Edge Insights for Autonomous Mobile Robots (EI for AMR) Robot Orchestration Get Started Guide

Contents

Robot Orchestration with the Edge Insights for Autonomous Mobile Robots Server Kit

1

This guide describes how to configure an Intel® Smart Edge Open Single-Node or Multi-Node setup to communicate with Edge Insights for Autonomous Mobile Robots, install the Edge Insights for Autonomous Mobile Robots (EI for AMR) server kit for robot orchestration, and set up the ThingsBoard* reference. For how to install the robot kit, see the Get Started Guide for Robots.

For more information about complex scenarios, advanced features, and debugging, see the Developer Guide.

The setup consists of:

- One control plane used to control all other nodes through Kubernetes*. It can be installed alongside the worker node on the same system resulting on a Single-Node deployment or on a different system for a Multi-Node deployment.
- Multiple worker nodes to be used by EI for AMR for different resource intensive actions like:
  - Remote inference
  - Collaborative SLAM
  - Other actions that help EI for AMR perform their purpose efficiently.
- One or more EI for AMR. This page focuses on the server parts of the setup. For how to install and configure an EI for AMR, see the Get Started Guide for Robots.

Intel® Smart Edge Open is built on top of Kubernetes*, a production-grade platform for managing containerized workloads and services. Intel® Smart Edge Open experience kits customize and extend the Kubernetes control plane and edge node with microservices, third-party applications, extensions, and optimizations. The control plane node and one or more edge nodes form an Intel® Smart Edge Open edge cluster.

- A Multi-Node cluster deployment consists of an Intel® Smart Edge Open Kubernetes control plane and of one or multiple Intel® Smart Edge Open edge nodes.

  The Multi-Node example below uses one machine for the control plane, Machine A, and uses Machine B and Machine C as two worker nodes.
- In a Single-Node cluster deployment, control plane services coexist on the same physical node as the edge node.

  The Single-Node example below uses Machine A as both the control plane and the worker node.

The currently supported versions are:

- Base OS: Ubuntu* 20.04 LTS
- ROS 2 with data distribution service: Foxy
- OpenVINO™: 2021.4
- Intel® oneAPI Base Toolkit: 2021.4
- Intel® RealSense™ SDK: v2.50
- Simulation: Gazebo* v11.8.1 + Agile Robotics for Industrial Automation Competition (ARIAC) world

EI for AMR is delivered as a compressed `.zip` file that is compatible with the operating system you selected during the download. The `.zip` contains a binary executable file, a manifest file that lists the modules to be installed, and a configuration file (config.ini).

See Troubleshooting if you run into problems installing the software.

Requirements

In addition to the Product Download, you must meet the following requirements.

Target System

- Intel® CPU Processors:
  - Intel® Xeon® processor E3, E5, and E7 family
  - 2nd Generation Intel® Xeon® Scalable Processors
  - 3rd Generation Intel® Xeon® Scalable Processors
- 16 GB RAM
- 128 GB hard drive
- Ubuntu* 20.04 LTS

Knowledge/Experience

- You are familiar with executing Linux* commands.
- You have basic Docker* experience.
- ROS 1 or ROS 2 background recommended.

Step 1: Prepare Your Machines

1. Install a clean Ubuntu* OS on all machines. This example uses Ubuntu* Linux Server, but Ubuntu* Desktop is also supported.
   a. Download the Ubuntu* Linux Server ISO file to your developer workstation for the control plane and for the worker nodes.
   b. Create a bootable flash drive using an imaging application, such as Startup Disk Creator, available on Ubuntu*.
   c. After flashing the USB drive, power off your target systems, insert the USB drive, and power on the target systems.
   d. If the target systems do not boot from the USB drive, change the boot priority in the system BIOS.
   e. Follow the prompts to install Ubuntu* with the default configurations. For detailed instructions, see the Ubuntu* guide.
   f. Power down your target systems, and remove the USB drive.
   g. Power up the target systems.

   Expected result: Ubuntu* Server is successfully installed.
2. Verify that PATH is configured in `/etc/environment` to contain, at a minimum:

```
PATH='/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin'
```

   If a proxy is required to connect to the Internet, add the proxy settings in `/etc/environment`:

```
export http_proxy="http://<http_proxy>:port"
export https_proxy="http://<https_proxy>:port"
export ftp_proxy="http://<ftp_proxy>:port"
export no_proxy="<no_proxy>"
```

   Source `/etc/environment`:

```
source /etc/environment
```

3. Install the following packages on all machines:

```
sudo apt-get update
sudo apt-get install ssh
sudo apt-get install net-tools wget pipenv git ansible
pip install sh
```

4. Configure the server with a static IP. This is needed to avoid IP changes during server reboots. Use your IP addresses in the following example. For help on how to set a static IP, go to Troubleshooting.

NOTE This step is needed because the deployment reboots your system multiple times, and the deployment assumes your system has the same IP every time.

5. Set `hostname` if the system does not already have one:

```
sudo hostname -b <hostname>
```

6. Make sure that password-less `ssh` access for `root` is set on all machines:

```
sudo nano /etc/ssh/sshd_config
```

Add the following line at the end of the file:

```
PermitRootLogin yes
```

After the `/etc/ssh/sshd_config` is updated, restart the `ssh` service:

```
sudo service ssh restart
```

NOTE Execute this step on all systems.

7. On all machines, log in as `root` and, if it does not have one already, set a password for `root`:

```
sudo su -
passwd
```

8. Generate, and add the `ssh` keys on the server:

```
ssh-keygen -f $HOME/.ssh/id_rsa -P ""
# Repeat the following two commands for all the machines.
# For Single-Node these steps still need to be done from Machine A to Machine A.
ssh-copy-id root@<hostname of the machine>      chmod -R 700 ~/.ssh
```

NOTEFor Multi-Node Deployment: The keys must be sent from the server (Machine A) to all nodes (Machine B, Machine C) and from server to server.
For Single-Node Deployment: The keys must be sent from the machine to itself. This is needed because Intel® Smart Edge Open uses the `ssh` connection from the controller node to a worker node even though they are on the same machine.

9. **For Multi-Node Deployment:** Set the correct permissions for `~/.ssh` directory on all worker nodes:

```
chmod -R 700 ~/.ssh
```

10. For all the machines, update `/etc/hosts`:

```
127.0.0.1       localhost  <enter the system's hostname here>
127.0.1.1       <enter the system's hostname here>

# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback <enter the system's hostname here>
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

11. **For Multi-Node Deployment:** Synchronize the date and time between the server and nodes:

```
date +%Y%m%d%T -s "`ssh root@node1 'date "+%Y%m%d %T"'`"
```

NOTE Run the command from Machine A to Machine B and from Machine A to Machine C.

Step 2: Install Intel® Smart Edge Open, Tailored for EI for AMR

The following steps install Intel® Smart Edge Open on the server.

1. Clone the `open-developer-experience-kits` repository to Machine A:

```
git clone -b smart-edge-open-21.12 https://github.com/smart-edge-open/open-developer-experience-kits.git ~/dek
cd ~/dek
git checkout 1848a355586d2c40420b6e5576efeac9396150de
```

2. Get the IPs from Machine A, Machine B, and Machine C and save them for later:

```
ifconfig
```

3. Machine A: Edit the `~/dek/inventory.yml` file by providing information about the cluster nodes and the intended deployment flavor.

   For Multi-Node Deployment: Set the `single_node_deployment` value to `false`.

   For Single-Node Deployment: Set the `single_node_deployment` value to `true`. In the following example, the controller and node must be the same machine (Machine A).

   Example:

```
# SPDX-License-Identifier: Apache-2.0
# Copyright (c) 2021 Intel Corporation

---
all:
   vars:
   cluster_name: dek_test        # NOTE: Use `_` instead of spaces.
   deployment: dek               # NOTE: Available deployment type: Developer experience kits (dek).
   single_node_deployment: false  # Request single node deployment (true/false).
   limit:                        # Limit ansible deployment to certain inventory group or hosts
controller_group:
   hosts:
   controller:
      ansible_host: <ip_from_Machine_A>
      ansible_user: root
edgenode_group:
   hosts:
   node01:
      ansible_host: <ip_from_Machine_B>
      ansible_user: root
   node02:
      ansible_host: <ip_from_Machine_C>
      ansible_user: root
```

   NOTE Do not forget to change the ansible_user from smartedge-open to root as shown in the above example.

4. Machine A: If a proxy is required to connect to the Internet, edit the proxy variables in the `~/dek/inventory/default/group_vars/all/10-default.yml` file.

Example:

```
# SPDX-License-Identifier: Apache-2.0
# Copyright (c) 2019-2021 Intel Corporation

---
# This file contains variables intended to be configured by user.
# It allows feature enabling and configuration.
# Per-host variables should be places in `inventory/default/host_vars` directory.
# Features should not be configured by changing roles' defaults (i.e. role/defaults/main.yml)

################################################
##### User settings

### Proxy settings
proxy_env:
   # Proxy URLs to be used for HTTP, HTTPS and FTP
   http_proxy: "http://proxy.example.org:3128"
   https_proxy: "http://proxy.example.org:3129"
   ftp_proxy: "http://proxy.example.org:3128"
   # No proxy setting contains addresses and networks that should not be accessed using proxy
(e.g. local network, Kubernetes CNI networks)
   no_proxy: "127.0.0.1/32"
```

5. The following changes are needed to avoid multiple errors:

a. Machine A: Update the `~/dek/inventory/default/group_vars/all/10-default.yml` file with:

```
sriov_network_operator_enable: false

## SR-IOV Network Operator configuration
sriov_network_operator_configure_enable: false

### Software Guard Extensions
# SGX requires kernel 5.11+, SGX enabled in BIOS and access to PCC service
sgx_enabled: false

# Install isecl attestation components (TA, ihub, isecl k8s controller and scheduler extension)
platform_attestation_node: false

install_hwe_kernel_enable: false
```

b. Machine A: Update the `~/dek/roles/telemetry/grafana/templates/prometheus-tls-datasource.yml` file. Open the file and replace `indentfirst` with `first`, or run the following command:

```
sed -i "s/indent(width=13, indentfirst=False)/indent(width=13, first=False)/g" ~/dek/roles/
telemetry/grafana/templates/prometheus-tls-datasource.yml
```

6. Machine A: Start deployment by executing the following script:

```
./deploy.sh
```

The script reboots the target. After the reboot, re-execute the script.

Example of a successful result:

```
kubernetes/harbor_registry/controlplane ------------------------------ 566.66s
infrastructure/docker ------------------------------------------------ 54.81s
infrastructure/install_dependencies ---------------------------------- 43.46s
baseline_ansible/infrastructure/install_packages --------------------- 15.76s
infrastructure/os_setup ---------------------------------------------- 13.75s
kubernetes/install --------------------------------------------------- 7.69s
kubernetes/cni/calico/controlplane ----------------------------------- 6.60s
```

```
gather_facts ------------------------------------------------------------ 5.45s
baseline_ansible/infrastructure/os_requirements/disable_swap ------------ 5.18s
kubernetes/harbor_registry/node ----------------------------------------- 4.97s
infrastructure/firewall_open_ports -------------------------------------- 4.95s
telemetry/grafana ------------------------------------------------------- 3.36s
baseline_ansible/infrastructure/os_requirements/dns_stub_listener ------- 3.06s
telemetry/collectd/node ------------------------------------------------- 2.60s
infrastructure/conditional_reboot --------------------------------------- 2.30s
telemetry/certs --------------------------------------------------------- 2.27s
baseline_ansible/infrastructure/configure_udev -------------------------- 2.08s
kubernetes/cni ---------------------------------------------------------- 1.91s
telemetry/prometheus ---------------------------------------------------- 1.79s
baseline_ansible/infrastructure/build_nic_drivers ----------------------- 1.75s
infrastructure/git_repo ------------------------------------------------- 1.75s
baseline_ansible/infrastructure/install_openssl ------------------------- 1.62s
kubernetes/default_netpol ----------------------------------------------- 1.18s
kubernetes/helm --------------------------------------------------------- 0.99s
telemetry/statsd-exporter ----------------------------------------------- 0.99s
kubernetes/nfd ---------------------------------------------------------- 0.97s
infrastructure/build_noproxy -------------------------------------------- 0.88s
baseline_ansible/infrastructure/os_proxy -------------------------------- 0.87s
kubernetes/custom_namespace --------------------------------------------- 0.78s
kubernetes/customize_kubelet -------------------------------------------- 0.78s
kubernetes/controlplane ------------------------------------------------- 0.73s
telemetry/collectd/controlplane ----------------------------------------- 0.69s
fail -------------------------------------------------------------------- 0.64s
baseline_ansible/infrastructure/os_requirements/enable_ipv4_forwarding --- 0.53s
shell ------------------------------------------------------------------- 0.52s
telemetry/cadvisor ------------------------------------------------------ 0.46s
baseline_ansible/infrastructure/install_golang -------------------------- 0.42s
baseline_ansible/kubernetes/operator/sriov_network_operator/install ----- 0.36s
infrastructure/grub ----------------------------------------------------- 0.29s
baseline_ansible/kubernetes/operator/sriov_network_operator/configure --- 0.26s
include_tasks ----------------------------------------------------------- 0.24s
debug ------------------------------------------------------------------- 0.22s
baseline_ansible/infrastructure/selinux --------------------------------- 0.21s
infrastructure/e810_driver_update --------------------------------------- 0.20s
set_fact ---------------------------------------------------------------- 0.16s
baseline_ansible/infrastructure/disable_fingerprint_authentication ------ 0.16s
baseline_ansible/infrastructure/time_setup_ntp -------------------------- 0.15s
baseline_ansible/kubernetes/operator/sriov_network_operator/prepare_node --- 0.09s
infrastructure/setup_baseline_ansible ----------------------------------- 0.08s
include_vars ------------------------------------------------------------ 0.07s
kubernetes/create_namespaces -------------------------------------------- 0.04s
infrastructure/setup_offline -------------------------------------------- 0.04s
stat -------------------------------------------------------------------- 0.03s
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
total ------------------------------------------------------------------ 767.77s
2021-11-05 22:45:45.898 INFO: dek_test single_node_network_edge.yml: succeed.
2021-11-05 22:45:46.899 INFO: ====================
2021-11-05 22:45:46.900 INFO: DEPLOYMENT RECAP:
2021-11-05 22:45:46.900 INFO: ====================
2021-11-05 22:45:46.900 INFO: DEPLOYMENT COUNT: 1
2021-11-05 22:45:46.900 INFO: SUCCESSFUL DEPLOYMENTS: 1
2021-11-05 22:45:46.901 INFO: FAILED DEPLOYMENTS: 0
2021-11-05 22:45:46.901 INFO: DEPLOYMENT "dek_test": SUCCESSFUL
2021-11-05 22:45:46.901 INFO: ====================
```

If you encounter errors, see Troubleshooting.

Step 3: Install the EI for AMR Server Kit

1. Machine A: Download the latest release.
   a. Go to Product Download.
   b. Select **Server Kit** or **Robot and Sever Complete Kit**.
   c. Click **Download**.
2. Machine A: Copy the zip file to your target system.
3. Machine A: Extract and install the software:

```
unzip edge_insights_for_amr.zip
cd edge_insights_for_amr
chmod 775 edgesoftware
export no_proxy="127.0.0.1/32,devtools.intel.com"
sudo ./edgesoftware install
sudo groupadd docker
sudo usermod -aG docker $USER
newgrp docker
source /etc/environment
```

4. Machine A: Get all Intel® Smart Edge Open nodes:

```
sudo su -
kubectl get nodes
```

5. Machine A: Delete Intel® Smart Edge Open node labels for all worker nodes:

```
kubectl label nodes <node-name-from-the-previous-step> tier-
kubectl label nodes <node-name-from-the-previous-step> environment-
```

6. Machine A: Go to `<edge_insights_for_amr_path>/Edge_Insights_for_Autonomous_Mobile_Robots_<release>`, and run. This playbook will set the node labels:

```
cd <edge_insights_for_amr_path>/Edge_Insights_for_Autonomous_Mobile_Robots_*
ansible-playbook AMR_server_containers/01_docker_sdk_env/docker_orchestration/ansible-playbooks/
02_edge_server/smart_edge_open/seo_cluster_labeling.yaml
```

7. Machine A: Verify that the labels were correctly updated:

```
kubectl get nodes -L environment
kubectl get nodes -L tier
```

Step 4: Install the ThingsBoard* Reference

This ThingsBoard* reference includes a pre-configured database customized for Intel's EI for AMR solution.

Configure the ThingsBoard* Reference Server

1. Create a Java* Keystore certificate. You can use this method: https://thingsboard.io/docs/user-guide/mqtt-over-ssl/#java-keystore.

   NOTE For testing purposes, you can set these values:

   - `DOMAIN_SUFFIX=localhost`
   - `SUBJECT_ALTERNATIVE_NAMES="ip:<ip_of_server>"`

   If errors are encountered during key generation, see Troubleshooting, "Keytool is not installed".

   After the certificate (.jks file) is generated, copy it to the following path in your installed EI for AMR.

NOTE This also creates a public-key (*.pub.pem file). You need to copy this file when generating the turtle_creek_client image (client-side image).

```
# Copy generated server.jks on the server side:
cp server.jks <edge_insights_for_amr_path>/Edge_Insights_for_Autonomous_Mobile_Robots_<version>/
AMR_server_containers/01_docker_sdk_env/artifacts/02_edge_server/edge_server_fleet_management/
mqttserver.jks

# Transfer (scp) server.pub.pem from server to client, then copy it to the following path on
client:
cp server.pub.pem <edge_insights_for_amr_path>/
Edge_Insights_for_Autonomous_Mobile_Robots_<version>/AMR_containers/01_docker_sdk_env/artifacts/
02_edge_server/edge_server_fleet_management/thingsboard.pub.pem
```

2. Set up the environment variables necessary to run docker-compose commands:

```
cd <edge_insights_for_amr_path>/Edge_Insights_for_Autonomous_Mobile_Robots_202*
source 01_docker_sdk_env/docker_compose/05_tutorials/config/docker_compose.source
```

Start the ThingsBoard* Reference Server Deployment

If Intel® Smart Edge Open Multi-Node is deployed, there will be two machines for orchestration.

- Machine A-1 is the controller.
- Machine A-2 is the server node where the ThingsBoard* server pod is deployed.

If Intel® Smart Edge Open Single-Node is deployed, Machine A-1 and Machine A-2 are the same machine.

1. If Intel® Smart Edge Open Single-Node is configured, run the following commands:

```
sed -i "s/number_of_nodes.stdout!=0/("{{number_of_nodes.stdout}}"!="0")/g" AMR_server_containers/
01_docker_sdk_env/docker_orchestration/ansible-playbooks/02_edge_server/fleet_management/
fleet_management_playbook_uninstall.yaml
sed -i "s/number_of_nodes.stdout!=0/("{{number_of_nodes.stdout}}"!="0")/g" AMR_server_containers/
01_docker_sdk_env/docker_orchestration/ansible-playbooks/02_edge_server/fleet_management/
fleet_management_playbook_install.yaml
sed -i "s/number_of_nodes.stdout==0/("{{number_of_nodes.stdout}}"=="0")/g" AMR_server_containers/
01_docker_sdk_env/docker_orchestration/ansible-playbooks/02_edge_server/fleet_management/
fleet_management_playbook_install.yaml
```

NOTE This step is needed to avoid an Intel® Smart Edge Open Single-Node Playbook known limitation.

2. Run following command on Machine A-1:

```
ansible-playbook AMR_server_containers/01_docker_sdk_env/docker_orchestration/ansible-playbooks/
02_edge_server/fleet_management/fleet_management_playbook_install.yaml
```

NOTE If the ansible-playbook fails, you can uninstall and try again. Also, verify if the installation was successful even if the ansible-playbook failed.

```
ansible-playbook AMR_server_containers/01_docker_sdk_env/docker_orchestration/ansible-playbooks/
02_edge_server/fleet_management/fleet_management_playbook_uninstall.yaml
ansible-playbook AMR_server_containers/01_docker_sdk_env/docker_orchestration/ansible-playbooks/
02_edge_server/fleet_management/fleet_management_playbook_install.yaml
```

3. Verify that the services, pods, and deployment are running on Machine A-1:

```
$ kubectl get all --output=wide --namespace fleet-management
NAME                                        READY   STATUS    RESTARTS      AGE
pod/fleet-deployment-8449fdc54f-m4fhb       1/1     Running   2 (21s ago)   81s

NAME                                        TYPE       CLUSTER-IP      EXTERNAL-IP
PORT(S)                                                    AGE
service/fleet-service                       NodePort   10.97.216.230   <none>
9090:32764/TCP,1883:32765/TCP,7070:32766/TCP,8883:32767/TCP   42s

NAME                                        READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/fleet-deployment            1/1     1            1           81s

NAME                                        DESIRED   CURRENT   READY   AGE
replicaset.apps/fleet-deployment-8449fdc54f   1         1         1       81s
```

NOTE CLUSTER-IP is a virtual IP that is allocated by Kubernetes* to a service. It is the Kubernetes* internal IP. Two different pods can communicate using this IP.

4. Verify that the Docker* container is running on Machine A-2:

```
docker ps | grep fleet
dd22be830f82   10.237.23.152:30003/intel/fleet-management              "/usr/bin/start-tb.sh"
52 minutes ago   Up 52 minutes             k8s_fleet_fleet-deployment-858494f866-7jmhh_fleet-
management_13d09334-4223-4409-8cd9-c0cac60cd04c_0
```

5. To test the image, install GDM3, open a browser on the host machine, and type the following URL on any host machine:

```
sudo apt-get install gdm3
sudo systemctl restart gdm3.service
sudo apt install firefox
# Open Firefox and go to:
<IP Address>:32764
```

A ThingsBoard* login interface appears.

NOTE The port is the mapped port of 9090 that the ThingsBoard* server uses (check the ports from `kubectl get all --namespace fleet-management` command). In this case, the port is 32764. If Single-Node orchestration is deployed, use the IP address of the Single-Node server.

If Multi-Node orchestration is deployed, use the IP address of the (Machine A-1) controller.

To get the IP address:

```
kubectl describe node | grep 'Addresses:' -A 4 | grep -B1 $(kubectl get nodes | grep
control-plane | awk '{print $1}') | grep InternalIP | awk '{print $2}'
```

Deepen Your Knowledge

For more information about complex scenarios, advanced features, and debugging, see the Developer Guide.

Troubleshooting

Setting a Static IP

Depending on your network setup, there are multiple ways to set a static IP.

- In a home network, check your local router on how to set static IP on your device using your MAC address.
- In a corporate network, please check with your local IT on how to set a static IP on a system.

Another option is to set it from your system's Operating System. A good tutorial on how to set a static IP using netplan can be found here.

Remember to run `netplan apply` after you are finished with the configuration.

- Make sure you that your system has the correct date:

```
date
```

If the date is incorrect, contact your local support team for help setting the correct date and time.

- To find the gateway:

```
ip route | grep default
```

- To find the name servers, find your interface name and replace it below:

```
nmcli device show <interface name> | grep IP4.DNS
```

virtualenv Error

If the following error is displayed:

```
Virtualenv location:
Warning: There was an unexpected error while activating your virtualenv.
Continuing anyway…
Traceback (most recent call last):
File "./deploy.py", line 24, in <module>
from scripts import log_all
ImportError: cannot import name 'log_all' from 'scripts' (/home/
test/.local/lib/python3.8/site-packages/scripts/__init__.py)
```

Remove the `~/.local/lib/python3.8/` directory and run the following commands:

```
pip install --user -U pip
pip freeze --user | cut -d'=' -f1 | xargs pip install --user -U
```

termcolor Error

If the following error is displayed:

```
Failed to install termcolor. b'/usr/local/lib/python3.8/dist-packages/pkg_resources/
__init__.py:122:
```

```
python3 -m pip uninstall setuptools
python3 -m pip install setuptools
```

Restart the target and run:

```
python3 -m pip install --upgrade setuptools
```

Failed OpenSSL Download

If the following error is displayed:

```
FAILED - RETRYING: OpenSSL download from https://www.openssl.org/source/openssl-1.1.1i.tar.gz (4
retries left).
FAILED - RETRYING: OpenSSL download from https://www.openssl.org/source/openssl-1.1.1i.tar.gz (4
retries left).
FAILED - RETRYING: OpenSSL download from https://www.openssl.org/source/openssl-1.1.1i.tar.gz (3
retries left).
FAILED - RETRYING: OpenSSL download from https://www.openssl.org/source/openssl-1.1.1i.tar.gz (3
retries left).
FAILED - RETRYING: OpenSSL download from https://www.openssl.org/source/openssl-1.1.1i.tar.gz (2
retries left).
FAILED - RETRYING: OpenSSL download from https://www.openssl.org/source/openssl-1.1.1i.tar.gz (2
retries left).
FAILED - RETRYING: OpenSSL download from https://www.openssl.org/source/openssl-1.1.1i.tar.gz (1
retries left).
FAILED - RETRYING: OpenSSL download from https://www.openssl.org/source/openssl-1.1.1i.tar.gz (1
retries left)
```

Run the following commands:

```
wget --directory-prefix=/tmp http://certificates.intel.com/repository/certificates/
IntelSHA2RootChain-Base64.zip

sudo unzip -o /tmp/IntelSHA2RootChain-Base64.zip -d /usr/local/share/ca-certificates/

rm /tmp/IntelSHA2RootChain-Base64.zip

update-ca-certificates
```

“Isecl control plane IP not set” Error

If the following error is displayed:

```
TASK [Check control plane IP]
*********************************************************************************************
*********************************************************************************************
*************
task path: /root/dek/roles/security/isecl/common/tasks/precheck.yml:7
Wednesday 16 February 2022  15:36:34 +0000 (0:00:00.047)       0:00:05.373 ****
fatal: [node01]: FAILED! => {
   "changed": false
}

MSG:

Isecl control plane IP not set!
fatal: [node02]: FAILED! => {
   "changed": false
}

MSG:

Isecl control plane IP not set!
```

```
fatal: [controller]: FAILED! => {
   "changed": false
}

MSG:

Isecl control plane IP not set!
```

Update the `~/dek/inventory/default/group_vars/all/10-default.yml` file with:

```
# Install isecl attestation components (TA, ihub, isecl k8s controller and scheduler extension)
platform_attestation_node: false
```

"PCCS IP address not set" Error

If the following error is displayed:

```
TASK [Check PCCS IP address]
*********************************************************************************************
*********************************************************************************************
*************
task path: /root/dek/roles/infrastructure/provision_sgx_enabled_platform/tasks/
param_precheck.yml:7
Wednesday 16 February 2022  15:39:59 +0000 (0:00:00.060)       0:00:05.688 ****
fatal: [node01]: FAILED! => {
   "changed": false
}

MSG:

PCCS IP address not set!
fatal: [node02]: FAILED! => {
   "changed": false
}

MSG:

PCCS IP address not set!
fatal: [controller]: FAILED! => {
   "changed": false
}

MSG:

PCCS IP address not set!
```

Update the `~/dek/inventory/default/group_vars/all/10-default.yml` file with:

```
### Software Guard Extensions
# SGX requires kernel 5.11+, SGX enabled in BIOS and access to PCC service
sgx_enabled: false
```

"no supported NIC is selected" Error

If the following error is displayed:

```
sriovnetwork.sriovnetwork.openshift.io/sriov-vfio-network-c1p1 unchanged
STDERR:
Error from server (no supported NIC is selected by the nicSelector in CR sriov-netdev-net-c0p0):
error when creating "sriov-netdev-net-c0p0-sriov_network_node_policy.yml": admission webhook
```

```
"operator-webhook.sriovnetwork.openshift.io" denied the request: no supported NIC is selected by
the nicSelector in CR sriov-netdev-net-c0p0
Error from server (no supported NIC is selected by the nicSelector in CR sriov-netdev-net-c1p0):
error when creating "sriov-netdev-net-c1p0-sriov_network_node_policy.yml": admission webhook
"operator-webhook.sriovnetwork.openshift.io" denied the request: no supported NIC is selected by
the nicSelector in CR sriov-netdev-net-c1p0
Error from server (no supported NIC is selected by the nicSelector in CR sriov-vfio-pci-net-
c0p1): error when creating "sriov-vfio-pci-net-c0p1-sriov_network_node_policy.yml": admission
webhook "operator-webhook.sriovnetwork.openshift.io" denied the request: no supported NIC is
selected by the nicSelector in CR sriov-vfio-pci-net-c0p1
Error from server (no supported NIC is selected by the nicSelector in CR sriov-vfio-pci-net-
c1p1): error when creating "sriov-vfio-pci-net-c1p1-sriov_network_node_policy.yml": admission
webhook "operator-webhook.sriovnetwork.openshift.io" denied the request: no supported NIC is
selected by the nicSelector in CR sriov-vfio-pci-net-c1p1
```

Update the `~/dek/inventory/default/group_vars/all/10-default.yml` file with:

```
sriov_network_operator_enable: false

## SR-IOV Network Operator configuration
sriov_network_operator_configure_enable: false
```

"Unexpected templating type error"

If the following error is displayed:

```
MSG:
AnsibleError: Unexpected templating type error occurred on (# SPDX-License-Identifier: Apache-2.0
# Copyright (c) 2020 Intel Corporation
apiVersion: v1
kind: ConfigMap
metadata:
   name: grafana-datasources
   namespace: telemetry
   labels:
      grafana_datasource: '1'
data:
   prometheus-tls.yaml: |-
      apiVersion: 1
      datasources:
      - name: Prometheus-TLS
         access: proxy
         editable: true
         orgId: 1
         type: prometheus
         url: https://prometheus:9099
         withCredentials: true
         isDefault: true
         jsonData:
            tlsAuth: true
            tlsAuthWithCACert: true
         secureJsonData:
            tlsCACert: |
               {{ telemetry_root_ca_cert.stdout | trim | indent(width=13, indentfirst=False) }}
            tlsClientCert: |
               {{ telemetry_grafana_cert.stdout | trim | indent(width=13, indentfirst=False) }}
            tlsClientKey: |
               {{ telemetry_grafana_key.stdout | trim | indent(width=13, indentfirst=False) }}
```

```
        version: 1
        editable: false
): do_indent() got an unexpected keyword argument 'indentfirst'
```

Update the `~/dek/roles/telemetry/grafana/templates/prometheus-tls-datasource.yml` file with:

```
-             {{ telemetry_root_ca_cert.stdout | trim | indent(width=13, indentfirst=False) }}
+             {{ telemetry_root_ca_cert.stdout | trim | indent(width=13, first=False) }}
-             {{ telemetry_grafana_cert.stdout | trim | indent(width=13, indentfirst=False) }}
+             {{ telemetry_grafana_cert.stdout | trim | indent(width=13, first=False) }}
-             {{ telemetry_grafana_key.stdout | trim | indent(width=13, indentfirst=False) }}
+             {{ telemetry_grafana_key.stdout | trim | indent(width=13, first=False) }}
```

"Wait till all Harbor resources ready" Message

If the following log is displayed:

```
TASK [kubernetes/cni : Wait till all Harbor resources ready]
*******************************************************************************************************
*************************************************************************
task path: /home/user/dek/roles/kubernetes/cni/tasks/main.yml:20
Tuesday 16 November 2021 14:41:58 +0100 (0:00:00.070) 0:04:39.646 ******
FAILED - RETRYING: Wait till all Harbor resources ready (60 retries left).
FAILED - RETRYING: Wait till all Harbor resources ready (59 retries left).
FAILED - RETRYING: Wait till all Harbor resources ready (58 retries left).
FAILED - RETRYING: Wait till all Harbor resources ready (57 retries left).
FAILED - RETRYING: Wait till all Harbor resources ready (56 retries left).
FAILED - RETRYING: Wait till all Harbor resources ready (55 retries left).
FAILED - RETRYING: Wait till all Harbor resources ready (54 retries left).
FAILED - RETRYING: Wait till all Harbor resources ready (53 retries left).
FAILED - RETRYING: Wait till all Harbor resources ready (52 retries left).
FAILED - RETRYING: Wait till all Harbor resources ready (51 retries left).
FAILED - RETRYING: Wait till all Harbor resources ready (50 retries left).
```

Wait approximately 30 minutes. The Intel® Smart Edge Open deployment script waits for the Harbor resources to be ready.

Installation Stuck

If the installation remains stuck with the following log:

```
TASK [infrastructure/os_setup : enable UFW]
*******************************************************************************************************
*************************************************************************************************
task path: /root/dek/roles/infrastructure/os_setup/tasks/ufw_enable_debian.yml:12
Wednesday 16 February 2022  15:53:04 +0000 (0:00:01.627)       0:08:03.425 ****
NOTIFIED HANDLER reboot server for controller
changed: [controller] => {
   "changed": true,
   "commands": [
      "/usr/sbin/ufw status verbose",
      "/usr/bin/grep -h '^### tuple' /lib/ufw/user.rules /lib/ufw/user6.rules /etc/ufw/
user.rules /etc/ufw/user6.rules /var/lib/ufw/user.rules /var/lib/ufw/user6.rules",
      "/usr/sbin/ufw -f enable",
      "/usr/sbin/ufw status verbose",
      "/usr/bin/grep -h '^### tuple' /lib/ufw/user.rules /lib/ufw/user6.rules /etc/ufw/
user.rules /etc/ufw/user6.rules /var/lib/ufw/user.rules /var/lib/ufw/user6.rules"
   ]
}
```

```
MSG:

Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), deny (routed)
New profiles: skip

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW IN    Anywhere
22/tcp (v6)                ALLOW IN    Anywhere (v6)
```

Type `Ctrl-c`, and restart the installation. (Run the `./deploy.sh` script again.)

Pod Remains in "Terminating" State after Uninstall

After uninstall, if the pod does not stop but remains in "Terminating" state, enter the following commands:

```
kubectl get pods -n fleet-management
kubectl delete -n <pod_name_from_above_command> --grace-period=0 --force
ansible-playbook AMR_server_containers/01_docker_sdk_env/docker_orchestration/ansible-playbooks/
02_edge_server/fleet_management/fleet_management_playbook_uninstall.yaml
```

docker-compose Failure

If you see an error message that `docker-compose` fails with some variables not defined, add the environment variables to `.bashrc` so that they are available to all terminals:

```
export DOCKER_BUILDKIT=1
export COMPOSE_DOCKER_CLI_BUILD=1
export DOCKER_HOSTNAME=$(hostname)
export DOCKER_USER_ID=$(id -u)
export DOCKER_GROUP_ID=$(id -g)
export DOCKER_USER=$(whoami)
# Check with command
env | grep DOCKER
```

Keytool Not Installed

The keytool utility is used to create the certificate store. Install any preferred Java* version. For development, Intel used:

```
sudo apt install default-jre
# Check your Java version:
java -version
```

Corrupt Database or Nonresponsive Server

Reset the ThingsBoard* server with the following steps.

1. Uninstall the playbook:

```
ansible-playbook AMR_server_containers/01_docker_sdk_env/docker_orchestration/ansible-playbooks/
02_edge_server/fleet_management/fleet_management_playbook_uninstall.yaml
```

2. After uninstalling the playbook, wait several seconds for all fleet related containers to stop. Verify that there are no fleet containers running:

```
docker ps | grep fleet
```

3. Reinstall the playbook:

```
ansible-playbook AMR_server_containers/01_docker_sdk_env/docker_orchestration/ansible-playbooks/
02_edge_server/fleet_management/fleet_management_playbook_install.yaml
```

ThingsBoard* Server Errors

These errors can be fixed directly on the hosting machine using Docker* Compose. However, this requires automated steps using Ansible* playbooks, so try these fixes last.

- Reset the database to a pristine state (without customizations from Intel):

```
# delete database and start the server
# The state of server will be - without any customization from Intel.
sudo rm -rf ~/.mytb-data/db ~/.mytb-data/.firstlaunch ~/.mytb-data/.upgradeversion
docker-compose -f 01_docker_sdk_env/docker_compose/02_edge_server/edge-server.all.yml down
CHOOSE_USER=thingsboard docker-compose -f 01_docker_sdk_env/docker_compose/02_edge_server/edge-
server.all.yml up fleet-management
```

NOTE This only restarts the ThingsBoard* server, without Intel® Smart Edge Open.

- Reset the database to the preconfigured state (with customizations from Intel), and restart the server:

```
# Start the server with old/corrupted database
CHOOSE_USER=thingsboard docker-compose -f 01_docker_sdk_env/docker_compose/02_edge_server/edge-
server.all.yml up fleet-management
# attach to running container from another terminal:
docker exec -it  edge-server-sdk-fleet-management bash

# inside the container: replace the database with Intel-customized-database:
# Just press tb<tab>. The tb-server-reset-db.sh is present in /usr/local/bin folder, so it is
accessible from anywhere.
tb-server-reset-db.sh
# When asked press y and enter. Done.
# Now exit the container. and run below commands again to re-launch the server with
preconfigured-state of database (With Intel Customizations):
docker-compose -f 01_docker_sdk_env/docker_compose/02_edge_server/edge-server.all.yml down
CHOOSE_USER=thingsboard docker-compose -f 01_docker_sdk_env/docker_compose/02_edge_server/edge-
server.all.yml up fleet-management
```

NOTE This only restarts the ThingsBoard* server, without Intel® Smart Edge Open.

- When you deploy the ThingsBoard* container using Intel® Smart Edge Open Ansible* playbook, sometimes the server cannot start due to following error:

```
edge-server-sdk-fleet-management | 2021-11-25 15:24:34,345 [main] ERROR
com.zaxxer.hikari.pool.HikariPool - HikariPool-1 - Exception during pool initialization.
edge-server-sdk-fleet-management | org.postgresql.util.PSQLException: Connection to
localhost:5432 refused. Check that the hostname and port are correct and that the postmaster is
accepting TCP/IP connections.
edge-server-sdk-fleet-management |      at
org.postgresql.core.v3.ConnectionFactoryImpl.openConnectionImpl(ConnectionFactoryImpl.java:303)
edge-server-sdk-fleet-management |      at
org.postgresql.core.ConnectionFactory.openConnection(ConnectionFactory.java:51)
edge-server-sdk-fleet-management |      at
org.postgresql.jdbc.PgConnection.<init>(PgConnection.java:223)
edge-server-sdk-fleet-management |      at org.postgresql.Driver.makeConnection(Driver.java:465)
edge-server-sdk-fleet-management |      at org.postgresql.Driver.connect(Driver.java:264)
```

If, after waiting for some time, the server is not up and running, and the server URL localhost:9090 is not showing the server page, uninstall and reinstall the playbook:

```
ansible-playbook AMR_server_containers/01_docker_sdk_env/docker_orchestration/ansible-playbooks/
02_edge_server/fleet_management/fleet_management_playbook_uninstall.yaml
ansible-playbook AMR_server_containers/01_docker_sdk_env/docker_orchestration/ansible-playbooks/
02_edge_server/fleet_management/fleet_management_playbook_install.yaml
```

Result: The database is reset to the preconfigured database provided by Intel.

Fleet Management Server Dashboard over LAN Issues

If the Dashboard is not accessible from the client, the first step is to make sure that the client and server nodes are in the same subnet. This helper page can be used to find out: https://www.meridianoutpost.com/resources/etools/network/two-ips-on-same-network.php

If the client and server are in the same subnet, then it is possible that you are using proxies that prevent the connection. To check this on Linux, run the following command:

```
wget -q -T 3 -t 3 --no-proxy http://<IP>:9090/ && echo "COMMAND PASSED"
```

Where `<IP>` is the IP of your server.

If `COMMAND PASSED` is displayed, then you should configure your browser to NOT use proxy when accessing the IP/hostname of the server.

Playbook Install Errors

If you start the basic fleet management server right after a server reboot, you may encounter the error:

```
fatal: [localhost]: FAILED! => {"changed": false, "msg": "Logging into 10.237.22.88:30003 for
user admin
failed - 500 Server Error for http+docker://localhost/v1.41/auth: Internal Server Error
(\"Get \"https://10.237.22.88:30003/v2/\": dial tcp 10.237.22.88:30003: connect: connection
refused\")"}
```

1. Wait two minutes until the server is up and running.
2. Verify that all pods are running and no errors are reported:

```
kubectl get all -A
```

3. After all pods and services are up and running, restart the basic fleet management server:

```
ansible-playbook AMR_server_containers/01_docker_sdk_env/docker_orchestration/ansible-playbooks/
02_edge_server/fleet_management/fleet_management_playbook_install.yaml
```

Terminology

Term	Description
ADBSCAN	Adaptive Density-Based Spatial Clustering of Applications with Noise
AOTA	Application Over the Air
ARIAC	Agile Robotics for Industrial Automation Competition
CNDA	Corporate Non-Disclosure Agreement
CPU	Central Processing Unit

Term	Description
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DI	Device Initialization Protocol
DL	Deep Learning
DMS	Device Management Service
DPC++	Data Parallel C++
DRM	Deterministic Road Map
EI for AMR	Edge Insights for Autonomous Mobile Robots
DPC++	Data Parallel C++
EOF	end-of-file
FDO	FIDO Device Onboard
FIDO	Fast IDentity Online
FLANN	Fast Library for Approximate Nearest Neighbors
FM	Fast Mapping
GEAR	Gazebo Environment for Agile Robotics
GPU	Graphics Processor Unit
GSLAM	General Simultaneous Localization and Mapping
GUI	Graphical User Interface
IDE	Integrated Development Environment
IE	Inference Engine
IMU	Inertial Measurement Unit
IPU	Image Processing Unit
ITS	Intelligent sampling and Two-way Search
KVM	Kernel-based Virtual Machine
LIDAR	Light Detection and Ranging
MQTT	Message Queuing Telemetry Transport
NFS	Network File System
NN	Neural Network
OSRF	Open Source Robotics Foundation
OTA	Over-The-Air
PCL	Point Cloud Library
PRM	Probabilistic Road Map
RDC	Resource and Documentation Center

Term	Description
RGBD	Red, Green, Blue plus Depth
ROS	Robot Operating System
RPLIDAR	360-degree 2D LIDAR solution developed by SLAMTEC
RPM	Red Hat* Package Manager
RTAB-Map	Real-Time Appearance-Based Mapping
RV	Rendezvous
SDK	Software Development Kit
SDO	Intel® Secure Device Onboard (Intel® SDO)
SLAM	Simultaneous Localization And Mapping
SOTA	Software Over the Air
SSD	Single-Shot multibox Detection
SSL	Secure Sockets Layer
TLS	Transport Layer Security
TMI	Test Module Interface
UEFI	Unified Extensible Firmware Interface
VNC	Virtual Network Computing
vSLAM	Visual Simultaneous Localization and Mapping

Notices and Disclaimers

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

All product plans and roadmaps are subject to change without notice.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

Intel technologies may require enabled hardware, software or service activation.

No product or component can be absolutely secure.

Your costs and results may vary.

Code names are used by Intel to identify products, technologies, or services that are in development and not publicly available. These are not “commercial” names and not intended to function as trademarks.

Performance varies by use, configuration and other factors. Learn more at www.Intel.com/PerformanceIndex.

© Intel Corporation. Intel, the Intel logo, and other Intel marks are trademarks of Intel Corporation or its subsidiaries. Other names and brands may be claimed as the property of others.

This software and the related documents are Intel copyrighted materials, and your use of them is governed by the express license under which they were provided to you (License). Unless the License provides otherwise, you may not use, modify, copy, publish, distribute, disclose or transmit this software or the related documents without Intel's prior written permission.

This software and the related documents are provided as is, with no express or implied warranties, other than those that are expressly stated in the License.