

Ansible AWX

- [Installation von AWX \(Docker Version\)](#)
- [☐ AWX on Single Node K3s](#)

Installation von AWX (Docker Version)

Um AWX zu installieren, folgen Sie diesen Schritten:

1. Installieren Sie zunächst einige notwendige Pakete:

```
sudo apt install docker.io docker-compose ansible
```

2. Starten Sie den Docker-Dienst und fügen Sie ihn dem Autostart hinzu:

```
sudo systemctl enable docker  
sudo systemctl start docker
```

3. Klonen Sie AWX in der neuesten Version von Github:

```
git clone https://github.com/ansible/awx.git
```

4. Wechseln Sie in den AWX-Ordner:

```
cd awx/
```

5. Erstellen Sie das Image:

```
sudo make docker-compose-build
```

6. Wenn Ihr System hinter einem Proxy steht, übertragen Sie ihn in die Docker-Konfiguration:

```
sudo mkdir -p /etc/systemd/system/docker.service.d/  
sudo nano /etc/systemd/system/docker.service.d/http-proxy.conf
```

Fügen Sie folgenden Inhalt ein:

```
[Service]  
Environment="HTTP_PROXY=http://<USERNAME>:<PASSWORD>@<PROXY-IP>:<PORT>"  
Environment="HTTPS_PROXY=https://<USERNAME>:<PASSWORD>@<PROXY-IP>:<PORT>"
```

7. Starten Sie die Container für AWX, Postgres und Redis:

```
sudo make docker-compose
```

Hinweis: Der Befehl blockiert die Kommandozeile. Verwenden Sie `sudo make docker-compose &`, um die Shell weiterhin nutzen zu können.

8. Warten Sie, bis der Dialog erscheint, bevor Sie fortfahren.

9. Starten Sie die grafische Oberfläche:

```
sudo docker exec tools_awx_1 make clean-ui ui-devel
```

10. Konfigurieren Sie den Proxy für NPM innerhalb des Docker Containers:

```
docker exec -it tools_awx_1  
npm config set proxy http://username:password@host:port  
npm config set https-proxy http://username:password@host:port
```

11. Starten Sie erneut die grafische Oberfläche:

```
docker exec tools_awx_1 make clean-ui ui-devel
```

12. Die grafische Oberfläche ist unter der Adresse `https://localhost:8043/` erreichbar.

13. Legen Sie vor dem ersten Login ein Administrator-Konto an:

```
sudo docker exec -ti tools_awx_1 awx-manage createsuperuser
```

Der darauf folgende Eingabeprompt verlangt von uns dann Benutzernamen und Passwort. Nach erfolgreicher Eingabe lässt sich dieser Benutzer auch direkt benutzen um die Login-Maske zu überwinden:

14. Erstellen Sie ein Skript, um AWX automatisch nach einem Neustart zu starten:

```
nano /opt/start_awx.sh  
chmod +x /opt/start_awx.sh
```

Inhalt der Datei `/opt/start_awx.sh`:

```
#!/bin/bash  
cd /<INSTALLATIONSPFAD_VON_AWX>/awx/ && make docker-compose
```

15. Fügen Sie das Skript dem Cron-Dienst hinzu:

```
crontab -e
```

Fügen Sie folgende Zeile am Ende der Datei hinzu:

```
@reboot /opt/start_awx.sh
```

Durch Ausführen von `make docker-compose` wird AWX gestartet.

☐☐ AWX on Single Node K3s

An example implementation of AWX on single node K3s using AWX Operator.

- Accessible over HTTPS from remote host

☐☐ Table of Contents

- [☐☐ Requirements](#)
- [☐ Install K3s](#)
- [☐ Install AWX Operator](#)
- [☐ Prepare required files to deploy AWX](#)
 - [☐ Deploy AWX](#)
- [☐ Troubleshooting Issues](#)

☐☐ Requirements

- **Computing resources**
 - Both **AMD64** (x86_64) with x86-64-v2 or cpu type Host support bc the DB would not start , and **ARM64** (aarch64) are supported.
 - **4 GiB RAM minimum.**
 - It's recommended to add more CPUs and RAM (like 4 CPUs and 8 GiB RAM or more) to avoid performance issue and job scheduling issue.
- **Storage resources**
 - At least **10 GiB** for `/var/lib/rancher` are safe for fresh install.
 - **The actual consumption highly depends on your environment and your use case**, so you should to pay attention to the consumption and add more capacity if required.

☐☐ Deployment Instruction

Disable firewalld if enabled if not use kubernetes-firewall first before installing K3S. This is [recommended by K3s](#).

```
cd firewall
. kubernetes-firewall.sh
```

❑ Install K3s

Install K3s with `--write-kubeconfig-mode 644` to make the config file (`/etc/rancher/k3s/k3s.yaml`) readable by non-root users.

```
curl -sfL https://get.k3s.io | sh -s - --write-kubeconfig-mode 644
```

Also add your subnet in noproxy in

```
vim /etc/systemd/system/k3s.service.env
#should look like this
no_proxy='localhost,127.0.0.1, .dkfz.heidelberg.de, .inet.dkfz-heidelberg.de, .dkfz.de, 10.131.196.0/22'
```

❑ Install AWX Operator

Clone this repository and change directory.

```
cd ~
git clone https://odcf-gitlab.dkfz.de/it/trainee/awx-test.git
cd awx-operator
```

then copy or create (if needed) an certificat to the kubernetes folder.

PS: is not needed you can also specify the location of the Cert and key.

```
cd kubernetes
openssl req -x509 -nodes -days 365 -newkey rsa:4096 -keyout yourdomain.key -out yourdomain.crt -subj
"/CN=example.com" -addext "subjectAltName=DNS:example.com"
```

After that you copied the Key and Cert file to the system. You can run the awx.sh script Also look for the [AWX-Operator version](#)

```
chmod +x awx.sh
. awx.sh
```

By default, the admin user is admin and the password is available in the `-admin-password secret`. To retrieve the admin password, run:

```
kubectl get secret odcf-awx-admin-password -n awx -o jsonpath="{.data.password}" | base64 --decode ; echo
```

☐ Troubleshooting Issues

1. Check Resources:

```
kubectl -n awx get awx,all,ingress,secrets
```

2. Create Admin Password Secret (if not created):

```
kubectl -n awx create secret generic odcf-awx-admin-password --from-literal=password=<your-admin-password>
```

3. Verify Ingress:

```
kubectl get endpoints -n awx
```

Check if the endpoint for `odcf-awx-service` is available on port 80. If not, update the `awx-ingress-tls.yaml` script.

Some Usefull CMD

```
kubectl get namespaces  
kubectl -n awx get awx,all,ingress,secrets  
kubectl get svc -n <namespace>  
kubectl describe pod <pod_name> -n <namespace>  
kubectl describe service <service_name> -n <namespace>  
kubectl get events -n <namespace>  
kubectl -n awx get all  
kubectl -n awx logs -f deployments/awx-operator-controller-manager
```

Credits: written by Johannes Nguyen

johannes.nguyen@dkfz-heidelberg.de