VENDOR CNF NETWORK FUNCTION INSTALLATION - MOP (METHOD OF PROCEDURE) VMWARE

Cognitive Assistant for Networks (CAN) Release 5.5

JULY 6, 2021

AVANSEUS TECHNOLOGY PVT. LTD.

REVISION HISTORY

Version	Date	Change description	Created by	Updated by	Reviewed by
V 1.0	July, 2021	Initial Release	Hemanth/Umesh	Sandeep Singh	Chiranjib

Table of Contents

1. Caas/PaaS Requirements

This section contains all the mandatory prerequisites and cluster requirements to onboard the NF to VMware Telco Cloud Platform:

- Any Linux Based System preferably CentOS, UBUNTU or RHEL.
- Docker v20.10.2 or higher.
- Kubernetes v1.20.0 or higher.
- Istio v1.8 or higher.
- Istio addons (grafana, kiali, prometheus) have to be enabled.
- Secrets (for Harbour or Docker registry).
- Domain Name for hosting the application.
- SSL Certificate Files for the registered Domain.

Note that Avanseus CAN application uses MongoDB as its persistent storage. Importing the default schema into the mongo database is one of the prerequisites which will be discussed in the later section.

2. Pre-Installation Steps (As Applicable)

2.1. Pushing Images to Harbour

Step 1. Login to harbour registry.

```
$docker login 192.168.100.38
```

Enter the password of the harbour registry (password:VMware1!)

Step 2: Loading the provided packed docker images.

```
$sudo docker load<canapp.tar.gz
$sudo docker load<pyvbi.tar.gz
$sudo docker load<ldap.tar.gz
$sudo docker load<predictionworker.tar.gz
$sudo docker load<predectioncontroller.tar.gz
```

For mongo images we will be using the image provided by the docker hub. Use the below command to load the mongo image:

```
$sudo docker pull mongo:3.4.6
```

Use the below command to check if the images are loaded correctly into the docker.

```
$sudo docker images
```

Output shows all the docker images loaded into the docker.

```
root@telco-harbor:~/AVANSEUS/IMAGES#
root@telco-harbor:~/AVANSEUS/IMAGES# sudo docker images
REPOSITORY                                          TAG                 IMAGE ID       CREATED         SIZE
192.168.100.38/avanseus_images/predictioncontroller   1                   2dd95225d3c9   5 weeks ago     671MB
predictioncontroller                                1                   2dd95225d3c9   5 weeks ago     671MB
192.168.100.38/avanseus_images/predictionworker     1                   b28e643bfa41   5 weeks ago     660MB
predictionworker                                    1                   b28e643bfa41   5 weeks ago     660MB
vbi                                                 1                   409223ef40e7   5 weeks ago     4.5GB
192.168.100.38/avanseus_images/vbi                  <none>              409223ef40e7   5 weeks ago     4.5GB
ldap                                                1                   ef867fbc76cd   5 weeks ago     679MB
192.168.100.38/avanseus_images/ldap                 1                   ef867fbc76cd   5 weeks ago     679MB
192.168.100.38/avanseus_images/canapp               1                   8b5deee54194   5 weeks ago     1.17GB
canapp                                              1                   8b5deee54194   5 weeks ago     1.17GB
192.168.100.38/avanseus_images/vbi                  1                   96ddd31daf73   4 months ago    4.48GB
avanseuscontainer.com/vmware/5.0/pyvbi              v1                  96ddd31daf73   4 months ago    4.48GB
goharbor/chartmuseum-photon                         v0.9.0-v1.10.1      0245d66323de   14 months ago   128MB
```

Step 3: Tag the loaded images. Use the below command to tag the loaded images:

```
$sudo docker tag ldap:1 192.168.100.38/avanseus_images/ldap:1
$sudo docker tag mongo:3.4.6 192.168.100.38/avanseus_images/mongo:1
$sudo docker tag vbi:1 192.168.100.38/avanseus_images/vbi:1
$sudo docker tag canapp:1 192.168.100.38/avanseus_images/canapp:1
$sudo docker tag predictionworker:1 192.168.100.38/avanseus_images/predictionworker:1
$sudo docker tag predictioncontroller:1 192.168.100.38/avanseus_images/predictioncontroller:1
```

Step 4: Push the tagged images into Harbour registry.

```
$sudo docker push 192.168.100.38/avanseus_images/ldap:1
$sudo docker push 192.168.100.38/avanseus_images/mongo:1
$sudo docker push 192.168.100.38/avanseus_images/vbi:1
$sudo docker push 192.168.100.38/avanseus_images/canapp:1
$sudo docker push 192.168.100.38/avanseus_images/predictionworker:1
$sudo docker push 192.168.100.38/avanseus_images/predictioncontroller:1
```

Now, all the Docker images have been pushed to the harbour registry and are ready to be used. Login to harbour and check in the user interface to confirm that all the images have been pushed.

2.2. Istio Installation

Follow the below steps to install the Istio:

1. You can use the link as a reference to install the Istio:

   https://istio.io/latest/docs/setup/getting-started/

2. Download the Istio installation files using the below command:

```
$curl -L https://istio.io/downloadIstio | sh -    //This will download the latest version.
```

3. Move to the Istio package directory.

```
$cd istio-1.9.3
$export PATH=$PWD/bin:$PATH
```

4. Install Istio:

```
$istioctl install --set profile=demo -y
```

5. The installation of Istio is complete. To check all the pods are up, use the below command:

```
$kubectl get all -n istio-system
```

```
capv@avensues02-master-control-plane-grwth [ ~/Avanseus_Files/Helm ]$
capv@avensues02-master-control-plane-grwth [ ~/Avanseus_Files/Helm ]$ kubectl get pods -n istio-system
NAME                                   READY   STATUS    RESTARTS   AGE
grafana-784c89f4cf-ft9rg               1/1     Running   0          31d
istio-ingressgateway-5bcdc9b77f-ttbh2  1/1     Running   0          31d
istiod-7dd96f56f8-9mcbn                1/1     Running   0          31d
jaeger-7f78b6fb65-gwrtd                1/1     Running   2          31d
kiali-dc84967d9-86qqd                  1/1     Running   0          31d
prometheus-7bfddb8dbf-7x9v8            2/2     Running   0          31d
capv@avensues02-master-control-plane-grwth [ ~/Avanseus_Files/Helm ]$
```

2.3. Istio Add-Ons

To install the Istio add-ons, execute the two commands:

```
$kubectl apply -f samples/addons
$kubectl rollout status deployment/kiali -n istio-system
```

Note: If there are errors while trying to install the add-ons, try to run the command again. There may be some timing issues which will be resolved, when you run the command again.

The relevant steps/commands that are applicable and to be performed/executed before installing the HELM charts are as follows:

1. Create a namespace.

```
$ kubectl create ns avanseus-workspace
```

2. Enable Istio injection on the desired namespace i.e., avanseus-workspace.

```
$ kubectl label namespace avanseus-workspace istio-injection=enabled
$ kubectl get namespace -L istio-injection
```

(Use this command to check if Istio-injection is enabled or not in the avanseus-workspace).

```
capv@avensues02-master-control-plane-grwth [ ~/Avanseus_Files/Helm ]$
capv@avensues02-master-control-plane-grwth [ ~/Avanseus_Files/Helm ]$ kubectl get namespace -L istio-injection
NAME                STATUS   AGE   ISTIO-INJECTION
avanseus-workspace   Active   32d   enabled
default             Active   32d
istio-system        Active   32d   disabled
kube-node-lease     Active   32d
kube-public         Active   32d
kube-system         Active   32d
nfs-client          Active   32d
tca-system          Active   32d
tkg-system-public   Active   32d
```

3. Installing Metric Server in the cluster.

Go to ***kubernetes_resources/Gateway_Metric_HPAI*** folder. Use the file "**metric_server.yaml"** and execute the below command:

```
$ kubectl apply -f metric_server.yaml
```

Alternatively, command:

```
$ kubectl apply -f
https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components. yaml
```

Use the below command to check, if metric server is running:

```
$kubectl get all -n kube-system|grep metric
```

```
capv@avensues02-master-control-plane-grwth [ ~/Avanseus_Files/Helm ]$
capv@avensues02-master-control-plane-grwth [ ~/Avanseus_Files/Helm ]$ kubectl get all -n kube-system| grep metric
pod/metrics-server-6d74fbf577-mqm6m                                1/1     Running   0          13d
service/metrics-server            ClusterIP   100.68.57.83     <none>        443/TCP                  13d
deployment.apps/metrics-server          1/1     1            1           13d
replicaset.apps/metrics-server-6d74fbf577          1         1         1       13d
capv@avensues02-master-control-plane-grwth [ ~/Avanseus_Files/Helm ]$
```

4. Secrets (for harbour or docker registry)

Run the below command to create the secret. You can find the config.json file in the Avanseus_files/Docker_sercret/ folder.

```
$kubectl create secret generic avanseus-imagepull-secret
--from-file=.dockerconfigjson=Avanseus_files/Docker_sercret/config.json
--type=kubernetes.io/dockerconfigjson -n avanseus-workspace
```

Use below command to check if secret has been created:

```
$kubectl get secrets -n avanseus-workspace
```

3. Installation Steps

3.1. Helm Installation Charts

There are 6 helm charts to be installed while deploying the CAN application. All the required helm charts are present under **Avanseus_files/Helm_Charts/** folder.

The sequences in which the charts have to be installed are as follows:

1. avanseus-mongo-chart -1 tgz
2. avanseus-ldapchart-chart 1 tgz
3. avanseus-pyvbi-chart 1 tgz
4. avanseus-workerapp-chart 1 tgz
5. avanseus-controllerapp-chart 1 tgz
6. avanseus-canmaster-chart 1 tgz

   Version 1

Minimum list of values that needs to be overridden per deployment along with explanation on how to derive these values.

Image-names: The image names in the chart have to be overridden with the image names pushed in the harbour registry. Image names can be obtained from the commands used above while pushing the images to the harbour registry.

3.2. Creating Configmaps for the charts

Go to Avanseus_files/ConfigMap_Files/ folder and execute the below command to create the configmap is as follows:

```
$kubectl create configmap controllerconfig --from-file=controller-config/ --namespace=avanseus-workspace
$kubectl create configmap workerconfig --from-file=worker-config/ --namespace=avanseus-workspace
$kubectl create configmap canconfig --from-file=can-config/  -n avanseus-workspace
```

```
capv@avensues02-master-control-plane-grwth [ ~ ]$
capv@avensues02-master-control-plane-grwth [ ~ ]$ kubectl get configmap -n avanseus-workspace
NAME                          DATA   AGE
canconfig                     3      32d
istio-ca-root-cert            1      32d
prediction-controller-config  1      13d
prediction-worker-config      1      13d
capv@avensues02-master-control-plane-grwth [ ~ ]$
```

Note:

- Please update domain details in the config map using kubectl edit configmap command.

```
$kubectl edit configmap canconfig -n avanseus-workspace
```

Edit the domain details in the canconfig, the domain details have to filled with domain in which the application will be hosted:

- **avanseus.app.cas.domain and avanseus.app.can.domain as "avanseus.telcocloud.com"** or whichever domain name has been provided by the vendor.

3.3. Creating Persistent Volume

Four persistence volumes have to be created for the whole application deployment.

SL No	Application name	PVC name
1	Mongodb(database)	mongopvc
2	CAN-logs	canlogpvc
3	CAN-prediction related Files	candatapvc
4	LDAP	ldappvc

Go to Avanseus_files/Persistance_Volume/ folder and execute the below commands for creating persistent volume claims:

```
$ kubectl apply -f mongo_pvc.yaml -n avanseus-workspace
$ kubectl apply -f can_pvc.yaml -n avanseus-workspace
$kubectl apply -f candatapvc.yaml -n avanseus-workspace
$ kubectl apply -f ldap_pvc.yaml -n avanseus-workspace
```

Verify whether all the 4 PVC are created using:

```
$ kubectl get pvc -n avanseus-workspace
```

```
capv@avensues02-master-control-plane-grwth [ ~/Avanseus_files/Helm_Charts/charts ]$ cd
capv@avensues02-master-control-plane-grwth [ ~ ]$ kubectl get pvc -n avanseus-workspace
NAME          STATUS   VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS   AGE
candatapvc    Bound    pvc-c2fbe366-b92b-440d-bae4-9045e84050c9   10Gi       RWO            vsphere-sc     32d
canlogpvc     Bound    pvc-c52df91b-6275-415d-9b2d-8d6a4021dbae   5Gi        RWO            vsphere-sc     32d
nfstest1      Bound    testpv                                     4Gi        RWO            nfs-client     32d
vspherepvc    Bound    pvc-6c8b8c8a-1906-4e26-bb73-7f04f907e08c   30Gi       RWO            vsphere-sc     32d
capv@avensues02-master-control-plane-grwth [ ~ ]$
```

Follow the order in details to comply with the dependencies between charts.

The order in which charts need to be installed and Commands to install those charts are as follows.

Go to Avanseus_files/Helm_Charts/ folder and execute the below command:

```
$helm install mongo avanseus-mongo-chart.-1.tgz  -n avanseus-workspace --set
mongo.container.image=<name_of_the_mongo_image_pushed>
```

In our case <name_of_the_mongo_image_pushed> has to be replaced with **192.168.100.38/avanseus_images/mongo:1**

Notes:

- After creating a mongo chart database has to be restored with master/mandatory tables. Please follow the below steps to do the same:

Go to Avanseus_file/Mongo_Files/mongodb-linux-x86_64-3.4.6/bin/ folder and execute the following commands:

```
./mongo <ip_address_of_any_nodes>:30001/admin -u "admin" -p "Avanseus$0"
use cangeneric
db.createUser({user: "cangeneric", pwd: "Avanseus$0", roles: ["userAdmin", "dbAdmin", "readWrite"]});
./mongorestore -h <ip_address_of_any_nodes>:30001 -u cangeneric -p '<password>' -d cangeneric Avanseus_file/Mongo_Files/Dump/cangeneric/
```

<ip_address_of_any_nodes>= ip address of any of the nodes from the cluster.

```
$helm install ldap avanseus-ldap-chart.-1.tgz -n avanseus-workspace --set ldap.container.image=<name_of_the_ldap_image_pushed>
$helm install vbi avanseus-vbi-chart.-1.tgz -n avanseus-workspace --set vbi.container.image=<name_of_the_vbi_image_pushed>
$helm install worker avanseus-workerapp-chart.-1.tgz -n avanseus-workspace --set avanseus_worker.container.image=<name_of_the_worker_image_pushed>
$helm install controller avanseus-controllerapp-chart.-1.tgz -n avanseus-workspace --set avanseus_controller.container.image=<name_of_the_controller_image_pushed>
$helm install can avanseus-canmaster-chart.-1.tgz -n avanseus-workspace --set can.container.image=<name_of_the_can_image_pushed>
```

Note: Names of the images has to be changed accordingly as they are pushed to the harbour registry.

All Helm charts must follow the helm guidelines shared as part of pre-assessment Questionnaire.

Yes, all the helm charts follow the guidelines mentioned.

4. Post Installation Steps (As Applicable)

The relevant steps/commands that are applicable and to be performed/executed after installing the HELM charts (configuration steps to complete the deployment process):

1. Applying Peer authentication.

   Go to **Avanseus_files/Helm/** folder and execute the below command:

   ```
   $kubectl apply -f peerAuthentication.yaml -n avanseus-workspace
   ```

2. Installing and Configuring Web Server.

   Install the NGINX web server on the master node using the command:

   ```
   $yum install -y nginx
   $sudo systemctl start nginx
   ```

   ssl-nginx.conf and **nginx.conf** files are present in the folder **Avanseus_file/Nginx_files** for web server configuration related changes.

   Place the provided ssl-nginx.conf folder in /**etc/ssl/ folder**.

   Go to the /etc/nginx folder.

   Rename the existing nginx.conf file to nginx.conf_bkp and add the nginx.conf present in **Avanseus_file/Nginx_Files/** folder to **/etc/nginx/ folder**.

   Few changes have to be made to the nginx.conf file.

   Changes to be made are as follows:

   a. domain_name: This has to be changed in the conf file as shown in the image below:

```
    server {
        listen       80;
        server_name avanseus.telcocloud.com;
        location / {
            proxy_pass http://allbackend;
            proxy_http_version 1.1;
        }
```

b. istio_http_port_number (mapping of port no 80 has to be added in the nginx.conf file).

c. Execute the below command to get the istio_http_port_number:

$kubectl get svc istio-ingressgateway -n istio-system

```
capv@avensues02-master-control-plane-grwth [ ~/Avanseus_files/Nginx_files ]$ kubectl get svc istio-ingressgateway -n istio-system
NAME                   TYPE           CLUSTER-IP     EXTERNAL-IP      PORT(S)                                                                      AGE
istio-ingressgateway   LoadBalancer   100.71.158.94  192.168.101.104  15021:30397/TCP,80:31757/TCP,443:31061/TCP,15012:32438/TCP,15443:31088/TCP   32d
capv@avensues02-master-control-plane-grwth [ ~/Avanseus_files/Nginx_files ]$
```

d. path_to_the_certificate_file: path of certificate file has to be added in the nginx.conf file.

e. path_to_the_key_file: path of key file has to be added in the nginx.conf file as seen in the below image:

```
    listen 443 ssl http2; # managed by Certbot
#    ssl_protocols TLSv1.3;
    ssl_certificate /etc/ssl/certs/nginx-selfsigned.crt;
    ssl_certificate_key /etc/ssl/private/nginx-selfsigned.key;
    include /etc/ssl/ssl-nginx.conf; # managed by Certbot
    ssl_dhparam /etc/ssl/certs/dhparam.pem; # managed by Certbot
```

After making the respective changes, restart the nginx server with the below command:

$sudo systemctl restart nginx

3. Istio VirtualService and Gateway

Go to Avanseus_files/Helm_Charts/ folder and execute the below command:

$kubectl apply -f gateway.yaml -n istio-system.

Now, the network function should be fully ready for interoperating with other network functions in the telco cloud.

5. Validation Steps

Steps to validate the successful deployment of the network functions. E.g., commands listing the desired state of PODs along with the screenshots.

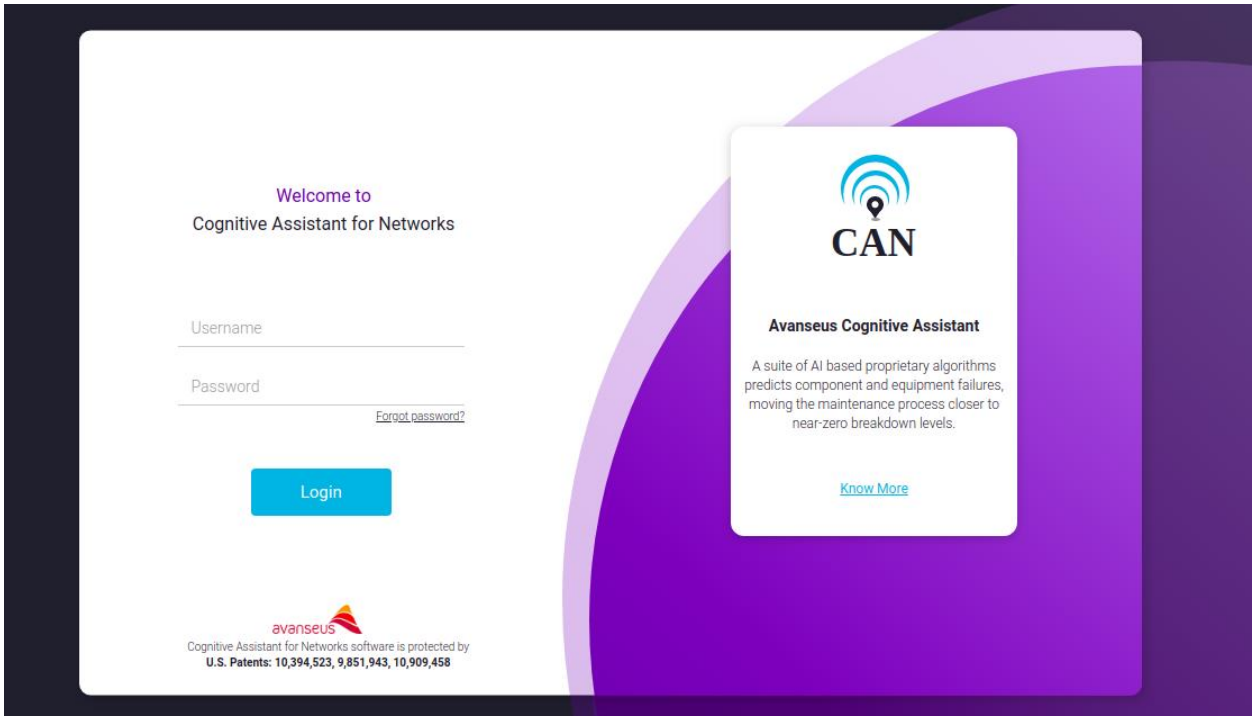
$kubectl get -f pods -n avanseus-workspace

```
capv@avensues02-master-control-plane-grwth [ ~ ]$
capv@avensues02-master-control-plane-grwth [ ~ ]$ kubectl get pods -n avanseus-workspace
NAME                          READY   STATUS    RESTARTS   AGE
can-79b46467f9-m5bws          2/2     Running   0          7d3h
controller-6989886c-lwdvz     2/2     Running   0          13d
ldap-69845bb64f-66fm5         2/2     Running   0          32d
mongo-0                       2/2     Running   0          32d
vbi-ccdf5b64c-6mn8r           2/2     Running   0          13d
worker-59b9577c5c-zrkzc       2/2     Running   0          13d
```

6. Accessing the Application and Monitoring Tools from the Browser

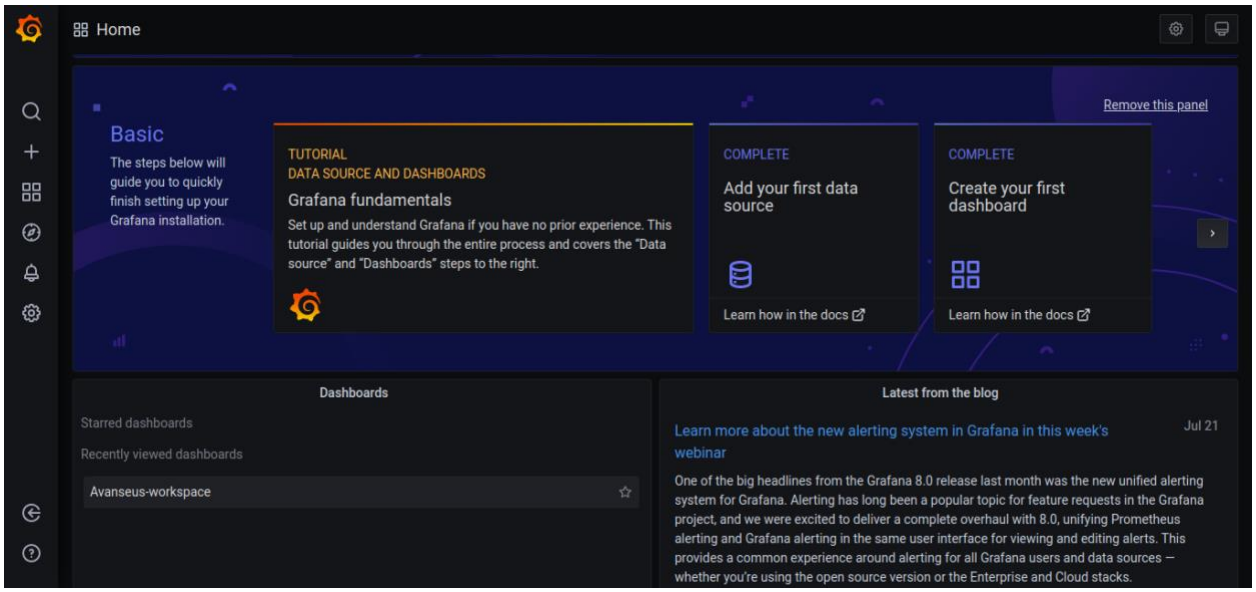
CAN Application: http://<domain_name>/CAN/

Ex: <domain_name> = 192.168.29.100 or can.avanseus.com



Grafana Dashboard: https://<domain_name>/grafana

Ex: <domain_name> = 192.168.29.100 or can.avanseus.com



Kiali Dashboard: https://<domain_name>/kiali

Ex: <domain_name> = 192.168.29.100 or can.avanseus.com