**Installing Ansible AWX on RedHat 8 With Kubernetes K3S**

https://redhatnordicssa.github.io/ansible-podman-containers-2

**1) K3s Introduction**

K3s is a lightweight Kubernetes distribution created by Rancher Labs, and it is fully certified by the Cloud Native Computing Foundation (CNCF). K3s is highly available and production ready. It has a very small binary size and very low resource requirements.

**Node Port Introduction**

A Node Port service is the most primitive way to get external traffic directly to your service. Node Port, as the name implies, opens a specific port on all the Nodes (the VMs), and any traffic that is sent to this port is forwarded to the service.

Basically, a Node Port service has two differences from a normal “Cluster IP” service. First, the type is “Node Port.” There is also an additional port called the node Port that specifies which port to open on the nodes. If you don’t specify this port, it will pick a random port. Most of the time you should let Kubernetes choose the port; as thockin says, there are many caveats to what ports are available for you to use.

When would you use this?

You can only have one service per port

You can only use ports 30000–32767

**Ingress**

Ingress is actually NOT a type of service. Instead, it sits in front of multiple services and act as a “smart router” or entrypoint into your cluster.You can do a lot of different things with an Ingress, and there are many types of Ingress controllers that have different capabilities.The default GKE ingress controller will spin up a [HTTP(S) Load Balancer](https://cloud.google.com/compute/docs/load-balancing/http/) for you. This will let you do both path based and subdomain based routing to backend services. For example, you can send everything on foo.yourdomain.com to the foo service, and everything under the yourdomain.com/bar/ path to the bar service.

**AWX Introduction**

The AWX project—AWX for short—is an open-source community project, sponsored by Red Hat, that enables users to better control their community Ansible project use in IT environments. AWX is the upstream project from which the automation controller component is ultimately derived.

**Find the below diagram for k3s awx structure with Node Port and Ingress**

Diagram

Description automatically generated

This isn’t the most technically accurate diagram, but I think it illustrates the point of how a Node Port works

2) Required 3 machines for K3s cluster and one machine for Postgres

Update your Machines for updates all the presently installed packages to their latest versions that are available in the repositories.

a) [eve@awx1 ~]$ sudo su –

b) [root@awx1 ~]# dnf -y update

c) Disable Firewalld. This is recommended by K3s.

[root@awx1 ~]# systemctl disable firewalld --now

d) Once the update is successful perform a system reboot

[root@awx1 ~]# sudo reboot

3. K3s Installation steps

a) switch user root user for k3s cluster configuration



b) Define security key for MYSECRET variable



c) bootstrapping k3s cluster for embedded etcd multi master HA mode

MYSECRET=XXXXXXX 🡪 DMSP

Anbk@domain@12345

MYSECRET=XXXXXXX 🡪 MMSP

[root@awx1 ~]# curl -fL https://get.k3s.io | K3S\_TOKEN=${MYSECRET} \

sh -s - server --cluster-init

### d) Deploy other k3s master nodes (optional)

Now that the first master is deploy, add additional masters (*remember to keep the total number of masters to an odd number*) by referencing the secret, and the IP address of the first master, on all the others:

MYSECRET= GO TO STEP 3a

[root@awx1 ~]# curl -fL https://get.k3s.io | K3S\_TOKEN=${MYSECRET} \

sh -s - server --server <https://10.1.214.208:6443>

above cmd for DMSP

curl -fL https://get.k3s.io | K3S\_TOKEN=${MYSECRET} \

sh -s - server --server <https://10.1.60.56:6443>

above command for MMSP

e) Repeat 3c step for adding additional master servers

f) [root@awx1 ~]# kubectl get nodes

Text

Description automatically generated

4) Install K3s Kubernetes Distribution

AWX is supported and can only be run as a containerized application using Docker images deployed to either an OpenShift cluster, a Kubernetes cluster, or docker-compose. We shall use K3s Kubernetes setup to run AWX on CentOS 8 / Rocky Linux 8.

Put SELinux in permissive mode:

1. [root@awx1 ~]# sudo setenforce 0
2. [root@awx1 ~]# sudo sed -i 's/^SELINUX=.\*/SELINUX=permissive/g' /etc/selinux/config
3. [root@awx1 ~]# cat /etc/selinux/config | grep SELINUX=

5) Install k3s by running the commands below:

a)

[root@awx1 ~]# curl -sfL https://get.k3s.io | sudo bash -

[root@awx1 ~]# sudo chmod 644 /etc/rancher/k3s/k3s.yaml

b) [root@awx1 ~]# systemctl status k3s.service

c) check version kubernetes version

[root@awx1 ~]# kubectl version –short

* Uninstalling k3s

**/usr/local/bin/k3s-uninstall.sh**

\_\_\_\_\_\_\_\_\_\_\_\_ /usr/local/bin/k3suninstall.sh\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

d)Install postgres on non-Master server

------postgres setup-------

https://www.tcien.com/install-postgresql-server-12-6-on-rhel-8/

Install postgres version 12 for awx

[root@awxapppostgres eve]# yum remove postgres\\*

[root@awxapppostgres eve]# dnf install <https://download.postgresql.org/pub/repos/yum/reporpms/EL-8-x86_64/pgdg-redhat-repo-latest.noarch.rpm>

[root@awxapppostgres eve]# dnf -qy module disable postgresql

[root@awxapppostgres eve]# dnf install postgresql12

[root@awxapppostgres eve]# dnf install postgresql12-server

[root@awxapppostgres eve]# /usr/pgsql-12/bin/postgresql-12-setup initdb

[root@awxapppostgres eve]# systemctl enable postgresql-12

[root@awxpostgres ~]# systemctl enable postgresql-12

Created symlink /etc/systemd/system/multi-user.target.wants/postgresql-12.service → /usr/lib/systemd/system/postgresql-12.service.

[root@awxpostgres ~]# systemctl start postgresql-12

[root@awxpostgres ~]# /usr/bin/psql --version

psql (PostgreSQL) 12.12

[root@awxpostgres ~]# psql

psql: error: FATAL: role "root" does not exist

[root@awxpostgres ~]# su -l postgres

[postgres@awxpostgres ~]$ psql

psql (12.12)

Type "help" for help.

postgres=# alter user postgres password 'Ansible';

ALTER ROLE

postgres=# CREATE USER awx;

CREATE ROLE

postgres=# ALTER USER awx WITH LOGIN ENCRYPTED PASSWORD ‘XXXXXXXX’; 🡪 DMSP

ALTER USER awx WITH LOGIN ENCRYPTED PASSWORD ‘XXXXXXXX’; 🡪 MMSP

ALTER ROLE

postgres=# CREATE DATABASE awx;

CREATE DATABASE

postgres=# GRANT ALL PRIVILEGES ON DATABASE awx TO awx;

GRANT

postgres=# ALTER USER awx WITH SUPERUSER;

ALTER ROLE

postgres=# \du

List of roles

Role name | Attributes | Member of

-----------+------------------------------------------------------------+-----------

awx | Superuser | {}

postgres | Superuser, Create role, Create DB, Replication, Bypass RLS | {}

postgres=#

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

droping db

REVOKE CONNECT ON DATABASE awx FROM awx;

Then use:

SELECT pg\_terminate\_backend(pg\_stat\_activity.pid)

FROM pg\_stat\_activity

WHERE pg\_stat\_activity.datname = 'awx';

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Update postgres config files

https://stackoverflow.com/questions/41125185/postgresql-could-not-connect-to-server-connection-refused-error#comment112301435\_59033144

/var/lib/pgsql/data

[root@awxdb data]# vi pg\_hba.conf

UPDATE BELOW LIES

host replication all ::1/128 ident

host all all 0.0.0.0/0 md5

host all all 10.1.214.0 255.255.255.0 trust 🡪 DMSP

host all all 10.1.60.0 255.255.255.0 trust 🡪 MMSP

#tcpip\_socket = true

https://stackoverflow.com/questions/41125185/postgresql-could-not-connect-to-server-connection-refused-error#comment112301435\_59033144

[root@awxdb data]# pwd

/var/lib/pgsql/data

[root@awxdb data]# vi postgresql.conf

UPDATE BELOW LINES

#------------------------------------------------------------------------------

# CONNECTIONS AND AUTHENTICATION

#------------------------------------------------------------------------------

# - Connection Settings -

listen\_addresses = '\*'

#listen\_addresses = 'localhost' # what IP address(es) to listen on;

# comma-separated list of addresses;

# defaults to 'localhost'; use '\*' for all

# (change requires restart)

port = 5432 # (change requires restart)

max\_connections = 150 # (change requires restart)

#superuser\_reserved\_connections = 3 # (change requires restart)

Restrt postgrs service

[root@awxapppostgres data]# systemctl start postgresql-12

[root@awxapppostgres data]# systemctl restart postgresql-12

[root@awxapppostgres data]# systemctl status postgresql-1

Install postges client

<https://www.cyberciti.biz/faq/install-and-setup-postgresql-on-rhel-8/>

[root@dmspawxapp20 ~]# psql -h 10.1.214.14 -p 5432 -d awx -U awx -W

[root@dmspawxapp20 ~]# psql -h 10.1.214.14 -p 5432 -d awx -U awx -W

Pwd: WDKCGVoGjB

6) Deploy AWX Operator on Kubernetes

This Kubernetes Operator has to be deployed for the management of one or more AWX instances in any namespace.

a)Install git and make tools:

[root@awx1 ~]# sudo yum -y install git make

[root@awx1 ~]# git clone <https://github.com/ansible/awx-operator.git>

b) Create namespace where operator will be deployed. I’ll name mine as awx:

[root@awx1 ~]#export NAMESPACE=awx

[root@awx1 ~]#kubectl create ns ${NAMESPACE}

Set current context to value set in NAMESPACE variable:

[root@awx1 ~]# kubectl config set-context --current --namespace=$NAMESPACE

Context "default" modified.

Switch to awx-operator directory:

[root@awx1 ~]#cd awx-operator/

c)Save the latest version from AWX Operator releases as RELEASE\_TAG variable then checkout to the branch using git.

[root@awx1 ~]#sudo yum -y install jq

[root@awx1 ~]#RELEASE\_TAG=`curl -s https://api.github.com/repos/ansible/awx-operator/releases/latest | grep tag\_name | cut -d '"' -f 4`

[root@awx1 ~]#echo $RELEASE\_TAG

d) Deploy AWX Operator into your cluster:

[root@awx1 ~]#git checkout $RELEASE\_TAG

[root@awx1 ~]#export NAMESPACE=awx

[root@awx1 ~]#make deploy

Check pods on awx workspace

[root@awx1 ~]# kubectl get pods -n awx

e) Install Ansible AWX on RHEL 8 /

1. Getting Certs from Ashish for connecting to git ,satellite, ldap and ingress(https)
2. Cert name: tls.key, tls.cert and bundle\_awx\_mso\_g3\_priv\_satellite.pem
3. And copy above certs to awx-operator
4. And then run below commands

kubectl create secret generic awx-custom-satellite-certs --from-file=bundle-ca.crt=./bundle\_awx\_mso\_g3\_priv\_satellite.pem -n awx

kubectl create secret tls awx-ingress-tls-secret --cert=./tls.crt --key=./tls.key -n awx

f) Create AWX deployment file:

[root@awx1 ~]# vi awx-instance-deployment.yml

Paste below contents to the file created

--

apiVersion: awx.ansible.com/v1beta1

kind: AWX

metadata:

name: awx

spec:

replicas: 4

service\_type: nodeport

ingress\_type: ingress

ingress\_tls\_secret: awx-ingress-tls-secret

hostname: itawx.idm.dev.entrust.com

#admin\_user: admin

#admin\_password\_secret: Ansible123

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: public-static-data-pvc

postgres\_configuration\_secret: postgres-configuration

bundle\_cacert\_secret: awx-custom-satellite-certs

create postgres secret file

[root@awx2 awx-operator]# vi public-static-pvc.yaml

---

apiVersion: v1

kind: PersistentVolumeClaim

metadata:

name: public-static-data-pvc

spec:

accessModes:

- ReadWriteOnce

storageClassName: local-path

resources:

requests:

storage: 50Gi

[root@awx2 awx-operator]# vi awx-postgres-secret-config.yml

---

apiVersion: v1

kind: Secret

metadata:

name: postgres-configuration

namespace: awx

stringData:

host: "10.1.214.214"

port: "5432"

database: awx

username: awx

password: WDKCGVoGjB

sslmode: prefer

type: unmanaged

type: Opaque

Install AWX on RedHat 8:

[root@awx1 ~]# kubectl apply -f awx-postgres-secret-config.yml -n awx

Run certs on MMSP and DMSP servers

[root@awx1~]# kubectl create secret generic awx-custom-satellite-certs --from-file=bundle-ca.crt=./bundle\_awx\_mso\_g3\_priv\_satellite.pem -n awx --> DMSP

[root@awx1~]# kubectl create secret generic awx-custom-satellite-certs --from-file=bundle-ca.crt=./bundle\_awx\_mso\_g3\_priv\_satellite.pem -n awx --> DMSP

kubectl create secret tls awx1-ingress-tls-secret --cert=./itawx1.entrust.com.cer.crt --key=./itawx1.entrust.com.key.crt -n awx 🡪 DMSP

kubectl create secret tls awx-ingress-tls-secret --cert=./itawxp.entrust.com\_CERT.crt --key=./itawxp.entrust.com\_KEY.crt -n awx --> MMSP

[root@awx1 ~]# kubectl apply -f awx-instance-deployment.yml -n awx

[root@awx1 ~]# watch kubectl get pods -l "app.kubernetes.io/managed-by=awx-operator" -n awx

[root@awx1 ~]# kubectl get pvc {when we can use internal postgres}

[root@awx1 ~]# kubectl get pods -n awx

[root@awx1 ~]# kubectl get deploy

[root@awx1 ~]# kubectl -n awx logs deploy/awx

[root@awx1 ~]#kubectl get service -n awx

Login to AWX

<https://itawx.idm.dev.entrust.com/#/login>

Graphical user interface, application, Teams

Description automatically generated

username: gudipus 🡪 enter your corp creds(like email or teams)  
password: xxxxx

Update Ldap:

Graphical user interface, text, application, email

Description automatically generated

LDAP Server URI:- ldaps://pmspad01.corporate.datacard.com:636

LDAP Bind DN :- CN=svc\_awxpatch,OU=Service Accounts,OU=Admin,DC=corporate,DC=datacard,DC=com

LDAP Bind Password : Enter svc\_awxpatch

LDAP User DN Template:- Not configured

LDAP Group Type:- ActiveDirectoryGroupType

LDAP Require Group:- CN=DL- Security Engineering,OU=DisGrp,DC=corporate,DC=datacard,DC=com

LDAP Deny Group:-Not configured

LDAP Start TLS:- Off

LDAP User Search:-

[

"DC=corporate,DC=datacard,DC=com",

"SCOPE\_SUBTREE",

"(sAMAccountName=%(user)s)"

]

LDAP Group Search:-

[

"DC=corporate,DC=datacard,DC=com",

"SCOPE\_SUBTREE",

"(objectClass=group)"

]

LDAP User Attribute Map:-

{

"email": "mail",

"first\_name": "givenName",

"last\_name": "sn"

}

LDAP Group Type Parameters:- {}

LDAP User Flags By Group:-

{

"is\_superuser": [

"CN=DL- Security Engineering,OU=DisGrp,DC=corporate,DC=datacard,DC=com"

]

}

LDAP Organization Map:-{}

LDAP Team Map-{}

And then save it

Configuring host groups in Satellite Inventory:

Create host group : edc\_ansible group under satellite integration job for excluding the awx and awxdb(postgres) servers during patching

After creating the group , we can add hosts(awx servers and postgredb)

Graphical user interface

Description automatically generated

A screenshot of a computer

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

---

validate\_certs: False

want\_hostcollections: True

group\_prefix: edc\_

want\_ansible\_ssh\_host: True

host\_filters: 'host\_collection="dmsp"'

* For MMSP host group creation :

Create host group : edc\_ansible group under satellite integration job for excluding the awx and awxdb(postgres) servers during patching

After creating the group , we can add hosts(awx servers and postgredb)

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

---

want\_hostcollections: True

group\_prefix: edc\_

want\_ansible\_ssh\_host: True

host\_filters: 'host\_collection="cmsp" || host\_collection="cmsp\_loadbuild" || host\_collection="pmsp\_loadbuild"'

validate\_certs: False

K3S AWX servers

New AWX Machines

https://itawx1.idm.dev.entrust.com/#/login

DNS Name: dmspawxapp20.idm.dev.entrust.com

IP Addresses: 10.1.214.10

DNS Name: dmspawxapp21.idm.dev.entrust.com

IP Addresses: 10.1.214.11

DNS Name: dmspawxapp22.idm.dev.entrust.com

IP Addresses: 10.1.214.12

DNS Name: dmspawxapp23.idm.dev.entrust.com

IP Addresses: 10.1.214.13

DNS Name: dmspawxdb20.idm.dev.entrust.com

IP Addresses: 10.1.214.14

https://itawx.idm.dev.entrust.com/#/login

DNS Name: awx1.idm.dev.entrust.com

IP Addresses: 10.1.214.80

DNS Name: awx2.idm.dev.entrust.com

IP Addresses: 10.1.214.208

DNS Name: awx3.idm.dev.entrust.com

IP Addresses: 10.1.214.209

DNS Name: awx4.idm.dev.entrust.com

IP Addresses: 10.1.214.211

DNS Name: awx5.idm.dev.entrust.com

IP Addresses: 10.1.214.221

DNS Name: awxpostgres.idm.dev.entrust.com

IP Addresses: 10.1.214.214

Old AWX Machines

DNS Name: dmspawxapp11.idm.dev.entrust.com

IP Addresses: 10.1.214.51

DNS Name: dmspawxapp12.idm.dev.entrust.com

IP Addresses: 10.1.214.52

DNS Name: dmspawxapp13.idm.dev.entrust.com

IP Addresses: 10.1.214.53

DNS Name: dmspawxapp14.idm.dev.entrust.com

IP Addresses: 10.1.214.54

MMSP server

New AWX MMSP servers

mmspawxapp05 (10.1.60.56)

mmspawxapp06 (10.1.60.54)

mmspawxapp07 (10.1.60.50)

mmspawxapp08 (10.1.60.53)

mmspawxdb010 (10.1.60.49)

when we provision , mmsp/dmsp server for awx, we need to configure the /etc/resolv.conf file same as old awx machines (like domain names and name server)

old AWX MMSP server

[mmspansapp04.idm.corp.entrust.com](http://mmspansapp04.idm.corp.entrust.com/) IP Addresses: 10.1.60.84 [mmspansapp03.idm.corp.entrust.com](http://mmspansapp03.idm.corp.entrust.com/) IP Addresses: 10.1.60.83 [mmspansapp02.idm.corp.entrust.com](http://mmspansapp02.idm.corp.entrust.com/) IP Addresses: 10.1.60.82 [mmspansapp01.idm.corp.entrust.com](http://mmspansapp01.idm.corp.entrust.com/) IP Addresses: 10.1.60.81