AWX 17 설치방법

AWX 설치방법

“AWX는 Redhat Ansible Tower의 Community 버전이다.”

AWX는 ANSIBLE을 WEB 에서 실행가능하도록 해주는 OPENSOURCE 프로그램이다.

현재 최신버전은 23버전이고 해당 매뉴얼에서는 17 버전을 UBUNTU 20.04에 설치하는 방법을 다룬다.

17버전까지는 DOCKER-COMPOSE로 설치가 가능하지만 18버전 부터는 KUBERNETES에서만 설치가 가능하다.

AWX 설치방법

필수 컴포넌트 설치

#git 설치

#Python3 설치 및 기본 구성

#docker 설치

위의 3가지 프로그램은 UBUNTU 기본 설치로 진행한다.

#필수 패키지 설치

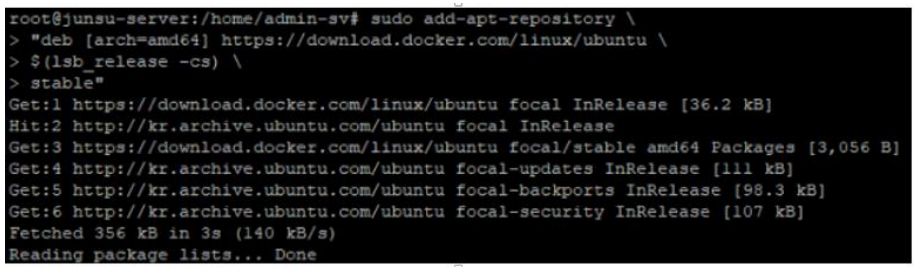
$ sudo apt-get install apt-transport-https ca-certificates curl gnupg-agent software-properties-common

#GPG Key 인증

$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

#docker 레포지토리 등록

$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"



AWX 설치방법

필수 컴포넌트 설치

#도커 설치

$sudo apt-get update && sudo apt-get install docker-ce docker-ce-cli containerd.io

#도커 버전확인

$docker -v

#시스템 재부팅시 도커가 시작되도록 설정

$sudo systemctl enable docker && service docker start

#docker-compose 설치

$sudo apt install docker-compose

AWX 설치방법

Ansible, AWX 설치

#Ansible 설치
$sudo add-apt-repository --yes --update ppa:ansible/ansible
$sudo apt update
$sudo apt install -y ansible
$ansible --version

#SSH-Key 생성
$ ssh-keygen

#AWX 설치

#git clone

$ git clone -b 17.1.0 https://github.com/ansible/awx.git

AWX 설치방법

AWX 설치

#inventory 변수 설정

```
$ cd awx/installer
$ vi inventory

###
…
admin_user=admin
admin_password=3w22kWelcom        # web에서 사용할 비밀번호 설정
…
project_data_dir #주석해제          # data를 저장할 디렉토리 설정

###
```

AWX 설치방법

AWX 설치

#AWX 플레이북 설치

$ ansible-playbook -i inventory install.yml -b

```
TASK [local_docker : Create Docker Compose Configuration] ***********************************************
changed: [localhost] => (item={'file': 'environment.sh', 'mode': '0600'})
changed: [localhost] => (item={'file': 'credentials.py', 'mode': '0600'})
changed: [localhost] => (item={'file': 'docker-compose.yml', 'mode': '0600'})
changed: [localhost] => (item={'file': 'nginx.conf', 'mode': '0600'})
changed: [localhost] => (item={'file': 'redis.conf', 'mode': '0664'})

TASK [local_docker : Render SECRET_KEY file] *************************************************************
changed: [localhost]

TASK [local_docker : Remove AWX containers before migrating postgres so that the old postgres container does not get used] ***
ok: [localhost]

TASK [local_docker : Run migrations in task container] ***************************************************
changed: [localhost]

TASK [local_docker : Start the containers] ***************************************************************
changed: [localhost]

TASK [local_docker : Update CA trust in awx_web container] ***********************************************
changed: [localhost]

TASK [local_docker : Update CA trust in awx_task container] **********************************************
changed: [localhost]

TASK [local_docker : Wait for launch script to create user] **********************************************
ok: [localhost]

TASK [local_docker : Create Preload data] ****************************************************************
changed: [localhost]

PLAY RECAP ***********************************************************************************************
localhost                  : ok=21   changed=12   unreachable=0    failed=0    skipped=73   rescued=0    ignored=1

ubuntu@ip-172-22-3-110:~/awx/installer$
```

AWX 설치방법

AWX 설치

#설치된 container 확인

$ sudo docker ps

```
ubuntu@ip-172-22-3-110:~/awx/installer$ sudo docker ps
CONTAINER ID   IMAGE                COMMAND                  CREATED              STATUS              PORTS
                NAMES
427icdbf79ef   ansible/awx:17.1.0   "/usr/bin/tini -- /u…"   About a minute ago   Up About a minute   8052/tcp
                awx_task
899bc00f20f1   ansible/awx:17.1.0   "/usr/bin/tini -- /b…"   4 minutes ago        Up About a minute   0.0.0.0:80->8052/tcp, :
:80->8052/tcp   awx_web
97f675ec7784   postgres:12          "docker-entrypoint.s…"   4 minutes ago        Up About a minute   5432/tcp
                awx_postgres
1d0e2cb511af   redis                "docker-entrypoint.s…"   4 minutes ago        Up About a minute   6379/tcp
                awx_redis
ubuntu@ip-172-22-3-110:~/awx/installer$
```

AWX 설치방법

AWX 설치

#샘플 yaml 파일 생성

설치시 설정한 디렉토리 (ex: /home/ubuntu/myplaybook) 에서 진행

$ cd /home/ubuntu/myplaybook

프로젝트 디렉토리 생성

$ mkdir test

샘플 yaml 파일 생성

$ vi debug.yml

AWX 설치방법

AWX 설치

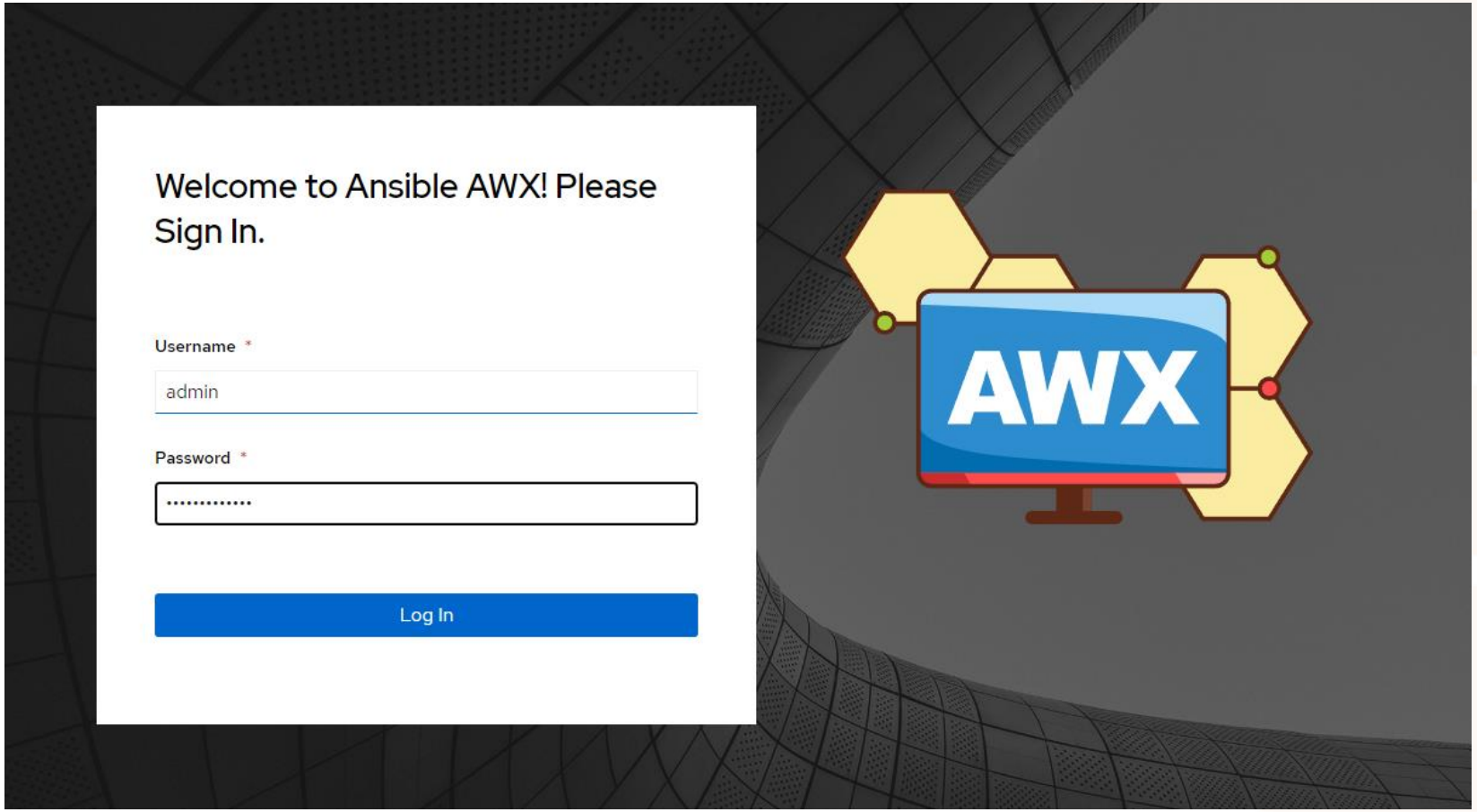
 # debug.yml
 # 해당 파일은 root 권한으로 /etc/hosts에 ip와 hostname을 추가한다.

```
###
---
- hosts: all
  become: yes
  tasks:
  - name : Add hostfile
    blockinfile:
          path: /etc/hosts
          block: |
             192.168.119.11 bm-inception
             192.168.119.13 bm-control
             192.168.119.15 bm-cluster-1
             192.168.119.17 bm-cluster-2
             192.168.119.19 bm-cluster-3
             192.168.119.21 bm-ceph-cluster-1
             192.168.119.23 bm-ceph-cluster-2
             192.168.119.25 bm-ceph-cluster-3
###
```

AWX 설치방법

AWX 설치 확인

#webpage 접속
#80포트 사용



AWX 사용방법

AWX 사용방법

AWX 테스트

AWX 를 테스트하기 위해 명령어를 실행한 TEST VM이 하나 더 필요하다.

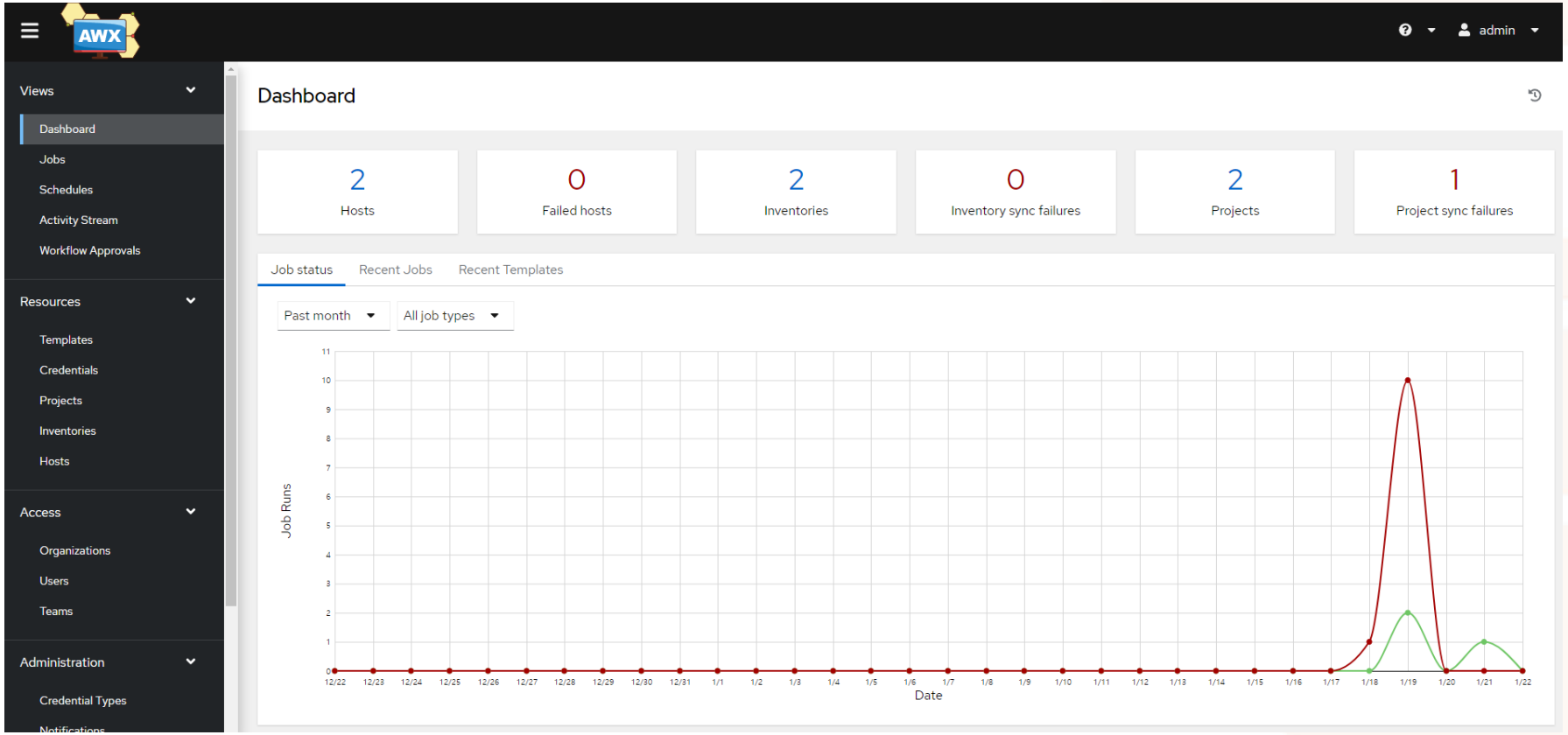
IP - 10.90.1.113
UBUNTU / PASSWORD로 생성하였다.

	Instance Name	Image Name	IP Address	Flavor
☐	awx-test-1	focal	10.90.1.113	m1.small
☐	awx-test	focal	10.90.1.195, 112.175.114.170	m1.medium

AWX 사용방법

AWX 대시보드

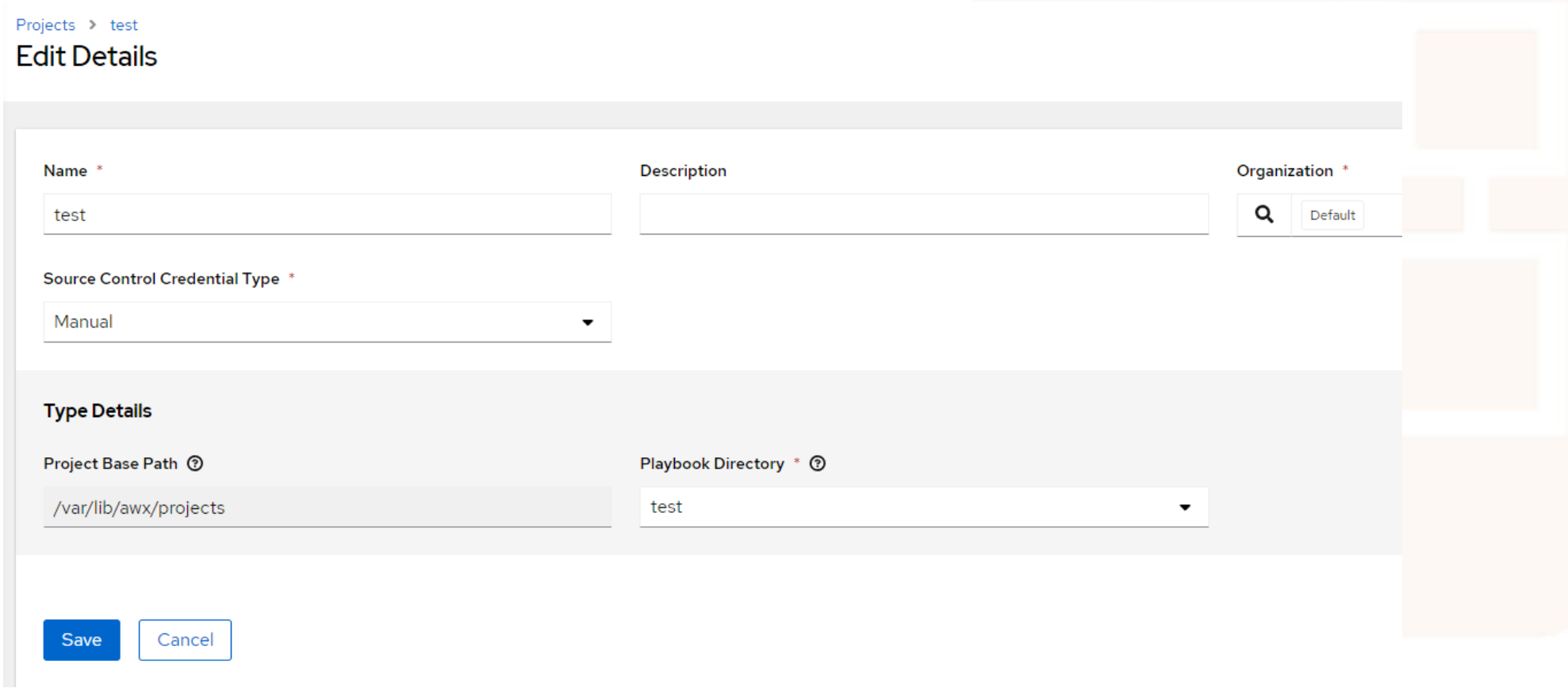
webpage 접속시 가장 먼저 보이는 대시보드 날짜 별 job 실행 결과와 현재 상태를 보여준다.



AWX 사용방법

AWX 테스트

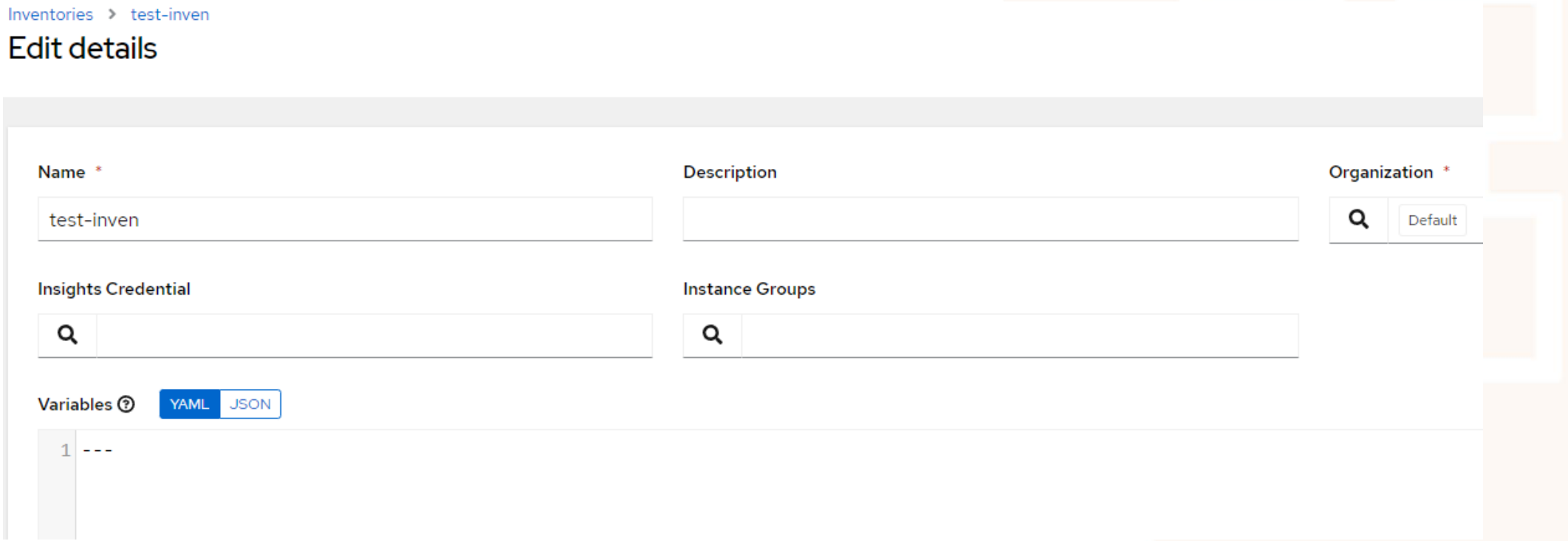
project 생성
아래와 같이 프로젝트를 생성한다. playbook directory에 자신이 만든 디렉토리 내에 프로젝트 디렉토리를 선택한다.



AWX 사용방법

AWX 테스트

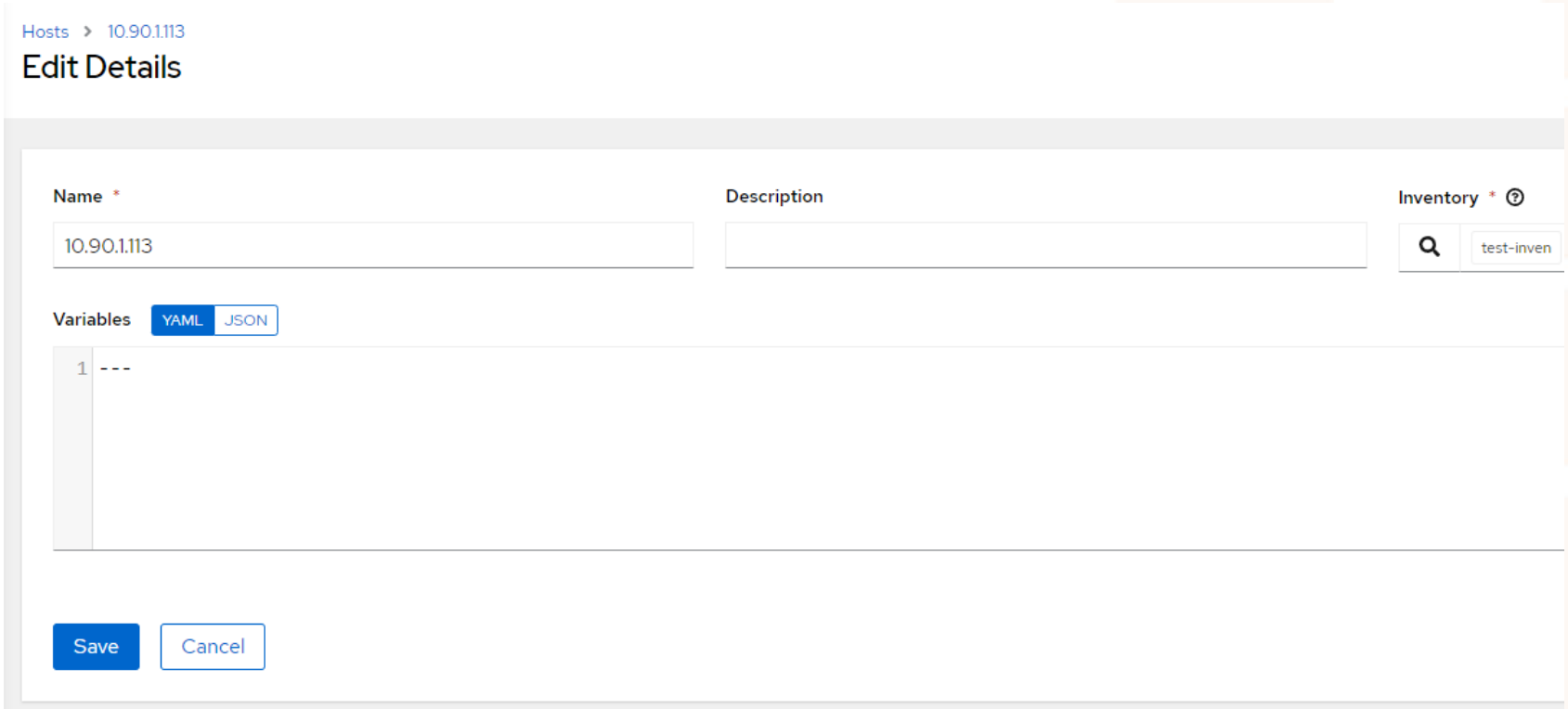
inventory 생성
inventory를 생성한다. 아무 옵션 없이 생성한 하면 된다.



AWX 사용방법

AWX 테스트

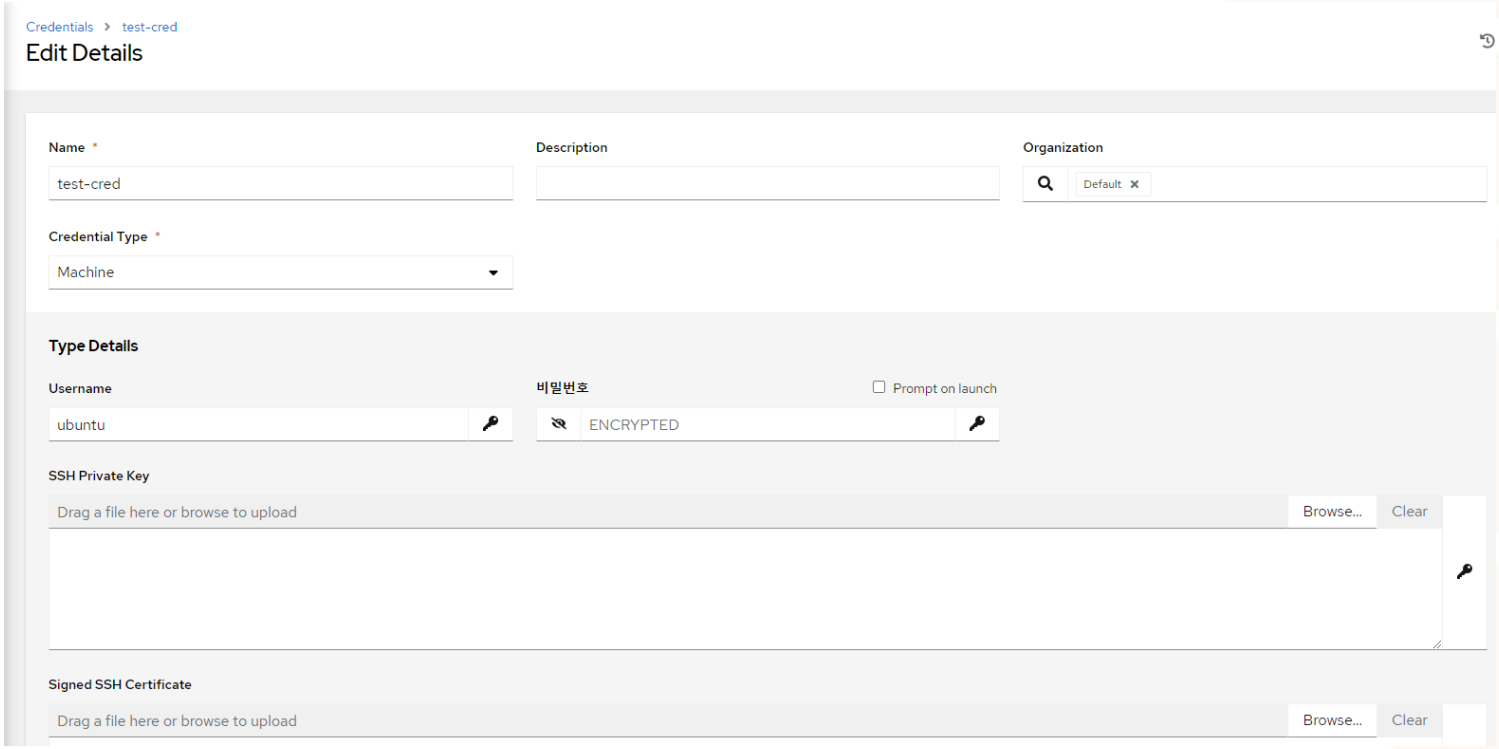
hosts 등록
이름에 ip를 입력하고, 앞에서 생성한 인벤토리를 선택한다.



AWX 사용방법

AWX 테스트

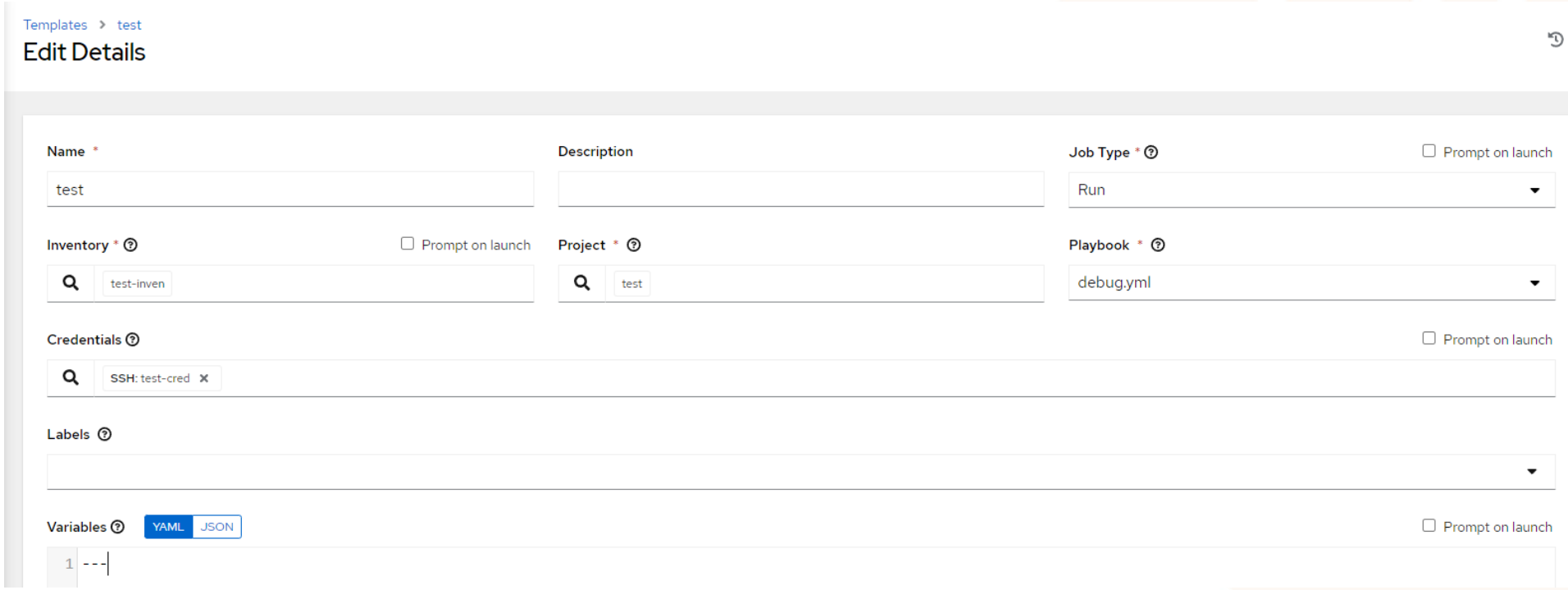
credential 생성
앞에 등록한 host에 맞는 credential을 생성한다.
해당 vm은 비밀번호로 생성하였기 때문에 manual - password로 생성



AWX 사용방법

AWX 테스트

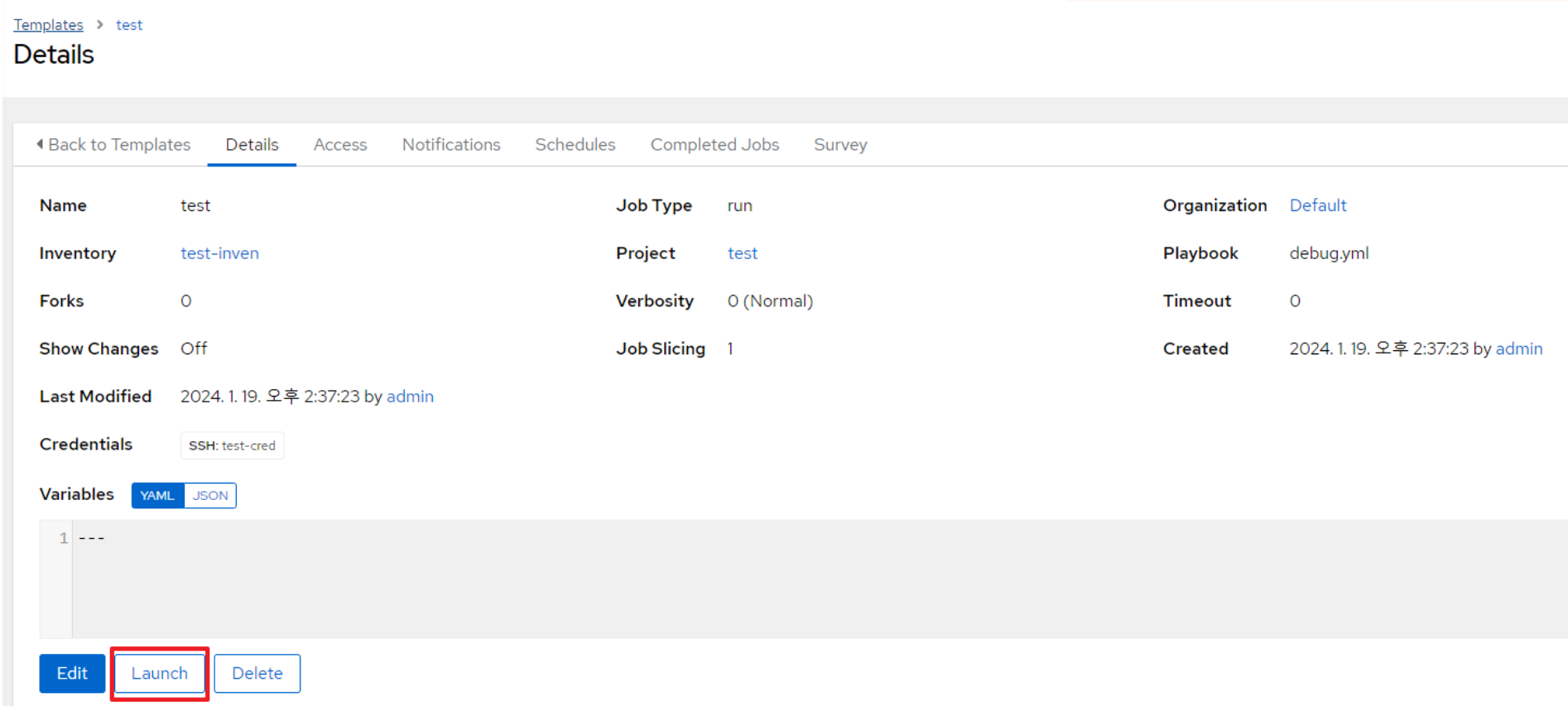
template 생성
기존에 생성한 poject, hosts, credential 정보를 사용한다.
앞에서 입력한 debug.yml을 사용하는 template를 생성한다.



AWX 사용방법

AWX 테스트

template 실행
template 의 launch 버튼을 클릭하면 해당 template를 실행한다.



AWX 사용방법

AWX 테스트

template 실행 결과
실행 후 output 화면에서 ansible 실행 결과를 확인할 수 있다.

AWX 사용방법

AWX 테스트

template 실행 결과
test vm에서 ansible로 /etc/hosts 파일이 변경된 것을 확인할 수 있다.

```
ubuntu@awx-test-1:~$ cat /etc/hosts
127.0.0.1 localhost

# The following lines are desirable for IPv6 capable hosts
::1 ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
ff02::3 ip6-allhosts
# BEGIN ANSIBLE MANAGED BLOCK
192.168.119.11 bm-inception
192.168.119.13 bm-control
192.168.119.15 bm-cluster-1
192.168.119.17 bm-cluster-2
192.168.119.19 bm-cluster-3
192.168.119.21 bm-ceph-cluster-1
192.168.119.23 bm-ceph-cluster-2
192.168.119.25 bm-ceph-cluster-3
# END ANSIBLE MANAGED BLOCK
```

AWX 23 설치방법

AWX 23 설치방법

설치방법

최신 23버전은 일반 docker에 설치가 불가능하며 kubernetes 위에만 설치가 가능하다.

만약 단독 node(vm)에 설치해야하는 경우 해당 node에 k3s를 설치하고

해당 k3s에 AWX의 operator를 생성하여 설치한다.

AWX 23 설치방법

최소 요구사항

- 우분투 22.04|20.04|18.04 LTS 서버
- **4vcpus**
- 최소 **16GB** RAM
- **20GB**의 디스크
- sudo 권한

AWX 23 설치방법

ubuntu 시스템 업데이트

$ sudo apt update

$ sudo apt -y upgrade [-f /var/run/reboot-required]

$ sudo reboot -f

AWX 23 설치방법

단일 노드 k3s Kubernetes 설치

- k3s 경량 도구를 사용하여 단일 노드 kubernetes를 배포합니다.
- **K3s**는 리소스가 제한된 환경에서 무인 환경의 프로덕션 워크로드를 위해 설계된 인증된 Kubernetes 배포판입니다.
- k3s의 좋은 점은 필요한 경우 나중에 단계에서 더 많은 작업자 노드를 추가할 수 있다는 것입니다.
- K3s는 systemd 또는 openrc 기반 시스템에 서비스로 설치하는 편리한 방법인 설치 스크립트를 제공합니다.

Ubuntu 시스템에 K3를 설치하려면 다음 명령을 실행합니다.

$ curl -sfL https://get.k3s.io | sudo bash -

AWX 23 설치방법

단일 노드 k3s Kubernetes 설치

/etc/rancher/k3s/k3s.yaml에 저장된 kubeconfig 파일은 Kubernetes 클러스터에 대한 액세스를 구성하는 데 사용됩니다.

전역 액세스를 위해 권한을 설정하고 KUBECONFIG를 설정할 수 있습니다.

```
$ sudo chmod 644 /etc/rancher/k3s/k3s.yaml

$ export KUBECONFIG=/etc/rancher/k3s/k3s.yaml

$ kubectl get pods --all-namespaces
```

클러스터 외부에 있는 머신에 ~/.kube/config로 /etc/rancher/k3s/k3s.yaml을 복사할 수도 있습니다.

```
$ mkdir ~/.kube
$ sudo cp /etc/rancher/k3s/k3s.yaml ~/.kube/config
```

AWX 23 설치방법

Kubernetes에 AWX Operator 배포

이 Kubernetes Operator는 Kubernetes 클러스터에 배포되어야 하며, 우리의 경우 K3s로 구동됩니다.
우리가 배포할 운영자는 모든 네임스페이스에서 하나 이상의 AWX 인스턴스를 관리할 수 있습니다.

Git을 설치하고 도구를 만듭니다.

```
$ sudo apt update
$ sudo apt install git build-essential -y
```

클론 운영자 배포 코드:

```
$ git clone https://github.com/ansible/awx-operator.git
Cloning into 'awx-operator'...
remote: Enumerating objects: 8773, done.
remote: Counting objects: 100% (1388/1388), done.
remote: Compressing objects: 100% (201/201), done.
remote: Total 8773 (delta 1223), reused 1275 (delta 1184), pack-reused 7385
Receiving objects: 100% (8773/8773), 2.35 MiB | 17.69 MiB/s, done.
Resolving deltas: 100% (5033/5033), done.
```

AWX 23 설치방법

Kubernetes에 AWX Operator 배포

다음으로 Operator가 배포될 네임스페이스를 만듭니다. 이름을 awx로 지정하겠습니다.

```
$ export NAMESPACE=awx
$ kubectl create ns ${NAMESPACE}
```

현재 컨텍스트를 NAMESPACE 변수에 설정된 값으로 설정합니다.

```
$ kubectl config set-context --current --namespace=$NAMESPACE
Context "default" modified.
```

awx-operator 디렉토리로 전환합니다.

```
$ cd awx-operator/
```

AWX 23 설치방법

Kubernetes에 AWX Operator 배포

AWX Operator 릴리스의 최신 버전을 RELEASE_TAG 변수로 저장한 다음 git을 사용하여 분기로 체크아웃하세요.

```
$ sudo apt install curl jq -y
RELEASE_TAG=`curl -s https://api.github.com/repos/ansible/awx-operator/releases/latest | grep tag_name | cut -d '"' -f 4` echo $RELEASE_TAG git checkout $RELEASE_TAG

$ git checkout $RELEASE_TAG

# 클러스터에 AWX Operator를 배포합니다.

$ export NAMESPACE=awx

$ make deploy

$ kubectl get pods
NAME                                               READY STATUS RESTARTS AGE
awx-operator-controller-manager-68d787cfbd-z75n4 2/2 Running     0          40s
```

AWX 23 설치방법

Operator를 사용하여 Ubuntu에 Ansible AWX 설치

AWX 데이터 지속성을 위한 정적 데이터 PVC 생성

```
$ cat <<EOF | kubectl create -f -
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: static-data-pvc
  namespace: awx
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: local-path
  resources:
    requests:
      storage: 5Gi
EOF

$  kubectl get pvc -n awx
NAME              STATUS VOLUME CAPACITY ACCESS MODES STORAGECLASS AGE
static-data-pvc     Pending                                       local-path   43s
```

AWX 23 설치방법

Operator를 사용하여 Ubuntu에 Ansible AWX 설치

AWX 배포 파일 생성

$ vim awx-deploy.yml

```
---
apiVersion: awx.ansible.com/v1beta1
kind: AWX
metadata:
  name: awx
spec:
  service_type: nodeport
  projects_persistence: true
  projects_storage_access_mode: ReadWriteOnce
  web_extra_volume_mounts: |
    - name: static-data
      mountPath: /var/lib/projects
  extra_volumes: |
    - name: static-data
      persistentVolumeClaim:
        claimName: static-data-pvc
```

AWX 23 설치방법

Operator를 사용하여 Ubuntu에 Ansible AWX 설치

```
$ kubectl apply -f awx-deploy.yml
awx.awx.ansible.com/awx created

$ watch kubectl get pods -l "app.kubernetes.io/managed-by=awx-operator"

# 실행한 후에는 다음 명령을 사용하여 확인할 수 있습니다.

$ kubectl get pods -l "app.kubernetes.io/managed-by=awx-operator"
NAME                  READY   STATUS    RESTARTS   AGE
awx-postgres-13-0     1/1     Running   0          7m58s
awx-task-6874d8656b-v75s4   4/4     Running   0          7m10s
awx-web-6c797b8657-sllzt    3/3     Running   0          5m52s

# 운영자 Pod 로그에서 설치 프로세스를 추적할 수 있습니다.

$ kubectl logs -f deployments/awx-operator-controller-manager -c awx-manager
```

AWX 23 설치방법

Ansible AWX 대시보드에 액세스

awx-service Nodeport를 확인

```
$ kubectl get svc -l "app.kubernetes.io/managed-by=awx-operator"
NAME              TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
awx-postgres-13   ClusterIP   None            <none>        5432/TCP       153m
awx-service       NodePort    10.43.179.217   <none>        80:30059/TCP   152m
```

서비스 유형이 NodePort로 설정된 것을 볼 수 있습니다.

웹브라우저에서 접속

http://hostip_or_hostname:30080