipped=0
rescued=0    ignored=0
```

Container

- 클러스터 생성
  - ✓ Inventory 설정: vi inventory.ini

```
[itstudy]
master
worker1
worker2

[worker_nodes]
10.0.2.201 # worker01
10.0.2.202 # worker02

[master_nodes]
10.0.2.200 # master01

[nodes]
10.0.2.201
10.0.2.202
10.0.2.200
```

Container

- 클러스터 생성
  - ✓ Inventory 구성 요소 내 통신 확인: ansible all -m ping

```
10.0.2.202 | 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"
}
10.0.2.200 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

Container

- 클러스터 생성
  - ✓ Master, Worker Nodes 스왑 메모리 해제: swap-off-playbook.yaml

```
- name: Disable all nodes swap memory
  hosts: nodes
  become: true
  tasks:
    - name: Gather facts
      ansible.builtin.setup:

    - name: Disable swap
      ansible.builtin.command: swapoff -a
      when: ansible_swaptotal_mb > 0

    - name: Ensure swap is disabled in /etc/fstab
      ansible.builtin.replace:
        path: /etc/fstab
        regexp: '^(.+₩s+swap₩s+.+)$'
        replace: '# ₩1'
```

Container

- 클러스터 생성
  - ✓ Master, Worker Nodes 스왑 메모리 해제: ansible-playbook swap-off-playbook.yaml

```
PLAY [Disable all nodes swap memory] ********************************************************************

TASK [Gathering Facts] **********************************************************************************
ok: [10.0.2.202]
ok: [10.0.2.201]
ok: [10.0.2.200]

TASK [Gather facts] *************************************************************************************
ok: [10.0.2.201]
ok: [10.0.2.202]
ok: [10.0.2.200]

TASK [Disable swap] *************************************************************************************
changed: [10.0.2.202]
changed: [10.0.2.201]
changed: [10.0.2.200]

TASK [Ensure swap is disabled in /etc/fstab] ************************************************************
changed: [10.0.2.202]
changed: [10.0.2.201]
changed: [10.0.2.200]

PLAY RECAP **********************************************************************************************
10.0.2.200                 : ok=4    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
10.0.2.201                 : ok=4    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
10.0.2.202                 : ok=4    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Container

- 클러스터 생성
  - ✓ Network 구성: install-containerd-playbook.yaml

```
- name: Install containerd
  hosts: nodes
  become: true
  tasks:
    ## Step 2.1
    - name: Gather facts
      ansible.builtin.setup:

    - name: Load overlay module
      ansible.builtin.shell: modprobe overlay
      register: overlay_result

    - name: Load br_netfilter module
      ansible.builtin.shell: modprobe br_netfilter
      register: br_netfilter_result
```

Container

- 클러스터 생성
  - ✓ Network 구성: install-containerd-playbook.yaml

```
- name: Ensure sysctl params are set for Kubernetes
  ansible.builtin.copy:
    dest: /etc/sysctl.d/k8s.conf
    content: |
      net.bridge.bridge-nf-call-iptables  = 1
      net.bridge.bridge-nf-call-ip6tables = 1
      net.ipv4.ip_forward                 = 1

- name: Apply sysctl params without reboot
  ansible.builtin.shell: sysctl --system

- name: Verify br_netfilter module is loaded
  ansible.builtin.shell: lsmod | grep br_netfilter
  register: br_netfilter_check
  failed_when: br_netfilter_check.rc != 0

- name: Verify overlay module is loaded
  ansible.builtin.shell: lsmod | grep overlay
  register: overlay_check
  failed_when: overlay_check.rc != 0
```

Container

- 클러스터 생성
  - ✓ Network 구성: install-containerd-playbook.yaml

```
- name: Verify sysctl settings
  ansible.builtin.shell: |
    sysctl net.bridge.bridge-nf-call-iptables net.bridge.bridge-nf-call-ip6tables
net.ipv4.ip_forward
  register: sysctl_check
  failed_when:
    - sysctl_check.stdout.find('net.bridge.bridge-nf-call-iptables = 1') == -1
    - sysctl_check.stdout.find('net.bridge.bridge-nf-call-ip6tables = 1') == -1
    - sysctl_check.stdout.find('net.ipv4.ip_forward = 1') == -1
```

Container

- 클러스터 생성
  - ✓ Network 구성: ansible-playbook install-containerd-playbook.yaml

```
TASK [Load br_netfilter module] ************************************************************************************
changed: [10.0.2.201]
changed: [10.0.2.202]
changed: [10.0.2.200]

TASK [Ensure sysctl params are set for Kubernetes] *****************************************************************
changed: [10.0.2.201]
changed: [10.0.2.202]
changed: [10.0.2.200]

TASK [Apply sysctl params without reboot] **************************************************************************
changed: [10.0.2.201]
changed: [10.0.2.202]
changed: [10.0.2.200]

TASK [Verify br_netfilter module is loaded] ************************************************************************
changed: [10.0.2.201]
changed: [10.0.2.202]
changed: [10.0.2.200]

TASK [Verify overlay module is loaded] *****************************************************************************
changed: [10.0.2.201]
changed: [10.0.2.202]
changed: [10.0.2.200]

TASK [Verify sysctl settings] **************************************************************************************
changed: [10.0.2.201]
changed: [10.0.2.202]
changed: [10.0.2.200]

PLAY RECAP *********************************************************************************************************
10.0.2.200                 : ok=9    changed=7    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
10.0.2.201                 : ok=9    changed=7    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
10.0.2.202                 : ok=9    changed=7    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Container

- 클러스터 생성
  - ✓ CRI(Container Runtime Interface – Containerd) 설치: install-containerd-playbook.yaml 뒤에 추가

```
 ## Step 2.2
- name: GPG 키 폴더 생성
  ansible.builtin.file:
    path: /etc/apt/keyrings
    state: directory
    mode: '0755'

- name: 도커 GPG 키 다운로드 (타임아웃 설정 추가)
  ansible.builtin.get_url:
    url: https://download.docker.com/linux/ubuntu/gpg
    dest: /etc/apt/keyrings/docker.asc
    mode: '0644'
    timeout: 30  # 30초 안에 응답 없으면 재시도
```

Container

- 클러스터 생성
  - ✓ CRI(Container Runtime Interface – Containerd) 설치: install-containerd-playbook.yaml 뒤에 추가

```
- name: Set up the Docker repository
  ansible.builtin.shell: |
    mkdir -p /etc/apt/keyrings
    echo ₩
      "deb [arch=$(dpkg --print-architecture) signed-
by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu ₩
      $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | ₩
      sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
    sudo apt-get update
```

Container

- ❖ 클러스터 생성
  - ✓ CRI(Container Runtime Interface – Containerd) 설치: install-containerd-playbook.yaml 뒤에 추가

```
- name: Install containerd.io
  ansible.builtin.apt:
    name: containerd.io
    state: present
```

Container

- 클러스터 생성
  - ✓ CRI(Container Runtime Interface – Containerd) 설치: ansible-playbook install-containerd-playbook.yaml

```
TASK [Verify overlay module is loaded] *****************************************************************************
changed: [10.0.2.201]
changed: [10.0.2.202]
changed: [10.0.2.200]

TASK [Verify sysctl settings] **************************************************************************************
changed: [10.0.2.201]
changed: [10.0.2.202]
changed: [10.0.2.200]

TASK [Add Docker's official GPG key] *******************************************************************************
changed: [10.0.2.200]
changed: [10.0.2.202]
changed: [10.0.2.201]

TASK [Set up the Docker repository] ********************************************************************************
changed: [10.0.2.202]
changed: [10.0.2.200]
changed: [10.0.2.201]

TASK [Install containerd.io] ***************************************************************************************
changed: [10.0.2.200]
changed: [10.0.2.202]
changed: [10.0.2.201]

PLAY RECAP *********************************************************************************************************
10.0.2.200                 : ok=12   changed=9    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
10.0.2.201                 : ok=12   changed=9    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
10.0.2.202                 : ok=12   changed=9    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Container

- 클러스터 생성
  - ✓ systemd cgroup driver 구성: install-containerd-playbook.yaml 뒤에 추가

```
## Step 2.3
- name: Setup containerd config directory
  ansible.builtin.file:
    path: /etc/containerd
    state: directory
    owner: root
    group: root
    mode: "0755"

- name: Generate default containerd configuration
  ansible.builtin.command:
    cmd: "containerd config default"
  register: containerd_default_config

- name: Save default containerd configuration to config.toml
  ansible.builtin.copy:
    dest: /etc/containerd/config.toml
    content: ""
```

Container

- 클러스터 생성
  - ✓ systemd cgroup driver 구성: install-containerd-playbook.yaml 뒤에 추가

```
- name: Update sandbox_image in config.toml
  ansible.builtin.replace:
    path: /etc/containerd/config.toml
    regexp: 'sandbox_image = ".*"'
    replace: 'sandbox_image = "registry.k8s.io/pause:3.9"'

- name: Update SystemdCgroup in config.toml
  ansible.builtin.replace:
    path: /etc/containerd/config.toml
    regexp: 'SystemdCgroup = false'
    replace: 'SystemdCgroup = true'

- name: Restart containerd
  ansible.builtin.systemd:
    name: containerd
    state: restarted
```

Container

- 클러스터 생성
  - ✓ systemd cgroup driver 구성: install-containerd-playbook.yaml 뒤에 추가

```
- name: 쿠버네티스 저장소용 GPG 키 폴더 생성
  ansible.builtin.file:
    path: /etc/apt/keyrings
    state: directory
    mode: '0755'

- name: 쿠버네티스 공식 GPG 키 다운로드
  ansible.builtin.get_url:
    url: https://pkgs.k8s.io/core:/stable:/v1.29/deb/Release.key
    dest: /etc/apt/keyrings/kubernetes-apt-keyring.asc
    mode: '0644'

- name: 쿠버네티스 저장소 추가
  ansible.builtin.shell: |
    echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.asc]
https://pkgs.k8s.io/core:/stable:/v1.29/deb/ /' | tee >

- name: 쿠버네티스 CRI 도구(crictl) 설치
  ansible.builtin.apt:
    name: cri-tools
    state: present
    update_cache: yes
```

Container

- ❖ 클러스터 생성
  - ✓ systemd cgroup driver 구성: install-containerd-playbook.yaml 뒤에 추가

```
- name: crictl 설정 파일 생성 (containerd 사용 시)
  ansible.builtin.copy:
    dest: /etc/crictl.yaml
    content: |
      runtime-endpoint: unix:///run/containerd/containerd.sock
      image-endpoint: unix:///run/containerd/containerd.sock
      timeout: 10
      debug: false
- name: Verify crictl configuration
  command: crictl stats
  register: crictl_stats
  ignore_errors: yes

- name: Display crictl stats output
  debug:
    var: crictl_stats.stdout_lines
  when: crictl_stats is defined and crictl_stats.stdout_lines is defined
```

Container

- 클러스터 생성
  - ✓ systemd cgroup driver 구성: ansible-playbook install-containerd-playbook.yaml

```
TASK [Display crictl stats output] *********************************************************************************************************
ok: [10.0.2.201] => {
    "crictl_stats.stdout_lines": [
        "\u001b[2J\u001b[HCONTAINER           NAME                CPU %               MEM                 DISK                INOD
ES"
    ]
}
ok: [10.0.2.202] => {
    "crictl_stats.stdout_lines": [
        "\u001b[2J\u001b[HCONTAINER           NAME                CPU %               MEM                 DISK                INOD
ES"
    ]
}
ok: [10.0.2.200] => {
    "crictl_stats.stdout_lines": [
        "\u001b[2J\u001b[HCONTAINER           NAME                CPU %               MEM                 DISK                INOD
ES"
    ]
}

PLAY RECAP *********************************************************************************************************************************
10.0.2.200                 : ok=26   changed=11   unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
10.0.2.201                 : ok=26   changed=13   unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
10.0.2.202                 : ok=26   changed=11   unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Container

- 클러스터 생성
  - ✓ kubeadm, kubelet, kubectl 설치: kube-series-playbook.yaml

```
- name: Install kubeadm, kubectl and kubelet
  hosts: nodes
  become: true
  tasks:
    - name: Gather facts
      ansible.builtin.setup:

    - name: Update apt package index
      ansible.builtin.apt:
        update_cache: yes

    - name: Install packages needed for Kubernetes apt repository
      ansible.builtin.apt:
        name:
          - apt-transport-https
          - ca-certificates
          - curl
          - gpg
        state: present
```

Container

- ❖ 클러스터 생성
  - ✓ kubeadm, kubelet, kubectl 설치: kube-series-playbook.yaml

```
- name: Create directory for apt keyrings if it does not exist
  ansible.builtin.file:
    path: /etc/apt/keyrings
    state: directory
    mode: '0755'

- name: Download Kubernetes public signing key
  ansible.builtin.shell: |
    curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.29/deb/Release.key | gpg --dearmor
-o /etc/apt/keyrings/kubernetes-apt-keyring.gpg

- name: Add Kubernetes apt repository
  ansible.builtin.shell: |
    echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg]
https://pkgs.k8s.io/core:/stable:/v1.29/deb/ /' | tee /etc/apt/sources.list.d/kubernetes.list

- name: Update apt package index again
  ansible.builtin.apt:
    update_cache: yes
```

Container

- ❖ 클러스터 생성
  - ✓ kubeadm, kubelet, kubectl 설치: kube-series-playbook.yaml

```
- name: Install kubeadm, kubelet and kubectl
  ansible.builtin.apt:
    name:
      - kubeadm
      - kubelet
      - kubectl
    state: present

- name: Hold kubelet, kubeadm and kubectl packages
  ansible.builtin.shell: |
    sudo apt-mark hold kubelet kubeadm kubectl

- name: Enable and start kubelet service
  ansible.builtin.systemd:
    name: kubelet
    enabled: yes
    state: started
```

Container

- 클러스터 생성
  - ✓ kubeadm, kubelet, kubectl 설치: ansible-playbook kube-series-playbook.yaml

```
TASK [Update apt package index again] ******************************************************************************************
changed: [10.0.2.202]
changed: [10.0.2.200]

TASK [Install kubeadm, kubelet and kubectl] ************************************************************************************
changed: [10.0.2.200]
changed: [10.0.2.202]

TASK [Hold kubelet, kubeadm and kubectl packages] ******************************************************************************
changed: [10.0.2.202]
changed: [10.0.2.200]

TASK [Enable and start kubelet service] ****************************************************************************************
changed: [10.0.2.200]
changed: [10.0.2.202]

PLAY RECAP *********************************************************************************************************************
10.0.2.200                 : ok=11   changed=7    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
10.0.2.202                 : ok=11   changed=7    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Container

- ❖ 클러스터 생성
  - ✓ 클러스터 초기화: init-cluster-playbook.yaml

```
- name: Initialize Kubernetes Cluster
  hosts: master_nodes
  become: true
  tasks:
    - name: 1. Swap 끄기
      ansible.builtin.shell: |
        swapoff -a
        sed -i '/swap/s/^/#/' /etc/fstab
      changed_when: false # 실행할 때마다 'changed'가 뜨는 것을 방지

    - name: 2. Containerd 설정 최적화 (SystemdCgroup 활성화)
      ansible.builtin.shell: |
        mkdir -p /etc/containerd
        containerd config default > /etc/containerd/config.toml
        sed -i 's/SystemdCgroup = false/SystemdCgroup = true/g'
/etc/containerd/config.toml
      notify: Restart Containerd # 핸들러 이름과 일치해야 함
```

Container

- ❖ 클러스터 생성
  - ✓ 클러스터 초기화: init-cluster-playbook.yaml

```
- name: 3. Kubelet 활성화 및 재시작
  ansible.builtin.systemd: # systemctl 대신 systemd 모듈 사용
    name: kubelet
    state: restarted
    enabled: yes
    daemon_reload: yes # 설정 변경 사항 반영

- name: Reset Kubernetes cluster before init
  command: kubeadm reset -f
  register: reset_cluster
  ignore_errors: yes

- name: Clean up Kubernetes manifests directory
  file:
    path: /etc/kubernetes/manifests
    state: absent
```

Container

- 클러스터 생성
  - ✓ 클러스터 초기화: init-cluster-playbook.yaml

```
- name: Initialize the Kubernetes cluster
  command: kubeadm init --pod-network-cidr=10.244.0.0/16 --apiserver-advertise-
address=10.0.2.200 --cri-socket=unix:///run/cont>
  register: kubeadm_init_output
  ignore_errors: true

- name: Create .kube directory
  file:
    path: "/home/adam/.kube"
    state: directory
    owner: "adam"
    group: "adam"
    mode: '0755'
```

Container

- ❖ 클러스터 생성
  - ✓ 클러스터 초기화: init-cluster-playbook.yaml

```
- name: Copy admin.conf to user's kube config
  copy:
    src: /etc/kubernetes/admin.conf
    dest: /home/adam/.kube/config  # 경로 확인 필요
    remote_src: yes
    owner: "adam"  # 또는 실행하려는 특정 유저명 (예: 'ubuntu')
    group: "adam"
    mode: '0644'
  when: kubeadm_init_output.changed

- name: Generate join command
  shell: kubeadm token create --print-join-command
  register: join_command
  become: yes


- name: Save join command to file
  copy:
    content: ""
    dest: "/kubeadm_join_cmd.sh"
  when: kubeadm_init_output.changed
```

Container

- 클러스터 생성
  - ✓ 클러스터 초기화: init-cluster-playbook.yaml

```
    - name: Output join command to console
      debug:
        msg: "Woker 노드에서 실행할 명령어: {{ join_command.stdout }}"

  handlers: # notify가 호출하는 섹션
    - name: Restart Containerd
      ansible.builtin.systemd:
        name: containerd
        state: restarted
```

Container

- ❖ 클러스터 생성
  - ✓ 클러스터 초기화: ansible-playbook init-cluster-playbook.yaml

```
TASK [Gathering Facts] *********************************************************************************************************************
ok: [10.0.2.200]

TASK [1. Swap 끄기] ************************************************************************************************************************
ok: [10.0.2.200]

TASK [2. Containerd 설정 최적화 (SystemdCgroup 활성화)] ************************************************************************************
changed: [10.0.2.200]

TASK [3. Kubelet 활성화 및 재시작] *********************************************************************************************************
changed: [10.0.2.200]

TASK [Reset Kubernetes cluster before init] ************************************************************************************************
changed: [10.0.2.200]

TASK [Clean up Kubernetes manifests directory] *********************************************************************************************
changed: [10.0.2.200]

TASK [Initialize the Kubernetes cluster] ***************************************************************************************************
changed: [10.0.2.200]

TASK [Create .kube directory] ***************************************************************************************************************
ok: [10.0.2.200]

TASK [Copy admin.conf to user's kube config] ************************************************************************************************
changed: [10.0.2.200]

TASK [Generate join command] ****************************************************************************************************************
changed: [10.0.2.200]

TASK [Save join command to file] ************************************************************************************************************
ok: [10.0.2.200]

TASK [Output join command to console] *******************************************************************************************************
ok: [10.0.2.200] => {
    "msg": "Woker 노드에서 실행할 명령어: kubeadm join 10.0.2.200:6443 --token 7shp6x.ml9dl8py3nrl9tsx --discovery-token-ca-cert-has
h sha256:af108609dd54b02cae00a6f873a6f1175a072d620393857730c7590d0ec180a3 "
}
```

Container

- 클러스터 생성
  - ✓ 네트워크 플러그인 설치(Calico): install-calico-playbook.yaml

```
- name: Install Calico
  hosts: master_nodes
  become: true
  tasks:
    - name: Download Calico manifest
      get_url:
        url: https://docs.projectcalico.org/manifests/calico.yaml
        dest: /usr/local/bin/calico.yaml

    - name: Apply Calico Manifest
      shell: |
        kubectl apply -f /usr/local/bin/calico.yaml --
kubeconfig=/etc/kubernetes/admin.conf
      become: yes

    - name: Install Calico network plugin
      command: kubectl apply -f /usr/local/bin/calico.yaml
      environment:
        KUBECONFIG: /etc/kubernetes/admin.conf
      become: yes
```

Container

- 클러스터 생성
  - ✓ 네트워크 플러그인 설치(Calico): ansible-playbook install-calico-playbook.yaml

```
BECOME password:

PLAY [Install Calico] **********************************************************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************************************
ok: [10.0.2.200]

TASK [Download Calico manifest] ************************************************************************************************************************
ok: [10.0.2.200]

TASK [Apply Calico Manifest] ***************************************************************************************************************************
changed: [10.0.2.200]

PLAY RECAP *********************************************************************************************************************************************
10.0.2.200                 : ok=3    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Container

- 클러스터 생성
  - ✓ 워커노드 추가: join-worker-nodes-playbook.yaml

```
- name: Join Worker Nodes to Kubernetes Cluster
  hosts: worker_nodes
  become: true
  tasks:
    - name: Join the node to the cluster
      command: {join commands - 수정필요}
      register: join_command
```

Container

- 클러스터 생성
  - ✓ 워커노드 추가: ansible-playbook join-worker-nodes-playbook.yaml

```
PLAY [Join Worker Nodes to Kubernetes Cluster] *********************************************************************

TASK [Gathering Facts] *********************************************************************************************
ok: [10.0.2.202]

TASK [Join the node to the cluster] ********************************************************************************
changed: [10.0.2.202]

PLAY RECAP *********************************************************************************************************
10.0.2.202                 : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Container

- ❖ 클러스터 생성
  - ✓ 확인
    - kubectl get pods -A

```
NAMESPACE     NAME                                       READY   STATUS    RESTARTS
AGE
kube-system   calico-kube-controllers-658d97c59c-brkw4   1/1     Running   0
2m42s
kube-system   calico-node-7dwp6                          1/1     Running   0
38s
kube-system   calico-node-mq5lk                          1/1     Running   0
2m42s
kube-system   coredns-76f75df574-c5fzs                   1/1     Running   0
6m23s
kube-system   coredns-76f75df574-pjrhc                   1/1     Running   0
6m23s
kube-system   etcd-master                                1/1     Running   2
6m37s
kube-system   kube-apiserver-master                      1/1     Running   2
6m39s
kube-system   kube-controller-manager-master             1/1     Running   7
6m37s
kube-system   kube-proxy-4k67g                           1/1     Running   1 (29s ago)
38s
kube-system   kube-proxy-f889d                           1/1     Running   0
6m23s
kube-system   kube-scheduler-master                      1/1     Running   6
6m37s
```

Monitoring

- 팩트를 이용한 시스템 모니터링
  - ✓ 사용을 할 facts
    - 호스트 이름
    - 커널 버전
    - 네트워크 인터페이스 이름
    - 네트워크 인터페이스 IP 주소
    - 운영체제 버전
    - CPU 개수
    - 사용 가능한 메모리
    - 스토리지 장치의 크기 및 여유 공간
  - ✓ 추출한 내용은 /var/log/daily_check 디렉토리에 저장

Monitoring

- 팩트를 이용한 시스템 모니터링
  - ✓ 플레이북: monitoring_facts.yml

```
---

- hosts: all
  vars:
    log_directory: /var/log/daily_check

  tasks:
    - name: Print system info
      ansible.builtin.debug:
        msg:
        - "################ Start ####################"
        - "Date: {{ ansible_facts.date_time.date }} {{ ansible_facts.date_time.time }}"
        - "HostName: {{ ansible_facts.hostname }}"
        - "OS: {{ ansible_facts.distribution }}"
        - "OS Version: {{ ansible_facts.distribution_version }}"
        - "OS Kernel: {{ ansible_facts.kernel }}"
        - "CPU Cores: {{ ansible_facts.processor_vcpus }}"
        - "Memory: {{ ansible_facts.memory_mb.real }}"
        - "Interfaces: {{ ansible_facts.interfaces }}"
        - "IPv4: {{ ansible_facts.all_ipv4_addresses }}"
        - "Devices: {{ ansible_facts.mounts }}"
        - "################ End ######################"
```

Monitoring

- 팩트를 이용한 시스템 모니터링
  - ✓ 플레이북: monitoring_facts.yml

```
    register: result
    become: yes
  - name: Create log directory
    ansible.builtin.file:
      path: "{{ log_directory }}"
      state: directory
    become: yes
  - name: Print logs to log file
    ansible.builtin.shell: |
      echo "{{ item }}" >> "{{ log_directory }}"/system_info.logs
    loop: "{{ result.msg }}"
    become: yes
```

Monitoring

❖ 팩트를 이용한 시스템 모니터링

✓ 플레이북 문법 체크: ansible-playbook --syntax-check monitoring_facts.yml -K register: result

playbook: monitoring_facts.yml

✓ 플레이북 실행 체크: ansible-playbook --check monitoring_facts.yml -K

PLAY RECAP
****
****

master : ok=4 changed=1 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0

worker2 : ok=4 changed=1 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0

Monitoring

- 팩트를 이용한 시스템 모니터링
  - ✓ 플레이북 실행: ansible-playbook monitoring_facts.yml -K

```
PLAY RECAP
*********************************************************************************************************************************
master                     : ok=4    changed=1    unreachable=0    failed=0    skipped=0
rescued=0    ignored=0
worker2                    : ok=4    changed=1    unreachable=0    failed=0    skipped=0
rescued=0    ignored=0
```

  - ✓ 호스트 컴퓨터에서 확인: sudo cat /var/log/daily_check/system_info.logs

```
################ Start ####################
Date: 2026-04-06 23:51:07
HostName: worker2
OS: Ubuntu
OS Version: 24.04
OS Kernel: 6.8.0-107-generic
CPU Cores: 1
Memory: {'total': 1953, 'used': 1769, 'free': 184}
Interfaces: ['tunl0', 'enp0s8', 'lo']
IPv4: ['10.244.189.64', '10.0.2.202']
```

Monitoring

- CPU, Memory, Disk 사용율 모니터링
  - ✓ 작성
    - 변수를 위한 vars_packages.yml

```
---

log_directory: /home/adam/logs
packages:
  - dstat
  - sysstat
```

Monitoring

- ❖ CPU, Memory, Disk 사용율 모니터링
  - ✓ 작성
    - ● 설치를 위한 monitoring_system.yml

```
---

- hosts: all
  vars_files: vars_packages.yml
  become: yes

  tasks:
    # 1. 로그 디렉토리 생성 태스크를 가장 먼저 추가하세요
    - name: Create log directory
      ansible.builtin.file:
        path: "{{ log_directory }}"
        state: directory
        owner: adam  # 디렉토리 소유자를 adam으로 설정
        group: adam
        mode: '0755'
```

Monitoring

- CPU, Memory, Disk 사용율 모니터링
  - ✓ 작성
    - 설치를 위한 monitoring_system.yml

```
- name: Install packages on RedHat
  ansible.builtin.dnf:
    name: "{{ item }}"
    state: present
  loop: "{{ packages }}"
  when: ansible_facts.os_family == "RedHat"

- name: Install packages on Ubuntu
  ansible.builtin.apt:
    name: "{{ item }}"
    state: present
  loop: "{{ packages }}"
  when: ansible_facts.os_family == "Debian"
```

Monitoring

- ❖ CPU, Memory, Disk 사용율 모니터링
  - ✓ 작성
    - ● 설치를 위한 monitoring_system.yml

```
- name: Monitoring dstat
  ansible.builtin.shell: |
    {{ item }} >> {{ log_directory }}/dstat.log
  loop:
    - dstat 2 10
    - dstat -cmdlt -D vda 2 10

- name: Monitoring iostat
  ansible.builtin.shell: |
    {{ item }} >> {{ log_directory }}/iostat.log
  loop:
    - iostat
    - echo "================"
    - iostat -t -c -dp vda
    - echo "================"
```

Monitoring

- CPU, Memory, Disk 사용율 모니터링
  - ✓ 작성
    - 설치를 위한 monitoring_system.yml

```
- name: Monitoring vmstat
  ansible.builtin.shell: |
    {{ item }} >> {{ log_directory }}/vmstat.log
  loop:
    - vmstat
    - echo "================"
    - vmstat -dt
    - echo "================"
    - vmstat -D
    - echo "================"

- name: Monitoring df
  ansible.builtin.shell: |
    df -h >> {{ log_directory }}/df.log
```

Monitoring

- CPU, Memory, Disk 사용율 모니터링
  - ✓ 작성
    - 문법 체크: ansible-playbook --syntax-check monitoring_system.yml

      playbook: monitoring_system.yml
    - 실행 체크: ansible-playbook --check monitoring_system.yml

      PLAY RECAP

      ********** ********** **********

      10.0.2.200 : ok=3 changed=0 unreachable=0 failed=0 skipped=5 rescued=0 ignored=0

      10.0.2.202 : ok=3 changed=0 unreachable=0 failed=0 skipped=5 rescued=0 ignored=0

      master : ok=3 changed=0 unreachable=0 failed=0 skipped=5 rescued=0 ignored=0

      worker2 : ok=3 changed=0 unreachable=0 failed=0 skipped=5 rescued=0 ignored=0

Monitoring

- ❖ CPU, Memory, Disk 사용율 모니터링
  - ✓ 작성
    - ● 실행: ansible-playbook monitoring_system.yml -K

      playbook: monitoring_system.yml