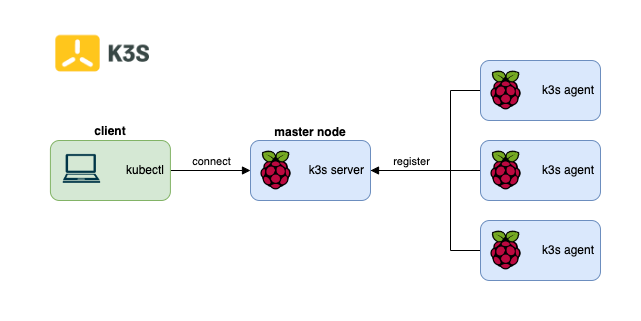
Installation Cluster

# Vorwort

Auf die RasPis wurde das Raspberry Pi OS aufgespielt. Die aktuelle Kombination der Software – k3s, Kubectl, Helm – wurde ausgewählt, da die Installationsanleitung im Internet einfach zu befolgen und verständlich war.

# Installation k3s

Die Installation von k3s entstammt größtenteils dem Folgenden Artikel: <https://medium.com/thinkport/how-to-build-a-raspberry-pi-kubernetes-cluster-with-k3s-76224788576c>

Auf die RasPis wird das OS aufgespielt. Über den Raspberry Pi Imager können ein Name und eine WiFi Verbindung vor dem flashen des Images konfiguriert werden. Die folgende Architektur wurde aufgebaut, allerdings mit nur einem Agenten

Auf dem Master-Node wird der k3s-Server installiert:

curl -sfL https://get.k3s.io | sh -

Die Installation wird nicht fehlerfrei Verlaufen. Im Anschluss muss im boot-Verzeichnis eine Änderung vorgenommen werden.

sudo nano /boot/cmdline.txt

In dem geöffneten Dokument muss am Ender der ersten Zeile folgendes eingefügt werden:

cgroup\_memory=1 cgroup\_enable=memory

Wichtig ist, dass die erste Zeile nicht verlassen und keinen Zeilenumbruch eingefügt wird. Mit strg X und y + Enter wird die Änderung gespeichert. Im Anschluss muss der RaspPi neugestartet werden.

Sudo reboot

Um zu überprüfen, ob der Master-Node korrekt aufgesetzt wurde, kann der folgende Befehl ausgeführt werden:

Sudo k3s kubectl get nodes

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Um einen Agenten oder Worker-Node wird die IP-Adresse des k3s Server und der Token benötigt. Befehl für die IP-Adresse:

hostname -I | awk '{print $1}'

Befehl für das Servertoken:

sudo cat /var/lib/rancher/k3s/server/node-token

Die resultierende Zeichenkette enthält die Zeichenkette „::string::“. Die gesamte Zeichenkette muss zwischengespeichert werden. Auf dem Agenten wird jetzt k3s installiert. IP-Adresse und Token werden als Parameter mitgegeben und müssen im folgenden String ersetzt werden:

curl -sfL https://get.k3s.io | K3S\_URL=https://<kmaster\_IP\_from\_above>:6443 K3S\_TOKEN=<token\_from\_above> sh -

Die Korrektur im Boot-Verzeichnis muss auch bei jedem der Agenten vorgenommen werden. Nach der Korrektur müssen die RasPis neugestartet werden. Durch den folgen Befehl kann die erfolgreiche Installation verifiziert werden.

Sudo k3s kubectl get nodes

# Web-App Check

Die Container für die Web-App sind im Docker hub gespeichert:

kaiser0815/webapp-server

kaiser0815/webapp-client

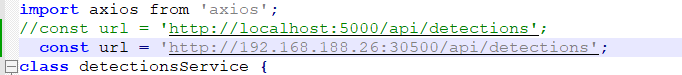
In den Container wurden zwei Anpassungen vorgenommen.

1. Die IP-Adresse der Maria DB wurde angepasst auf die im manifest definierte POD-ID  
   (Rat-Detector\RasPiCluster\WebApp\server\DBConnector\db\_connector.js

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

1. Die IP-Adresse in dem Client – Teil der Anwendung wurde angepasst  
   Grund dafür ist, dass „localhost“ im Cluster nicht funktioniert und der Port 5000 nicht angesprochen werden kann.



Sollte das Cluster neu aufgebaut werden und die Netzwerkonfigurationen abweichen müssen die Container Web-App Server und Webapp-Client mit den angepassten konfigurationen neu erstellt werden.

docker build --platform linux/arm/v7 D:\WorkSpace\WebApp\client -t kaiser0815/webapp-client:v5

docker push kaiser0815/webapp-client:v5

docker build --platform linux/arm/v7 D:\WorkSpace\WebApp\server -t kaiser0815/webapp-server:v5

docker push kaiser0815/webapp-server:v5

Bitte nicht Version v4 überschreiben. Sollte die Version angepasst werden, müssen im manifest „test\_depl.yaml“ die Versionsnummern geändert werden.

# Storage Setup

<https://medium.com/codex/reliable-kubernetes-on-a-raspberry-pi-cluster-storage-ff2848d331df>

Die inhärenten Eigenschaften des Clusters und von Kubernetes erschweren das Speichern der Daten. Daher wurde auf dem Master-Node ein NFS-Server installiert. Da es nur einen Masternode gibt, vermehrt sich nicht die Anzahl der „Single-Point of Failures“. Mit diesem Server kann von jedem Node aus Storage geclaimt werden.

Ein Bild, das Text, orange, Screenshot enthält.

Automatisch generierte Beschreibung

**Der Master-Node muss mit dem passenden Label getaggt werden**:

Sudo kubectl label node <hostname\_master> hdd=enabled

Kopiere das manifest Verzeichnis auf das Cluster

scp -r <WINDOWS\_PATH> pi@masternode:/home/pi/cluster/v1

Mit dem Folgenden Befehl kann der Pod überprüft werden:

Sudo kubectl get pods -n work-space

Aktuell wird noch über die Optionen „NAS“ und „Longhorn“ beraten.

# Setup MariaDB

Um das manifest anzuwenden:

Sudo kubectl apply -f ./manifest

Das Passwort der Mariadb ist in dem „secret.yaml“ gespeichert und ist „password“. Es wurde in im base 64 encoded. Das Passwort ist wichtig, da es im Serverteil der Webanwendung hardcode drin steht.

Bash MariaDB:

Sudo exec -it <Containername> -n work-space – bash

MariaDB Console:

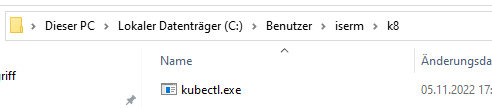
Mariadb -p

password

Ausführen des SQL-Skripts in dem „data“ Ordner

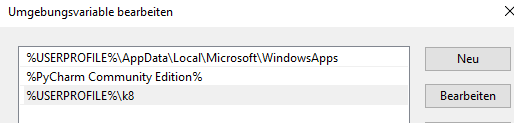
# Installation Kubectl (optional)

Kubectl wird auf dem Gerät installiert, mit dem man dem Cluster Befehle geben möchte. Im Folgenden wird leicht von dem Artikel abgewichen. Die nächste Installation wurde auf Windows durchgeführt. Hier sind alle Anleitung zur Installation von kubectl <https://kubernetes.io/docs/tasks/tools/> . Für Windows wird die aktuelle Version heruntergeladen. Es handelt sich bei der Anwendung um eine Command Line Anwendung, daher muss die Anwendung in die Path Umgebungsvariable hinzugefügt werden. Im Folgenden ist ein Beispiel aufgeführt.



Ein Bild, das Text enthält.

Automatisch generierte Beschreibung



Weiteren muss eine Konfigurationsdatei aus dem Master-Node kopiert werden:

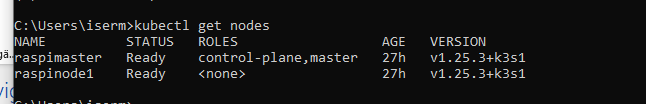
sudo cat /etc/rancher/k3s/k3s.yaml

Greift man über SSH auf den RasPi zu, kann man mit strg + einfg kopiert werden. Die resultierende Datei muss zwischengespeichert und bearbeitet werden. Allerdings darf auf dem Master Node keine Änderung vorgenommen werden. Die IP-Adresse des Master Nodes muss durch die tatsächliche Adresse ersetzt werden. Im Homeverzeichnis des Benutzers muss das Verzeichnis ./kube eingefügt werden. Die bearbeite Datei muss dort unter dem Namen „config“ (ohne Endung) eingefügt werden.

Mit dem folgenden Befehl kann die Installation überprüft werden

kubectl config use-context

kubectl get nodes



# Installation Helm (optional)

Helm bietet eine Möglichkeit das automatische Erstellen, Paketieren, Konfigurieren und Einsetzen von Anwendungen in Kubernetes. Hier muss zunächst die gewünschte Version heruntergeladen werden: <https://helm.sh/docs/intro/install/> . Helm wird auf dem gleichen System wie Kubectl installiert. Hier muss ebenfalls eine Umgebungsvariable angelegt werden, dieses Mal aber als System-Umgebungsvariable.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Mit dem Folgendem Befehl kann überprüft werden, ob Helm installiert ist.

helm version

Um Mosquitto zu installieren, müssen die folgenden Befehle ausgeführt werden.

helm repo add k8s-at-home https://k8s-at-home.com/charts/

helm repo update

# create a dedicated namespace

kubectl create namespace mqtt

# install mosquitto

helm install mosquitto k8s-at-home/mosquitto -n mqtt

# check installation

helm list -n mqtt

# check the resources installed

kubectl get all -n mqtt