Oosto OnWatch V2

Basic Troubleshooting Manual T1/T2

Customer Maintenance Level

From:

Oosto AI LTD.

(“Oosto” or “company”)

May 26, 2022

Table of Contents

Troubleshooting

Camera Connection Failure

This chapter aids with troubleshooting camera connection failure. There are 3 main reasons why camera connections fail:

- **Camera Setup Problem**
- **Camera Credentials**
- **Insufficient Licenses**

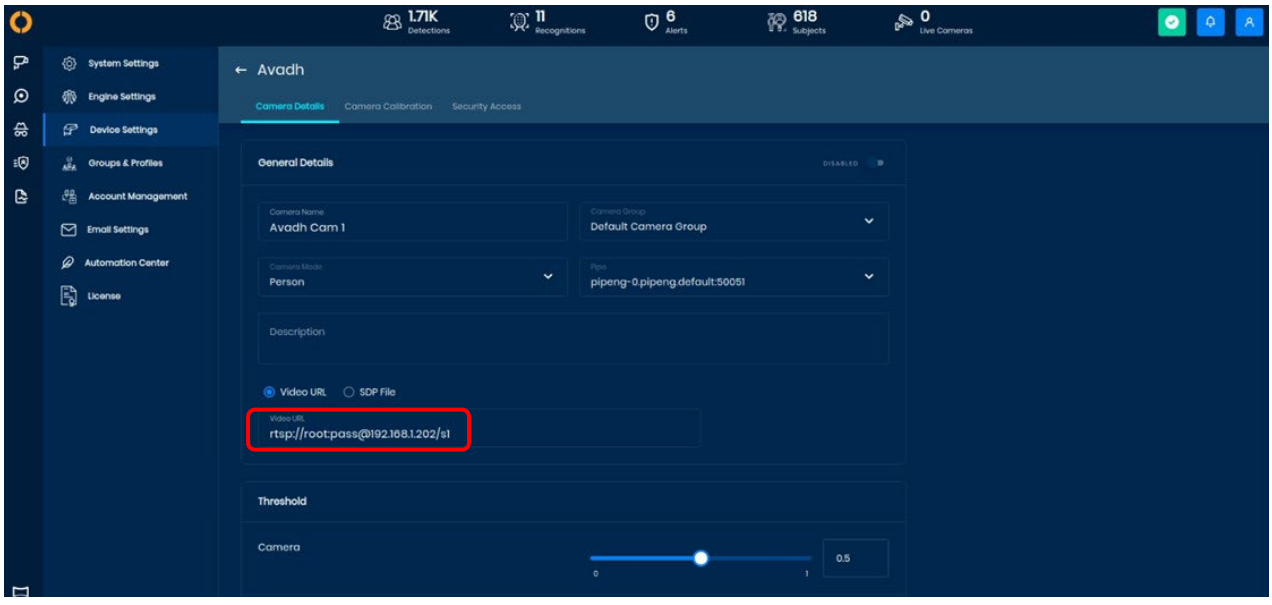
Camera Setup Problem

If you are experiencing a camera setup problem, check the camera connection to the POE switch and all the relevant physical connections. You can also try connecting the camera via Browser or VLC or Ping the camera.

Camera credentials

Check the camera credentials that you have specified while adding the camera in OnWatch from the Device Settings Tab as shown below.

Check if the camera username and password are mentioned in the RTSP URL and are correct.

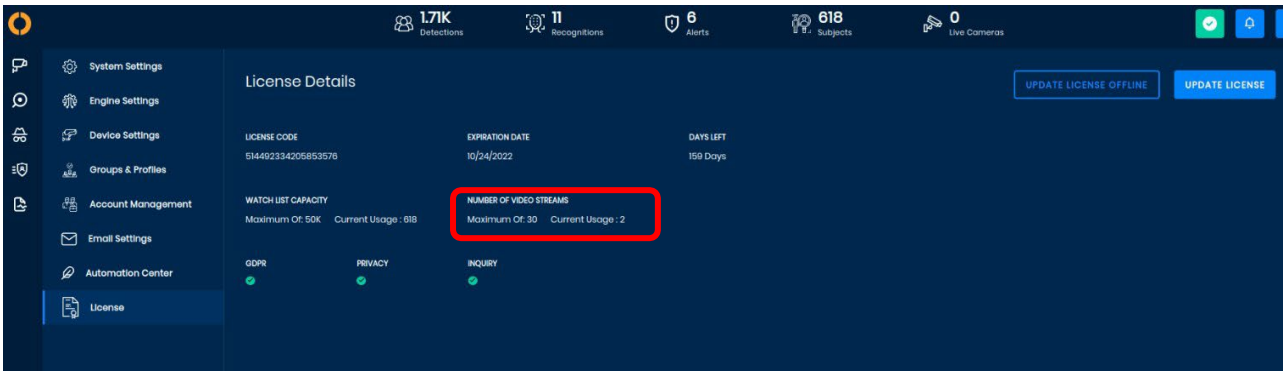


If it still does not work, check if this URL works on VLC player from the same network/

VLAN.

Insufficient Licenses

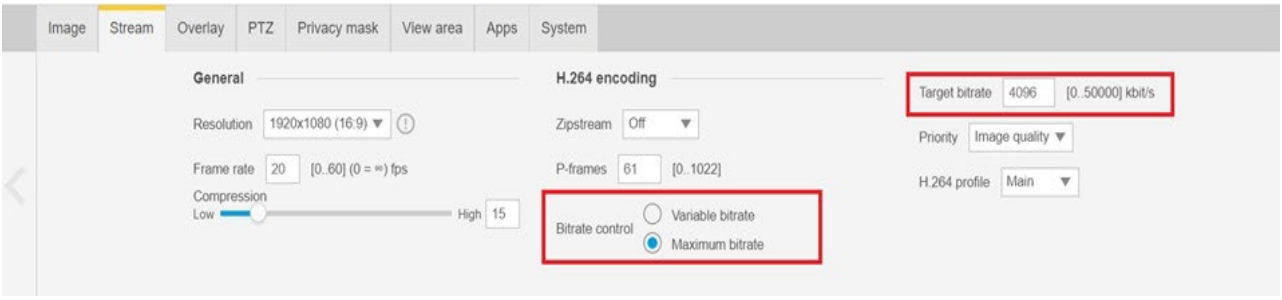
It is important to determine the number of active streams/cameras does not exceed the license capacity and the same can be done by checking the below as the system will not allow you to have more active cameras than the number of video streams supported by your license:



Corrupted Video streams with artifacts

This chapter aids with troubleshooting corrupted video streams. This could be due to the following:

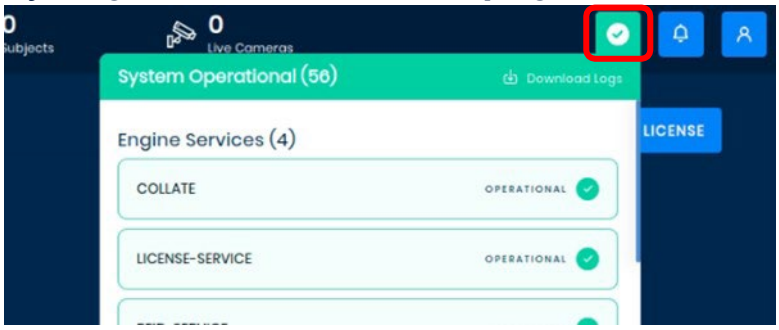
- **Insufficient network bandwidth / bitrate: It is important to ensure that there is sufficient network bandwidth provisioned for the camera to stream. In many cases, the camera bitrate configured is sub-optimal and might create artifacts due to over-compression. Make sure you configure sufficient bitrate (roughly 4-5 Mbps for a Full HD camera stream). This can be generally done from the camera webpage as shown below:**



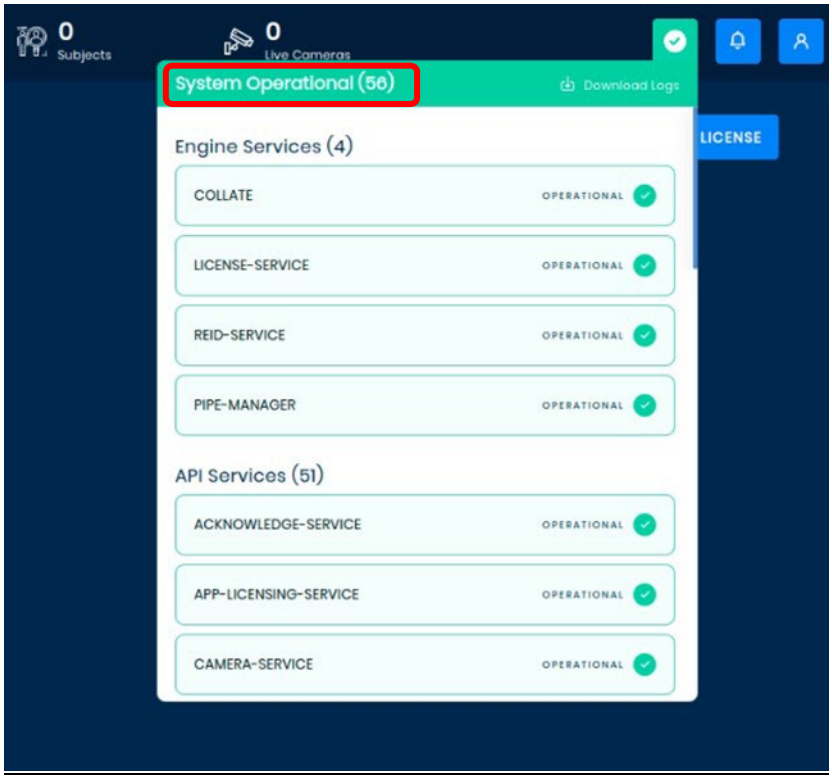
Tools for basic troubleshooting

OnWatch Dashboard: https://kong.tls.ai

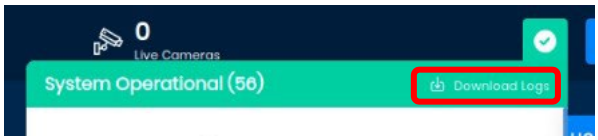
- **Within the OnWatch interface, you can see if all services are running correctly, indicated by the green checkmark in the top right corner.**



- **After selecting the icon, you will see a full list of services and their operational status.**

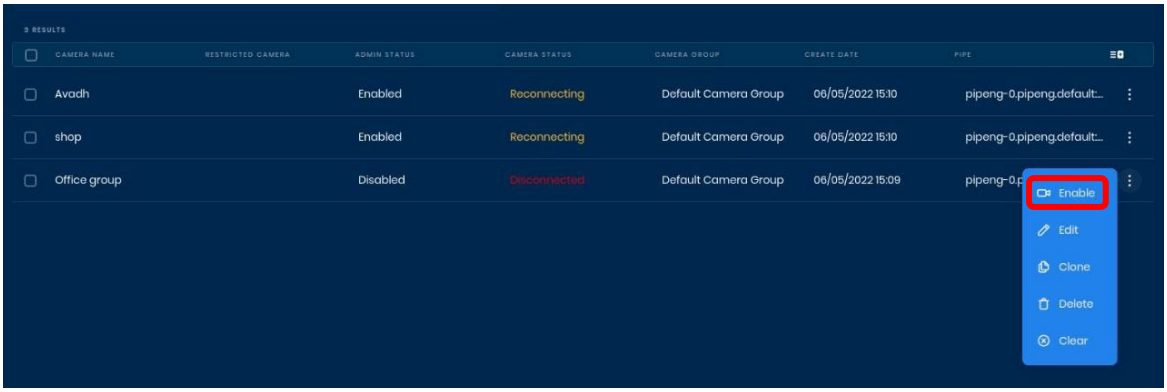


- **There is also a button to download the logs for the services, in order to view recent logs that may indicate what the issue is.**



- **If Services are up, but you are not receiving detections, ensure that your devices are connected**

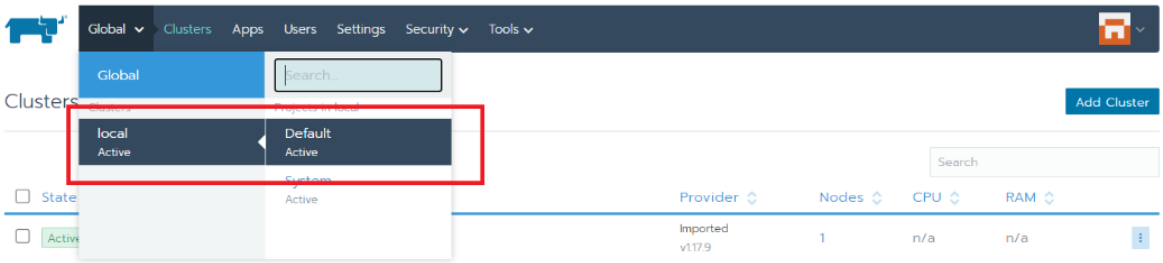
by checking the total number of "Live Cameras":
In Settings → Device Settings, you can see an error notification and try to reconnect/enable



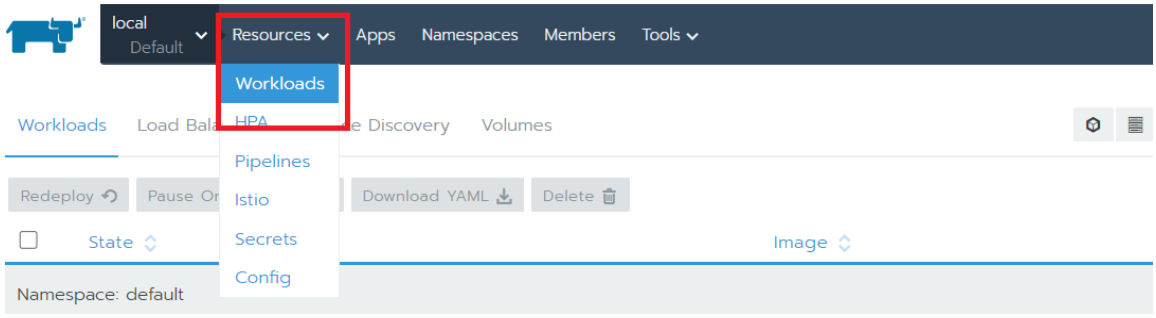
cameras.

Rancher Dashboard: https://rancher.anv

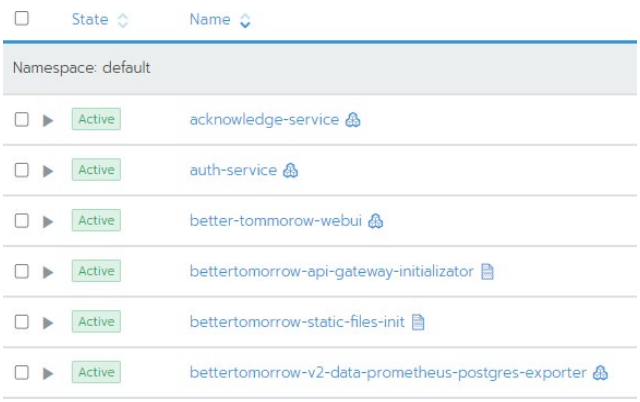
- **Choose local → default project**



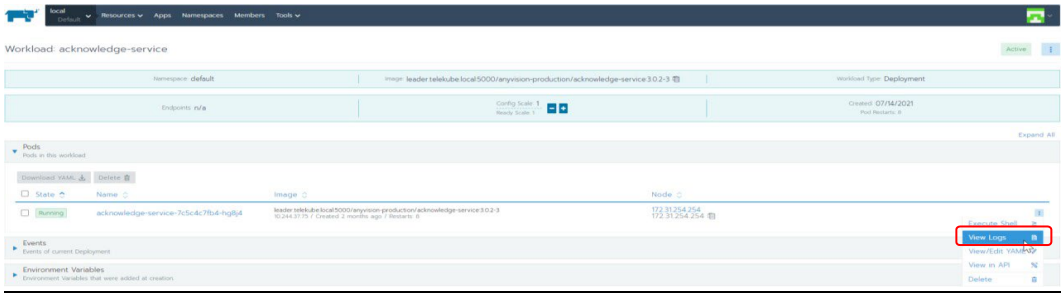
- **Open Resources → Workloads**



- **From here you can confirm that all services are up and "active".**



- **If a service is not "active" you can check the logs by clicking the service's blue link. Within the pod/service select the 3-dot menu and click "View Logs". These logs may give an indication as to why a service is down.**


Workload: acknowledge-service
Pods
Events
Environment Variables
View Logs
View in API
Delete

Terminal commands for basic troubleshooting

How to check GPU Usage - nvidia-smi

Connect to the OnWatch server over SSH to access the CLI

(command line interface) You can get a snapshot of the current

usage using the *nvidia-smi* command

```
user@DK:~/docker-compose$ nvidia-smi
Thu Aug 30 10:55:29 2018
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 390.77                 Driver Version: 390.77                    |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  TITAN X (Pascal)    Off  | 00000000:01:00.0  On |                  N/A |
| 23%   39C    P8    15W / 250W |    440MiB / 12194MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|    0      1681      G   /usr/lib/xorg/Xorg                            18MiB |
|    0      1758      G   /usr/bin/gnome-shell                          50MiB |
|    0      2486      G   /usr/lib/xorg/Xorg                           106MiB |
|    0      2639      G   /usr/bin/gnome-shell                         101MiB |
|    0      9491      C   /pyconcrete/python                           159MiB |
+-----------------------------------------------------------------------------+
```

Or if you want a steady stream, you can use the watch command - helpful when you want to monitor usage over time.

Use watch -n1 nvidia-smi

```
user@DK:~/docker-compose$ watch -n1 nvidia-smi
```

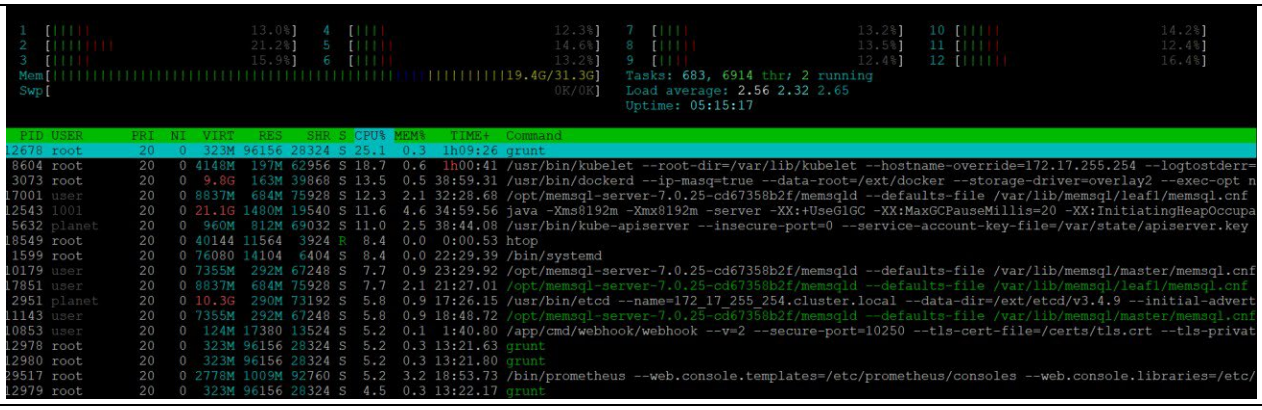
This will show you the updated usage every second.

```
Every 1.0s: nvidia-smi                                          DK: Thu Aug 30 10:56:32 2018

Thu Aug 30 10:56:32 2018
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 390.77                 Driver Version: 390.77                    |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  TITAN X (Pascal)    Off  | 00000000:01:00.0  On |                  N/A |
| 23%   39C    P8    11W / 250W |    440MiB / 12194MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|    0      1681      G   /usr/lib/xorg/Xorg                            18MiB |
|    0      1758      G   /usr/bin/gnome-shell                          50MiB |
|    0      2486      G   /usr/lib/xorg/Xorg                           106MiB |
|    0      2639      G   /usr/bin/gnome-shell                         101MiB |
|    0      9491      C   /pyconcrete/python                           159MiB |
+-----------------------------------------------------------------------------+
```

systemctl status k3s.service **- This command will let you know if your cluster is up and healthy. If there is an issue with the cluster, it will give you some indication as to what this issue may be such as "Low Disk space".**

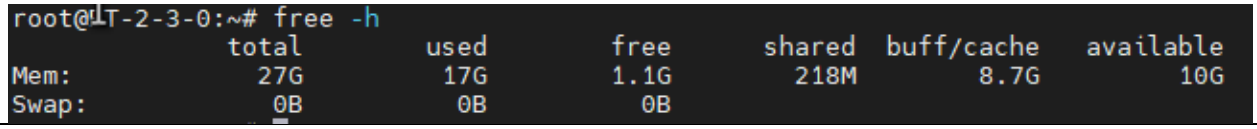
```
root@master1:~# systemctl status k3s.service
● k3s.service - Lightweight Kubernetes
     Loaded: loaded (/etc/systemd/system/k3s.service; enabled; vendor preset: e>
     Active: active (running) since Wed 2022-05-18 12:03:29 IST; 6h ago
       Docs: https://k3s.io
    Process: 1464 ExecStartPre=/sbin/modprobe br_netfilter (code=exited, status>
    Process: 1471 ExecStartPre=/sbin/modprobe overlay (code=exited, status=0/SU>
   Main PID: 1474 (k3s-server)
      Tasks: 2437
     Memory: 3.4G
     CGroup: /system.slice/k3s.service
             ├─   1474 /usr/local/bin/k3s server
             ├─   1511 containerd
             ├─   3755 containerd-shim -namespace k8s.io -workdir /var/lib/ranc>
             ├─   3777 /pause
             ├─   5532 containerd-shim -namespace k8s.io -workdir /var/lib/ranc>
             ├─   5579 /pause
             ├─   5732 containerd-shim -namespace k8s.io -workdir /var/lib/ranc>
             ├─   5838 tini -- rancher --http-listen-port=80 --https-listen-por>
             ├─   5969 rancher --http-listen-port=80 --https-listen-port=443 -->
             ├─   6086 containerd-shim -namespace k8s.io -workdir /var/lib/ranc>
             ├─   6096 containerd-shim -namespace k8s.io -workdir /var/lib/ranc>
             ├─   6132 /pause
             ├─   6133 /pause
lines 1-23...skipping...
● k3s.service - Lightweight Kubernetes
     Loaded: loaded (/etc/systemd/system/k3s.service; enabled; vendor preset: enabled)
     Active: active (running) since Wed 2022-05-18 12:03:29 IST; 6h ago
       Docs: https://k3s.io
    Process: 1464 ExecStartPre=/sbin/modprobe br_netfilter (code=exited, status=0/SUCCESS)
    Process: 1471 ExecStartPre=/sbin/modprobe overlay (code=exited, status=0/SUCCESS)
   Main PID: 1474 (k3s-server)
      Tasks: 2437
     Memory: 3.4G
     CGroup: /system.slice/k3s.service
             ├─   1474 /usr/local/bin/k3s server
             ├─   1511 containerd
             ├─   3755 containerd-shim -namespace k8s.io -workdir /var/lib/rancher/k3s/agent/containerd/io.containerd.runtime.v1.linux/k8s.io/18c01c775b8b312a083f5ba78189
b98 -a
             ├─   3777 /pause
             ├─   5532 containerd-shim -namespace k8s.io -workdir /var/lib/rancher/k3s/agent/containerd/io.containerd.runtime.v1.linux/k8s.io/ad398893c9b5b27f9f716102ceae
782 -a
             ├─   5579 /pause
             ├─   5732 containerd-shim -namespace k8s.io -workdir /var/lib/rancher/k3s/agent/containerd/io.containerd.runtime.v1.linux/k8s.io/087fb3ecd6f26335a0705467485b
7a2 -a
             ├─   5838 tini -- rancher --http-listen-port=80 --https-listen-port=443 --audit-log-path=/var/log/auditlog/rancher-api-audit.log --audit-level=0 --audit-log-
0 --a
             ├─   5969 rancher --http-listen-port=80 --https-listen-port=443 --audit-log-path=/var/log/auditlog/rancher-api-audit.log --audit-level=0 --audit-log-maxage=1
t-log
             ├─   6086 containerd-shim -namespace k8s.io -workdir /var/lib/rancher/k3s/agent/containerd/io.containerd.runtime.v1.linux/k8s.io/2938866667e1096fcd6d722580e6
945 -a
             ├─   6096 containerd-shim -namespace k8s.io -workdir /var/lib/rancher/k3s/agent/containerd/io.containerd.runtime.v1.linux/k8s.io/d93861d1b9794e639dbfa5afef69
```

htop **or** *top* **- This will show you resource usage and allow you to see if the server itself is being stressed,**
and if there is some process taking too much CPU or memory.



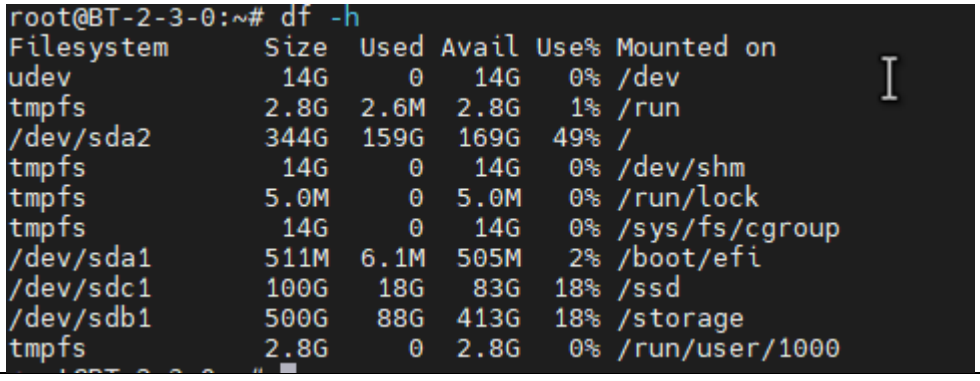
free -h **- Shows the amount of free memory available on your system**

- **Total: Total amount of memory installed in the system.**
- **Used: Total amount of memory currently in use by processes.**
- **Free: Total amount of memory that is not currently in use.**
- **Available: Cache reserved to make some processes faster but can still be used by another process.**



df -h **- Shows your current mounted drive structure and the available space for each mounted drive.**

- **Your '/' and '/ssd' should both be below 80% use.**
- **It is important to pay attention to the following mounted locations:**
- **/ (root)**
- **/ssd (Oosto database location)**
- **/storage (Oosto data storage location)**



Important Terminal Commands for analysis and Troubleshooting:

- **Check current Linux Kernel version:** *uname -r*
- **Execute a command as super-user (in case you face any permission issues while running a command:** *sudo <command...>*
- **Switch terminal to super-user (on successful execution $ prompt will change to #):** *sudo -i*
- **Check network connection status:** *ping <ip address>*

- **Check port open status:** *telnet <ip address> <port no>*
- **Check CPU usage (requires the htop package to be installed using - sudo apt install htop -y):** *htop*
- **Check CPU usage:** *top*
- **Show uptime information and load average:** *uptime*
- **RAM / Memory monitoring:** *vmstat / free -m*
- **Show disk utilization:** *df -h*
- **List all block devices (HDD/USB etc.) including empty devices:** *lsblk -a*
- **List CPU information:** *lscpu*
- **Show current directory:** *pwd*
- **List all files:** *ls -al*
- **Change directory:** *cd <path>*
- **Copy a file:** *cp <FileName> /home/user/...*
- **Copy a directory with all files:** *cp -r DirToCopy/ /home/user/...*
- **Rename a file:** *mv OldFileName NewFileName*
- **Move a file:** *mv File2Move /home/user/...*
- **Make directory / directories:** *mkdir Dir1 Dir2 Dir3*

Tools for advanced troubleshooting

Grafana Dashboard: https://grafana.anv

- **Using Grafana, you can view various performance metrics of both server and the Oosto internals.**

- **Using the "Backend" dashboard you will find some historical metrics on server performance that can help track down what may have occurred.**

Terminal commands for advanced troubleshooting

Run these commands in the terminal window within the server itself.

- *kubectl get pods* **- Lists various pods and their associated status similar to rancher.**
- *kubectl logs ${POD_NAME}* **- Displays the logs associated with the specified pod name.**
- *kubectl delete pod ${POD_NAME}* **- This will remove and restart the service with the specified pod name.**
- *ip a* **- Displays all connected network interfaces.**

  There will be several virtual interfaces for Oosto backend communications. You will want to focus on the first few physical network interfaces and make sure your

```
2: enp6s18: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether d2:87:41:50:65:1a brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.217/24 brd 192.168.1.255 scope global dynamic enp6s18
       valid_lft 3957sec preferred_lft 3957sec
    inet6 fe80::d087:41ff:fe50:651a/64 scope link
       valid_lft forever preferred_lft forever
```

  interface is listed as "up" and contains a valid IP address.

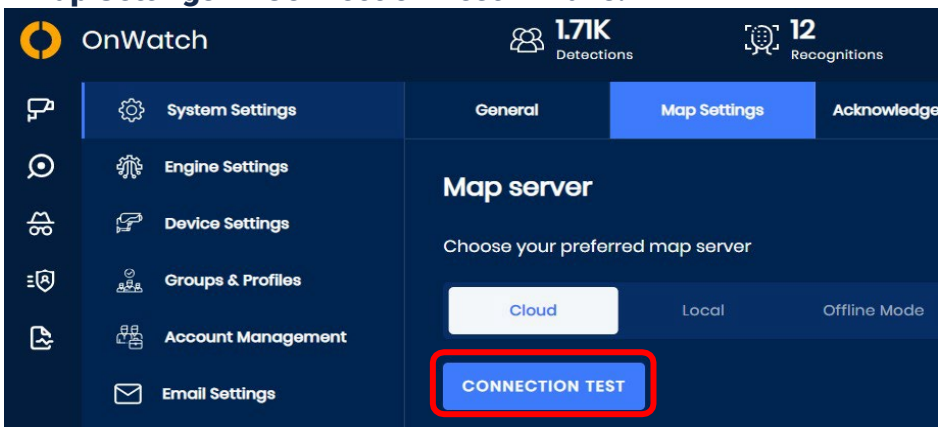
Extracting Information and Logs:

- **These are run in the terminal window within the server itself.**
- **/storage/logs - Here you will find the location of all Oosto logs including the historical logs kept after reboots or log rotation.**
- **/var/log - Here you will find the location of various Linux system logs.**
- **Searching System Logs: Some search commands to search through system logs in case of unscheduled restarts or boot issues:**
    - *journalctl -xe* **- This will show extra information for the entire latest system log.**
    - *journalctl -p 0..3* **- This will search system logs for anything classified as Emergency, Alert, Critical, or Error.**
    - *grep -i -e fail -e error -e corrupt /var/log/syslog* **- This will search system logs for lines containing the words fail, error, and corrupt.**

Get Linux disk and partition information: *lsblk*

Mapbox service is not available

Settings -> Map Settings -> Connection Test -> Fails!



<u>Troubleshooting steps:</u>

Open DEV Tools (F12) on the browser, go to Networks and see what happens Most likely there will be a timeout.

Open the terminal and check the ping to the service, domain & IP:

As root user: ping api.mapbox.com

Access map-service pod and ping the service by its domain:

```
root@Demo-V2-EMEA-1:~# kubectl exec -it map-service-96d7ff895-6b4vg bash
bash-5.0# ping api.mapbox.com
PING api.mapbox.com (13.224.191.114): 56 data bytes
64 bytes from 13.224.191.114: seq=0 ttl=239 time=0.904 ms
64 bytes from 13.224.191.114: seq=1 ttl=239 time=0.894 ms
64 bytes from 13.224.191.114: seq=2 ttl=239 time=0.921 ms
64 bytes from 13.224.191.114: seq=3 ttl=239 time=1.016 ms
```

If it works, likely you don't have an issue with the maps. If it doesn't work, ping the IP itself, Mapbox has few Ips, ping the ones you've received when trying to ping the domain.

How to Resolve?

Access configmap coredns:

kubectl -n kube-system edit configmap coredns

Put a valid DNS server:

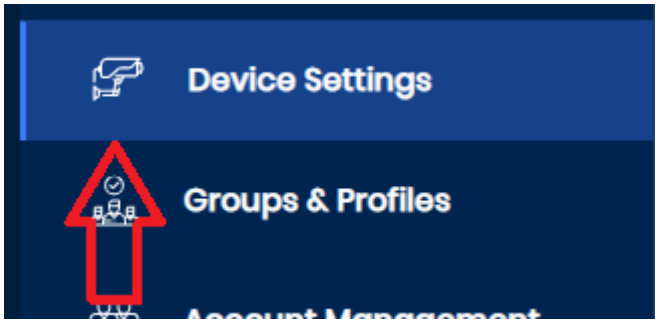
```
hosts /etc/coredns/NodeHosts {
  reload 1s
  fallthrough
}
prometheus :9153
8.8.8.8 forward . /etc/resolv.conf
cache 30
loop
reload
```

Exit and save the file by pressing *escape* and then *:wq!*

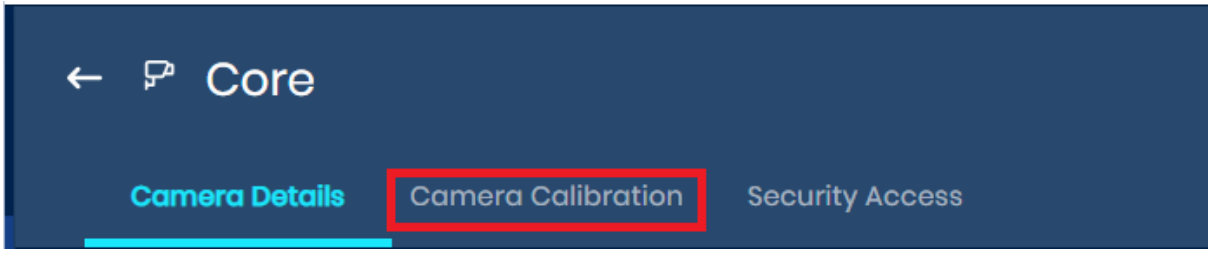
Platform

Configuring Padding on OnWatch Server

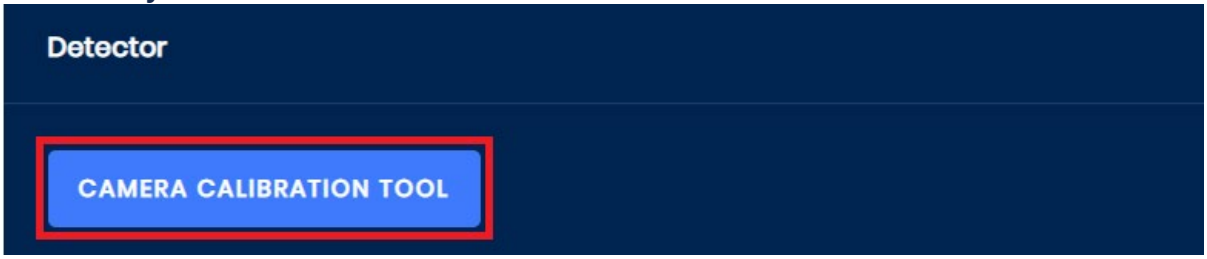
On the OnWatch UI , Select Settings, then Device settings :



Click on the Camera you would like to configure.
Inside The camera settings , select Camera Calibration:

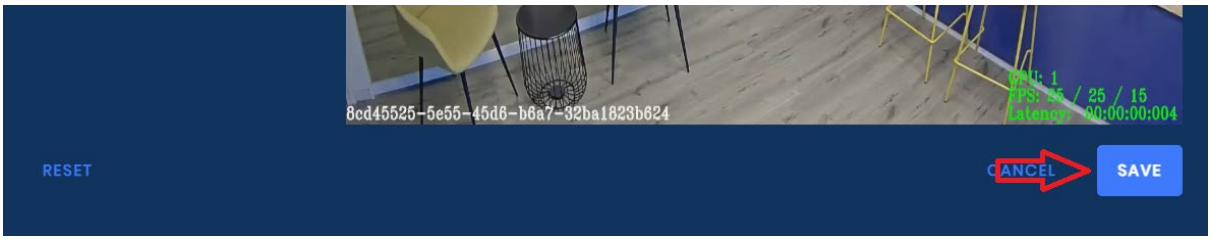


Scroll all the way down and choose the "camera calibration tool."

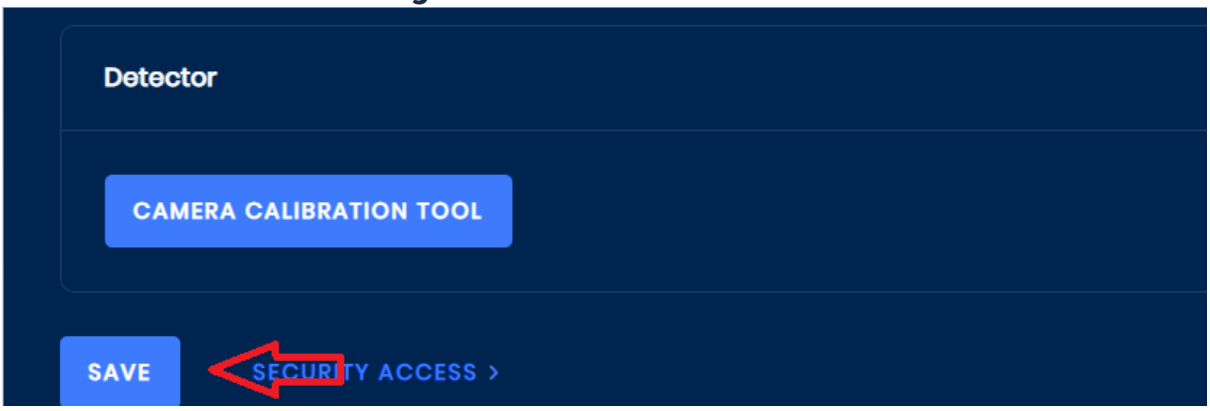


Using this tool, you can configure the pixels on the stream that the camera would focus on, with dragging a square shape across the stream, or manually updating the pixels with relevant parameters.

***Notice that to save the calibrations you have configured, you will have to save twice – Once inside the Camera Calibration tool:**



And once inside the camera settings itself:

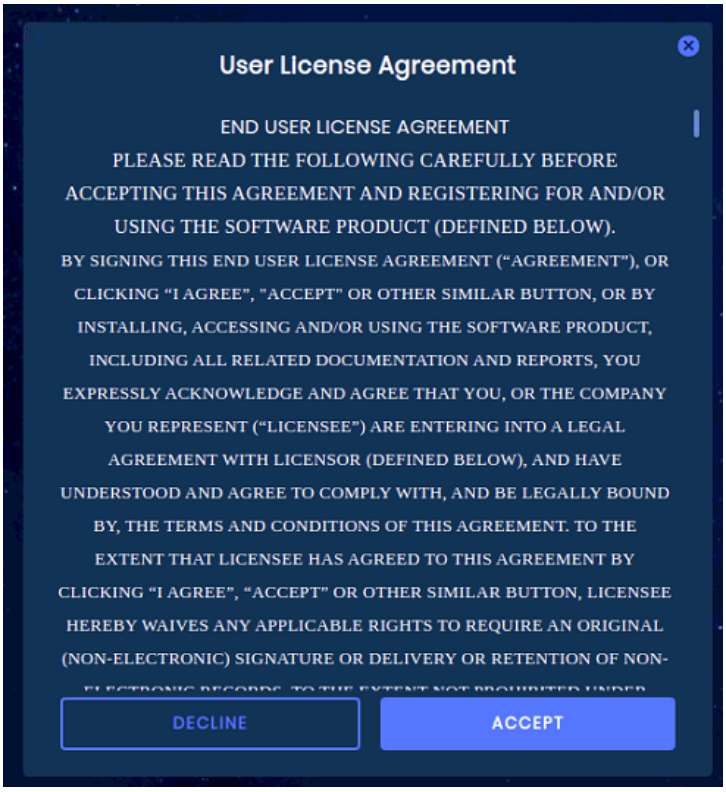


OnWatch License Activation

Access License panel:

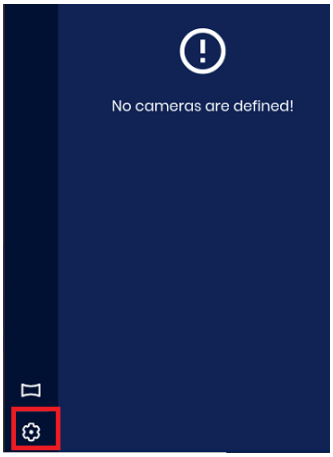
- **Login to OnWatch**
- **Accept EULA message:**

At first login, need to accept the user license agreement:



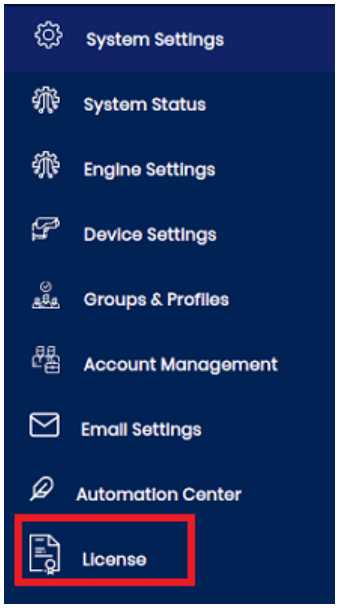
- **Access to Settings:**

  At the bottom left corner, click on the settings icon:

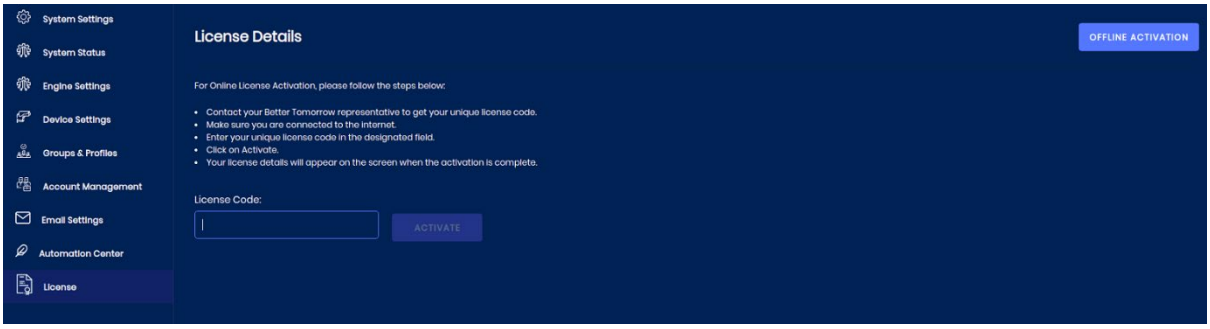


- **Access License Settings:**

On the left side menu click on “License”:

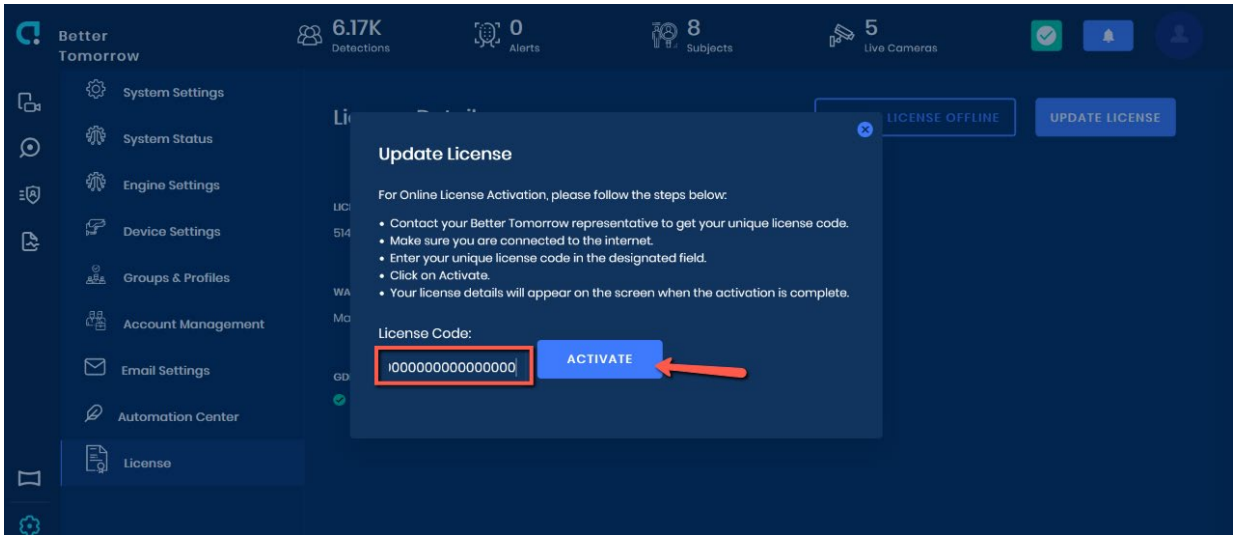


⇨ **This is how the license screen will look like:**



Online Activation

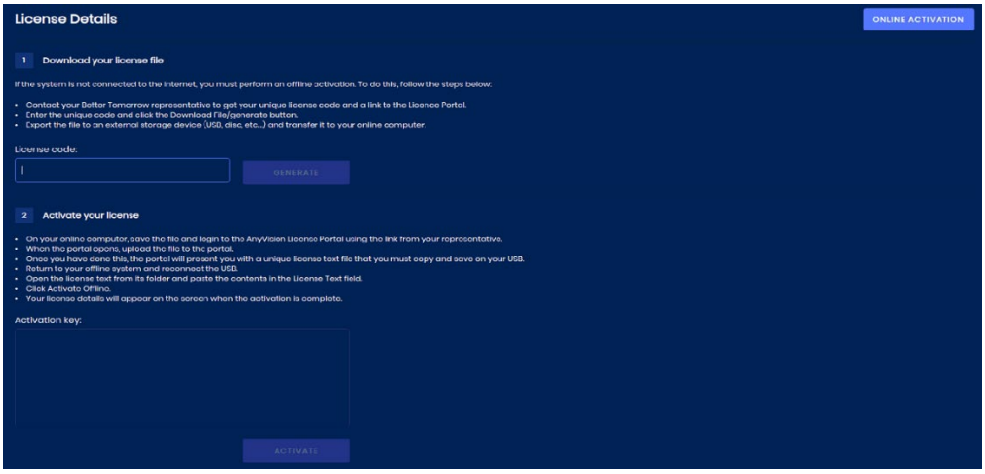
- **On the License Screen, enter the License Code you received from Oosto Support and paste it in the box, then click Activate**



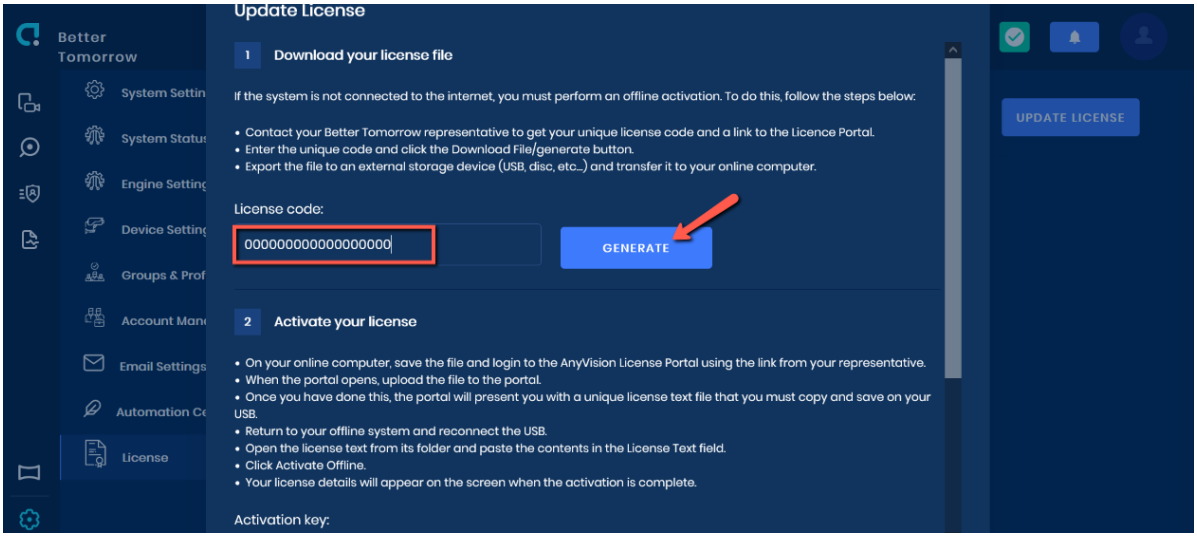
- **Congratulations, your license is now activated!**

Offline Activation

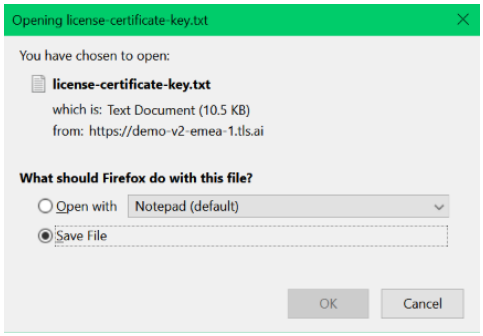
- **To perform an offline activation, click on the "OFFLINE ACTIVATION" button on the top right side:**



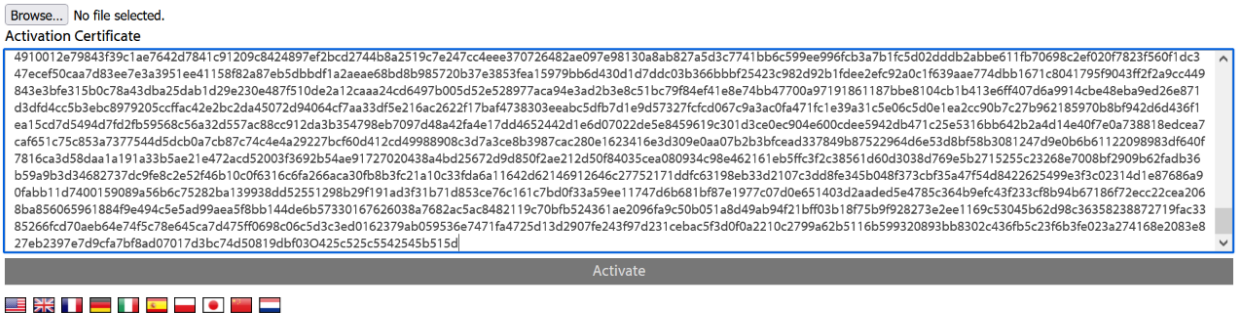
- **Paste the License Code you received from Oosto support, then click on GENERATE.**



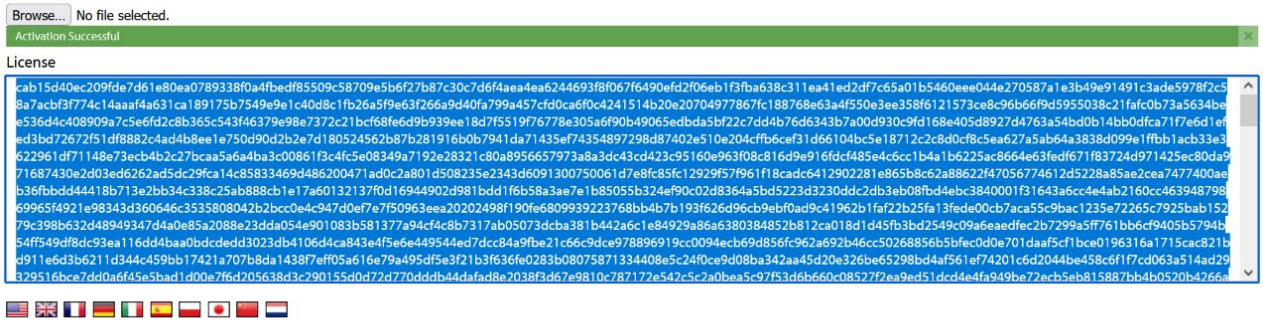
- **Once you do, a pop-up will appear. Save or open the text file:**



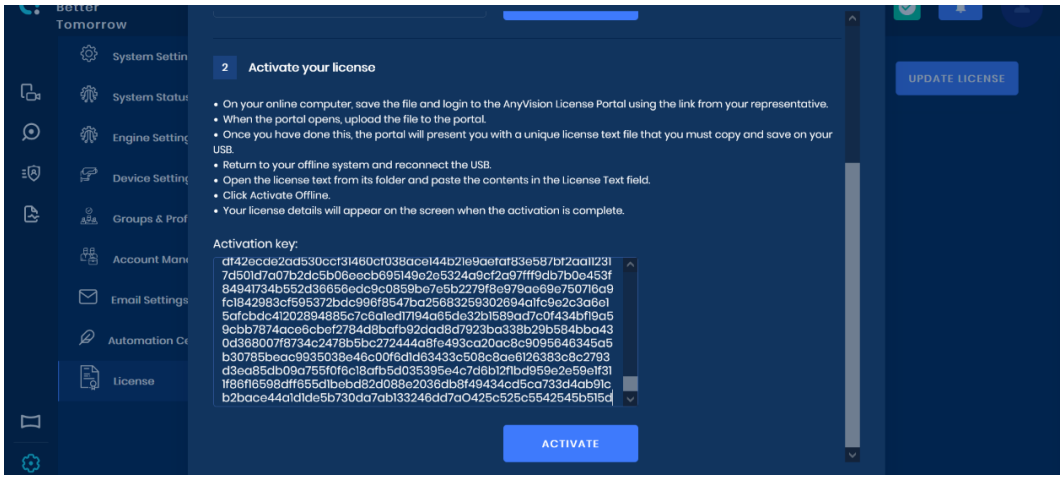
- **Copy the content of the text file, then navigate to http://www.activationportal.me/selfservice/activation.aspx?Type=1&cid=7362&pid=8592**
- **Once there, paste the content of the text file in the box as showed below, then click on activate:**



- **You will see the sentence Activation successful on the top. Copy the text that came out, as shown below:**



- **Go back to the OnWatch UI, where you previously generated the text file, and paste the content under Activation Key, then click Activate:**



Congratulations, your license is now successfully activated!

Latency in detections

Case – Incoming detections appears with significant delay.

First, make sure there is a connection to the camera – and that the connection is stable enough to create valid detections.

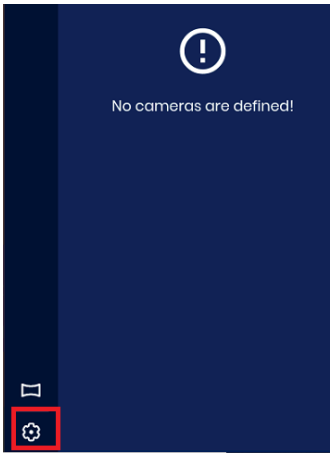
How to check – Apply the following ping command to validate the connection:

ping -c4 <CAMERA IP>
Example for camera with ip 10.1.50.58: *ping -c4 10.1.50.58*

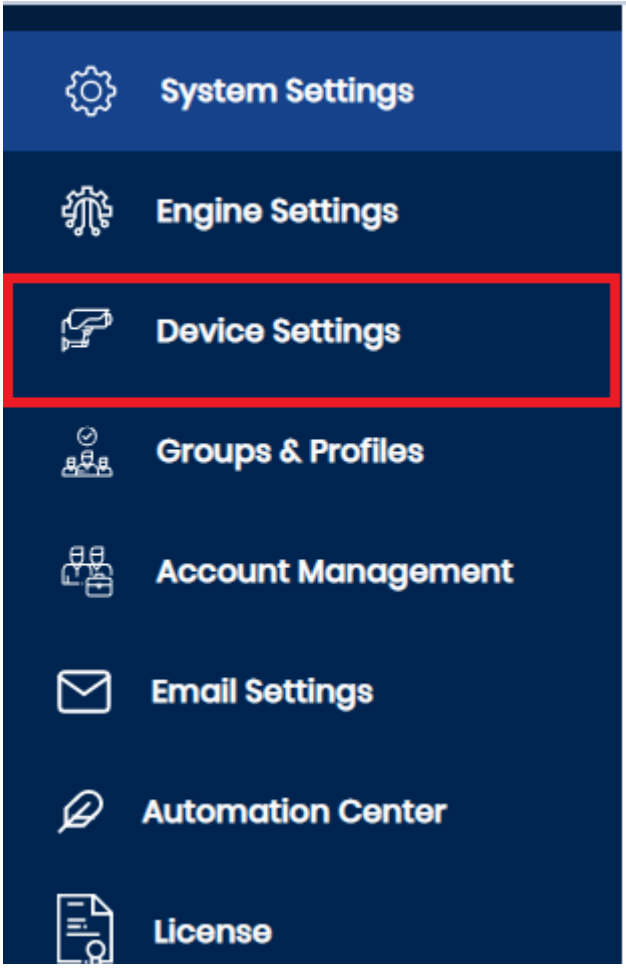
```
root@master1:~# ping -c 4 10.1.50.58
PING 10.1.50.58 (10.1.50.58) 56(84) bytes of data.
64 bytes from 10.1.50.58: icmp_seq=1 ttl=63 time=2.21 ms
64 bytes from 10.1.50.58: icmp_seq=2 ttl=63 time=1.63 ms
64 bytes from 10.1.50.58: icmp_seq=3 ttl=63 time=0.670 ms
64 bytes from 10.1.50.58: icmp_seq=4 ttl=63 time=0.451 ms

--- 10.1.50.58 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 0.451/1.240/2.212/0.714 ms
```

As long as the timer does not reach 10MS or higher (As marked with the yellow brackets).
If the timer is reasonable, the issue should be on the camera settings , inside the UI.

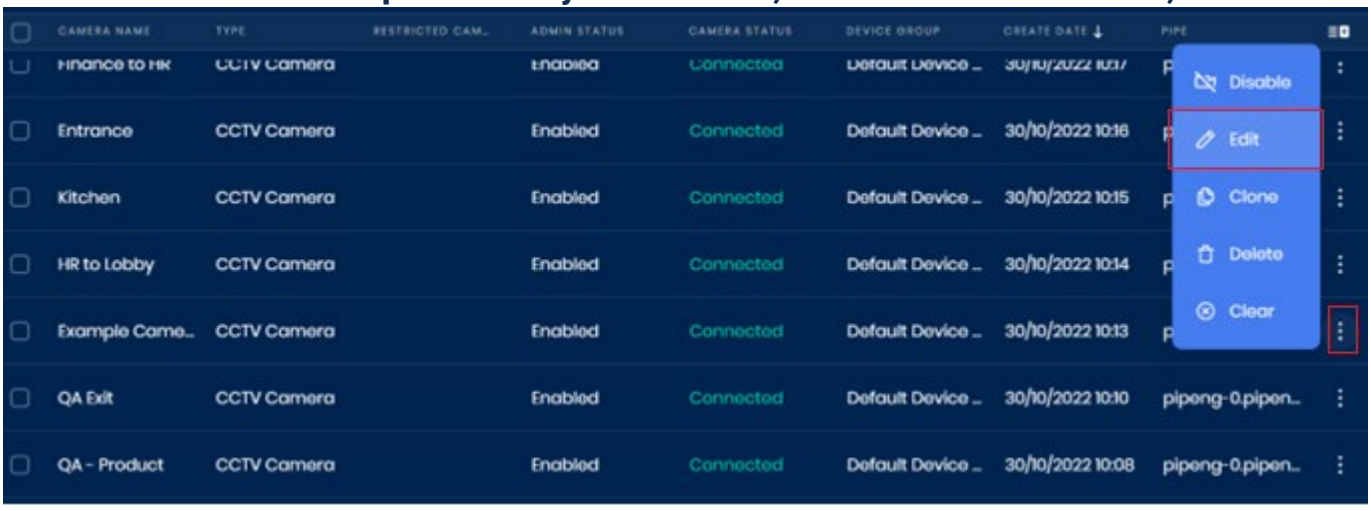
Log in to the OnWatch UI.
Access to Settings:
At the bottom left corner, click on the settings icon:



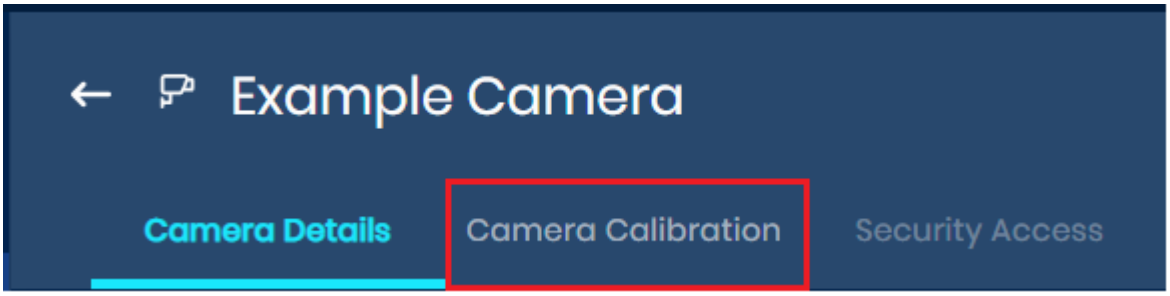
Choose the Device Settings



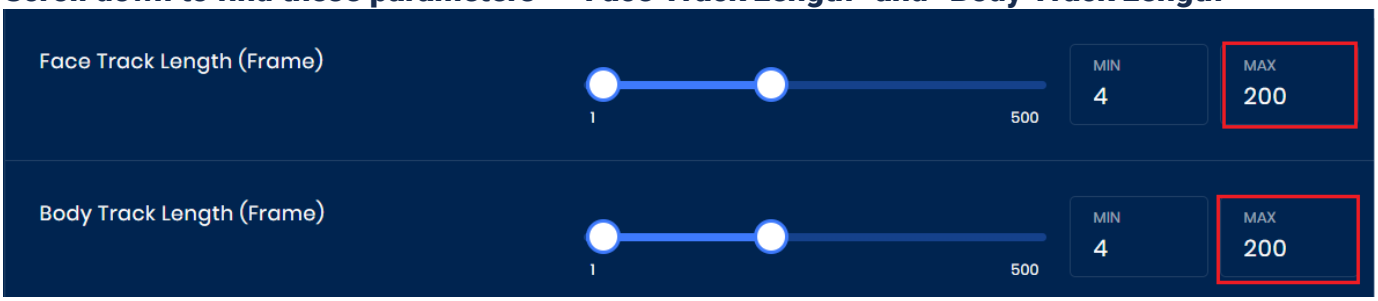
Choose the camera which provide delayed detections, click on the 3 dots on side, and "edit":



Proceed to "Camera Calibration"



Scroll down to find these parameters – “Face Track Length” and ”Body Track Length”



Change the MAX parameter to half of the current value. In my case it would be to “100”.

Scroll down and Save the settings.

These parameters reduce \ increase the time it take to send detection, and may solve the issue.

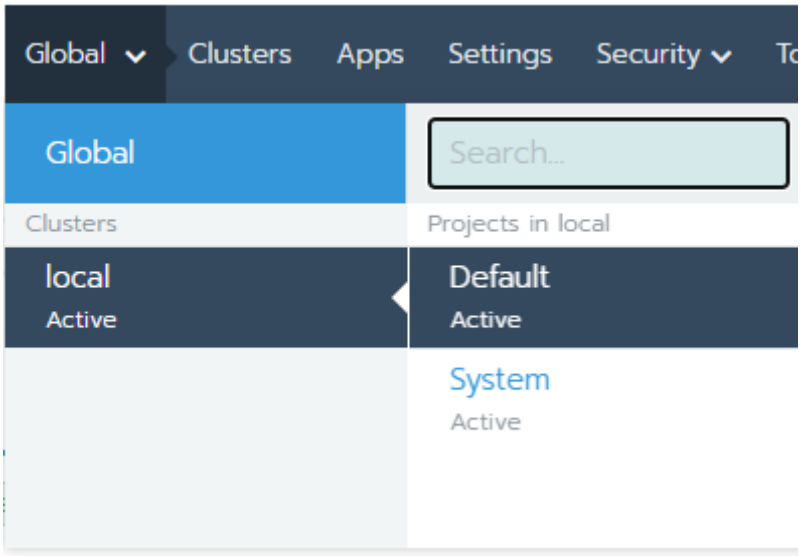
After applying those changes , the delays should reduce on each detection from the selected camera. Apply these changes to every camera with this issue.

Infrastructure

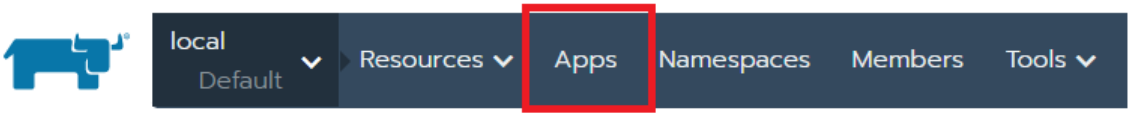
Removing and redeploying the software layers

This method is a drastic measure, to be applied when system failure occurs.
To begin, open the rancher using the onwatch's ip server and adding :9443
To the URL as follow : https://<IP>:9443 , for example https://10.1.20.20:9443

Inside The Rancher – Put the cursor on "Global" , then "local", then click "default" like so:



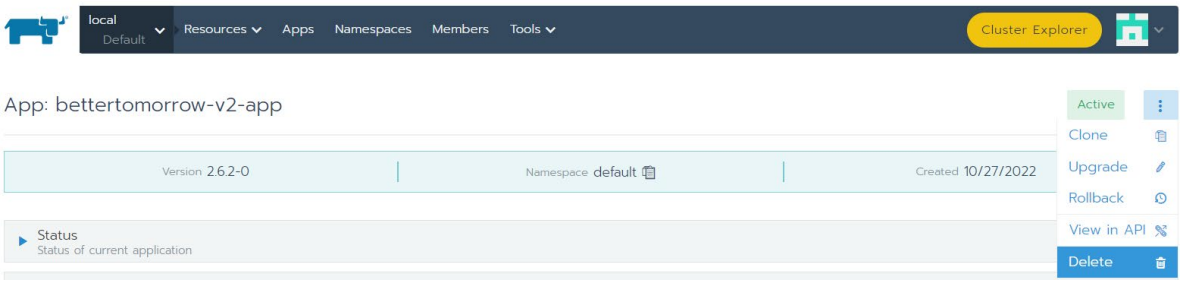
Now click on "Apps"



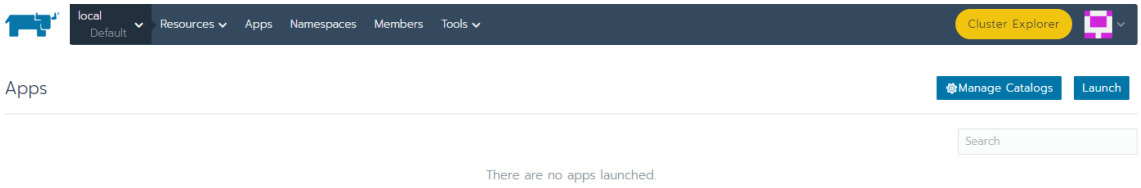
On this page you should see 3 or 4 layers.
To remove and redeploy then continue as follows:

First, the removal and the redeployment will be configured with the same specific order – the removal will be first removing App, init , data. The Redeploy will be data, init, App.
Another thing to bear in mind is that only when a layer finishes deployment/removal, we can proceed to the next layer.

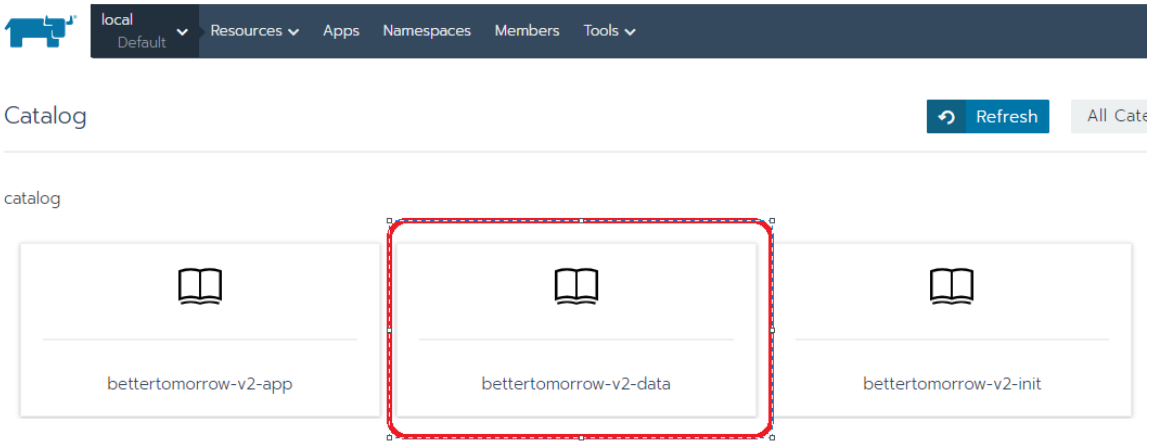
<u>How To Remove Layers:</u>
Starting With the App layer, press on the name of the layer you would like to remove. Then click on the 3 dots sign on the right side of the page, and select "delete" as follows:



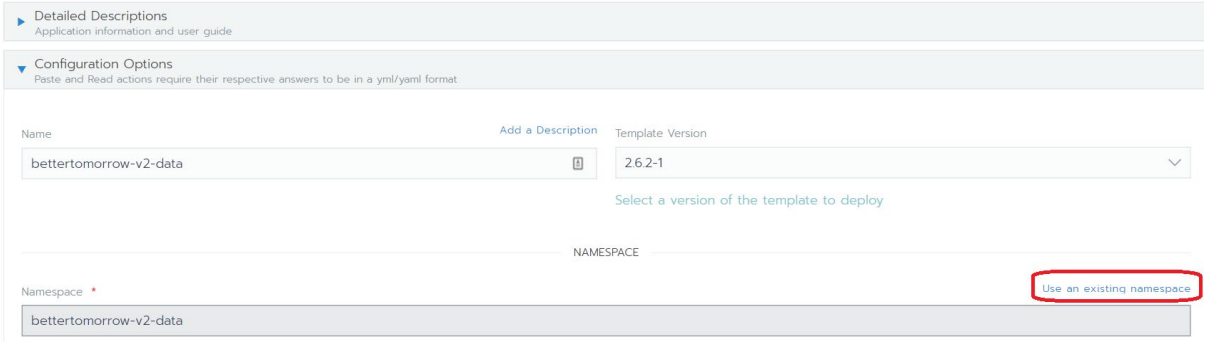
Only when you can't see the layer you deleted anymore, you can proceed to the next one. (By Order – APP →INIT → DATA)
After deleting all the layers – you should see the Apps page as follows:



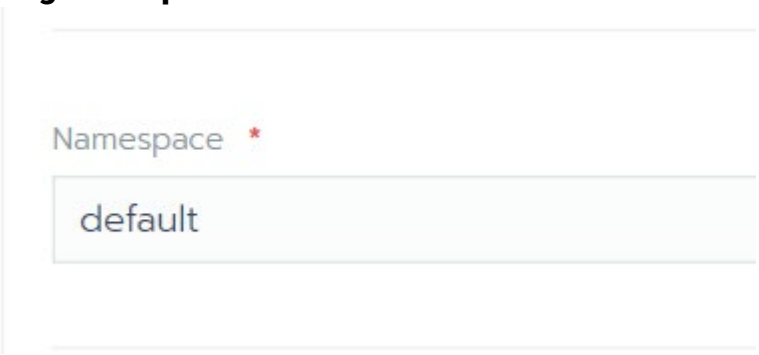
How To Redeploy the layers:
First, click on "Launch", and you will get to select from the 3 layers you have deleted.
Start With pressing on the Data layer:



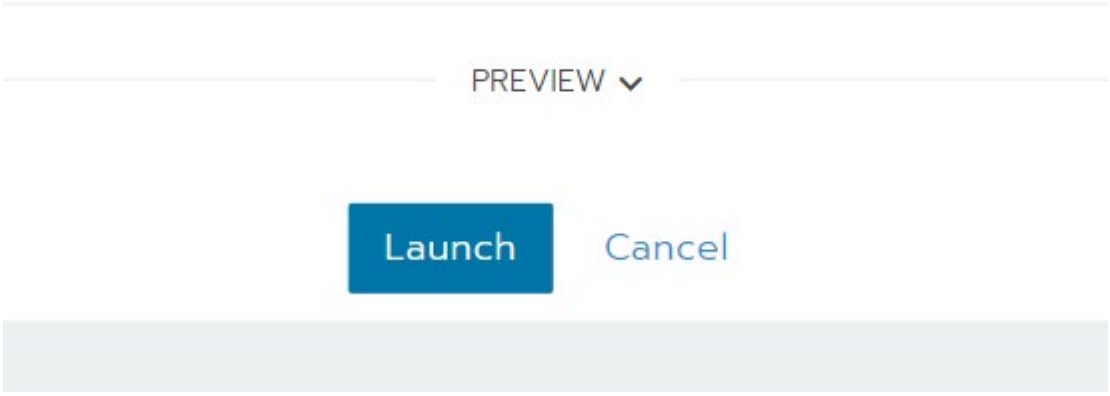
Inside the Layer, there is a section regarding namespace –



Press on " use an existing namespace" and select "default" as follows:



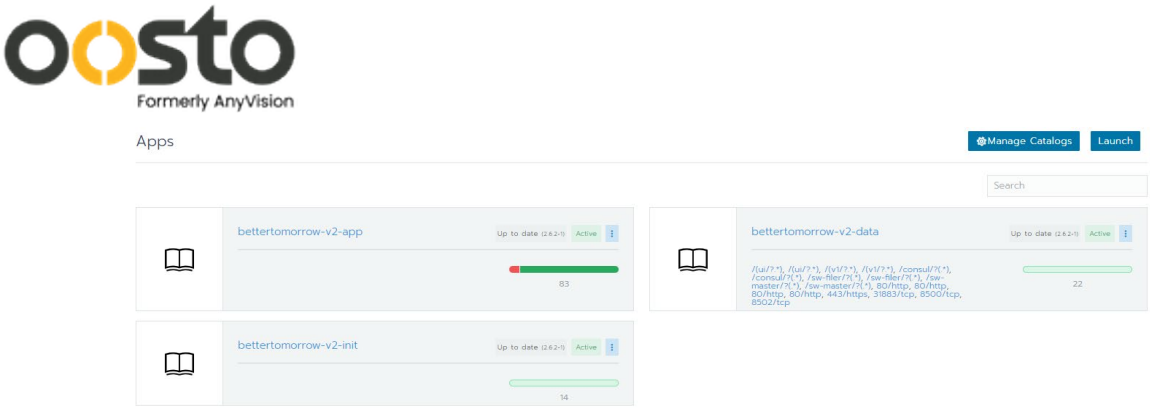
Now Scroll all the way down and select "Launch."



After launched, the layer will need a couple of minutes to deploy itself, and become "Active"

Once the layer is Active, you may proceed to the next layer. (A reminder – first Data layer → init layer → App Layer)
Don't forget to apply the same steps to every layer.

Final results example:



Kernel Hold

Notice that you use this method to lock the Correct kernel version, and not just any version.
Correct Versions:
On Ubuntu Desktop – 5.11.0-27-generic
On Ubuntu Server – 5.4.0-81 Generic
To identify your kernel version, apply the following command:

uname -r

After validating the kernel version, lock it:

apt-mark hold $(uname -r)
apt remove unattended-upgrades -y
cat > /etc/apt/preferences <<EOF
*Package: linux-image-**
*Pin: release **
Pin-Priority: -1

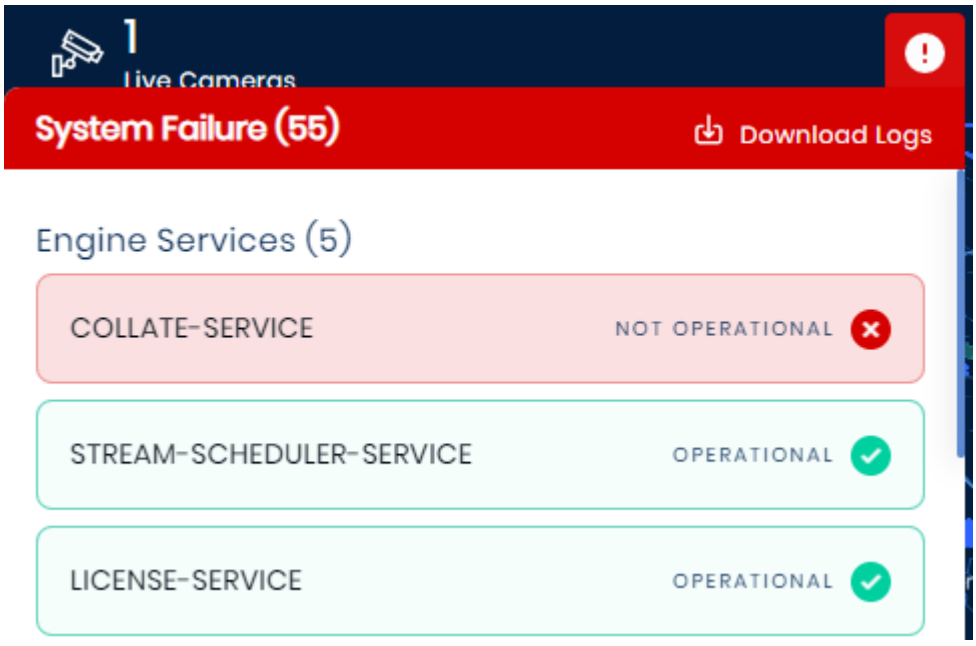
*Package: linux-headers-**
*Pin: release **
Pin-Priority: -1
EOF

Logs Inspection

How to determine if there's something wrong with the system, and how to determine the cause?
The health-Check feature on the UI is the solution.

If this icon appears on the top right part of the UI: 
The system works and everything is up.

If this icon appears instead : 
It means that there is an issue with a/some services.
Clicking on the icon will reveal all the services and their status and will mark the problematic ones. For Example:



On This example the problematic service is "Collate Service".
To inspect further into the issue, connect via ssh or directly to the onwatch server.

How to view the logs of the issued service?
-From the OnWatch's server terminal – obtain admin access (sudo su -)
-Use the following command to obtain the full name of the service you would like to inspect :

kubectl get po -A | grep "<SERVICE NAME>"
Example:

```
root@master1:~# kubectl get po -A | grep collate
default                 collate-service-8467f8846f-q9kwr
            1/1     Running           0          7m13s
default                 collate-service-init-pkmfw
            0/1     Completed         0          28h
```

We can see In this case we have more than one result, in cases like this, always choose the option without the "init" in the name, in my case – the first service.

-To View the logs of the service, use the following command:

***The -f is not mandatory , it just will make the logs to keep being reported live, until you press CONTROL + C to stop it**
****The -n declares the namespace of the service, as it needs to be stated in the command. You can see at the beginning of the collate service name the "default", it may very between services.**

kubectl logs -n <NAMESPACE> -f <SERVICE NAME>

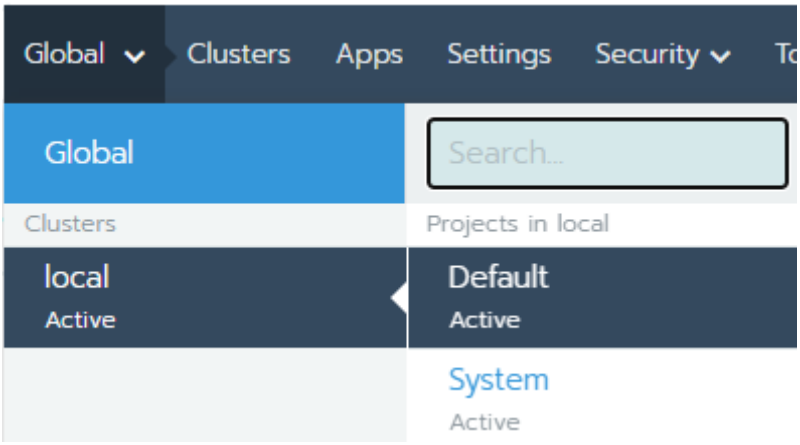
Sometimes you will not receive logs from a service, as it is down. In this case you need to pull the logs of the pods itself, instead of the service. Use the following command to do so(The pod name will be similar to the service name):

kubectl describe pod -n <NAMESPACE> <POD NAME>

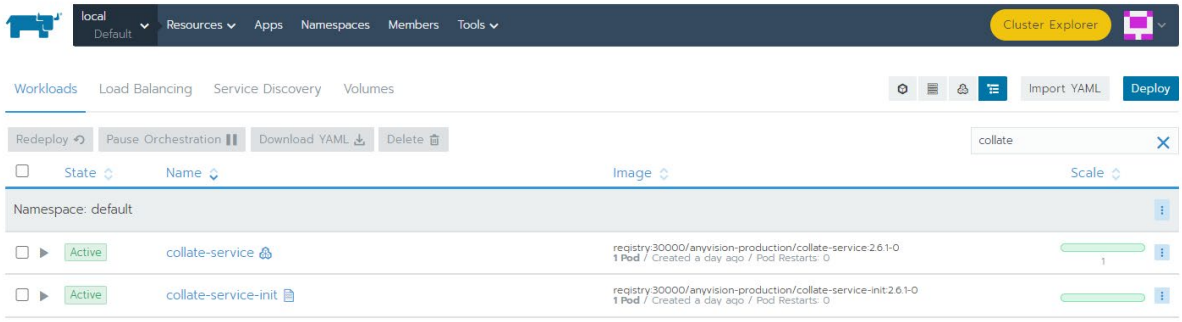
View Logs Using the Rancher:

Access the Rancher of the said OnWatch Server by applying the server's ip in the following URL:

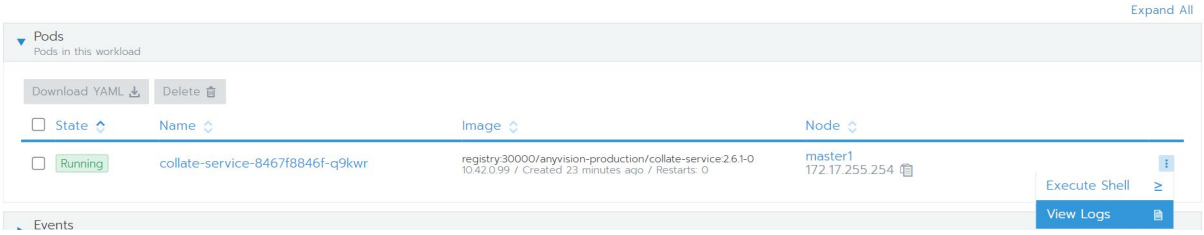
https://<SERVER_IP>:9443 , Username and Password will always be "admin".
Inside The Rancher – Put the cursor on "Global" , then "local", then click "default" like so :



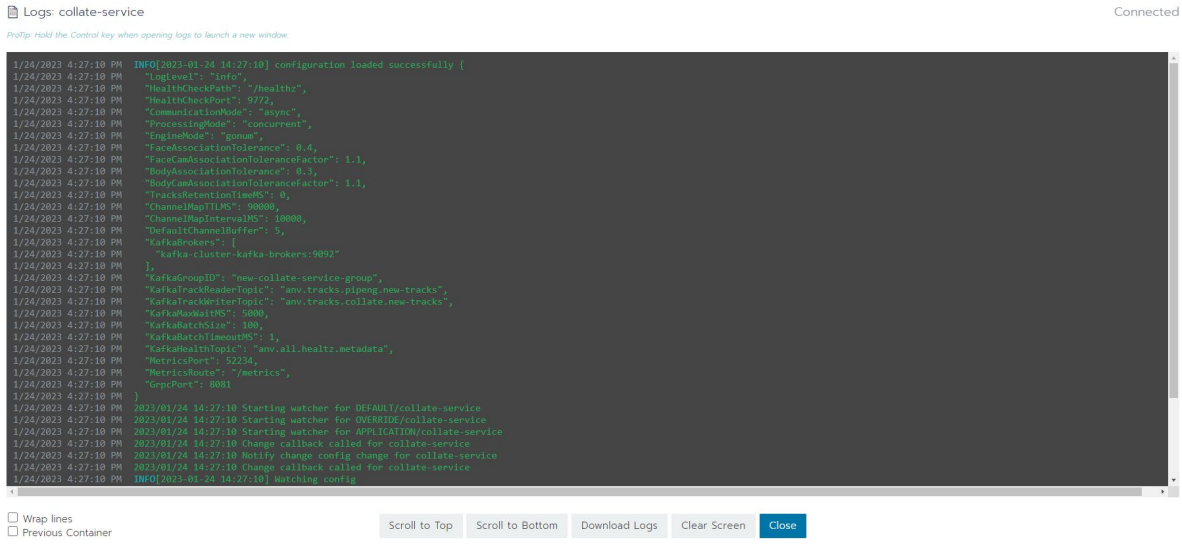
On the right side of the page, there is a search bar, search there for the wanted service, for example a search for collate service:



Next click on the service name you would like to inspect, there you will see the full name of the service. Select the three dots at "Pods" Section And choose "View Logs". For Example:



Example Logs for service :



Guide for “Live Updates are currently down.”

When you encounter the following report:

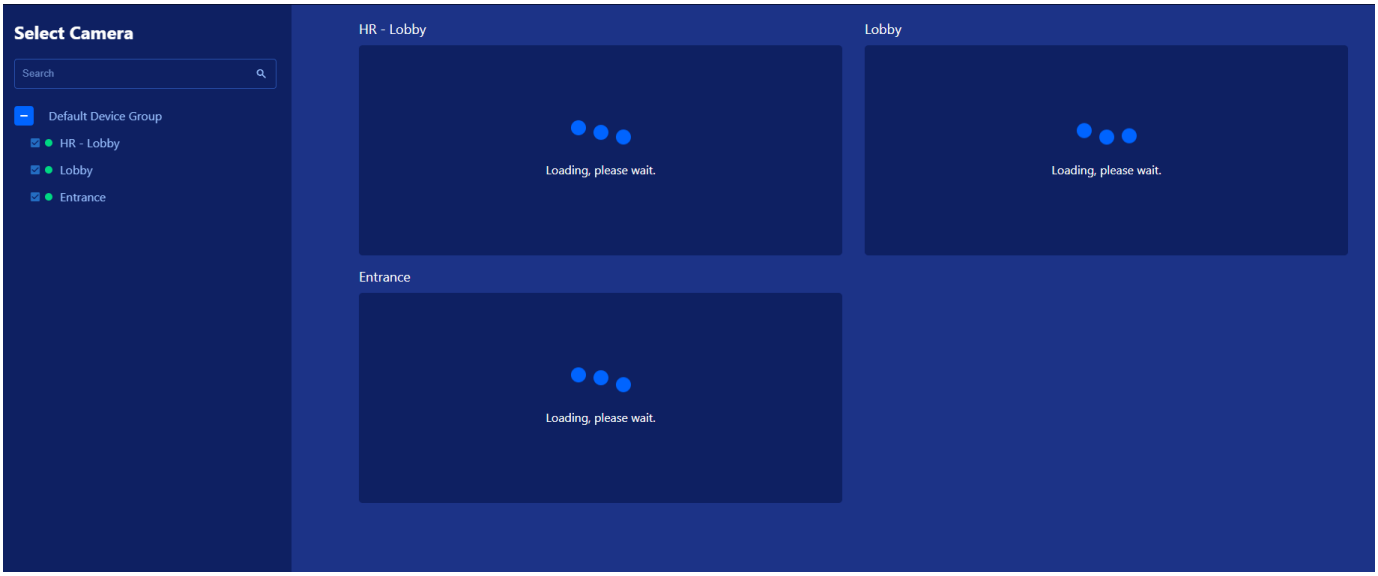
This mostly means that something in the network configuration has changed, causing a confusion in the system.
A quick solution could be deleting the services that were affected by those changes, with the following set of commands:

kubectl delete pod $(kubectl get po -A | grep node-rtsp-s | awk '{print $2}')
kubectl delete pod $(kubectl get po -A | grep pipe-manager | awk '{print $2}')
kubectl delete pod $(kubectl get po -A | grep redis | awk '{print $2}')
kubectl delete pod $(kubectl get po -A | grep socket-service | awk '{print $2}')

From the moment this solution is applied , it could take up to 10 minutes for the OnWatch application to go up.

Video wall camera stuck on Loading state

Case - Live video display : camera connected to the system. video does not come => make a clone of the camera then video comes.



Solution :

Create the script that will fix this issue.

```
sudo su -
cd /root
nano videowallfix.sh
```

An empty text file will open, fill it with the following script by copying and pasting.

```
#!/bin/bash
kubectl exec -it redis-master-0 -- bash -c 'redis-cli DEL $(redis-cli KEYS "*" | grep "node-rtsp")'
kubectl delete pod $(kubectl get pods | grep node-rtsp-service | awk '{print $1}')
sleep 10
```

Save the script by pressing CONTROL + S, and exit by pressing CONTROL + X
Allow execute permission:

```
chmod +x videowallfix.sh
```

- **Place This script in crontab to run it every day at 00:25 (you can also confirm with the customer for the best time to implement this solution and setup the cron time accordingly)**

  crontab -e

- **Notice that if this is the first time this system uses crontab it will prompt you to choose between 3 text editors, choose 1.**

```
dave@howtogeek:~$ crontab -e
no crontab for dave - using an empty one

Select an editor.  To change later, run 'select-editor'.
  1. /bin/nano        <---- easiest
  2. /usr/bin/vim.tiny
  3. /bin/ed

Choose 1-3 [1]:
```

- **Upon choosing an option the next text file will appear, it will also appear if an option already chosen before this procedure**

```
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
```

- **Copy & paste the following line to the END of this file.**

*25 0 * * * /root/videowallfix.sh*

- **Save & exit again by pressing CONTROL +S, followed by CONTROL +X**
-
- **Run the script:**

  ./videowallfix.sh & disown
  (This will start the script in the background and detach it from the shell).

Error: Failed to get EULA

Follow this procedure if you encounter a blank EULA when a user logs in.

- **Go to https://<server_ip>/sw-filer**
  Example : https://10.1.70.73/sw-filer
- **Credentials are admin/Passw0rd123**
- **Navigate to buckets > static > eula**

- **The en_us.html file may be there, but if you click on it, you will see a blank screen.**
- **Download the html file from our Drive:**

  https://drive.google.com/drive/folders/1aseOmDpVFSmDfvdzR431h3ibZihJvbUt?usp=share_link

- **Click the Upload button and upload file.**
- **There is no need to restart any services. You will now see the EULA when logging in for the first time.**

Reset Rancher Password

- **Gain admin access:**
  sudo su –
- **Enter the Rancher pod:**

kubectl exec -it -n cattle-system $(kubectl get pods -A | grep rancher | grep -v webhook | grep -v operator | awk '{print $2}') bash

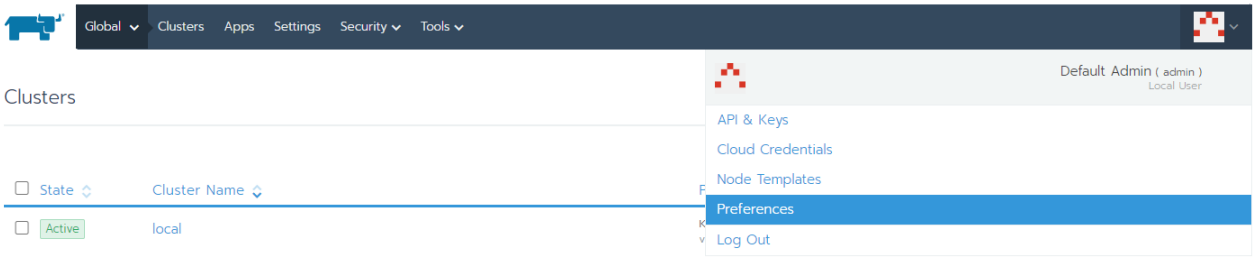
- **Reset the password (inside the pod's bash)**

reset-password

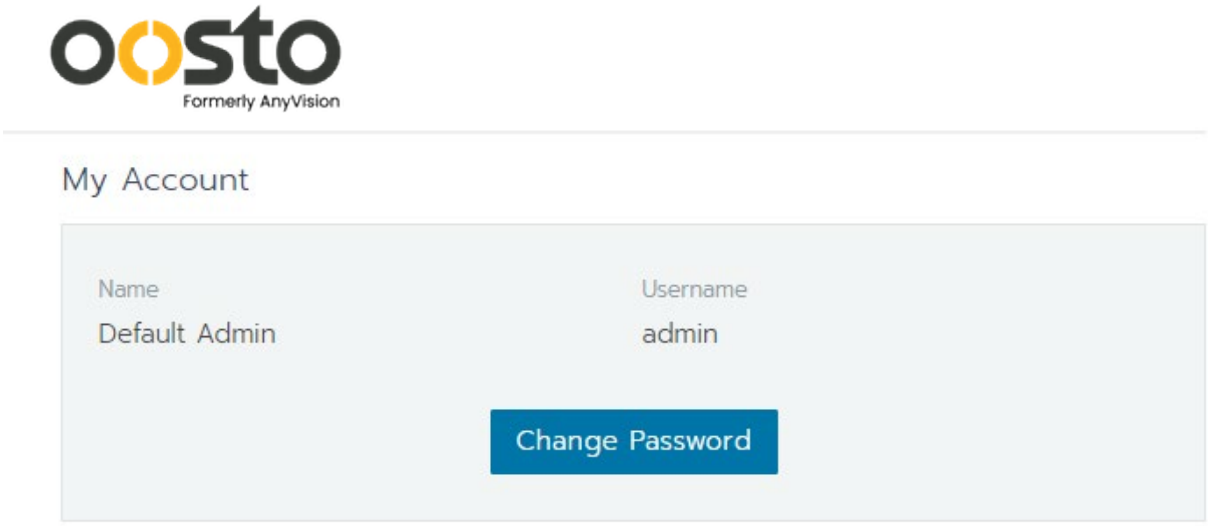
- **You will now get a new hash as a password**

```
root@master1:~# kubectl exec -it -n cattle-system $(kubectl get pods -A | grep r
ancher | grep -v webhook | grep -v operator | awk '{print $2}') bash
kubectl exec [POD] [COMMAND] is DEPRECATED and will be removed in a future versi
on. Use kubectl exec [POD] -- [COMMAND] instead.
root@rancher-7d6c96d844-w4zn4:/var/lib/rancher# reset-password
W0129 16:31:15.659659     754 client_config.go:608] Neither --kubeconfig nor --m
aster was specified.  Using the inClusterConfig.  This might not work.
New password for default admin user (user-gfqkj):
wErJ2cMhVzCL5i6--O64
```

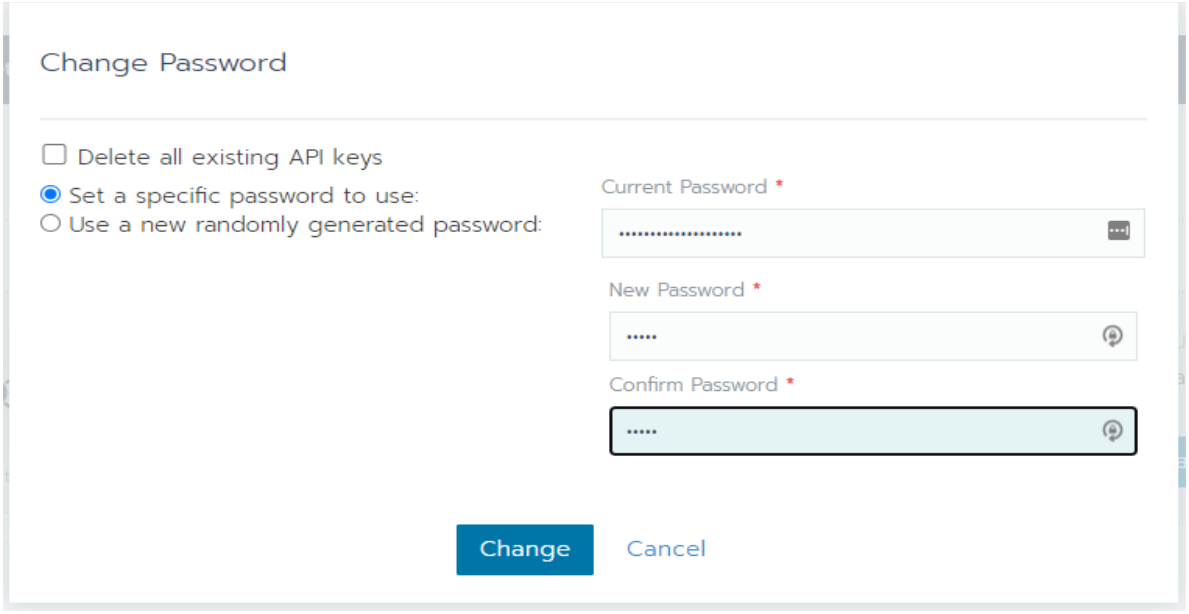
- **Login with this hash as the new password.**
- **On the top right in the rancher, click the account**



- **Preferences → change password**



- **Use the hash to set the new password**



- **You are all set! Rancher password has changed.**

Reset OnWatch UI Login Credentials

- **The following procedure will describe how to reset the password of the admin users of OnWatch.**
- **Access to Postgres anv_db**

kubectl exec -it $(kubectl get po | grep postgres | grep -v "abraxas\|sea\|better" | awk '{print $1}' | sort | head -1) bash
pgcli anv_db

- **For OnWatch (version 2.5.0 and higher):**

  username: Administrator

  Password: pa$$word!

  update users set password='$2a$10$Emp2bxdvFwqV0zK7Gvp2QOKOWEwzql6wNzZkQUmovYqM1wbJi7X5G' where username='Administrator'

- **For OnWatch (version 2.4.0 and lower):**

  username: AnyVisionAdmin

  Password: AVpa$$word!

  update users set password='$2b$10$Zxs1/l/dQS/ZbUJFrcdmr.WDiJ6vvT02cScQJrLAg41Obf27JQ0N6' where username='AnyVisionAdmin'

Username and password have been reset!

Network

k3s Changed IP in 2.6.6

- **Case - system was installed without advertise IP, and then the IP had changes.**

Identification:

- **Identify the OLD IP the system was installed with:**

  systemctl status k3s

- **You will see a line that looks like that, the RED IP is the old one:**

  abr 05 09:47:56 master1 k3s[4803]: time="2022-04-05T09:47:56.236327477+02:00" level=info msg="Failed to test data store connection: this server is a not a member of the etcd cluster. Found [master1-c684b123=https://192.168.2.10:2380], expect: master1-c684b123=192.168.2.15"

Create nic:

- **Use the 2.4. create_nic.sh script with the old IP**

  ./create_nic.sh <OLD_IP>

- **If you can't find the create_nic.sh script, you can use it as a code block:**

  #!/bin/bash

  DUMMY_IP=$1

  /sbin/ip link add dummy0 type dummy

  /sbin/ip addr add $DUMMY_IP/32 dev dummy0

  /sbin/ip link set dummy0 up

- **Edit k3s Service:**

edit k3s service /etc/systemd/system/k3s.service

- **include the following: '--node-ip YOUR_PREVIOUS_IP' , As in the following example:**

```
[Service]
Type=notify
ExecStartPre=-/sbin/modprobe br_netfilter
ExecStartPre=-/sbin/modprobe overlay
ExecStart=/usr/local/bin/k3s server   --cluster-init   --disable='servicelb,traefi
belet-arg 'max-pods=150' --node-name master1 --node-ip 192.168.2.10 --resolv-conf
llMode=process
```

From:

ExecStart=/usr/local/bin/k3s server --cluster-init --disable='servicelb,traefik,local-storage,metrics-server' --kubelet-arg 'max-pods=150' --node-name master1 --resolv-conf /run/systemd/resolve/resolv.conf KillMode=process

To:

ExecStart=/usr/local/bin/k3s server --cluster-init --disable='servicelb,traefik,local-storage,metrics-server' --kubelet-arg 'max-pods=150' --node-name master1 --node-ip 192.168.2.10 --resolv-conf /run/systemd/resolve/resolv.conf KillMode=process

- **Services Restarts:**

systemctl daemon-reload

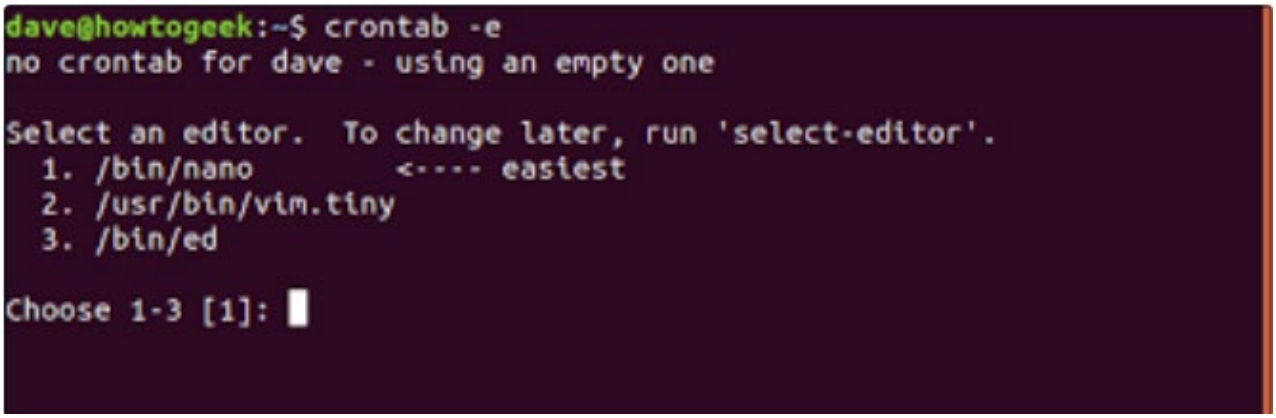
systemctl restart k3s

- **Auto load on Reboot:**

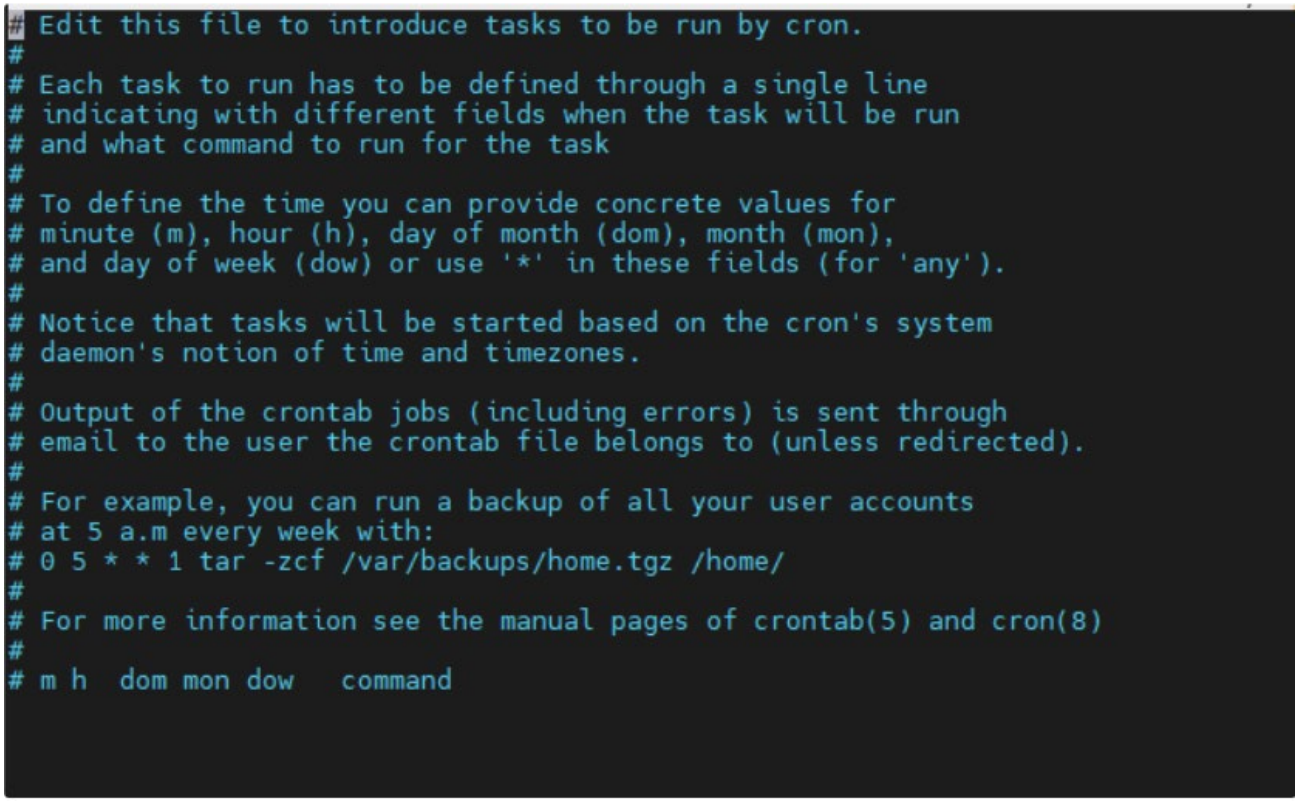
Add cronjob so that the advertise IP (old IP) will be reobtained every server restart.

crontab -e

Notice that if this is the first time this system uses crontab it will prompt you to choose between 3 text editors, choose 1.



- **After choosing an option the next text file will appear, it will also appear if an option already chosen before.**



- **Copy & paste the following line to the END of this file.**

@reboot ./create_nic.sh OLD_IP

Configuring Static IP

Static IP = permanent IP address for the machine

To set the ip to static, there are 3/4 parameters you need beforehand:

IP address, Default Gateway address, subnet mask(numeric form – for example "24") and not mandatory but crucial is DNS server IP.

Another relevant parameter is the name of the Network Interface Card the machine is using (NIC) which can be found with the following command:

ip r | head -n 1 | awk '{print $5}'
Usage Example:

```
user@master1:~$ ip r | head -n 1 | awk '{print $5}'
eno1
```

First – create a copy of the netplan you currently have for backup:

sudo su – (**apply admin password**)
netfile="/etc/netplan/"$(ls /etc/netplan/ | awk '{print $1}')
touch /etc/netplan/netplan_backup.yaml
cat $netfile > /etc/netplan/netplan_backup.bu

Next – Edit the Network configuration file:

(Copy and paste the following paragraph as it is one command, from the "echo", until the "$netfile")

If you cannot copy and paste this command, It is recommended to get this format on Portable storage device, and transfer. Trying to write this file by yourself may cause a lot of indentation errors.

echo "network:
version: 2
renderer: networkd
ethernets:

```
    nic
        addresses:
          - ip/subnetmask
        nameservers:
          addresses: [8.8.8.8, dns]
        gateway4: default_gateway
" > $netfile
```

nano $netfile

After the nano command, you will get to edit the new network configuration file. Change the "nic" to the NIC you discovered earlier, "ip", "default_gateway", "subnetmask" with the relevant values.

Also change "dns" to the new dns server ip (If DNS is not necessary, just remove the dns and the sign leaving this line to be:

"addresses: [8.8.8.8]"

***Be Careful with editing this file, as its dictation is very fragile. A misplaced tab or space could damage the file, leaving the network unconnected.**

Once you are done editing the file, save the file with CONTROL+S and exit with CONTROL + X.

Now, Apply the changes:

netplan apply
ip r

Using the ip r command should show you the new network configuration on the machine.

If this method has caused the machine to lose its internet connection, please make sure the netplan file was configured as instructed, and the addresses are accurate.

Wish to return the previous Network settings? Apply the following commands:

mv $netfile /etc/netplan/netplan.old
mv /etc/netplan/netplan_backup.bu /etc/netplan/netplan.yaml
netplan apply

Coredns stuck on Crashloopbackoff

In case of having a Coredns pod that's stuck on Crashloopbackoff status, applying this script will resolve the issue.
Follow those steps for creating and automating the script to run on each boot event.

Create the script that will fix this issue.

sudo su –

cd /root

nano corednsfix.sh

An empty text file will open, fill it with the following script by copying and pasting.

```
#!/bin/bash
#
# Title: Coredns_CrashLoop_Startup
#
# Summary: This script should be set to a cronjob that executes
# upon reboot and modifies coredns pod during a crashloop.
#
# By: Reginald Jackson
# modified by Deb Jena
# Date: 06/05/2022
#
# Version: Built for OnWatch-2.5.x - 2.6.x
#
#
if [ $(id -g) -ne 0 ]; then
    echo "Error: Please run as root. Quiting."; exit 1
fi

dir="/tmp"
backupdir="/etc"
```

```
backupyaml="$backupdir/coredns_original_backup.yaml"
issues="reginald[dot]jackson[at]oosto[dot]com"

function help_menu() {
    echo "Title: Coredns_CrashLoop_Startup"
    echo ""
    echo "-h, --help   This menu."
    echo ""
    echo "How to use:"
    echo "1. crontab -e"
    echo "2. @reboot /<path>/$0"
    echo ""
    echo "Issues: $issues"
    exit 0
}

function wait_until_crashloop()
{
    while [ True ] ; do
            if [ "$(kubectl get pods -A | grep coredns | awk '{print $4}')" ==
    "CrashLoopBackOff" ] ; then
                    edit_coredns_config
            else
                    sleep 2
            fi
    done
}

function backup_original_config()
{
    if [ -f "$backupyaml" ]; then
            return
    else
            kubectl get cm/coredns -n kube-system -o yaml > "$backupyaml"
```

```
    fi
}

function edit_coredns_config()
{
    backup_original_config
    corednsconfig="$(kubectl get pods -A | grep coredns | awk '{print
    $2}').yaml"
    kubectl get cm/coredns -n kube-system -o yaml > "$dir/$corednsconfig"
    replace="$(kubectl get cm/coredns -n kube-system -o yaml | grep -oP
    '(?<=forward).*?(?=/)' | head -n 1)"

    sed -e "s/forward$replace\//forward 8.8.8.8 \//" -i "$dir/$corednsconfig"

    kubectl replace -f "$dir/$corednsconfig"
    kubectl delete pod -n kube-system $(kubectl get pod -A | grep coredns |
    awk '{print $2}')
    rm -f "$dir/$corednsconfig"

    sleep 140
    if [ "$(kubectl get pods -A | grep coredns | awk '{print $4}')" == "Running" ] ;
    then
            engine="$(kubectl get pods | awk '{print $1}' | grep -w -e 'pipeng-0' -e
    'cv-engine-0')"
            for e in $engine ; do
              kubectl delete pod $e
            done
            exit 0
    fi
}

function main()
{
    wait_until_crashloop
```

```
}

case "$@" in
    -h|--help) help_menu ;;
          "") main ;;
              *) echo "Bad Input." ; help_menu ;;
esac

    exit 0
```

Save the script by pressing CONTROL + S, and exit by pressing CONTROL + X

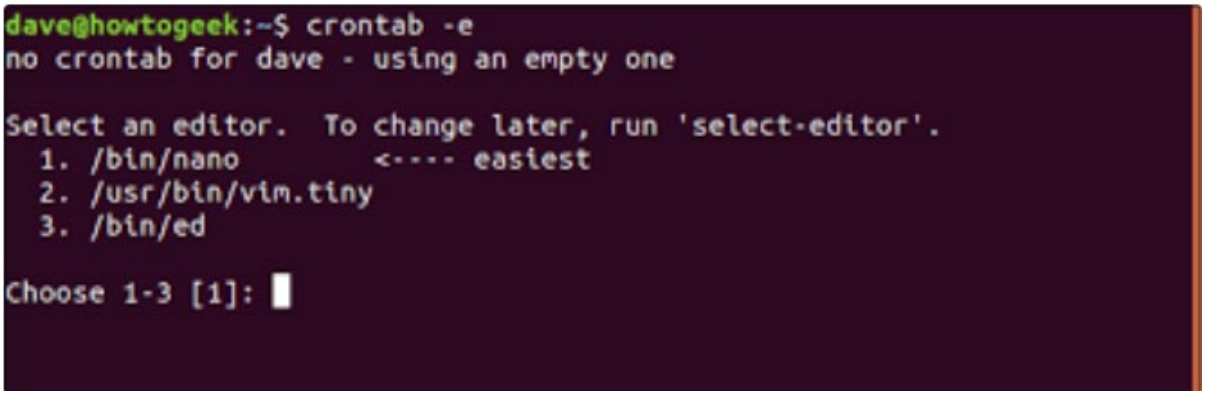
Allow execute permission.

```
chmod +x corednsfix.sh
```

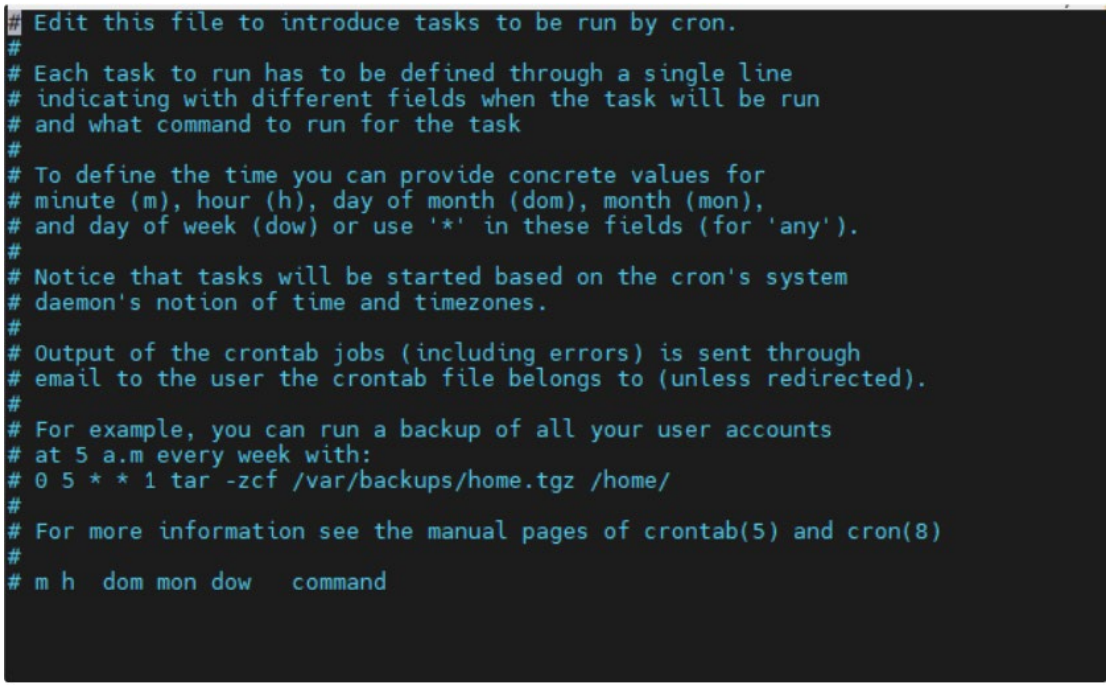
Place This script in crontab to run it every time the system reboots.

```
crontab -e
```

Notice that if this is the first time this system uses crontab it will prompt you to choose between 3 text editors, choose 1.



After choosing an option the next text file will appear, it will also appear if an option already chosen before.



Copy & paste the following line to the END of this file.

@reboot /root/corednsfix.sh

Save & exit again by pressing CONTROL +S, followed by CONTROL +X

Finally, you can run the script without needing to reboot.

./corednsfix.sh & disown

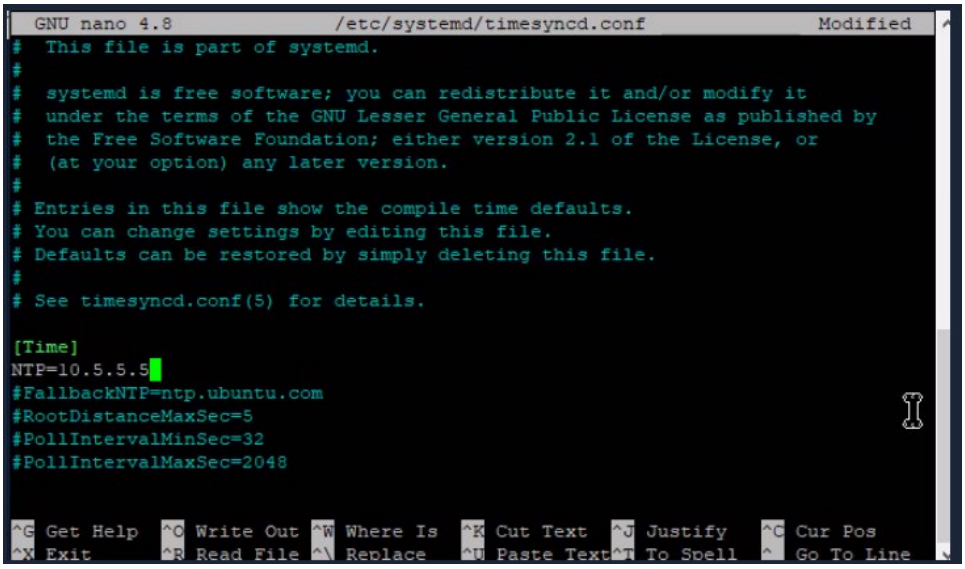
This will start the script in the background and detach it from the shell. You can then monitor pods to watch them come up (It may take 5-10 minutes to see pods start to recover after the coredns service recovers).

Configure NTP in Linux client

- **Edit the NTP File using nano:**

nano /etc/systemd/timesyncd.conf

- **Uncomment and Set NTP server IP as shown:**



- **Save the file with CONTROL+S and exit with CONTROL + X.**

- **Related Commands:**

timedatectl set-ntp no **#stop ntp service.**
timedatectl set-ntp on **#start ntp service.**
timedatectl set-ntp true **#start automatic time sync with server.**
watch timedatectl **#compare time to ntp server.**

How To Edit hosts file

In multiple processes you are required to edit your machine's hosts file.
This file's job is to connect between host names an IP addresses.
(Example use cases in the company - connecting to a v2 server from your private machine, loading up rancher in a server, loading up rancher of a different server, etc)

This is a General guide on how to edit this file on your machine:

For Linux Machines

Open up the terminal and run the following command:

sudo vi /etc/hosts

press I to enter insert mode

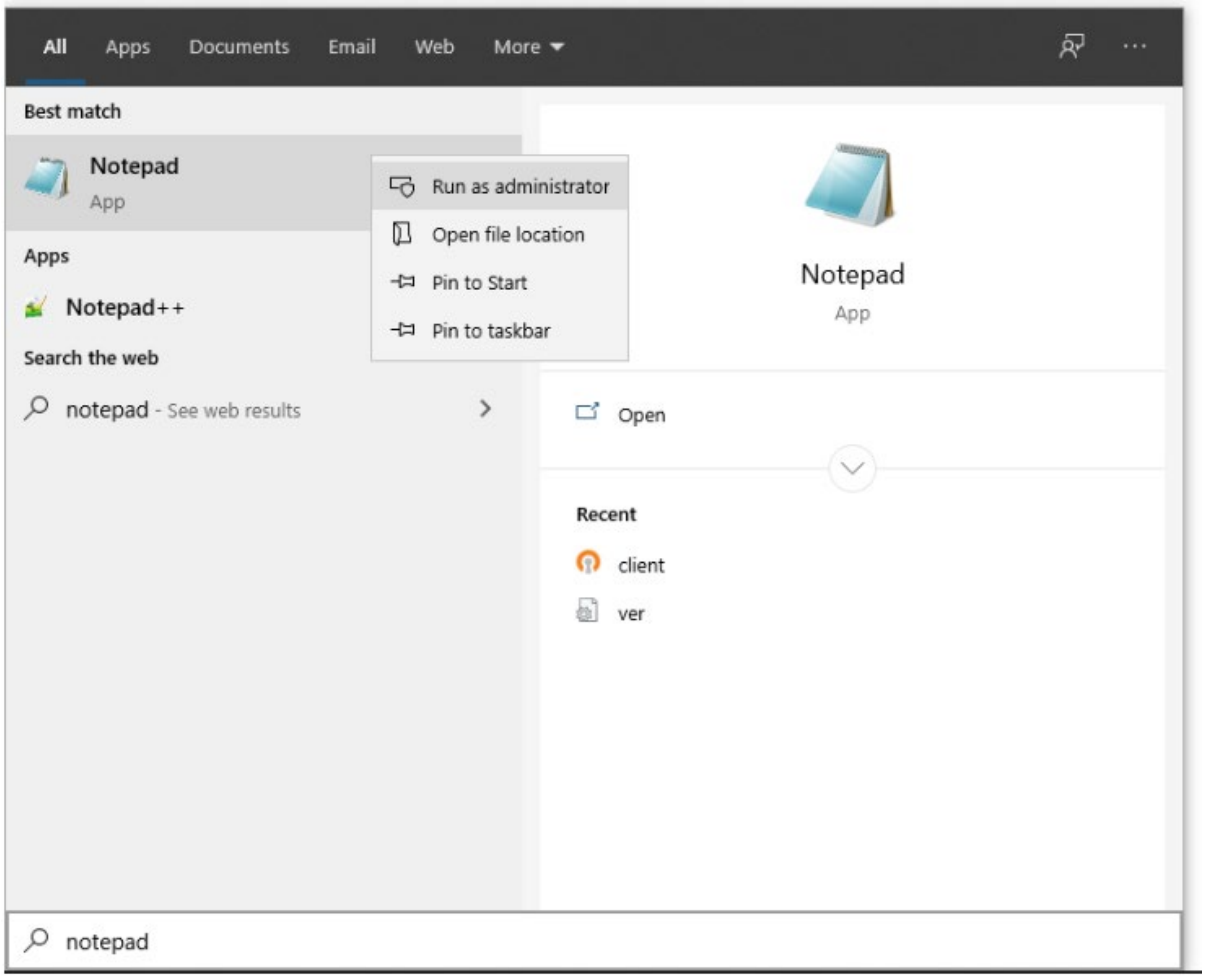
Add the required lines using this format: 'ipaddr hostname'
Example : 127.0.0.1 rancher.anv

Make sure the lines you want active are not commented (#).
To Save the changes : simply press ESC and then write :wq and then press enter.

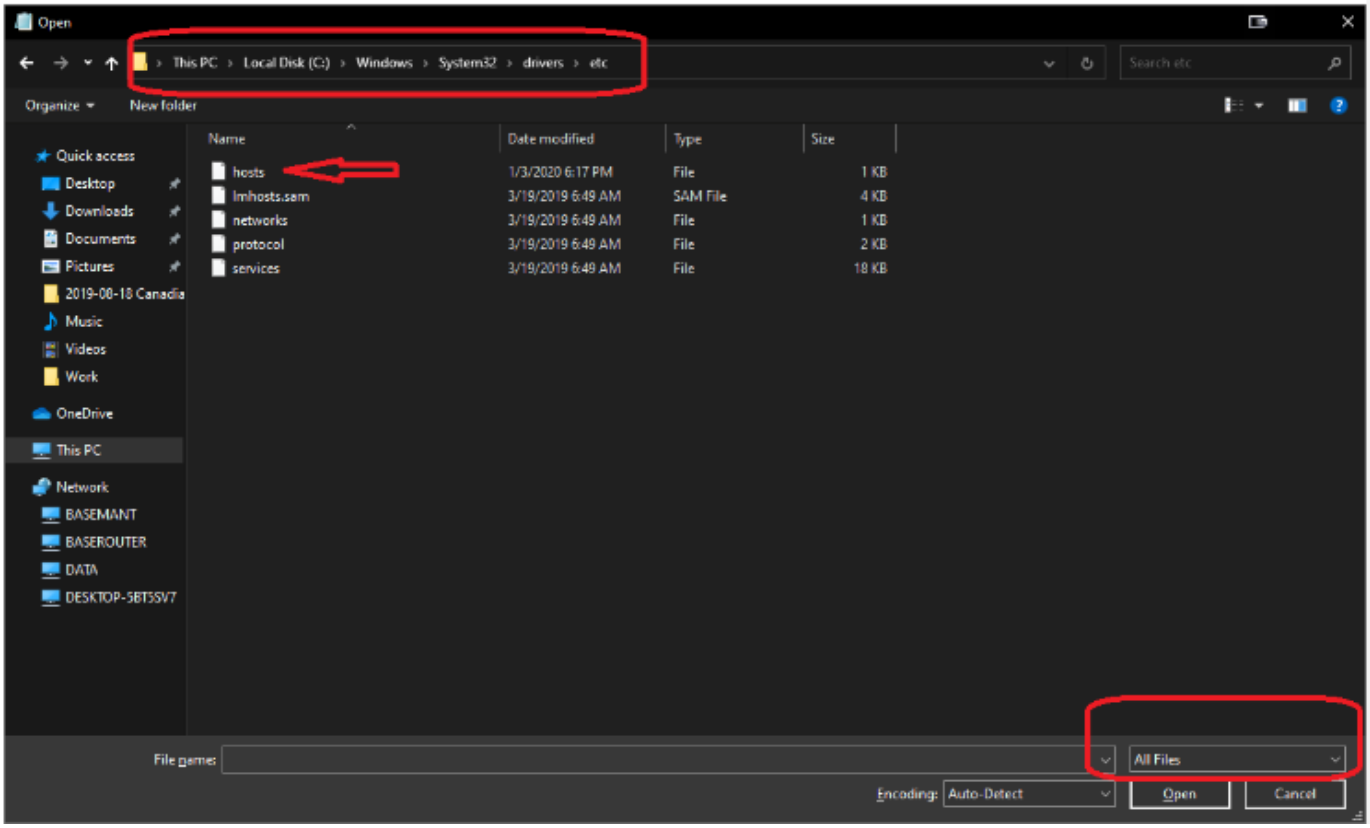
For Windows Machines

Press the start menu and start typing notepad
Right Click it and press Run as administrator.

If a window appears, click yes.



In Notepad, click File in the top left, Then Open.
Once the dialog box opens, you need to navigate to:
C:\Windows\System32\drivers\etc
and choose on the bottom right All files Instead of Text Files (*.txt)



Then click the file named hosts and press open.
Then you can add what you need in this format: 'ipaddr hostname'
Example : 127.0.0.1 rancher.anv

Make sure the lines you want active are not commented (#).
Save Your changes and exit the file.

Congratulations! You're now ready to edit the hosts file on any machine.

Jetson

Check the status of jetsons in the cluster:

In master, run command –

kubectl get nodes -owide | grep jetson

If cv-engine-arm pod fails to load:

In order to login via ssh to the Jetson you need to have corresponding key files. Next , use the following command to apply the SSH connection: (Change to correct Ip Address of jetson):

ssh oosto@<jetson_ipAddress>

- **Restart k3s service**

systemctl restart k3s-node

Remove jetson from existing cluster: (Suggested to perform in the presence of the Support Team)

Use the following command to apply the SSH connection: (Change the XXX to the correct key files, and the Ip Address to the jetson IP):

ssh oosto@<jetson_ipAddress>

- **Stop k3s service**

service k3s-node stop

- **Delete k3s service in Jetson**

rm /etc/systemd/system/k3s-node.service

- **Reboot jetson**

  sudo reboot

- **Wait until you will be able to connect again via ssh and the user environment loaded (able to run Linux command)**

- **In Main Master:**
  Remove jetson from an existing cluster:

  get jetson name from the command *kubectl get no*

- **Following with:**

  kubectl delete no <jetson-name>

- **Example (Notice that the Jetson name will change between setups):**

```
root@master1:~# kubectl get no
NAME            STATUS   ROLES         AGE   VERSION
jetson-5a9aef   Ready    edge          82d   v1.19.11+k3s1
master1         Ready    etcd,master   93d   v1.19.11+k3s1
root@master1:~# kubectl delete no jetson-5a9aef
```

- **delete jetson line from k3s node-passwd**

- **edit the following file – remove the line with the jetson (make sure to delete the correct jetson line if you have many jetsons and you wish to remove just one – pick the line to remove by searching the correct jetson name, which you pulled on the previous step.**

- **Example:**

```
root@master1: ~
root@master1:~# cat /var/lib/rancher/k3s/server/cred/node-passwd
9e58395d6cb2443006d6611863e1e13c,master1,master1,
ae58c01e5bf7c457df339f86307dce4e,jetson-5a9aef,jetson-5a9aef,
```

- **On this case, delete the second line only.**

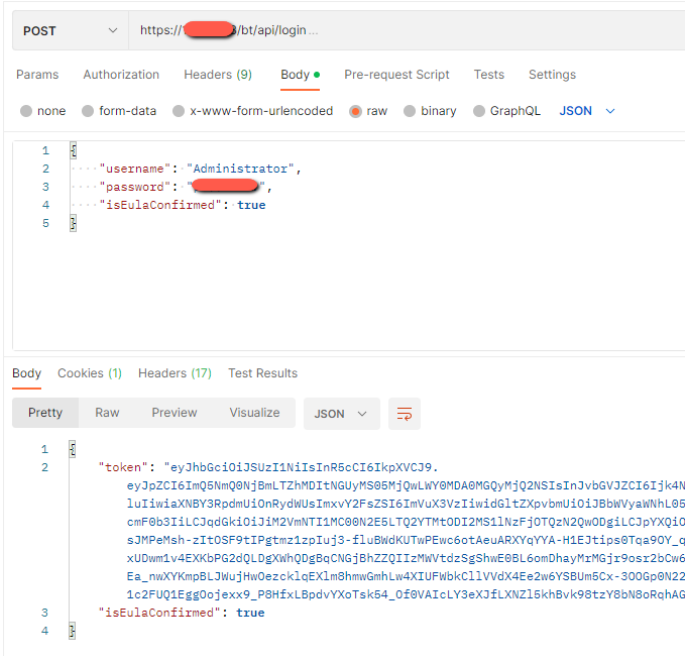
- **That's it, the jetson is no longer connected to the OnWatch.**

API

Login and receive a token

Send a POST request to https://<server-ip>/bt/api/login with the body containing 3 fields:

username – The same username used for connecting to the platform with the browser.
password – The same password used for connecting to the platform with the browser.
isEulaConfirmed – always set to true.



Save the token and use it for authenticating in future requests.

Adding a camera to OnWatch using API

To add a camera to the platform using the API, you will have to send a POST request to the https://<server-ip>/bt/api/cameras.
The body should look like the following:

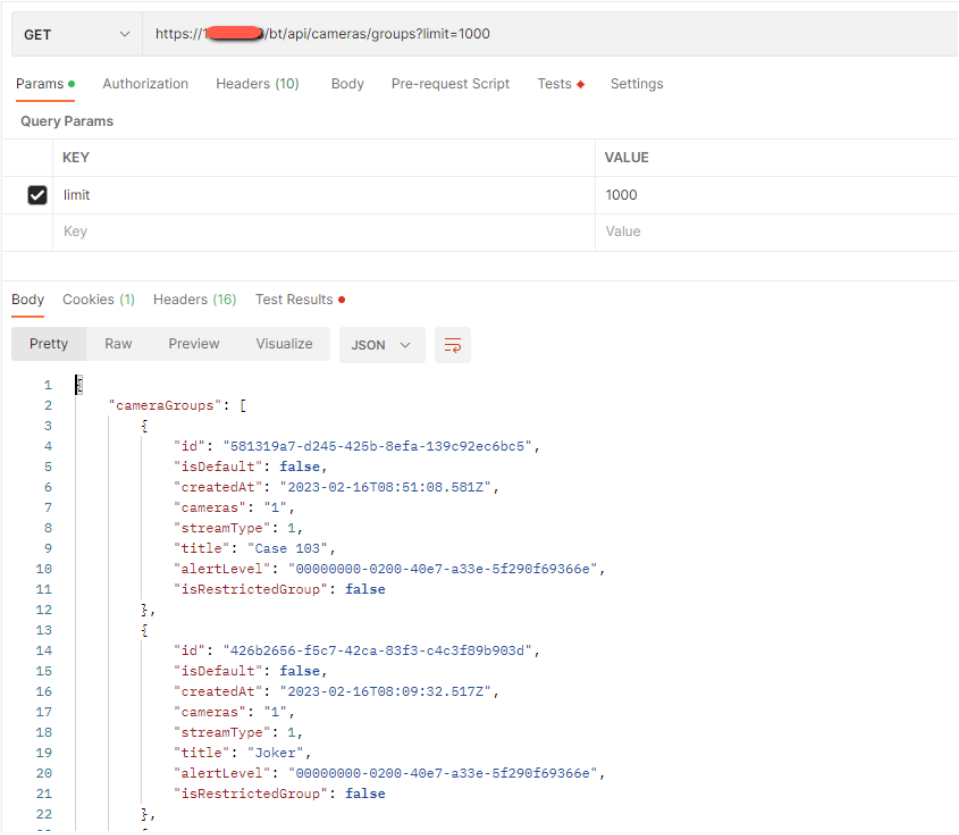
```
{
  "title": "Test camera",
  "description": "This is a test camera added via API",
  "cameraGroupId": "{{default_camera_group}}",
  "restriction": false,
  "isEnabled": true,
  "location": [0.1276, 51.5072],
  "videoUrl": "{{camera_url}}",
  "threshold": 0.5,
  "configuration": {
    "preview": false,
    "frameRotation": -1,
    "webRTC": false,
    "ffmpegOptions": "",
    "frameSkip": {
      "autoSkipEnabled": true,
      "percent": 0
    },
    "cameraMode": [1]
  },
  "pipe": "",
  "isLoadBalancingEnabled": true,
  "isAlternativeThresholdEnabled": false,
  "alternativeThreshold": null,
  "timeProfileId": null
}
```

The fields that we recommend tweaking according to your needs are the following:
“title” – This will be the title of the camera.
“description” – This will be the description of the camera.
“cameraGroupId” – This field will dictate which Camera Group this new camera will belong to. Its value should be a Camera Group ID, which can be fetched as shown in the next chapter.
“location” – The camera’s coordinates (longitude, latitude. not the other way around). This will dictate the position of the camera on the UI’s map
“videoUrl” – The camera’s RTSP URL.
Ex: rtsp://IPADDRESS:554/axis-media/media.amp

The rest of the fields are for advanced configurations which won't be covered here.

Listing Camera Groups using API

To list all your Camera Groups, send a GET request to https://<server-ip> /bt/api/cameras/groups?limit=1000



This way you will receive a list of all the groups you created and the relevant metadata. When looking for a specific group, you can use the search query parameter.

Ex: To find the Default Device Group's info, you should send the following request:

GET https://[redacted]/bt/api/cameras/groups?search=default

Params • Authorization Headers (10) Body Pre-request Script Tests ◆ Settings

Query Params

KEY	VALUE
limit	1000
search	default
Key	Value

Body Cookies (1) Headers (16) Test Results •

Pretty Raw Preview Visualize JSON


```
{
    "cameraGroups": [
        {
            "id": "00000000-0200-4c1b-4e12-1ba74bff4a4b",
            "isDefault": true,
            "createdAt": "2022-10-27T16:26:45.016Z",
            "cameras": "13",
            "streamType": 0,
            "title": "Default Device Group",
            "alertLevel": "00000000-0200-40e7-a33e-5f290f69366e",
            "description": "",
            "isRestrictedGroup": false
        }
    ],
    "total": 1
}
```