Intel® In-Band Manageability Framework

User Guide – Azure*

June 2021

Revision 2.8

Intel Confidential

Document Number: 626762-2.7

Contents

Figures

Tables

Revision History

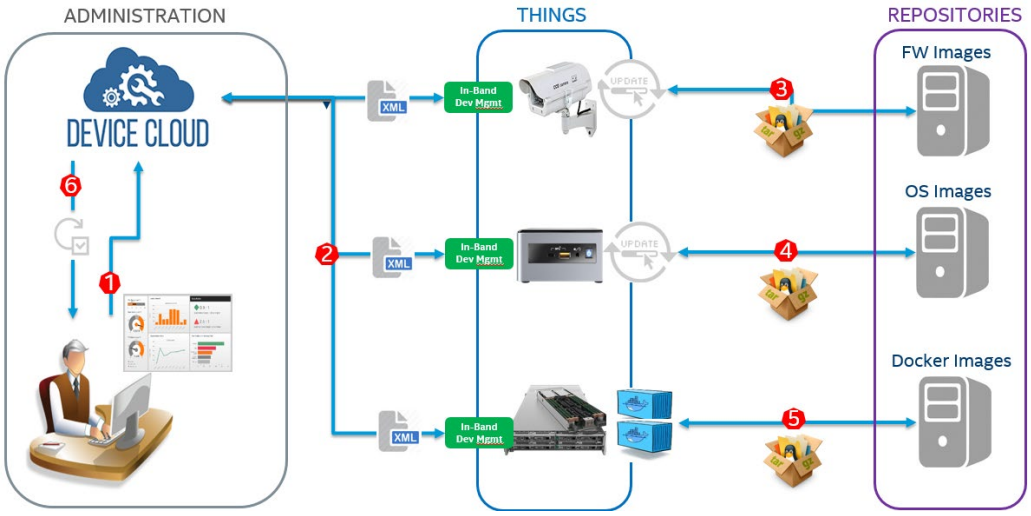
Date	Revision	Description
June 2021	2.8	Added provision-tc parameters info. Added clarity on AOTA package generation.
May 2021	2.x	Added the X509 authentication mechanism instructions and X509 based OTA package verification.
April 2021	2.x	Added customer NOTE on Trusted repositories.
September 2020	2.7	Updated on AOTA command.
August 2020	2.6	EIS 2.3, ECS 1.5 and Platform releases.
May 2020	2.1.1	EIS 2.2 release.

1.0 Introduction

The Intel® In-Band Manageability Framework (also known as INB) is a software running on the Edge IoT Device which enables an administrator to perform critical Device Management operations over-the-air remotely from the cloud. It also facilitates the publishing of telemetry and critical events and logs from the Edge IoT device to the cloud enabling the administrator to take corrective actions if, and when necessary. The framework is designed to be modular and flexible ensuring scalability of the solution across preferred Cloud Service Providers (for example, Azure* IoT Central, Telit DeviceWISE, ThingBoard.io, and so on).

Some of the key advantages of Intel® In-Band Manageability solutions are:

1. Out-of-box cloud support: Azure* IoT Central, Telit DeviseWise, ThingsBoard.io.
2. Single interface to handle OS, FW and Application (Docker container) updates.
3. Scalable across Intel x86 (Intel® Atom® and Intel® Core®) architectures SoCs and on Vision platforms from Intel.



This document provides detailed instructions on how to provision a device with **Azure* IoT Central.**

The Device Management use-cases covered by the Intel® In-Band Manageability Framework are listed in the table below:

Use-cases	Notes
Update	- System (OS), Software-over-the-air (SOTA) - Firmware-over-the-air (FOTA) - Application-over-the-air (AOTA)
Telemetry	- System attributes - Events - Devices States - Usage data
Recovery	- Rollback post updates. - System Reboot/Shutdown

Embedded within the Intel® In-Band Manageability Framework are features which ensure Security and Diagnostics aspects:

Feature	Notes
Security	- ACL for trusted repositories - Mutual TLS authentication between services - TPM to store framework secrets
Diagnostics	- Pre and Post OTA update checks - Periodic system checks

1.1 Purpose

This User Guide serves to provide the reader an overview on how to:

- Login and setup Azure* IoT Central portal
- Provision the Edge IoT device running the Intel® In-Band Manageability Framework
- Perform OTA updates through Azure* IoT Central portal.

It also provides examples of the Web-UI configuration, reported Telemetry from device and commands for performing OTA updates.

1.2 Audience

This guide is intended for

- Independent BIOS Vendors providing Firmware Update packages to ensure FW update binary packaging.
- Independent Software Vendors (ISV) providing OS and Application update packages.
- System Integrators administrating devices running the Intel® In-Band Manageability framework.

1.3 Terminology

Term	Description
AOTA	Application Over the Air (Docker)
BIOS	Basic Input Output System
Device	A device is any equipment that is installed to be monitored or controlled in a building. Examples of devices include light switches, thermostats, cameras, other mechanical loads, chillers, cooler, and so on.
FOTA	Firmware Over the Air
FW	Firmware
INB	Intel® In-Band Manageability Framework
IoT	Internet of Things
OS	Operating System
OTA	Over-the-air
SMBIOS	System Management BIOS
SOTA	Software Over the Air (OS update)

§

2.0 Azure* Overview

2.1 Azure* Setup Overview

Creating an Azure* account and obtaining the connection tokens from Azure* is required for provisioning or enabling Over-the-Air updates. For reference and quick setup, you will also need to import INB's IoT Central Application which will provide the same UI interface described in this document to monitor the device and perform OTA commands.

This section will walk through the setup steps:

- Create your Azure* portal account
- Importing the Intel® In-Band Manageability Framework IoT Central Application
- Creating a device and obtaining its Connection (SAS) tokens
- Provisioning the Intel® In-Band Manageability Framework on Edge Device with the SAS tokens
- Working with Azure* to perform OTA operations

2.2 Create Your Azure* account

In order to setup an Azure* account, follow the steps below:

Creating an Azure* Account

- If not done already, an Azure* account can be created through the link below: https://azure.microsoft.com/en-us/free/

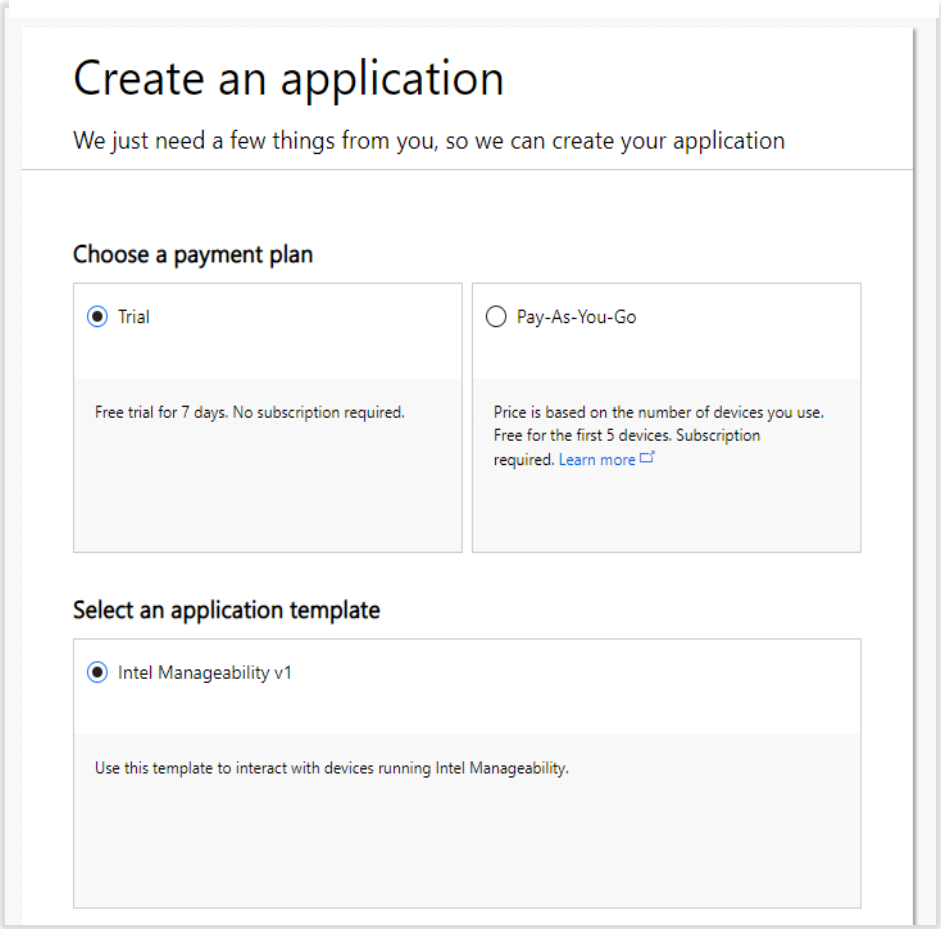
Accessing Azure*:

- Azure* portal can be accessed at: https://portal.azure.com/#home
- If an Azure* IoT Central has already been set up, it can be accessed at: https://apps.azureiotcentral.com
- Otherwise, refer to Section 2.2.1 to set up an IoT Central application

2.2.1 Setting up an Azure* IoT Central Application

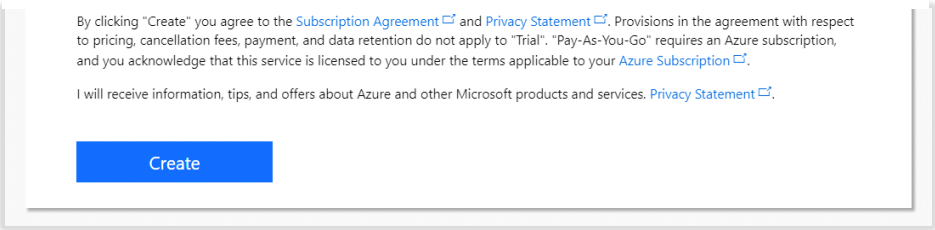
- To use the reference Intel® In-Band Manageability Framework IoT Central application, go to the link contained in */usr/share/cloudadapter-agent/azure_template_link.txt* on your edge device.
- Log in with an Azure* Account when prompted.

Figure 1. Create an Application



- The following form will appear:

Figure 2. Click Create



- Fill out the form accordingly, then click **Create.**
- After provisioning, the IoT Central application with premade device templates and dashboards will appear. As noted before, this can be accessed at

https://apps.azureiotcentral.com under *My Apps* tab or through the Azure* portal.

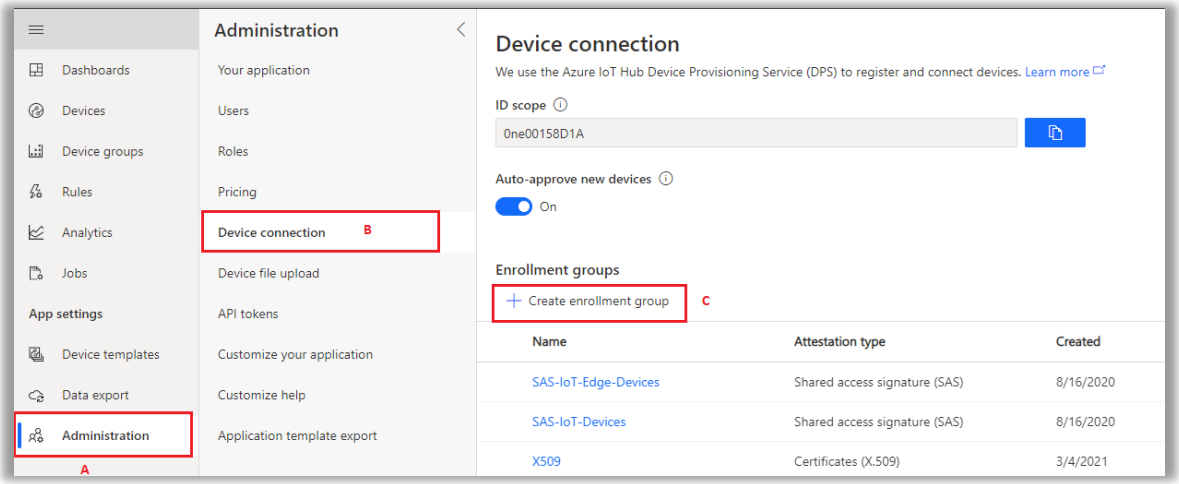
2.3 X509 based enrollment

The following Dashboard screen appears once the application is created. The user can enroll for a X509 based enrollment group to enroll the intermediate or root CA signed certificate, to authenticate the device further by using the X509 authentication mechanism.

This step is necessary only if the user requires the X509 authentication on the devices, else this step can be ignored.

To create an Enrollment Group, click **Administration** [A], **Device Connection** [B], **Create enrollment group** [C]. Refer to Figure 3.

Figure 3. Device Enrollment Groups

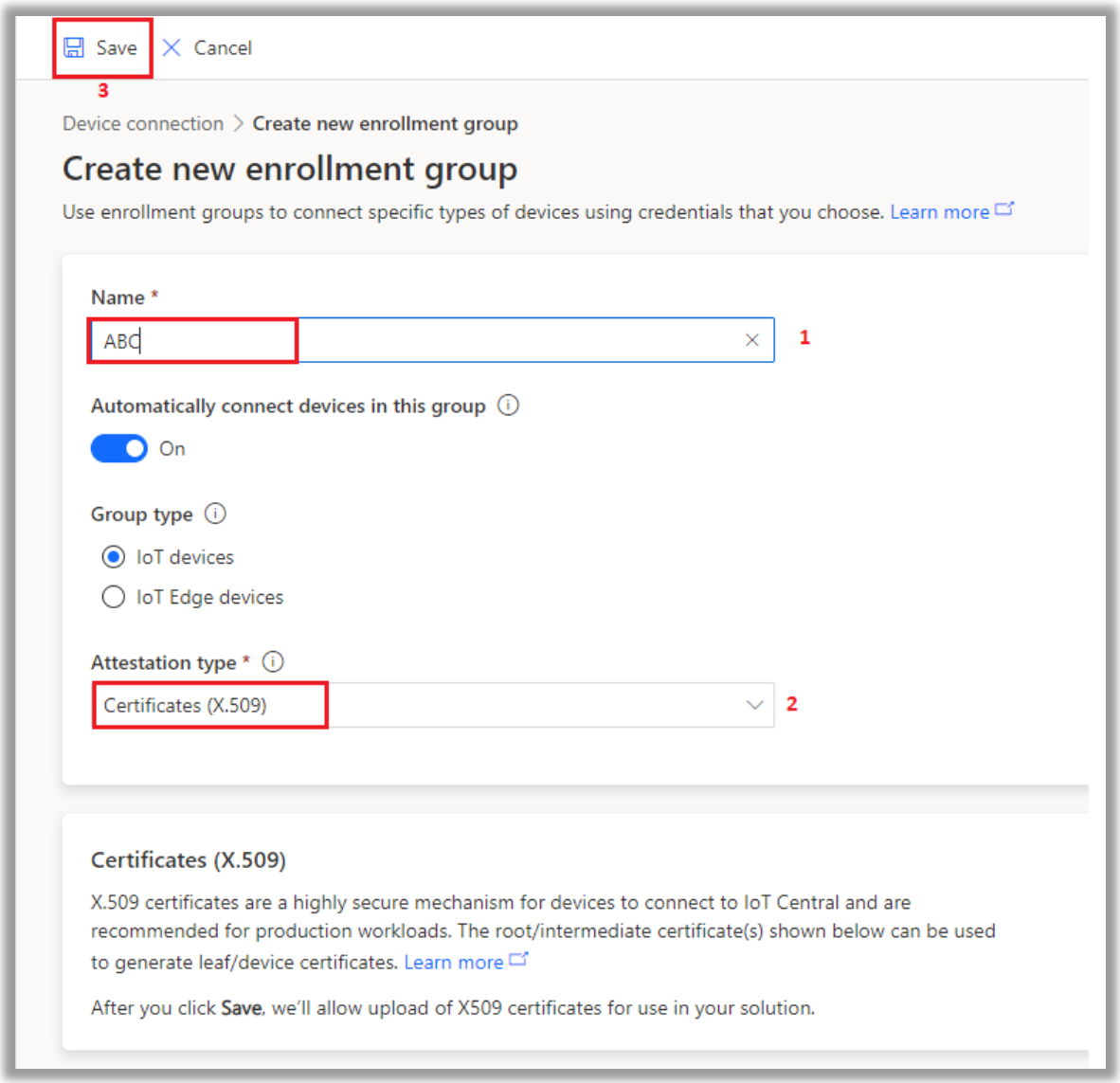


This opens a form as shown in Figure 4.

1. Fill in the Enrollment group name.
2. Select **Attestation type** as *Certificates (X509).*

3. Click **Save**.

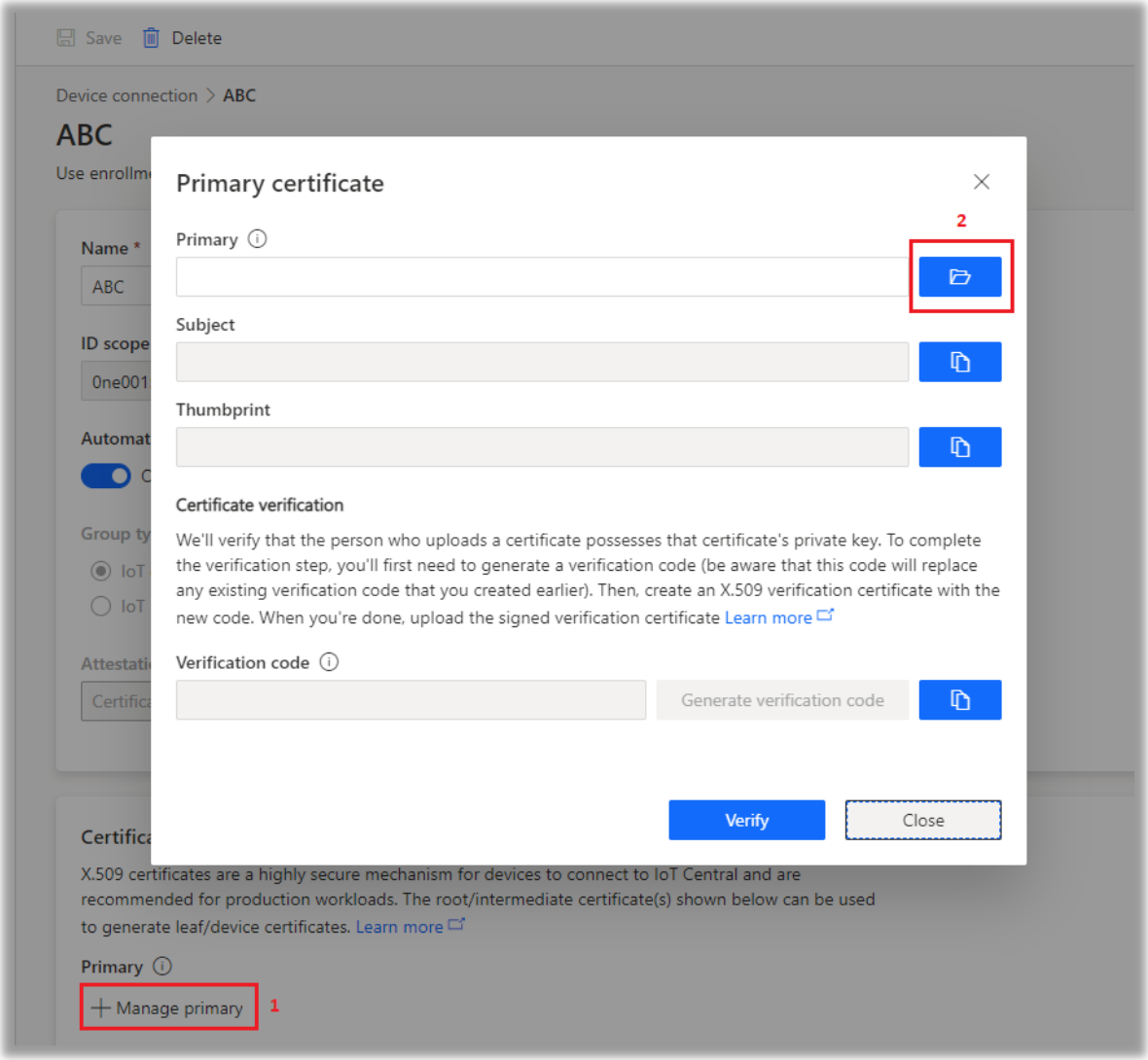
Figure 4. Create New Enrollment Group



Once the group is saved, the user needs to upload root or intermediate certificate as shown in Figure 5:

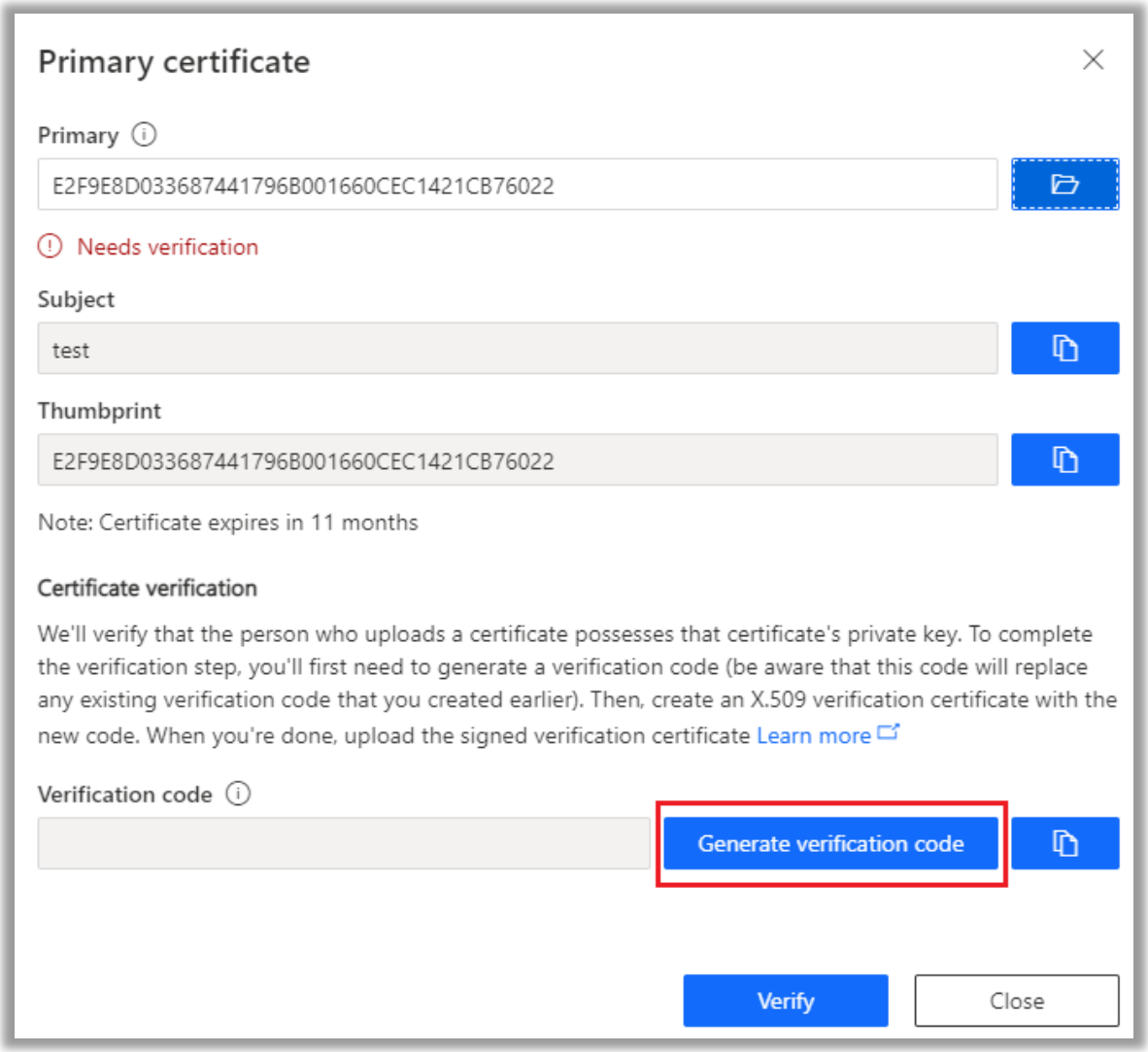
1. Click **Manage Primary**.
2. Select the folder button as show below. This opens a window where user chooses the certificate from the currently operated user device.

Figure 5. Primary Root or Intermediate Certificates



After uploading the certificate, click **Generate verification code** as shown in Figure 6.

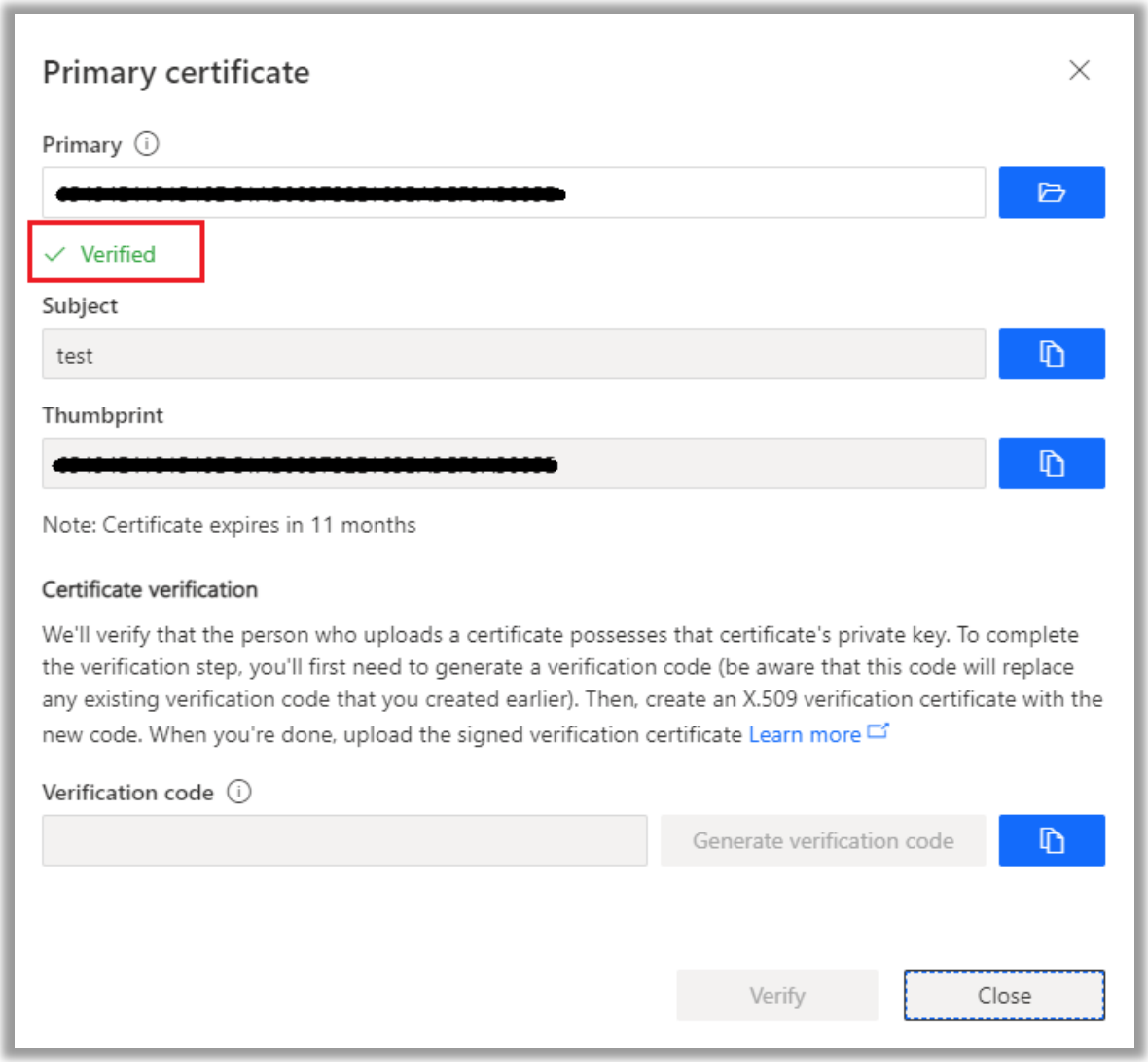
Figure 6. Generate Verification Code



After the verification code is populated in the text box adjacent to the **Generate verification code** button, the user needs to use this verification code to generate a verification certificate which will later be uploaded after clicking the **Verify** button in the form shown in Figure 6.

Once the verification is done by the portal, the following screen displays stating verification is successful. Next, click **Close** button to close the form.

Figure 7. Verification success

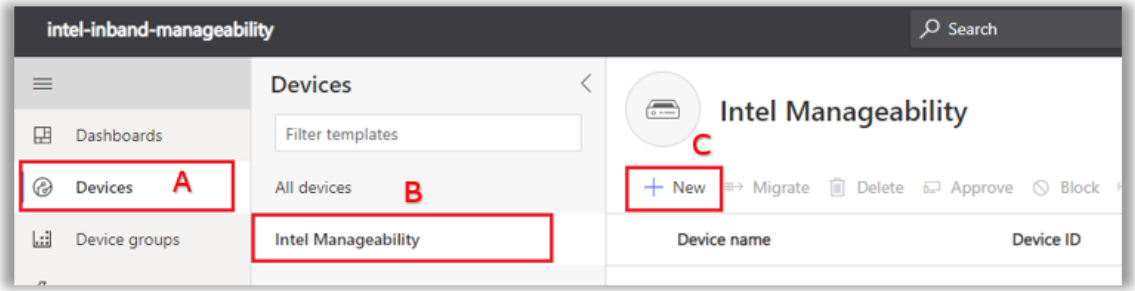


2.4 Obtaining Device Credentials

To connect a device to the Azure* portal, a device needs to be created first on the portal with the template that the user wishes to associate the device with. The device created will have a name and an auto-generated device-id, device-scope-id, and a shared access primary key, which will be later used on the user's device, while provisioning the device to Azure* cloud.

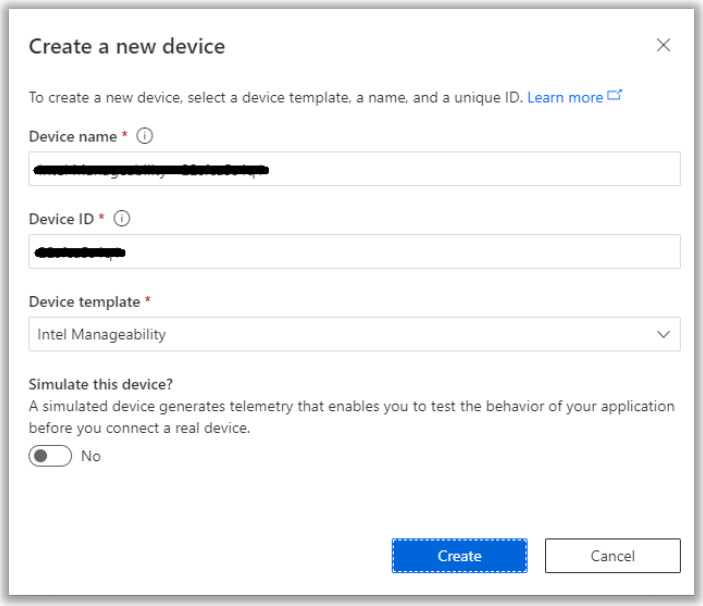
When accessing the dashboard for the IoT Central application, the following screen will appear. In **Devices** tab **A**, select Template (**Intel Manageability**) **B** and click **New C**. Refer to Figure 8.

Figure 8. Devices



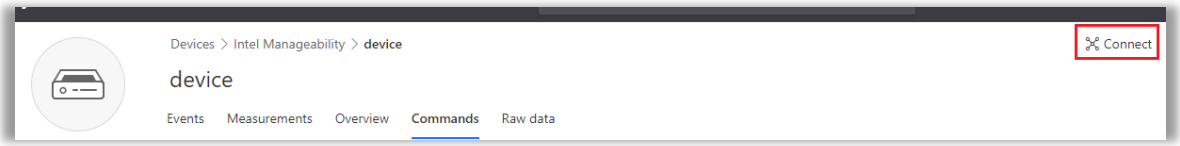
A new device registration form appears as shown in Figure 9. Fill in the **DeviceID** and **Device Name** information and click **Create**.

Figure 9. Create a New Device



The newly created device will appear on the Dashboard with the specified Device Name. Click the device created, the status will be shown as *Registerer*. Then, click **Connect** as shown in Figure 10.

Figure 10. Create a new device

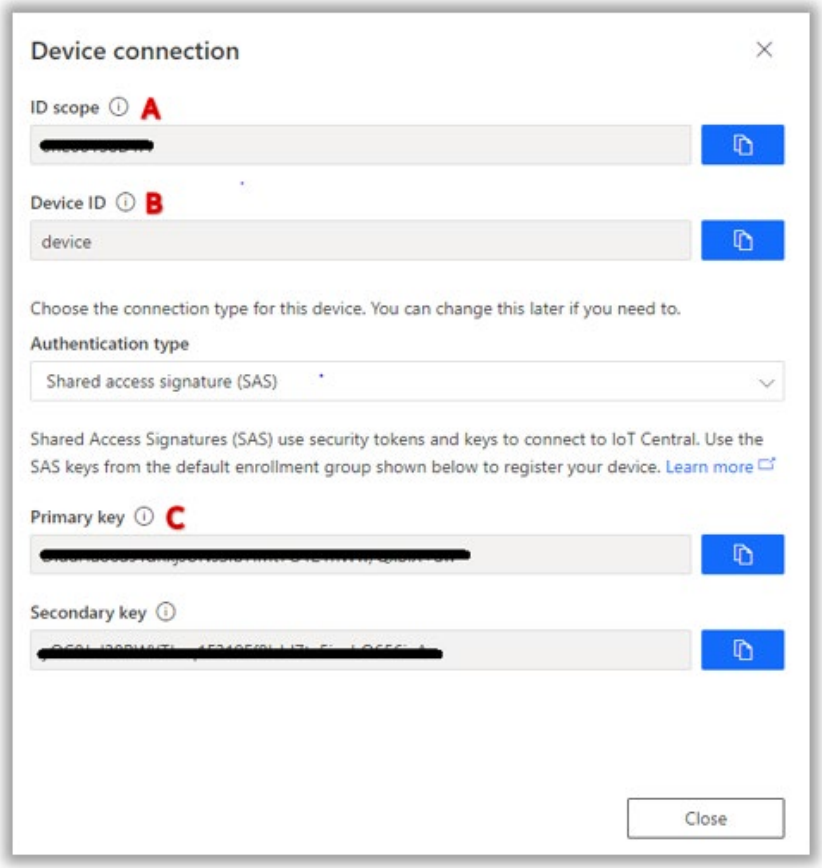


As INB supports both SAS and X509 authentication types, the user must choose one of the Authentication types supported. If the user intends to select SAS based authentication, refer to Shared Access Signature (SAS) authentication. Else, if the user wants X509 based Authentication, refer to X509 Authentication.

2.4.1 Shared Access Signature (SAS) authentication:

By clicking the 'Connect' button shown in Figure 5, in the dialog that appears, **note** the **Scope ID [A]**, **Device ID [B]**, and **Shared Access Key(SAS) [C]**, as these information will be used to provision the device as depicted in Figure 11:

Figure 11. Scope ID, Device ID, SAS



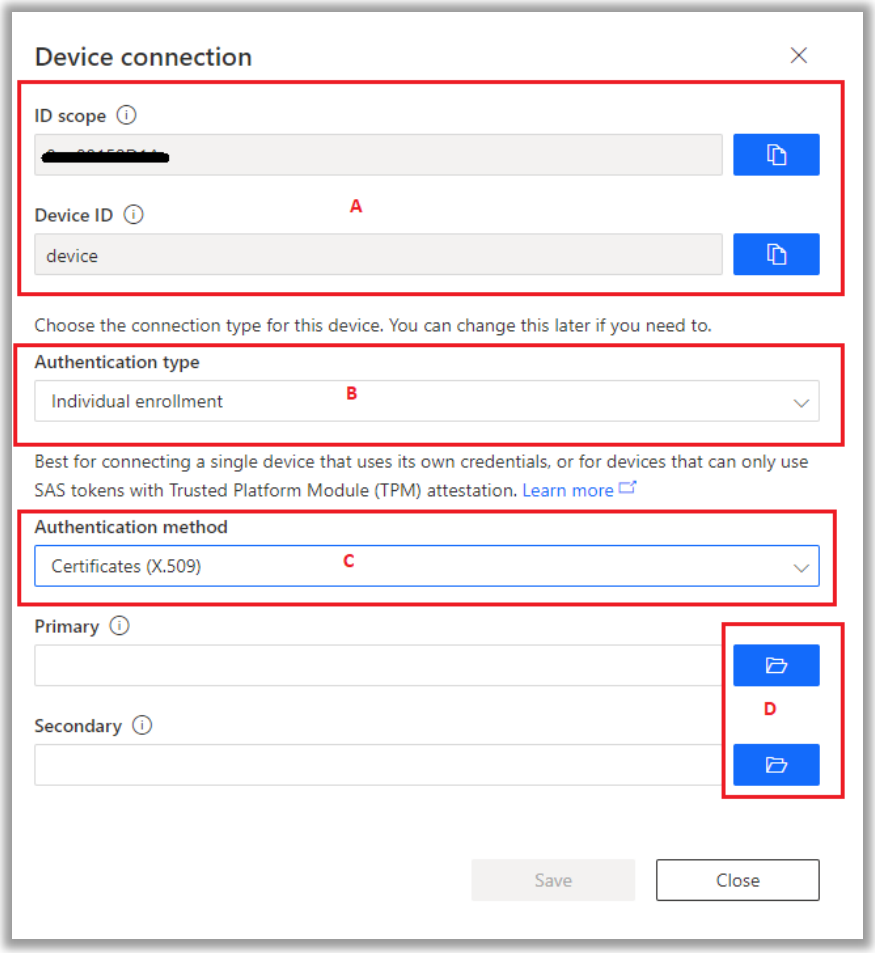
2.4.2 X509 authentication:

Note: To authenticate a device using X509 mechanism, a X509 based enrollment group needs to be created with CA signed root or intermediate certificates. The verification of the private key possession needs to be done as shown in Section 2.3.

The user needs to generate a primary and secondary device certificates using the root or intermediate certificate used to enroll in Section 2.3.

Once the device certs are generated, visit the Azure* portal, select the device created earlier. Then click the **Connect** button shown in Figure 5. In the dialog that appears, **note** the **Scope ID and Device ID [A]** as the information will be used to provision the device. Next, select Authentication type as **Individual Enrollment [B]**, Authentication Method as **Certificates (X509) [A]**, and use the folder icons **[D]**, to upload the device primary and secondary certificates as shown in Figure 12:

Figure 12. Scope ID, Device ID, Authentication Method and Enrolling Device Certificates



2.5 Provisioning a Device

Provisioning is a Device Management phase during which the Edge IoT Device is configured with credentials to ensure that it can establish a secure session with the Device Management backend. This usually involves assigning Device ID's and Secure tokens/keys which the Device may use to identify and authenticate itself to the remote Device Management Portal.

NOTE ON PREREQUISITE AND ASSUMPTIONS:

1. The Intel® In-Band Manageability Framework is installed on the Edge IoT device.
2. The date and time on the edge device needs to be set correct
3. Device credentials (for example, Device ID, Scope ID, SAS token) that have been obtained from the Azure* portal.

- Launch the provisioning script using the command:

```
$ sudo provision-tc
```

- If the device was previously provisioned, the following message appears. To override the previous cloud configuration, press **Y**:

```
A cloud configuration already exists: "telit"
Replace configuration?
[Y/N] Y
```

- Press **2** and **[ENTER]** for Azure* to choose a cloud service :

```
Please choose a cloud service to use:

1) Telit Device Cloud  3) ThingsBoard
2) Azure IoT Central   4) Custom
#? 2
```

- Next, enter the information for **Scope ID**, **Device ID**, and the **Shared Access Key**. Use the information collected in Section 2.4**:**

```
Please enter the device Scope ID:
dEviCeScopeID1234

Please enter the Device ID:
Device-ID-1234
```

- Then, the user is required to select the authentication mechanism.

```
Please choose provision type.
1: SAS key authentication
2: X509 authentication
```

- When the user selects 1: SAS key authentication, a prompt to enter SAS key is seen, the SAS key information can be obtained by following the steps in section 2.4.1:

```
Please enter the device SAS primary key (Hint: https://docs.microsoft.com/en-us/azure/iot-central/howto-generate-connection-string
```

- If the user selects 2: X509 authentication, the following prompt appears to confirm that the user has the device certificates generated.

```
Configuring device to use X509 auth requires device certificate verification.


Are device certs and keys generated?  [Y/N]
```

- If the user selects '**N**', the provisioning exists stating that the device certificates are required to proceed further.
- If the device certificates are already generated, select '**Y**' and the user is requested to upload the certificates.

```
Please enter a filename to import
Input path to Device certificate file (*cert.pem):
/home/certs/device_cert.pem
```

- The user would be required to enter the path to the device key file:

```
Input Device Key from file? [Y/N] y

Please enter a filename to import
Input path to Device Key file (*key.pem):
/home/certs/device_key.pem
```

- Once the information is provided by the user and if the cloud provisioning is successful, the following message appears:

```
Successfully configured cloud service!
```

- A Yes/No user prompt appears requesting for a certificate verification on an OTA package. Choose **'Y'** if FOTA or Config load packages need to be verified using signature, else choose **'N'**.

```
Signature checks on OTA packages cannot not be validated without provisioning a cert file.
Do you wish to use a pre-provisioned cert file for signature checks for OTA packages? [Y/N]
```

- The script will then start the INB services; when the script finishes, the device should be able to interact with its associated IoT Central Application. To verify whether the device is provisioned to the right device on the Azure* portal, check the status of the device created in Section 2.4. The device will be shown as 'Provisioned' on the top-right corner. Refer to Section 2.6.

- To verify the connectivity,
  - Check to see if telemetry or events appear; see Section 2.6.
  - Alternatively, trigger a command like Reboot; see Section 2.6.

- If at any time the cloud service configuration needs to be changed or updated, run this provisioning script again.

2.5.1 Provisioning Command Parameters

Provisioning can be done with or without TPM security by setting 'PROVISION_TPM'. 'PROVISION_TPM' can be set to:

- auto: use TPM if present; disable if not present; do not prompt.
- disable: do not use TPM.
- enable: use TPM; return error code if TPM not detected.
- (unset): default behavior; use TPM if present, prompt if not.

To run provisioning that automatically detecting the present of the TPM security:

```
$sudo PROVISION_TPM=auto provision-tc
```

To run without the TPM security:

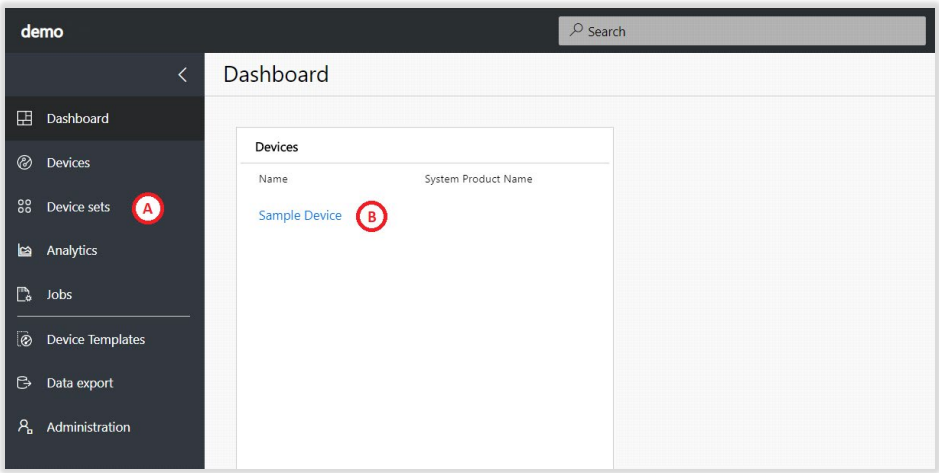
```
$sudo PROVISION_TPM=disable provision-tc
```

2.6 Using the IoT Central Application

2.6.1 Viewing and Managing Devices

- To view and manage devices, go to the **Devices** tab on the side panel Ⓐ
- Alternatively, to quickly view a device, use the **Devices** panel Ⓑ

Figure 13. Device Panel

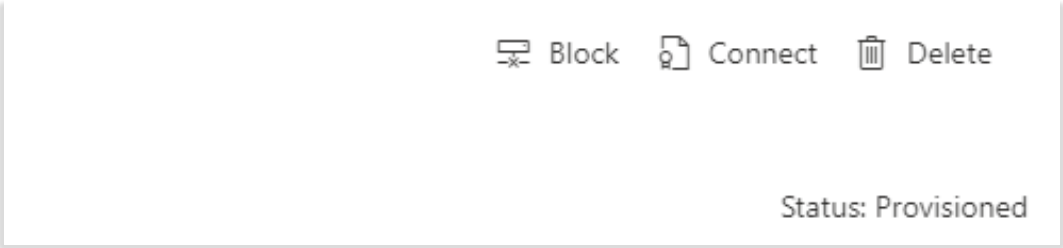


If the device list is showing an error: Refer to Section 5.1.

2.6.2 Navigating the Device Interface

First, view a device using instructions from Section 2.5.1.

Figure 14. Provisioned Status

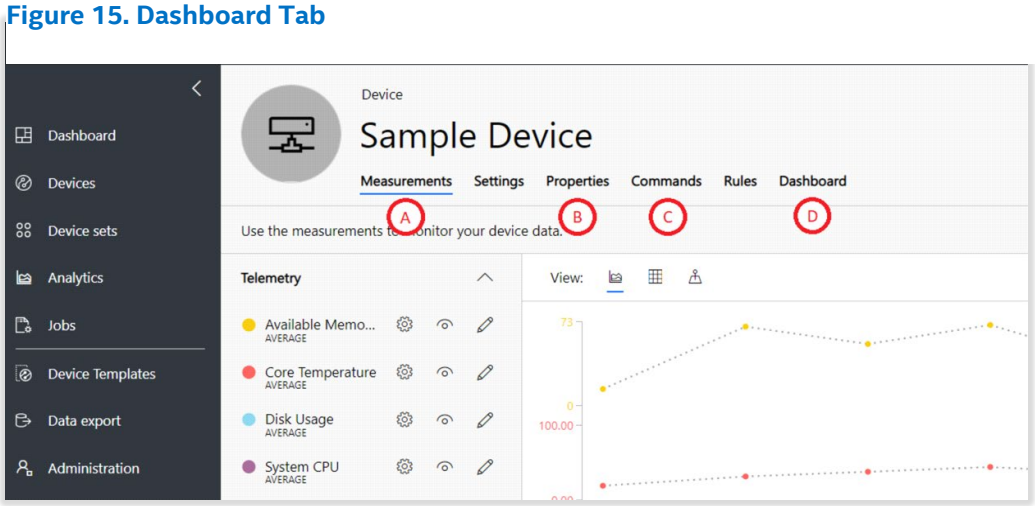


- If the device is successfully provisioned, the status of the device will be shown as Provisioned on the top-right corner.

- Upon viewing a device, the **Measurement** tab Ⓐ is displayed, where the device's telemetry and events can be seen
- To see the device's Attributes, click the **Properties** tab Ⓑ
- To trigger methods from the cloud, click the **Commands** tab Ⓒ.

  Refer to Section 3.3 for additional instructions on how to trigger methods.
- To see an overview of the device, including the Properties and the Event log, click the **Dashboard** tab Ⓓ

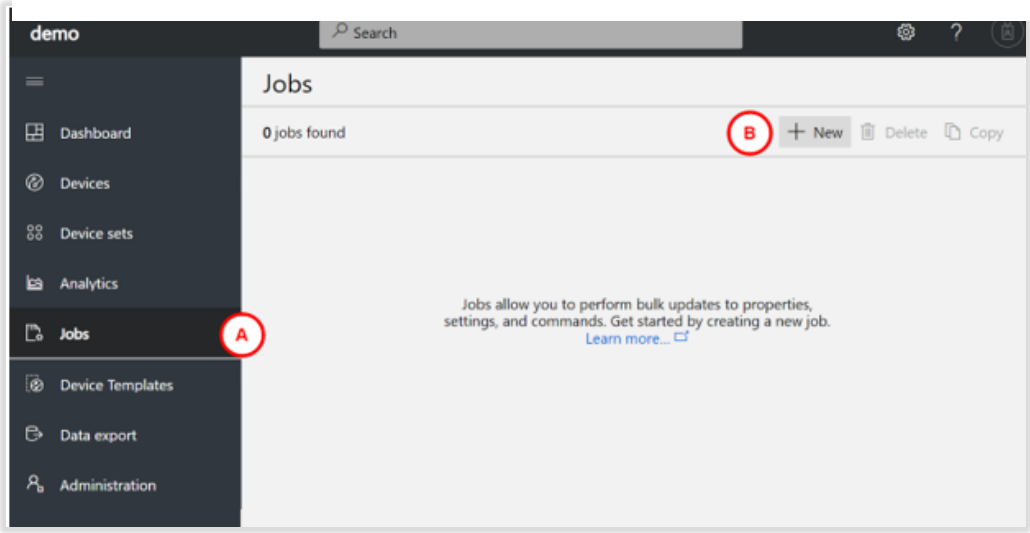
Figure 15. Dashboard Tab



2.6.3 Performing batch operations

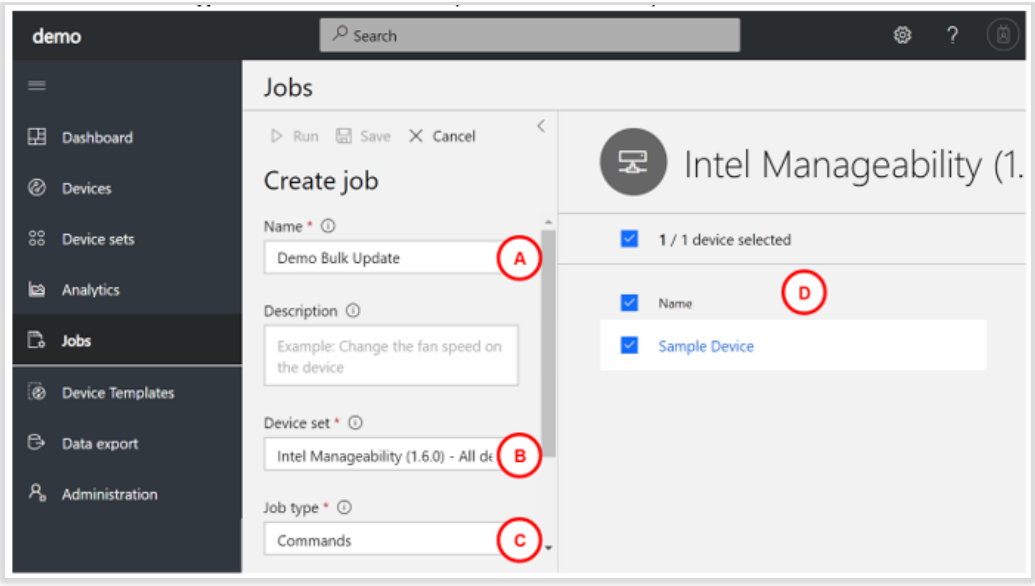
1. To perform a batch OTA operation, i.e. send the same OTA command to multiple IoT Devices at the same time, click the **Jobs** tab Ⓐ, then click **New** Ⓑ:

Figure 16. Jobs Tab



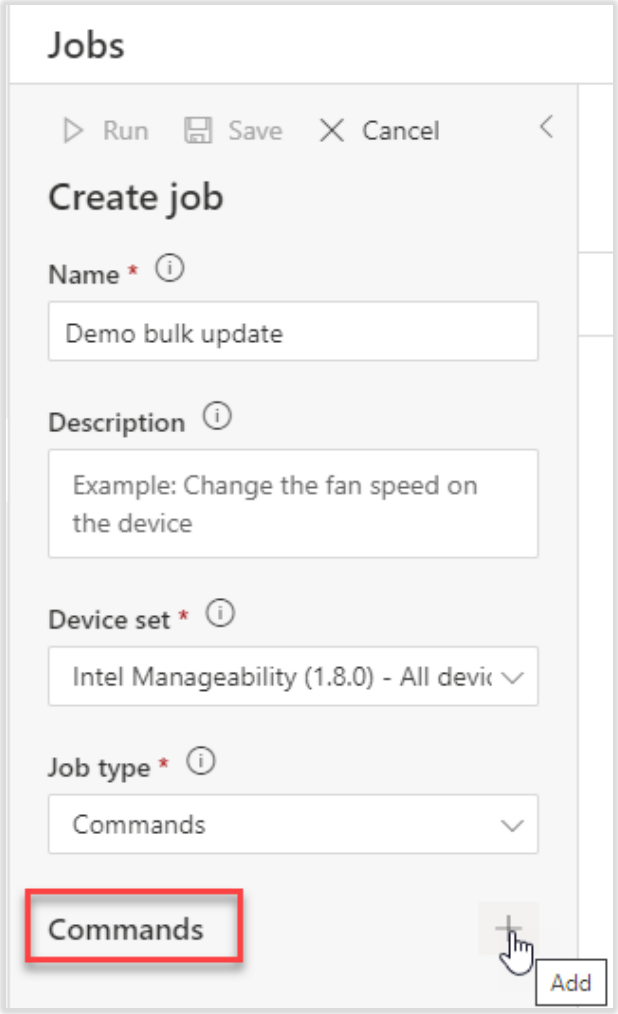
2. Type out a meaningful name Ⓐ, then select an Intel Manageability device set to use Ⓑ, the "Commands" **Job type** Ⓒ, and the devices to perform the batch operation Ⓓ.

Figure 17. Intel Manageability Device



3. The **Commands** header should now appear in the **Create Job** panel; click the adjacent plus-sign button, then select an operation to perform:

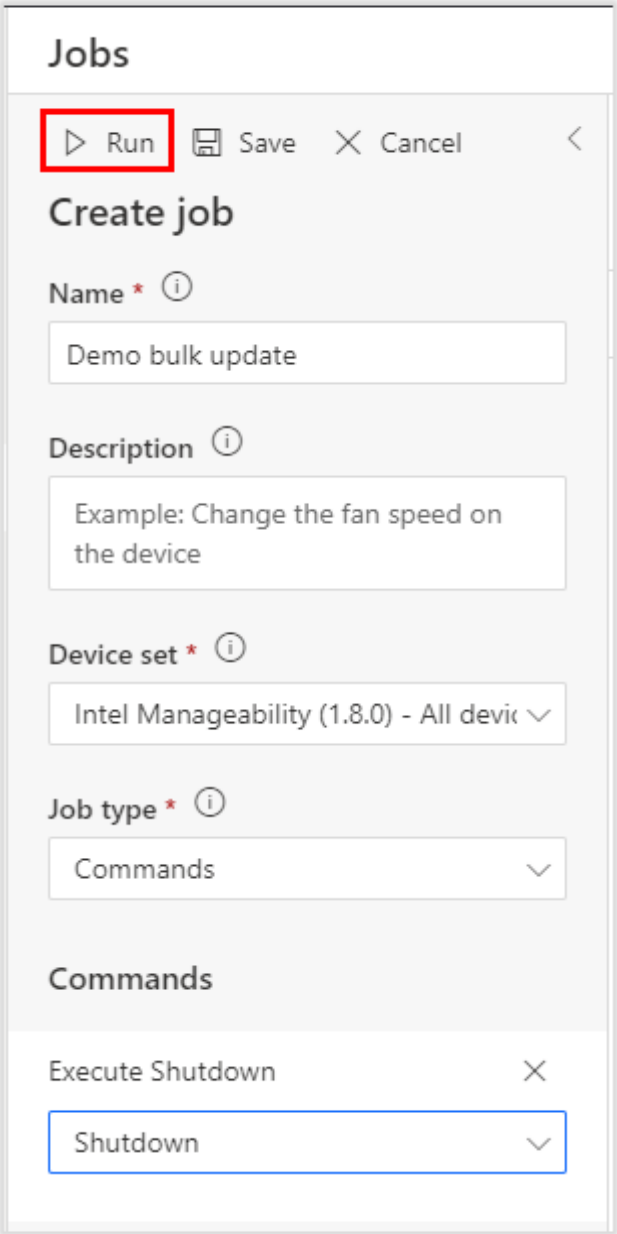
Figure 18. Select Operation



4. Fill out any necessary fields that appear after selecting the command; see Section 3.3 for more info.

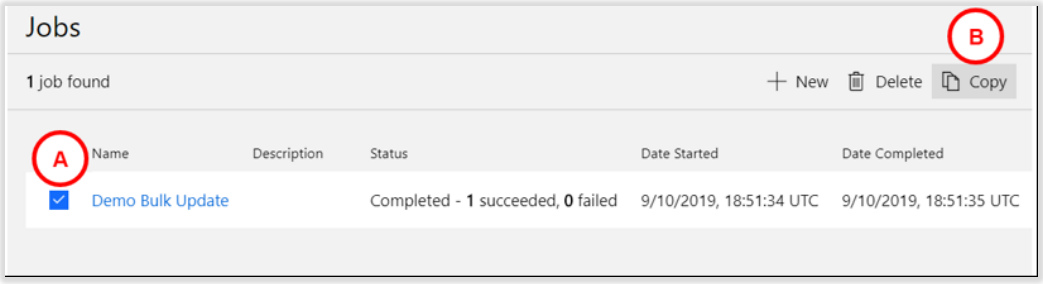
5. Finally, click the **Run** button at the top of the panel to run the bulk operation:

Figure 19. Run Job



6. To run the same batch command again, click **Jobs** tab on the left side panel. Then check box next to the batch command Ⓐ, then click **Copy** Ⓑ and follow step 5:

Figure 20. Run the Same Batch Command



§

3.0 OTA Updates

After the Intel® In-Band Manageability Framework running on the Edge IoT Device is provisioned, it will establish a secure session with the Azure* portal and the status of the device can is visible as 'Provisioned' – refer to Section 2.5.2.

Users shall be able to perform the updates listed below on the device that is provisioned:

- AOTA (Application Over the Air update)
- FOTA (Firmware-over-the-Air update)
- SOTA (Software/OS-over-the-Air update)
- Config Update (configuration parameter update)
- Power Management (Remote Shutdown and Restart)

3.1 Trusted Repositories

As part of a security measure, INB requires that the location of the OTA update repository be included in a "trusted repository list" which is maintained internally. Hence, it is **mandatory** that the OTA download URL be included in the "trusted repository list" prior to initiating an OTA update command. This can be achieved via manually updating a configuration file which includes the trusted repository list, or via an OTA command itself.

IMPORTANT NOTE: It is critical for the user to ensure that the OTA packages are hosted in secure repositories. This is outside the scope of INBM.

OTA Configuration Update

Follow the steps for Config Append in Section 3.7 to append the URL to the existing list.

To delete the entire list and add a new entry, follow the steps for Config Set in Section 3.7.

To remove an entry from the list, follow steps for Config Remove in Section 3.7.

NOTE: *If the URL from which the package for an OTA update is being fetched doesn't exist in the trustedRepositories list, INB would abort the update since the fetch URL is not in the trusted list.*

Manual Configuration Update

Refer to **Developer Guide Documentation**.

3.2 Preparing OTA Update Packages

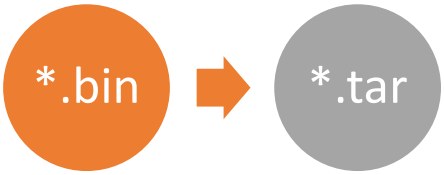
Before updates can be dispatched to the endpoint, some preparation needs to be done at the repository server to facilitate the updates.

3.2.1 Creating FOTA Package

The FOTA package structure remains the same when signature is used. For a more secure FOTA update, users can provision a device with a PEM file containing the signing certificate to validate the downloaded file against a signature provided as part of the OTA command, refer to How to generate Signature to generate signature. Users may create a PEM file using the OpenSSL and Cryptography libraries.

1. **With Signature:** FOTA package structure with signature accepts a *tar* (archive) file or just a binary file as a FW update package. If using a *tar* file, the *tar* file should consist of the firmware update binary (e.g., *.bin, *.cap, and so on) file as a capsule. Archiving the *.bin file with a *tar* archive tool can be performed with the below command:

```
$ tar cvf ifwi_update.tar ifwi_update.bin
```



When a device is provisioned with a PEM file to check the signature, the expectation is that every FOTA method triggered with a firmware package is validated against the signature using the provisioned PEM file.

Note: When using the secure method, do ensure to send the signature generated for the *.tar file. Refer How to generate Signature

2. **Without Signature:** FOTA package structure without signature only accepts a single firmware update binary (e.g., *.bin, *.cap, and so on) file as a capsule.

3.2.2 Creating SOTA Package

SOTA on Ubuntu Operating System does not require any SOTA package.

SOTA on Yocto is handled by INB based on OS implementation:

1. Debian package manager: in does not require any SOTA package creation but instead requires the APT repositories set correctly and path included in the apt resources.
2. Mender.io: These involve OS update images, also known as **mender artifacts**, generated by the build infrastructure. More information on mender integration can be found at https://docs.mender.io .

3.2.3 Creating AOTA Package

AOTA Package structure for the below commands should follow the below format

Table 1. Creating AOTA Package

AOTA Command	AOTA Package structure

AOTA Docker-Compose package
 	
(Same format for up/pull)
	Container Tag == Container Image Name
	
	Example: The container Image name and the tar file name should be the same
	
	Container Tag =CPU
	
	Tar file = CPU.tar.gz
	
	Note: The Tar file should contain a folder with the same name CPU. This folder CPU, needs to have the docker-compose.yml file.
	
	Steps:
	
	1.Make a folder
	
	`$ mkdir CPU`
	
	2.Copy the docker-compose.yml file into the folder
	
	`$ cp docker-compose.yml CPU/.`
	
	3.Tar the folder

	`$ tar -cvzf CPU.tar.gz CPU`
AOTA Docker Load/Import	Package needs to be **tar.gz** format The package needs to have a folder within with the same name as the package.

3.2.4 Creating Configuration Load Package

The Configuration load package structure remains unchanged when signature field is used. For a more secure OTA update, users can provision a device with a PEM file containing the certificate to validate the downloaded file against a signature provided as part of the OTA command, refer to **How to generate Signature**. Users may create a PEM file using the OpenSSL and Cryptography libraries.

1. **With Signature**: Configuration Load package structure with signature accepts both tar file with the intel_manageability.conf file or just the intel_manageability.conf file alone. Archiving the intel_manageability.conf file with a *tar* archive tool can be performed with the below command:

```
$ tar cvf conf_update.tar intel_manageability.conf signing_cert.pem
```

When a device is provisioned with a PEM file to validate the downloaded config file or package, it is expected that every Config Load method triggered with a firmware package will be having a signature that is validated against the signature using the provisioned PEM file.

2. **Without Signature**: Configuration Load package structure with no signature only contains intel_manageability.conf file.

3.2.5 How to Generate Signature

To generate certificate, private key and signatures, OpenSSL or Cryptography libraries can be used.

Once the above are generated, to validate the OTA package for FOTA/Config Load, we need to have the device provisioned with a certificate (cert.pem). While triggering OTA command from cloud, fill the signature field in the OTA form before clicking 'Execute' to trigger OTA.

Note: While creating a signature INB strictly enforces to use shar-256 or sha-384 based encryption mechanism.

3.3 OTA Commands

To trigger OTA commands on the device provisioned with Azure*, navigate to the 'Commands' tab of the device on the portal as stated in Section 2.5.2.

Table 2. Commands - Definition and Usage

Command	Definition
Trigger AOTA	Remotely launch/update docker containers on the Edge IoT Device
Trigger FOTA	Update the BIOS firmware on the system
Trigger SOTA	User-friendly, parameter driven updates to OS software packages on the system
Trigger Config Update	Update the Intel® In-Band Manageability configurations
Reboot	Remotely reboot the Endpoint
Shutdown	Remotely shut down the Endpoint
Decommission	Decommission a device from the cloud.
Manifest Update	Any OTA update type can be done via the Manifest Update, by entering XML text to update the Endpoint. **(Refer Developer Guide)**

3.4 AOTA Updates

Supported AOTA commands and their functionality:

'docker-compose' commands currently supported:

Table 3. 'docker-compose' Commands

'docker-compose' Command	Definition
Up	Deploying a service stack on the device
Down	Stopping a service stack on the device
Pull	Pulls an image or a repository from a registry
List	Lists containers
Remove	Removes docker images from the system

'docker' commands currently supported:

Table 4. 'docker' Commands

'docker' Command	Definition
Import	Importing an image to the device
Load	Loading an image from the device
Pull	Pulls an image or a repository from a registry
Remove	Removes docker images from the system
Stats	Returns a live data stream for all the running containers

'Application' command currently supported:

'application' Command	Definition
Update	Updating an application package

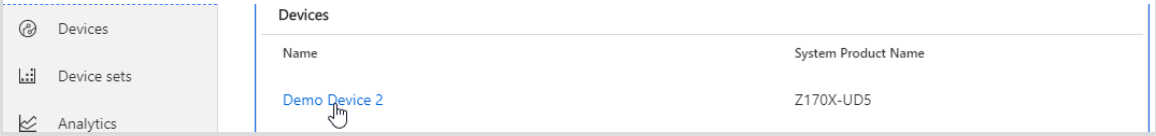
Table 5. List of AOTA Commands that are Not Supported

Docker-Compose	Import
	Load
	Update
	Stats
Docker	Up
	Down
	Update
	List
Application	Up
	Down
	List
	Remove
	Pull
	Load
	Stats
	Import

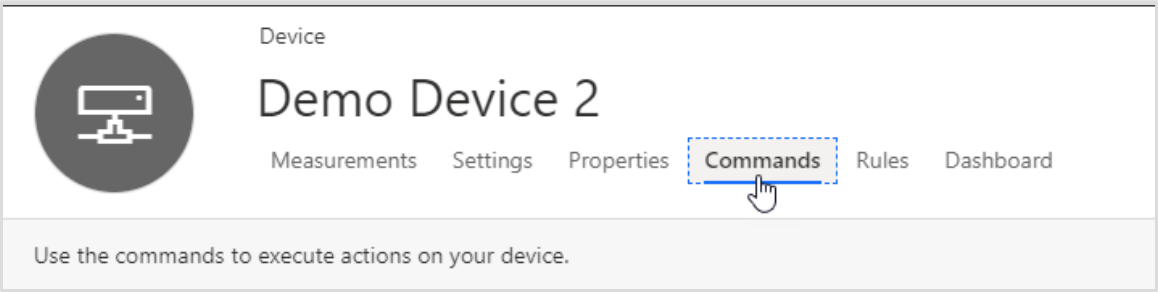
In order to trigger Application Over-the-Air updates:

- Select Edge Device by clicking on **Dashboard** tab and by clicking on the **device name**.

Figure 21. Dashboard



- Now select the **Commands** tab



Scroll the page to the text area named *Trigger AOTA:*

Figure 22. Trigger AOTA

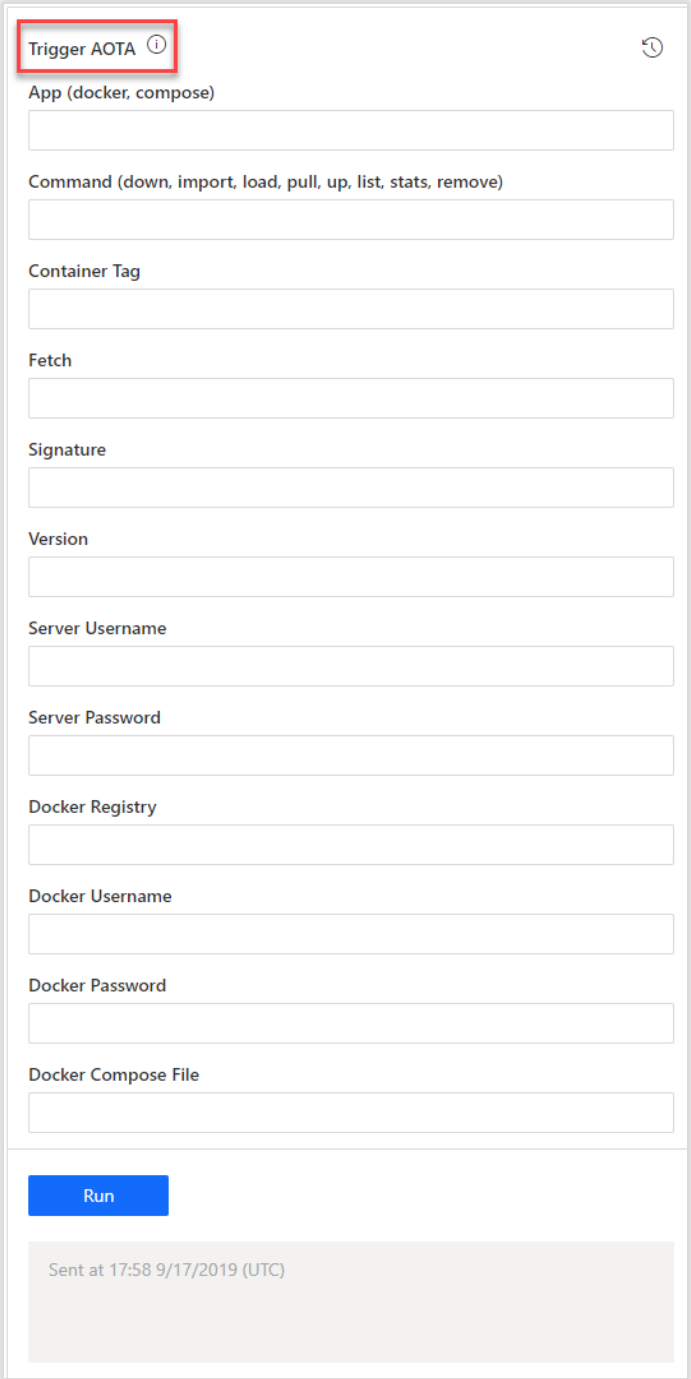


Table 6. AOTA Field Details

Field	Description
App	Docker or Docker-compose
Command	Docker-Compose supported operations: Up, Down, Pull, List and Remove. Docker supported operations: Load, Import, Pull, Remove and Stats
Container Tag	Name tag for image/container.
Docker Compose File	Field to specify the custom yaml file for docker-compose command. Example: custom.yml
Fetch	Server URL to download the AOTA container tar.gz file Note: If the server requires username/password to download the file, you can provide in server username/ server password
Server Username/ Server Password	If server needs credentials, we need to specify the username and password
Version	Each container will have a tag with the version number. It is recommended that you use this version number under version in the AOTA trigger. Command: sudo docker images. See image below to see result of this command
Docker Registry Docker Registry Username/Password	Specify Docker Registry if accessing any registry other than the default 'index.docker.io'. Optional fields Docker Registry Username/Password can be used to access docker private images in AOTA through docker and docker-compose up, pull commands.

Note: Following sections demonstrate what fields to fill for respective AOTA operations with required and optional fields.

The arrow in **green** indicates – **Mandatory field**

The arrow in **blue** indicates – **Optional field**

This symbol states that the fields are not used

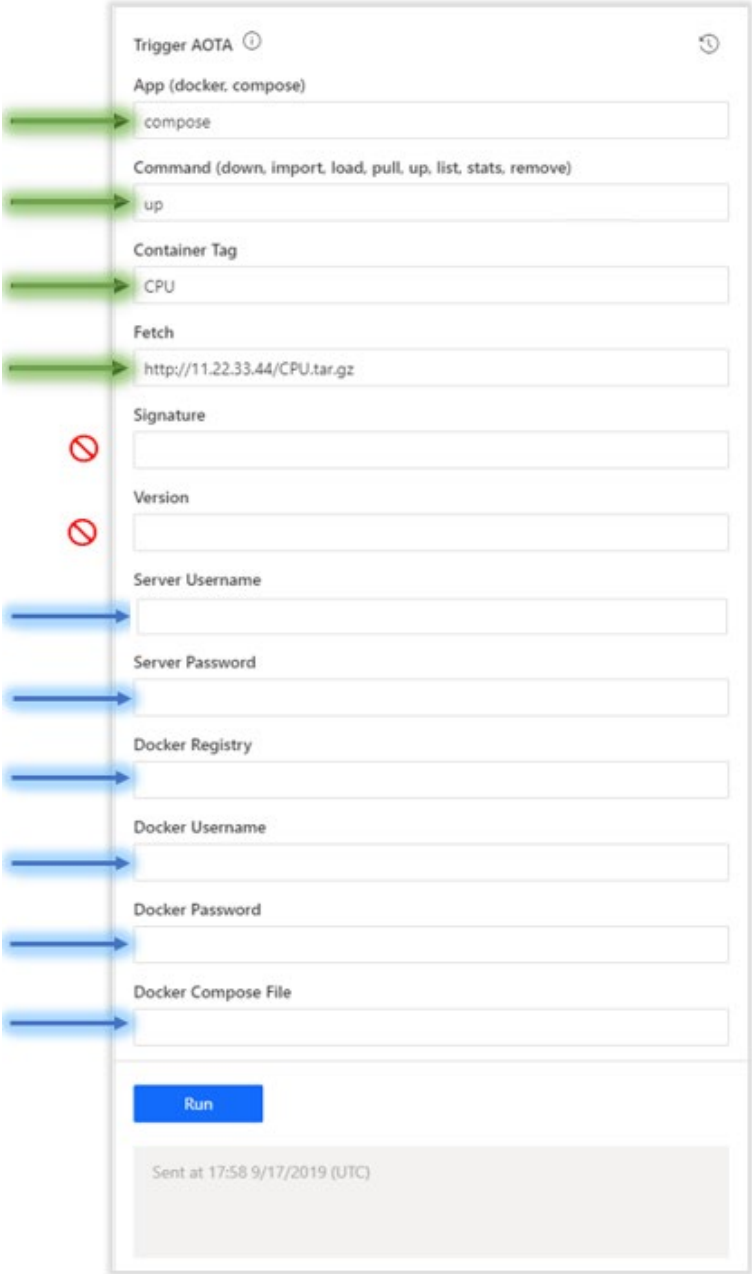
For each of the AOTA functions, insert the correct parameters as described and click **Run**. The result log can be viewed by clicking on the **Dashboard** tab.

3.5 AOTA Docker-Compose Operations

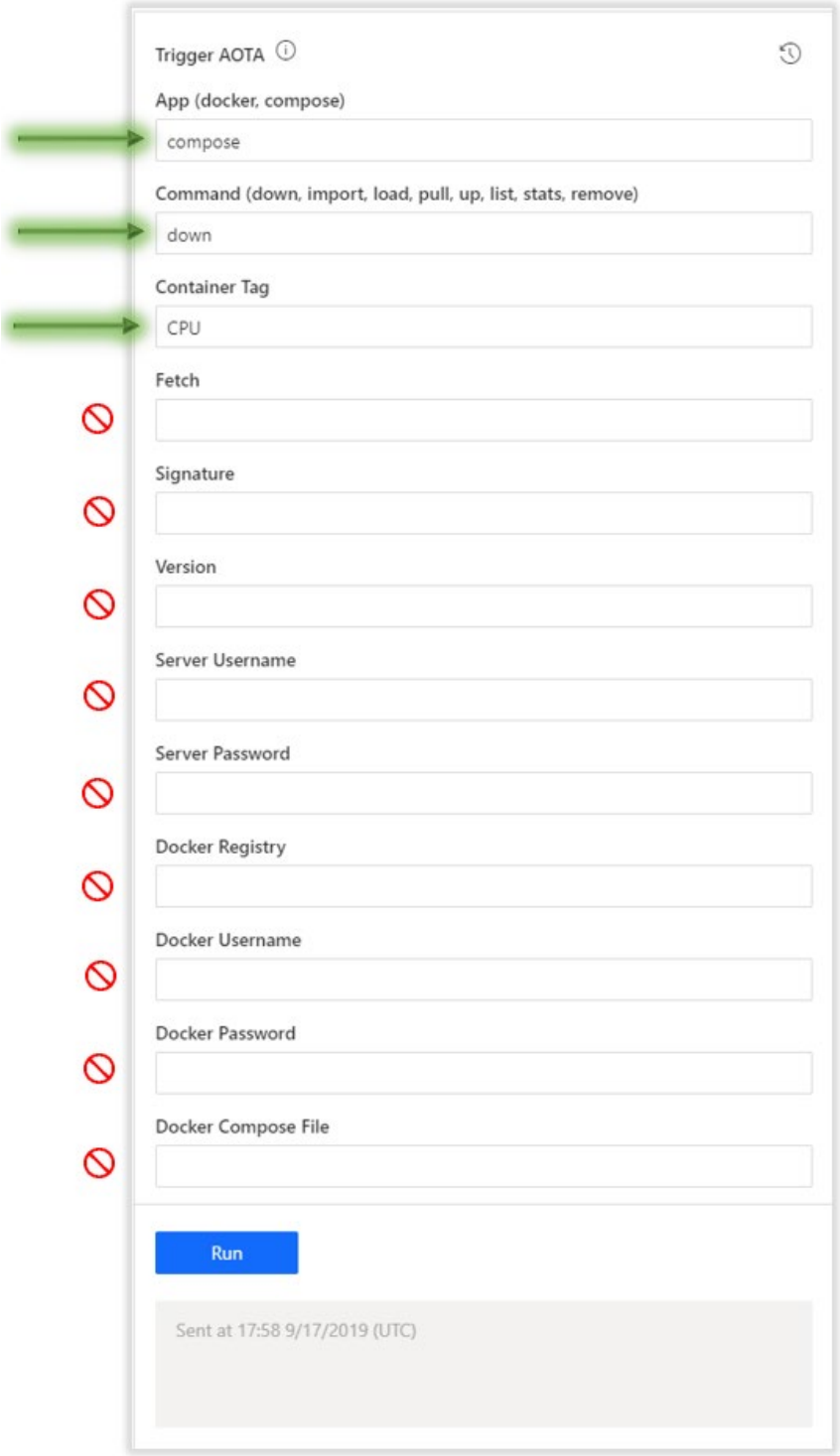
3.5.1 Docker Compose Up

NOTE:

1. The Container Tag name should be same as the file name in the fetch field. Example: Container Tag: CPU Downloaded fetch file: CPU.targ.gz.
2. Docker-Compose yml file should have the correct docker version.

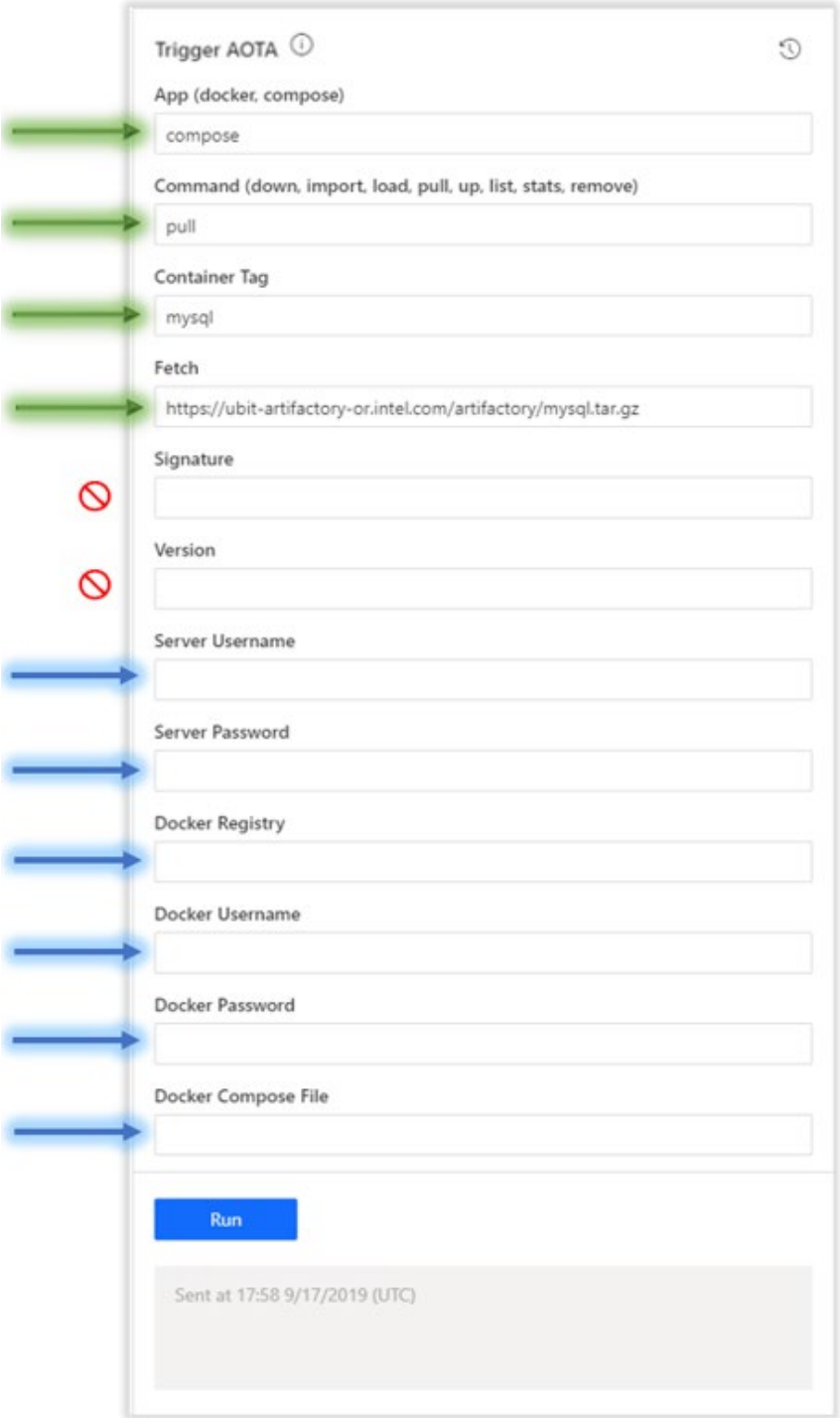


3.5.2 Docker-Compose Down

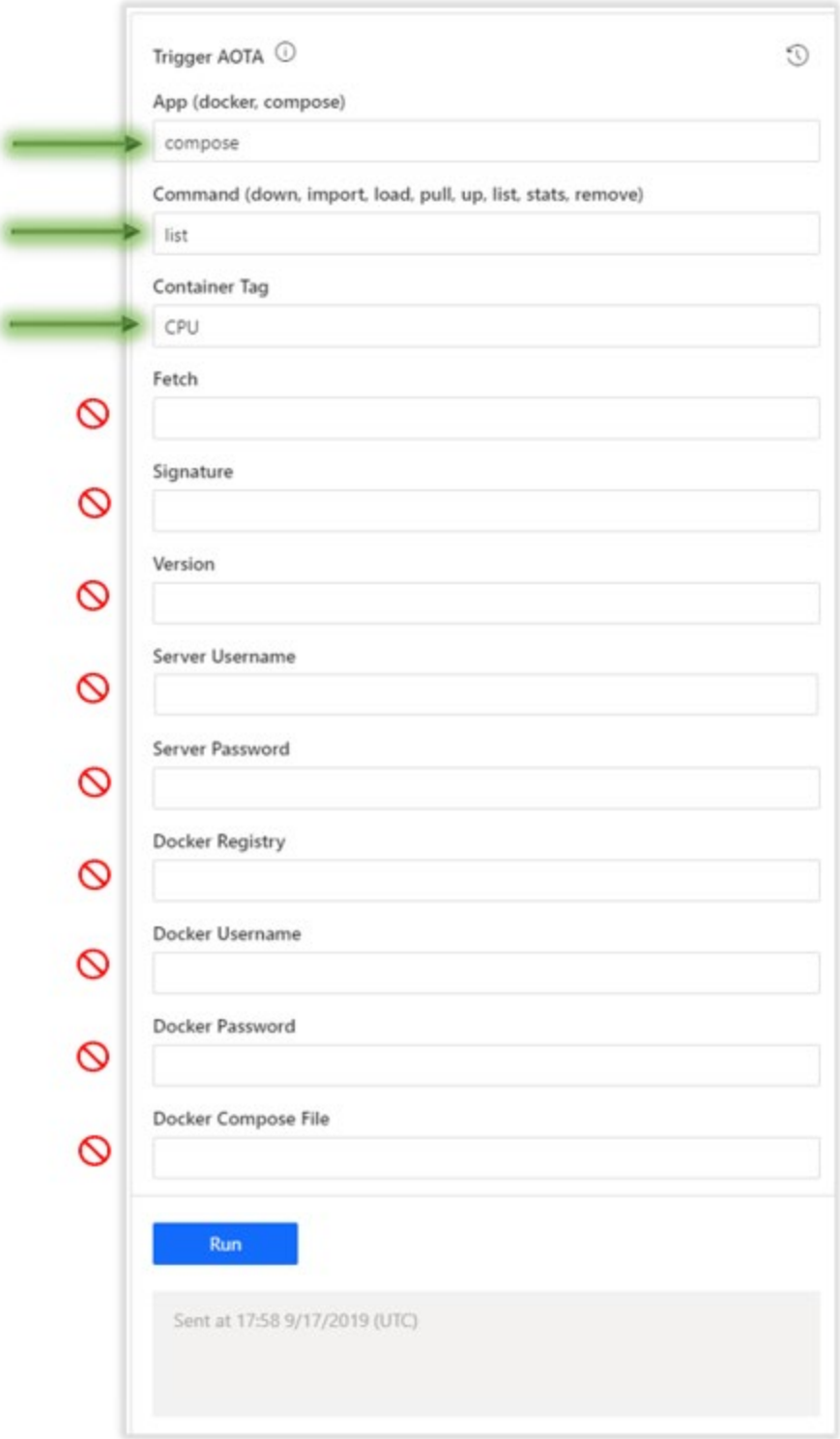


3.5.3 Docker-Compose Pull

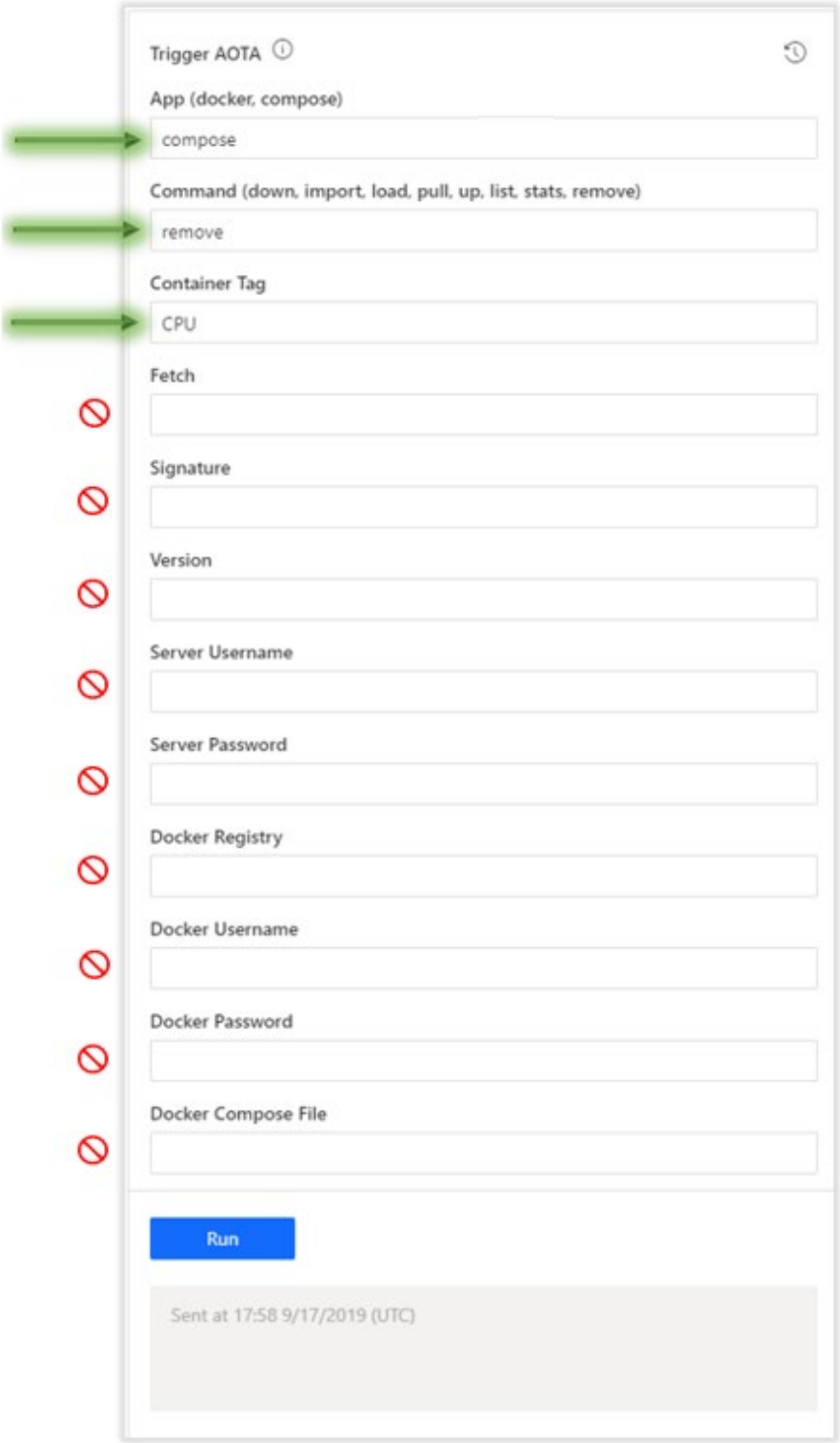
NOTE: The Container Tag name should be same as the file name in the fetch field. Example: Container Tag: CPU Downloaded fetch file: CPU.targ.gz



3.5.4 Docker-Compose List



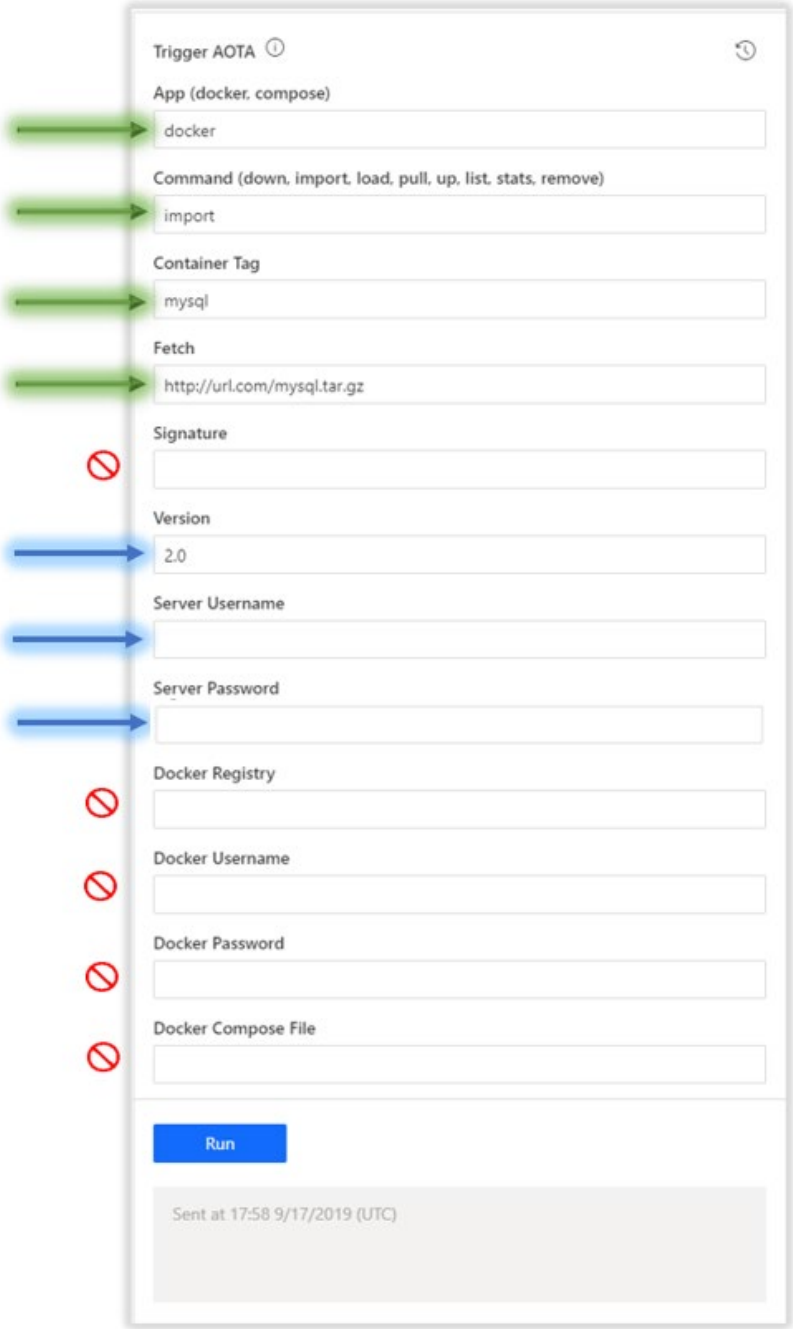
3.5.4 Docker-Compose Remove



3.6 AOTA Docker Operations

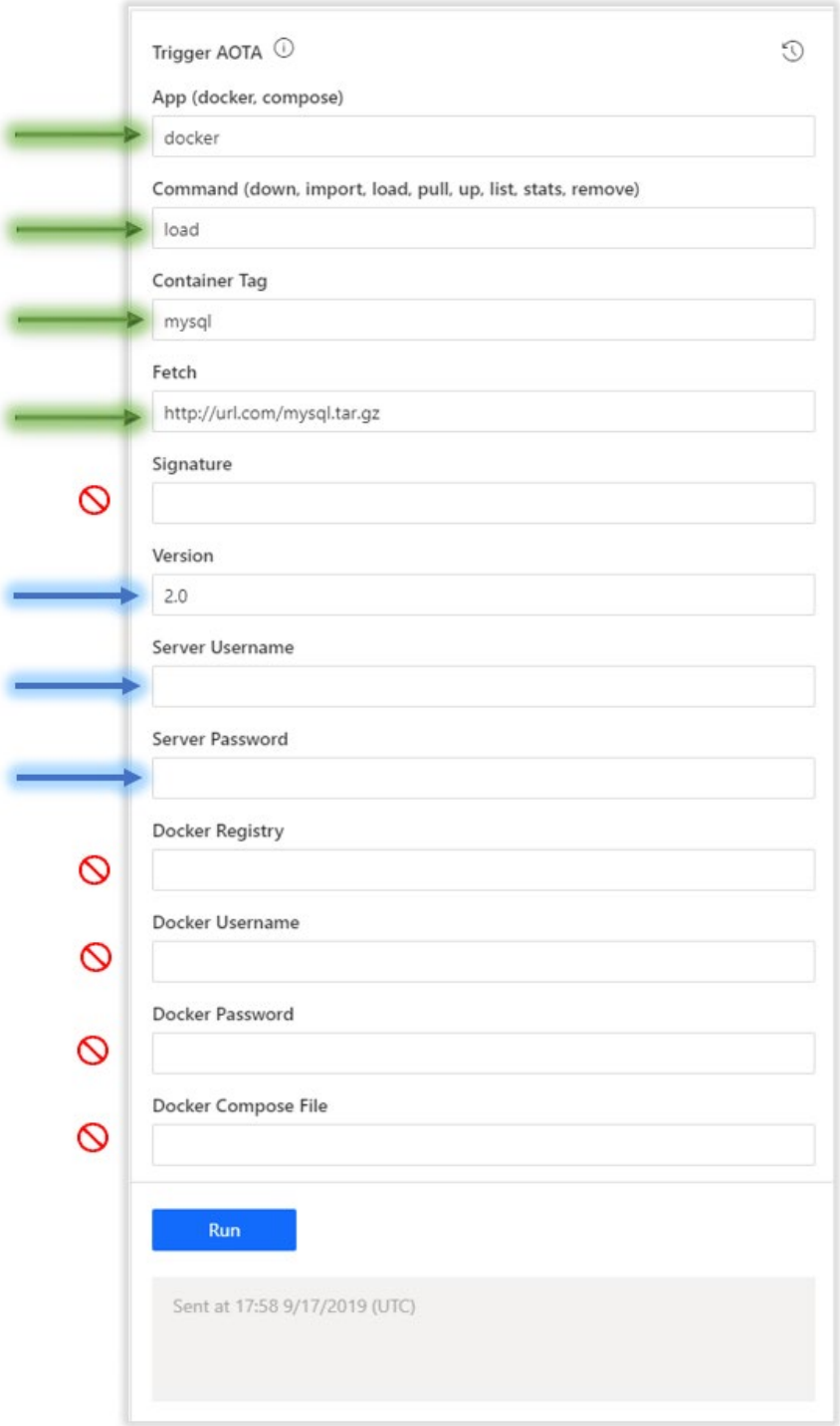
3.6.1 Docker Import

NOTE: The Container Tag name should be same as the file name in the fetch field.
Example: Container Tag: CPU, Downloaded fetch file: CPU.targ.gz

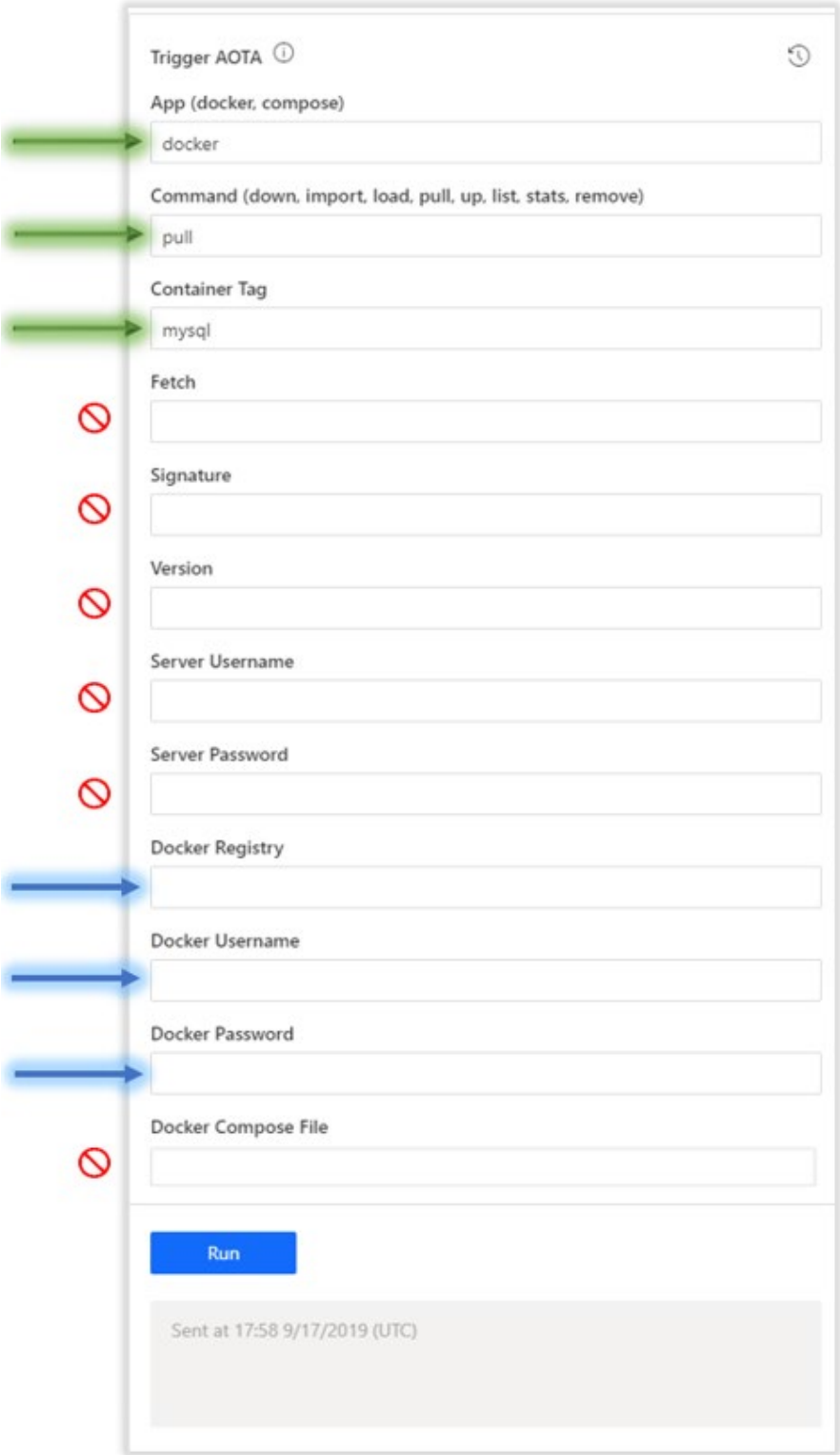


3.6.2 Docker Load

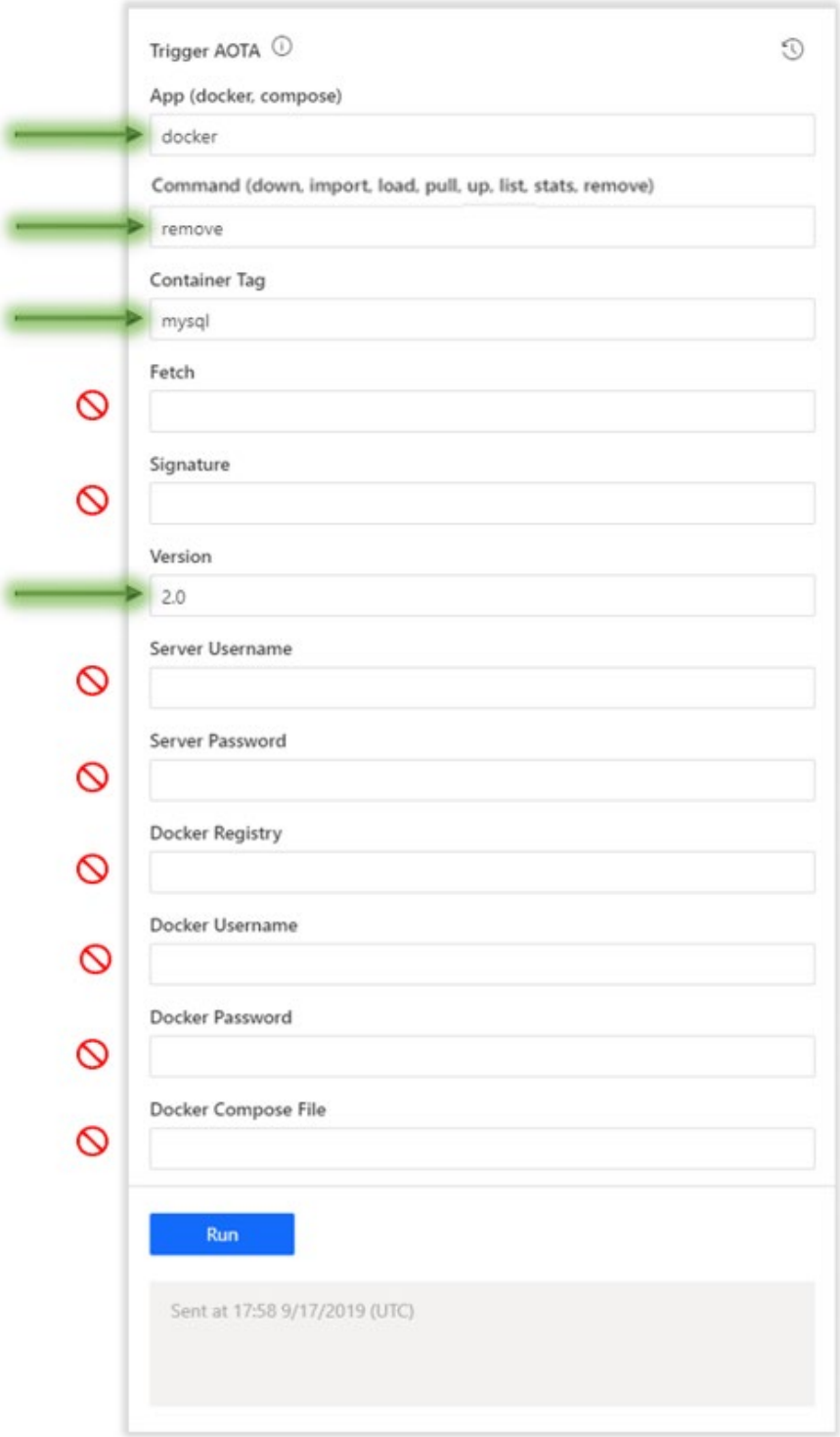
NOTE: The Container Tag name should be same as the file name in the fetch field. Example: Container Tag: CPU Downloaded fetch file: CPU.targ.gz



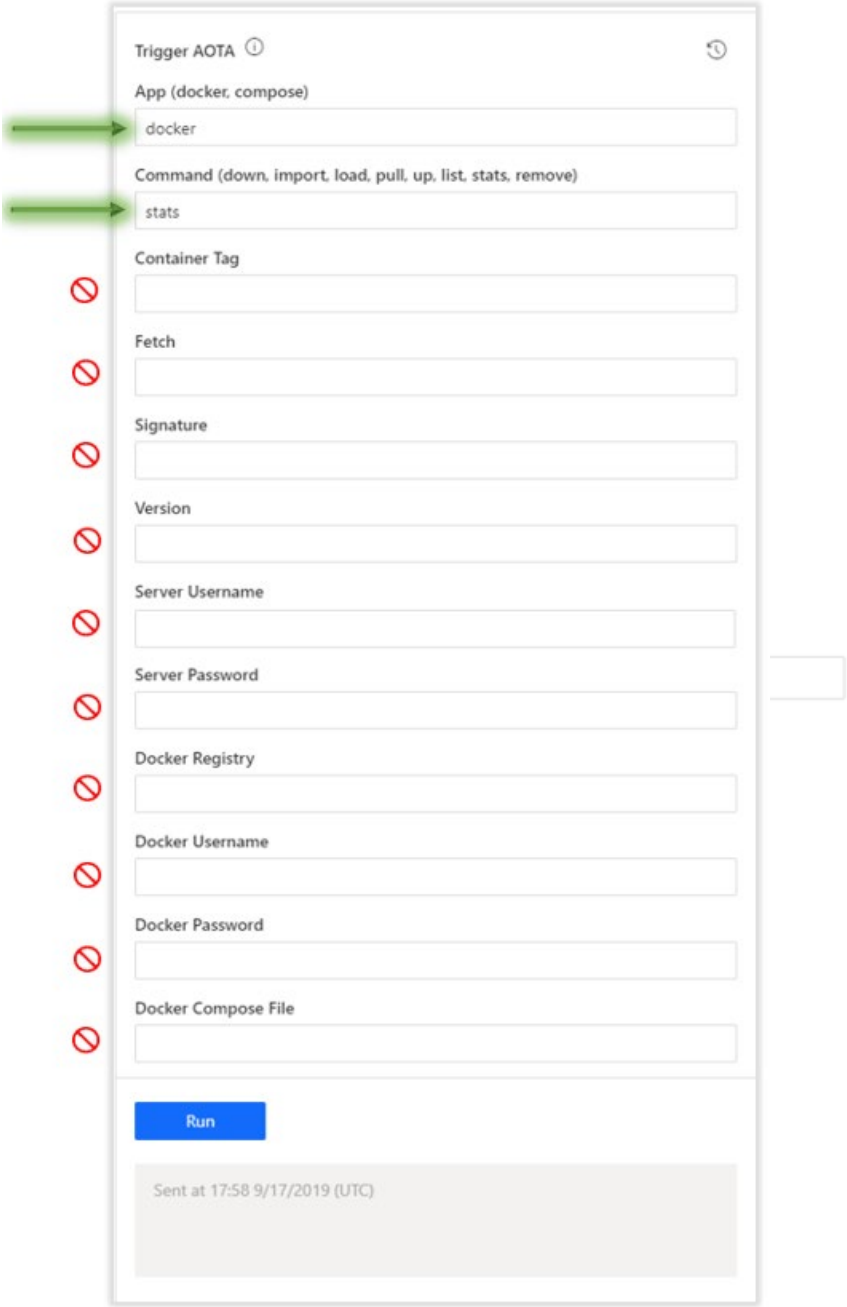
3.6.3 Docker Pull



3.6.4 Docker Remove



3.6.5 Docker Stats



3.7 AOTA Application Operations

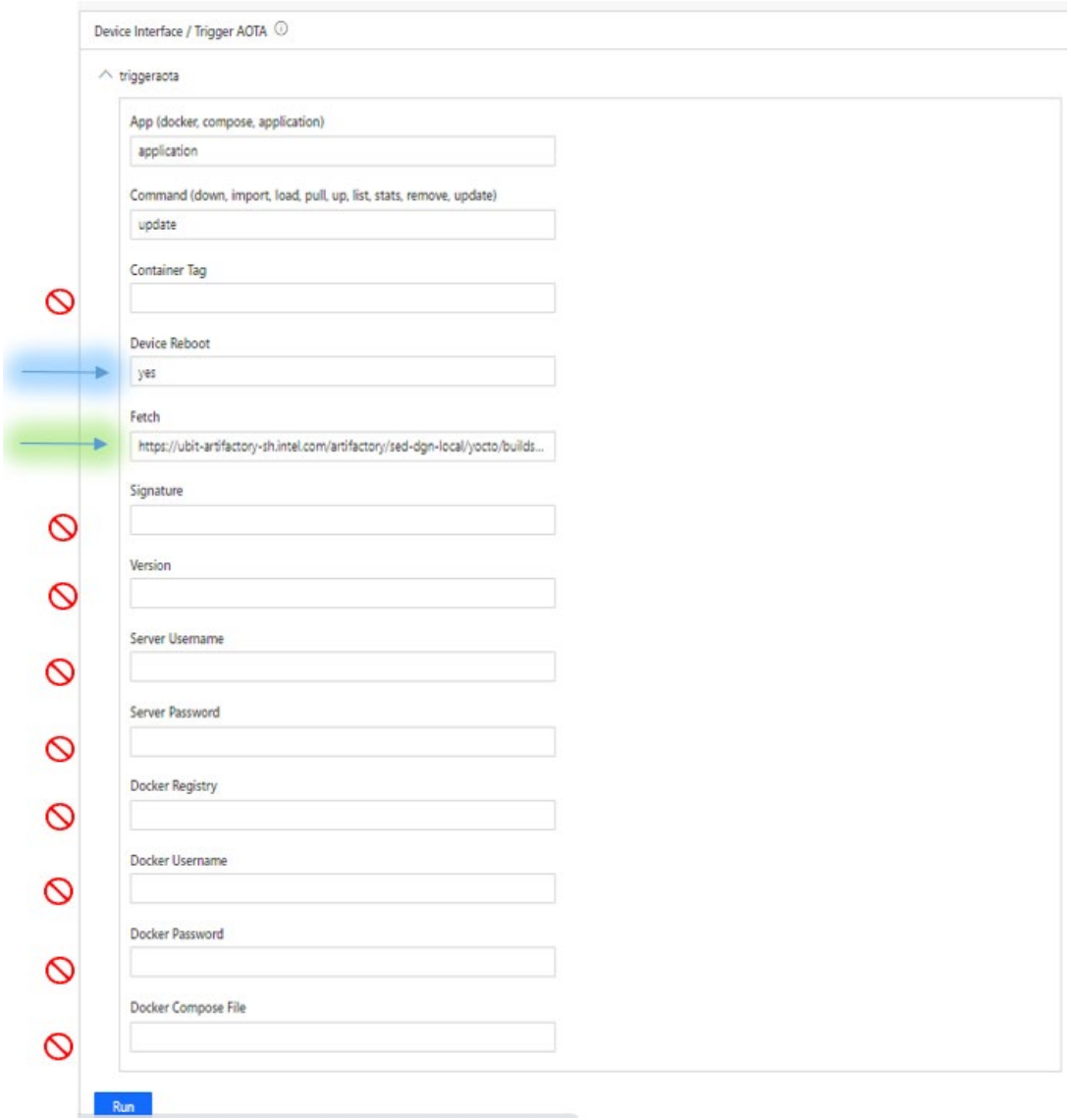
3.7.1 Application Update

NOTE: The Device Reboot is an optional field.

For any Xlink driver update it is mandatory to reboot the device.

Input "yes" for Device Reboot as seen below.

You can only use signed packages to update Xlink Driver application



3.7.2 AOTA Docker/Docker-Compose Operations via Manifest

Refer to **Developer Guide Documentation.**

3.8 FOTA Updates

To perform FOTA updates, IBVs must supply the SMBIOS or Device Tree info that is unique to each platform SKU and fulfill the vendor, version, release date, manufacturer and product name that matches the endpoint as shown below.

Note: The following information must match the data sent in the FOTA update command for the Intel® In-Band Manageability Framework to initiate a Firmware update process.

Table 7. FOTA Update Info

Information	Field	Checks
FW	Vendor	Exact string match
	Version	Unused
	Release Date	Checks if the FOTA date is newer than current
System	Manufacturer	Exact string match
	Product Name	Exact string match

To find the FW and System fields at the endpoint, run the commands below:

Intel x86 UEFI-based Products

For UEFI-based platforms, the firmware and system information can be found by running the following command.

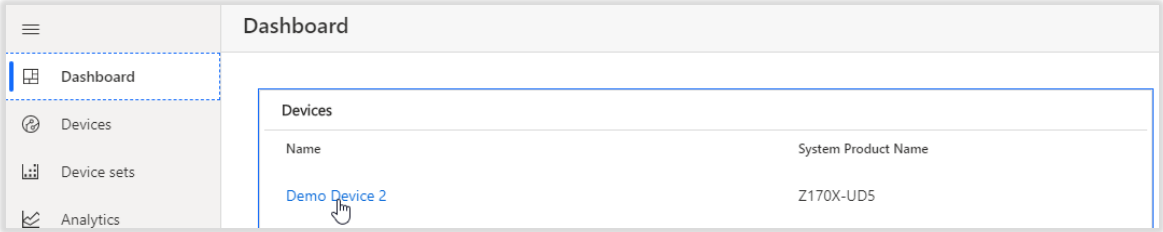
```
Command-line: sudo dmidecode -t bios -t system
```

3.8.1 FOTA Update via Button Click

In order to trigger Firmware-Over the Air updates:

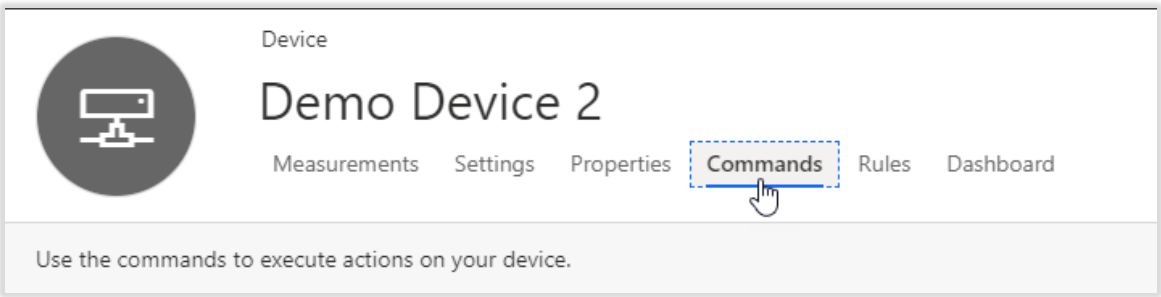
- Select Edge Device by clicking on **Dashboard** tab and by clicking on the **device name**.

Figure 23. Dashboard Tab



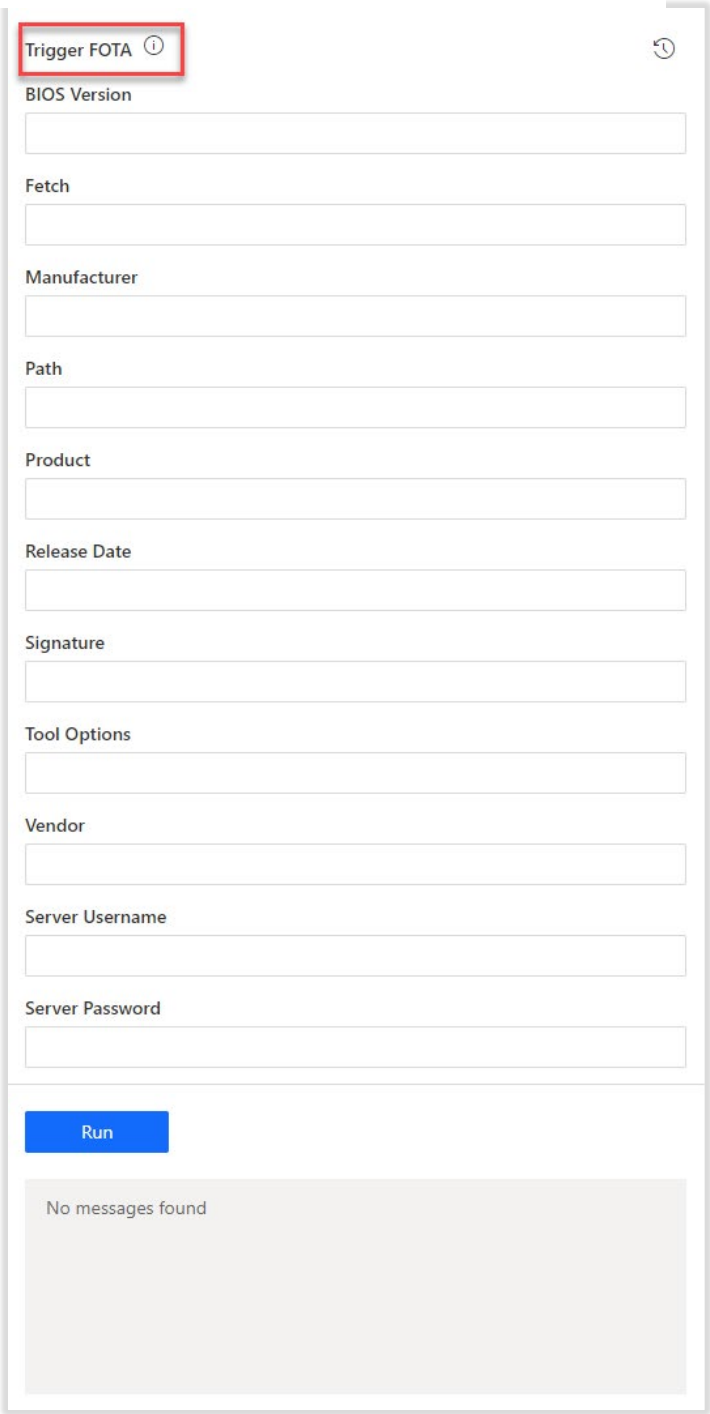
- Now select the 'Commands' tab

Figure 24. Commands Tab



- Scroll the page to the text area named **'Trigger FOTA'**:

Figure 25. Trigger FOTA



- Populate the text fields within the 'Trigger FOTA' block with the parameters in the table below.

  (Note: If triggering a secure FOTA update with a *.pem file within the *tar*, a signature needs to be given in the respective field. The signature can be generated using OpenSSL, or Cryptography libraries along with the key.pem file.

- After filling the correct parameters as described in the table, click **Run** to commission the FOTA update.

- The result log can be viewed by clicking on the **Dashboard** tab.

Table 8. Parameter Details

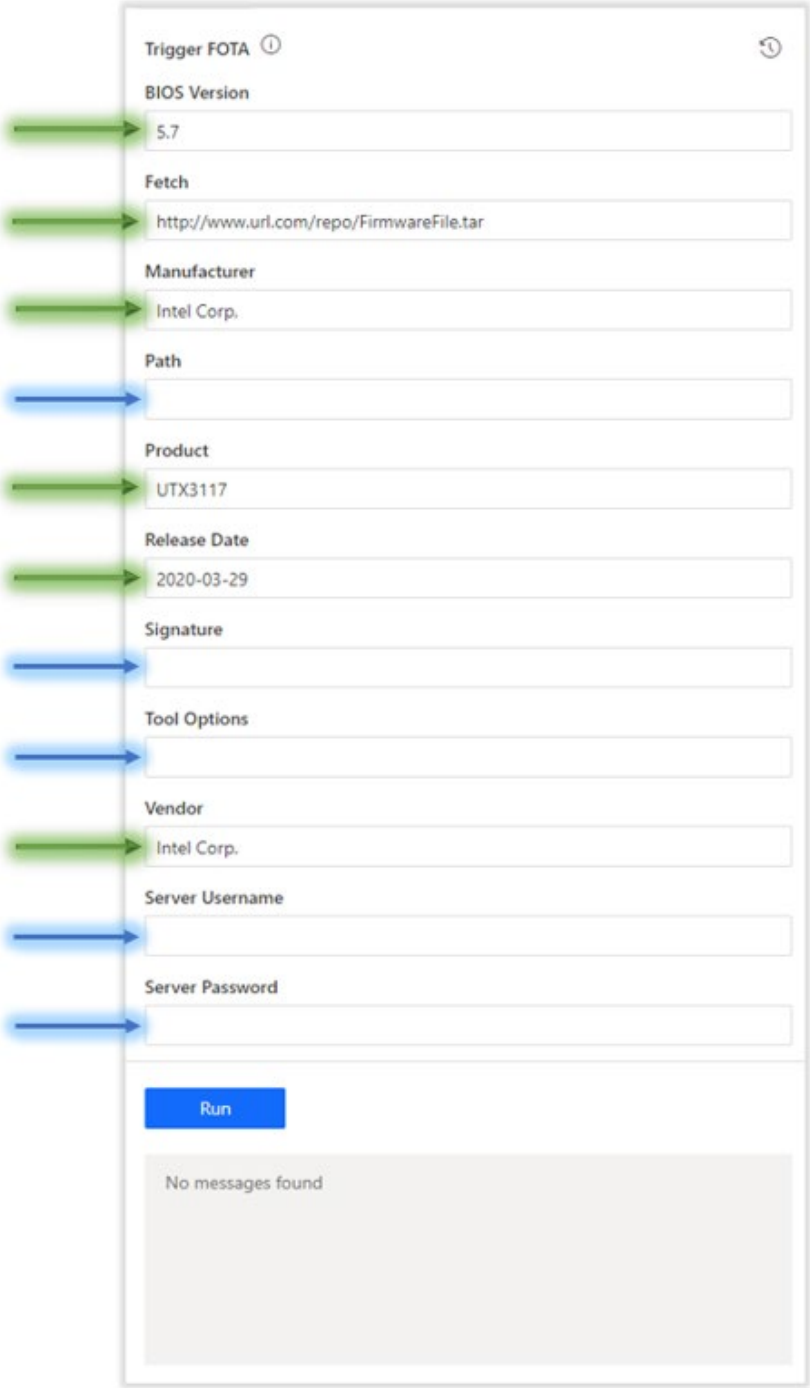
Parameter	Description
BIOSVersion	Verify with BIOS Vendor (IBV)
Fetch	Repository URL
Manufacturer	Endpoint Manufacturer Name
Path	FOTA path created in repository
Product	Product name set by Manufacturer
Release Date	Verify with BIOS Vendor (IBV) and specify the release date of the BIOS file you are applying **IMPORTANT NOTE: Date format: yyyy-mm-dd**
Signature	Digital signature of *.tar file.
Vendor	BIOS Vendor (IBV) Name

Note: The following screenshot demonstrate what fields to fill for a FOTA operation with required and optional fields.

The arrow in **green** indicates – **Mandatory field**

The arrow in **blue** indicates – **Optional field**

This symbol states that the fields are not used


Trigger FOTA
BIOS Version
5.7
Fetch
http://www.url.com/repo/FirmwareFile.tar
Manufacturer
Intel Corp.
Path
Product
UTX3117
Release Date
2020-03-29
Signature
Tool Options
Vendor
Intel Corp.
Server Username
Server Password
Run
No messages found

3.8.2 FOTA Update via Manifest

Refer to **Developer Guide Documentation.**

3.9 SOTA Updates

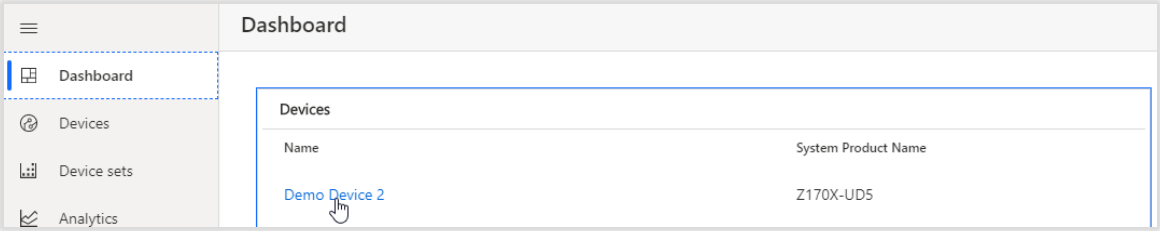
SOTA commands vary based on OS type and update mechanisms supported by it. Ubuntu OS or Yocto based OS which include Debian package manager do not require any package preparation, while a Yocto based OS with Mender.io based solution does. This changes the interface slightly as explained below.

3.9.1 SOTA Update Via 'Trigger SOTA' Button Click (Debian Package Manager, Ubuntu OS)

In order to trigger Software-Over the Air updates:

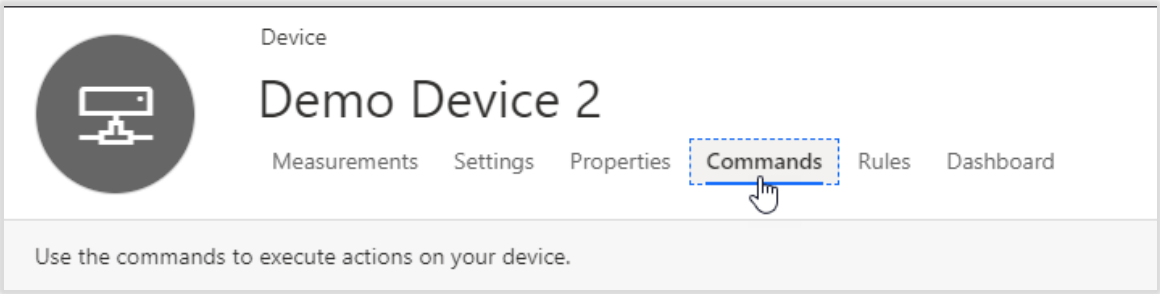
- Select Edge Device by clicking on **Dashboard** tab and by clicking on the **device name**.

Figure 26. Dashboard Tab



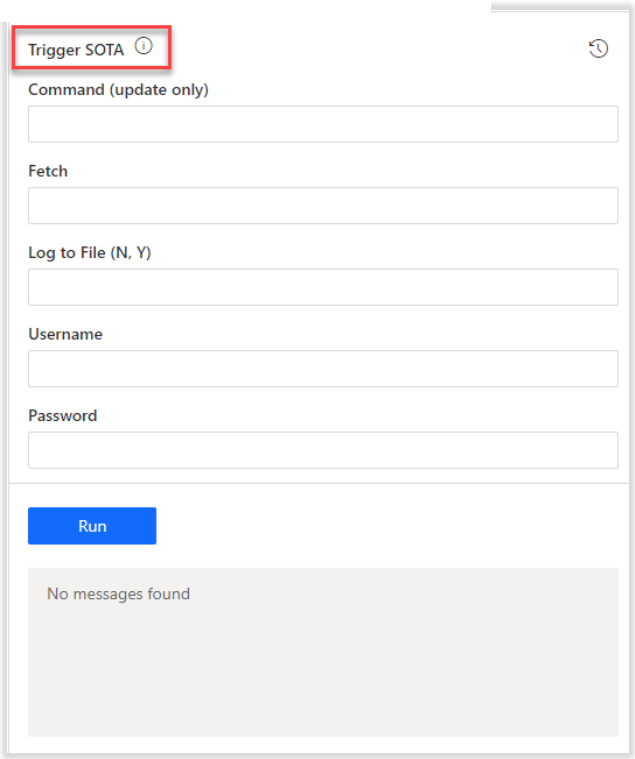
- Now select the 'Commands' tab

Figure 27. Commands Tab



- Scroll the page to the text area named **'Trigger SOTA'**:

Figure 28. Trigger SOTA



Populate the SOTA text fields on screen with the parameters below:

Table 9. SOTA Parameters

Command	Specifies the SOTA 'update' command.
Log to File	Specifies if the logs be written to a file or to the cloud. Values "Y" or "N" SOTA log files can be located at the endpoint */var/cache/manageability/repository-tool/sota/*

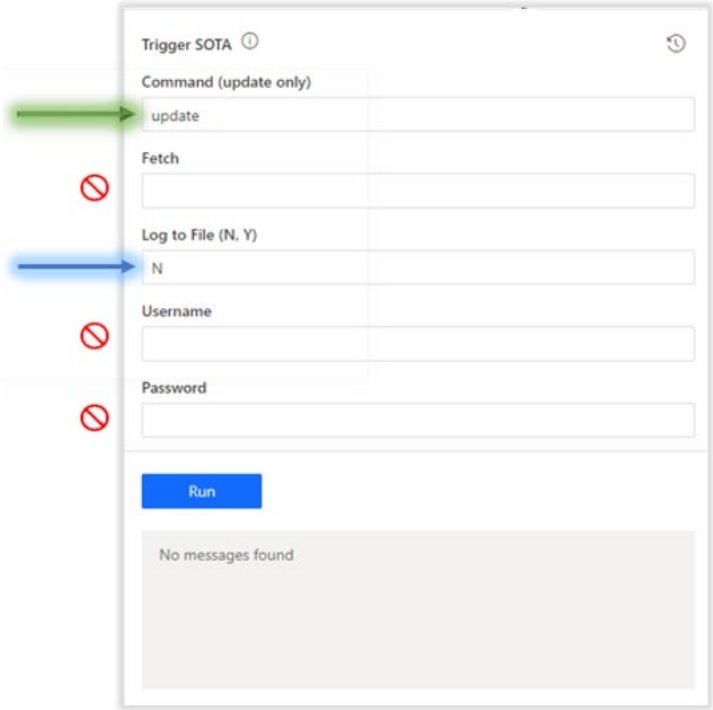
- Click **Run** to commission SOTA update.

Note: Following screenshot demonstrates what fields to fill for a SOTA operation with required and optional fields.

The arrow in **green** indicates – **Mandatory field**

The arrow in **blue** indicates – **Optional field**

This symbol states that the fields are not used

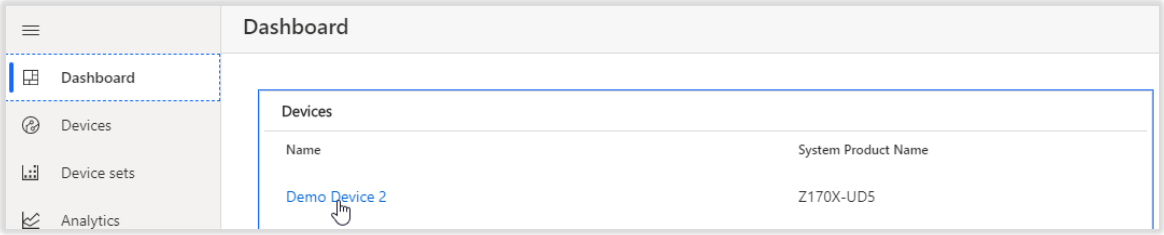


3.9.2 SOTA Update Via 'Trigger SOTA' Button Click (Mender)

In order to trigger Software-Over the Air updates:

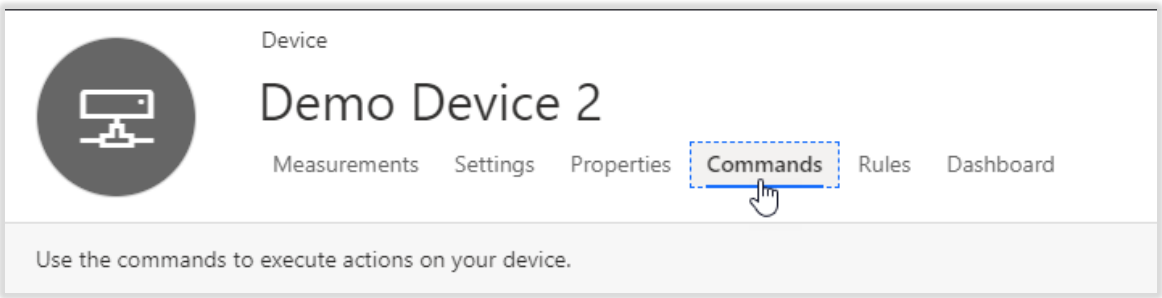
- Select Edge Device by clicking on **Dashboard** tab and by clicking on the **device name**.

Figure 29. Dashboard Tab



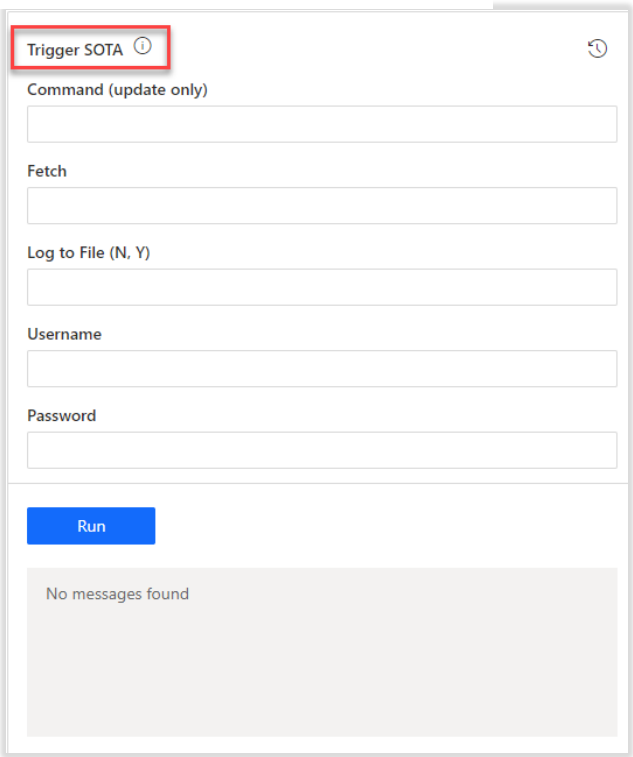
- Now select the '**Commands'** tab

Figure 30. Commands Tab



- Scroll the page to the text area named **'Trigger SOTA'**:

Figure 31. Trigger SOTA



Populate the SOTA text fields on screen with the parameters below:

Figure 32. Parameter Details

Command	Specifies the SOTA 'update' command.
Fetch	URL patch to download the Mender artifact from
Log to File	Specifies if the logs be written to a file or to the cloud. Values "Y" or "N" SOTA log files can be located at the endpoint */var/cache/manageability/repository-tool/sota/*
Username	Mender artifact repository Username
Password	Mender artifact repository Password

- Click **Run** to commission SOTA update.

Note: Following screenshot demonstrate what fields to fill for a SOTA operation with required and optional fields.

The arrow in **green** indicates – **Mandatory field**

The arrow in **blue** indicates – **Optional field**

This symbol states that the fields are not used

3.9.3 SOTA Update via Manifest

Refer to **Developer Guide Documentation.**

3.10 Configuration Update

Configuration update is used to change/retrieve/append/remove configuration parameters value from the Configuration file located at **/etc/intel_manageability.conf**. Refer to table below to understand the configuration tags, it's values and the description.

Table 10. Default Configuration Parameters

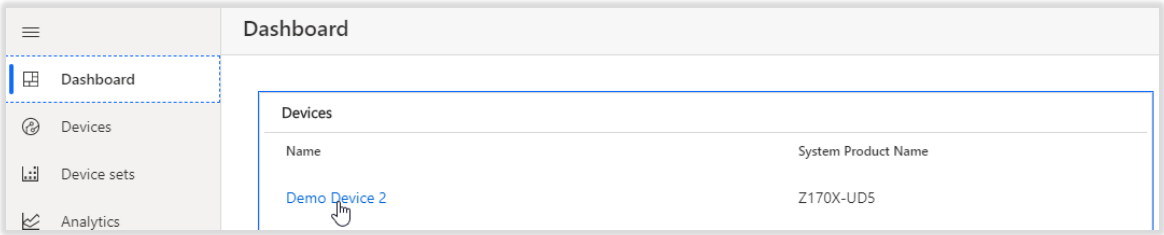
Telemetry		
Collection Interval Seconds	60 seconds	Time interval after which telemetry is collected from the system.
Publish interval seconds	300 seconds	Time interval after which collected telemetry is published to dispatcher and the cloud
Max Cache Size	100	Maximum cache set to store the telemetry data. This is the count of messages that telemetry agent caches before sending out to the cloud
Container Health Interval Seconds	600 seconds	Interval after which container health check is run and results are returned.
Diagnostic Values		
Min Storage	100 MB	Value of minimum storage that the system should have before or after an update
Min Memory	200 MB	Value of minimum memory that the system should have before or after an update
Min Power Percent	20%	Value of minimum battery percent that system should have before or after update
Mandatory SW	docker, trtl, telemetry	List of software that should be present and are checked for.
Docker Bench Security Interval Seconds	900 seconds	Time interval after which DBS will run and report back to the cloud.
Network Check	true	This configures network check on the platforms based on their Ethernet capability.
Dispatcher Values		

DBS Remove Image on Failed Container	False	Specifies if the image should be removed in the event of a failed container as flagged by DBS.
Trusted Repositories		List of repositories that are trusted and packages can be fetched from them
SOTA Values		
Ubuntu Apt Source	http://linux-ftp.jf.intel.com/pub/mirrors/ubuntu/	Location used to update Ubuntu
Proceed Without Rollback	True	Whether SOTA update should go through even when rollback is not supported on the system.

In order to trigger Configuration updates:

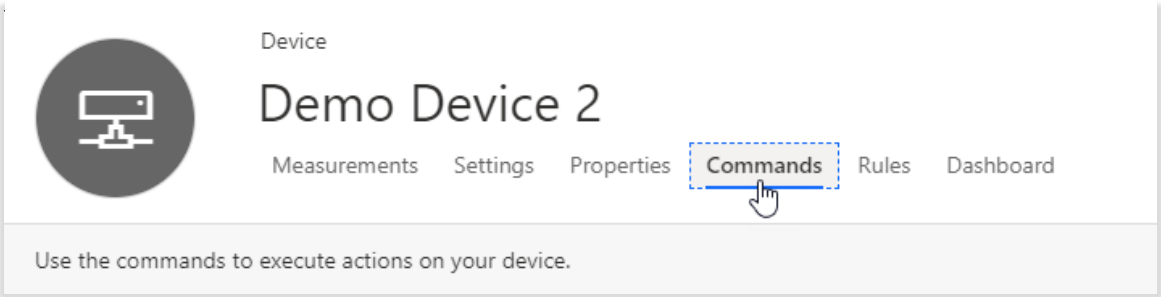
- Select Edge Device by clicking on **Dashboard** tab and by clicking on the **device name**.

Figure 33. Dashboard Tab



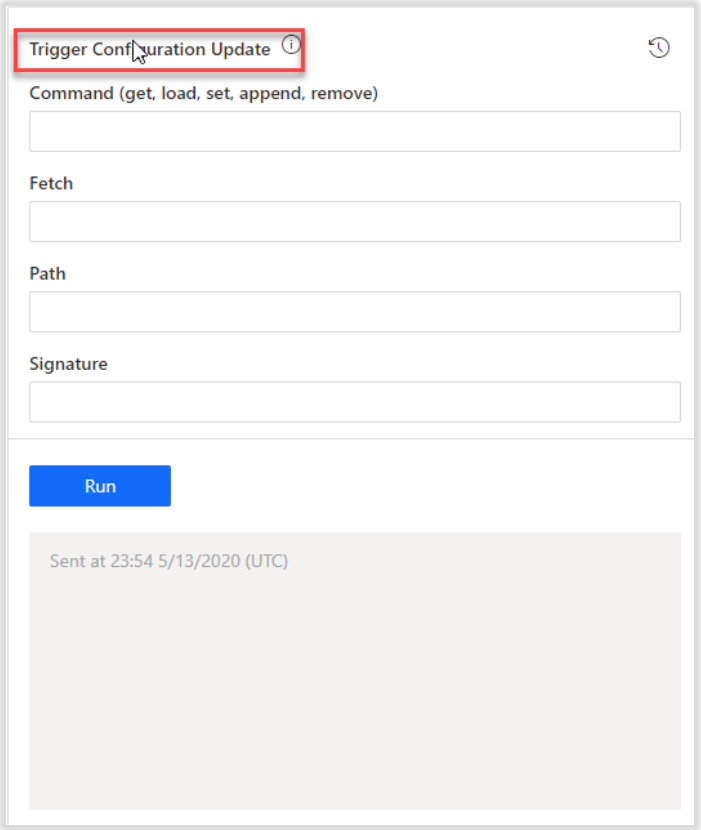
- Now select the '**Commands'** tab

Figure 34. Commands Tab



- Scroll the page to the text area named **'Trigger Configuration Update'**:

Figure 35. Trigger Configuration Update



- Populate the Config Update pop-up window with required parameters. Refer to the table below to know what commands are available.

  (Note: If triggering a secure Config update load with a *.pem file within the *tar*, a signature needs to be given in the respective field. The signature can be generated using OpenSSL, or Cryptography libraries along with the key.pem file.

- Click **Run** to trigger the Config update.

- The result log can be viewed by clicking on the **Dashboard** tab.

Below are the configuration update commands/input fields with its description:

Table 11. Configuration Update Command/Input Fields

Trigger Configs	Description of field
Command	**Set**: Command to change the configuration value from an old value to new value using key:value pair. **Get**: Command used to retrieve a specific configuration value using key:value pair **Load**: Command used to replace entire configuration file **Append**: Command used to append additional values to a configuration parameter **Remove**: Command used to remove a specific value from the configuration parameter
Fetch	The URL to fetch config file from in the case of a load
Path	Specifies the path of element to get or set in key:value format
Signature	Digital signature

Note: Following screenshots demonstrate what fields to filled for various configuration operations with required and optional fields.

The arrow in **green** indicates – **Mandatory field**

The arrow in **blue** indicates – **Optional field**

This symbol states that the fields are not used

3.10.1 Configuration Operation via Button Click

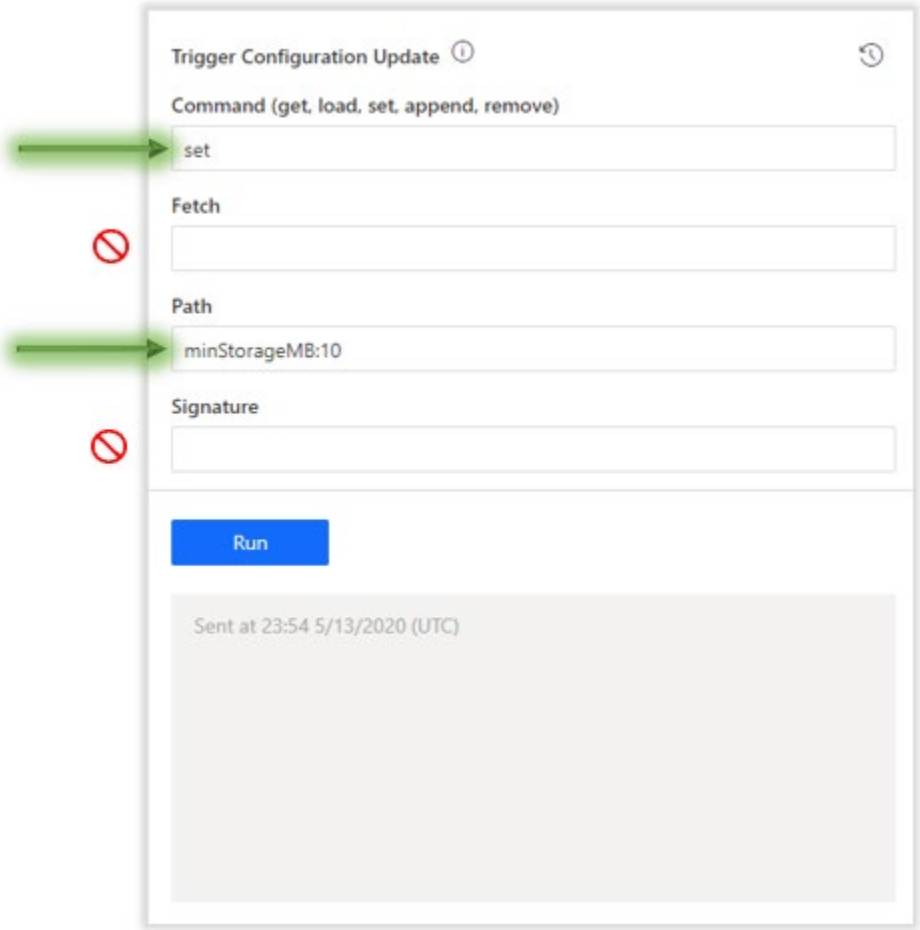
3.10.2 Configuration Set

Examples:

To set one value: **minStorageMB:10**

To set multiple values at once: **minStorageMB:10; minMemoryMB:250**

NOTE: Path takes in key value pairs as an input with key as the configuration parameter tag and value to be set as the value. Also, to set multiple key:value pairs, use; to separate one pair from another as shown above in the example.



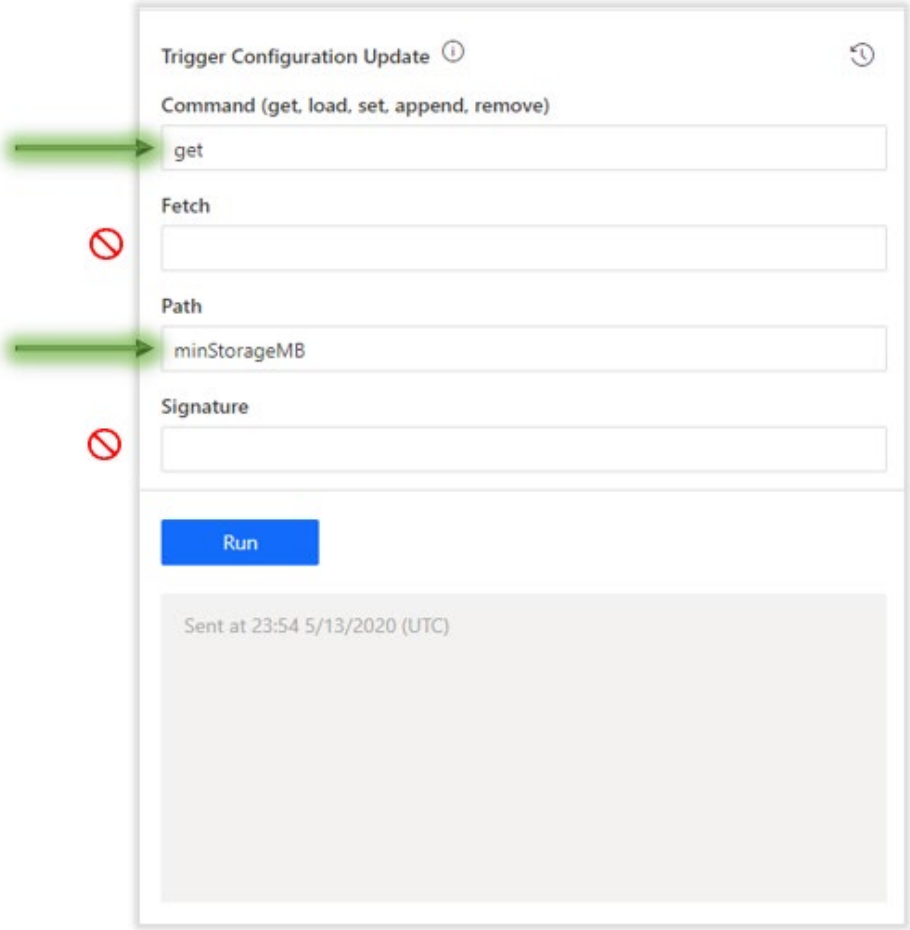
3.10.3 Configuration Get:

Examples:

To set one value: **minStorageMB**

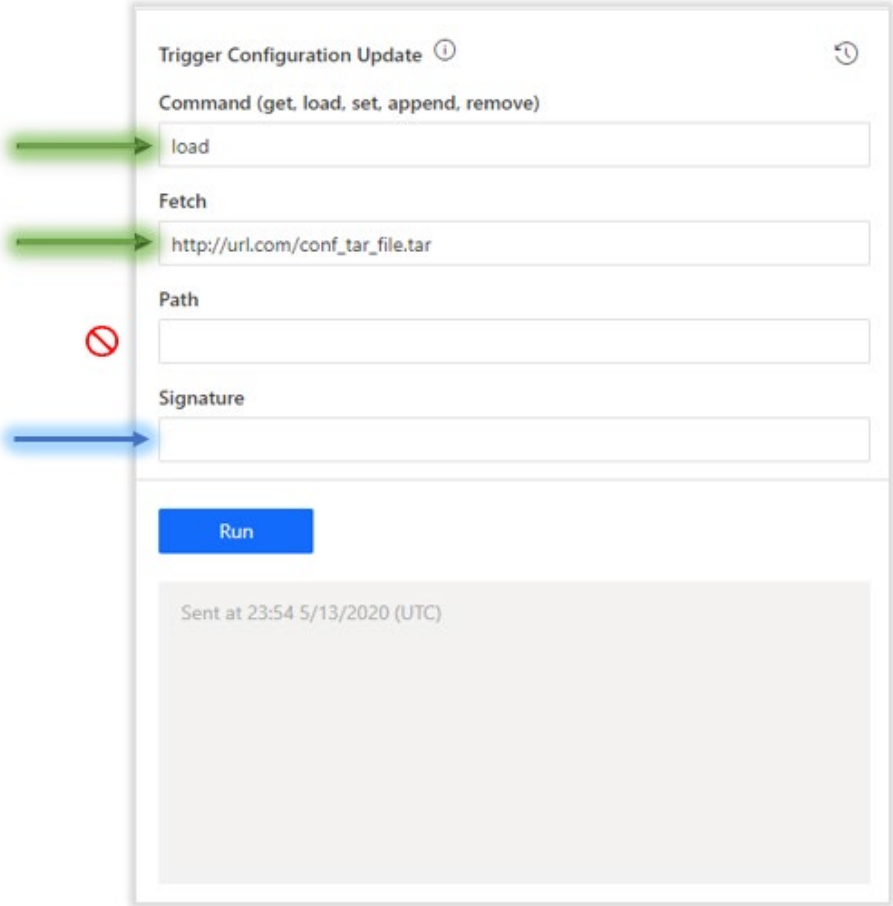
To set multiple values at once: **minStorageMB; minMemoryMB**

NOTE: Path takes in keys as an input with key as the configuration parameter tag whose value needs to be retrieved. Also, to retrieve multiple values at once use ';' to separate one tag from another as shown above in the example.



3.10.4 Configuration Load:

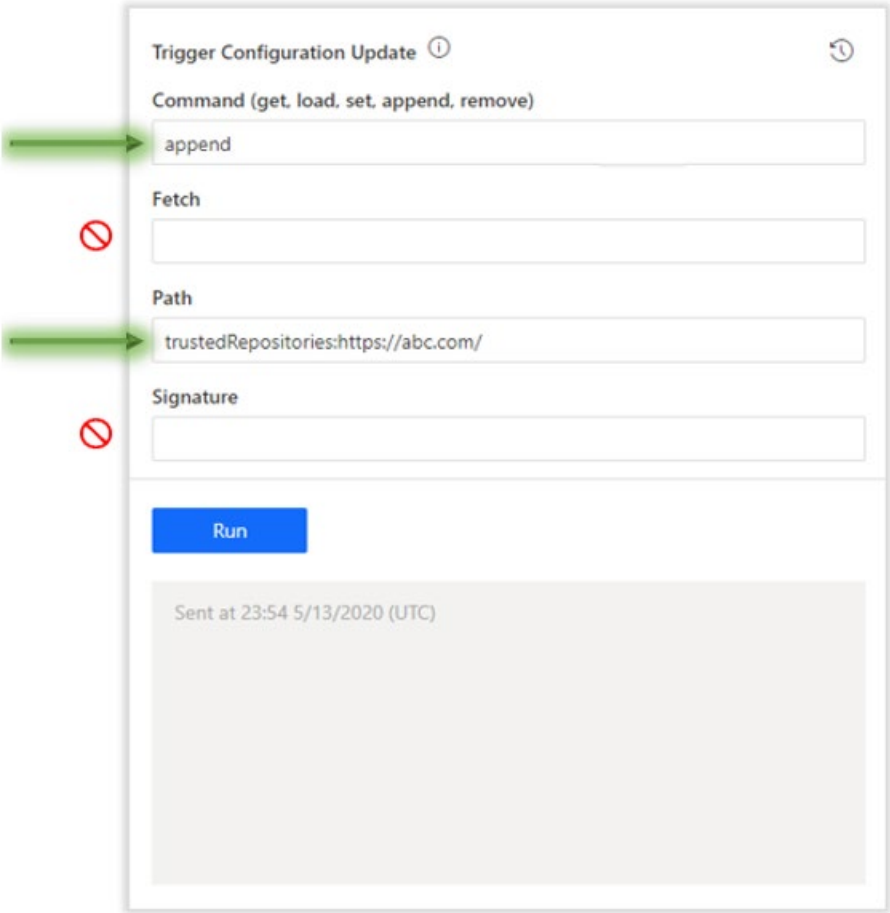
NOTE: The configuration file you provide in Fetch needs to be named as intel_manageability.conf file. If you wish to send with signature, tar both the pem file and the intel_manageability.conf in a tar file.



3.10.5 Configuration Append:

NOTE: Append is only applicable to three configuration tags i.e trustedRepositories, sotaSW and ubuntuAptSource

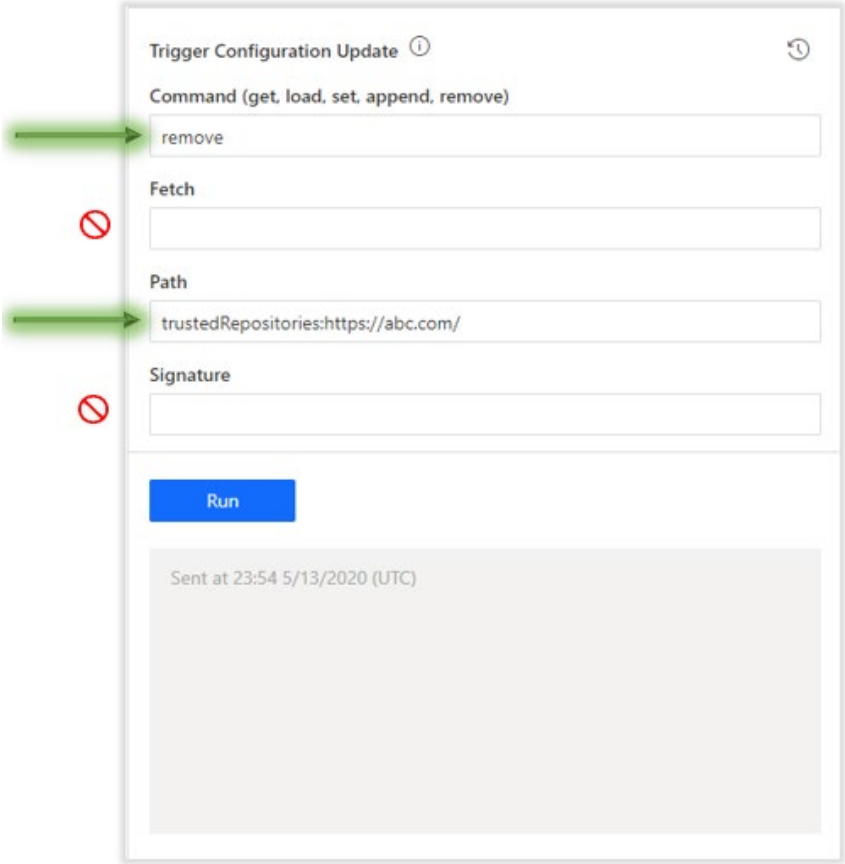
Path takes in key value pair format, example: trustedRepositories:https://abc.com/



3.10.6 Configuration Remove:

NOTE: Remove is only applicable to three configuration tags i.e trustedRepositories, sotaSW and ubuntuAptSource

Path takes in key value pair format, example: trustedRepositories:https://abc.com/



3.10.7 Configuration Operation via Manifest

Refer to the **Developer Guide Documentation.**

3.11 Power Management

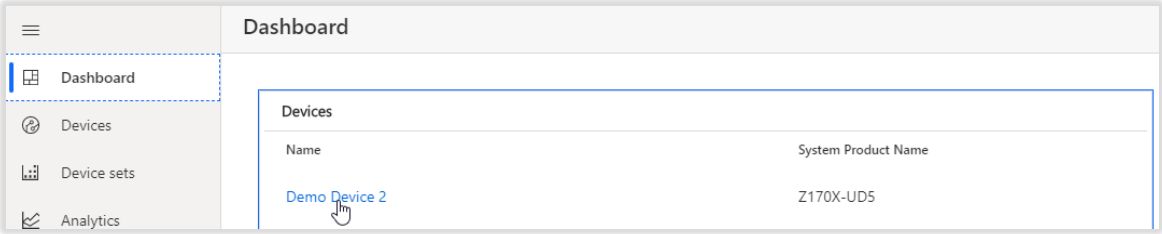
The Shutdown and Restart capabilities are supported via button click or through manifest.

3.11.1 Power Management via Button Click

In order to trigger Reboot/Shutdown:

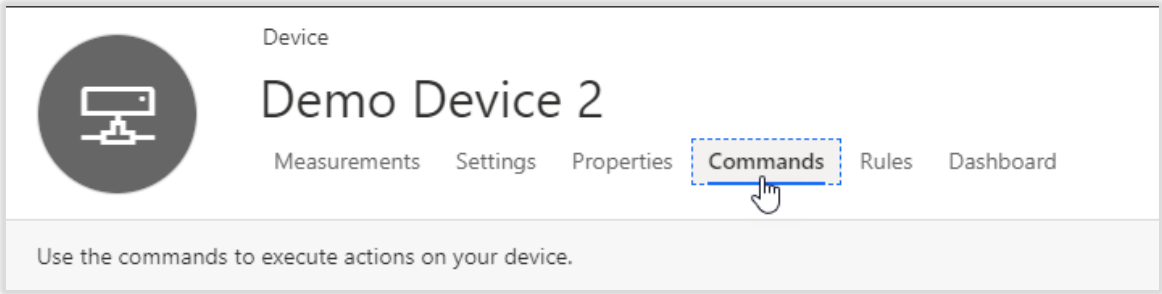
- Select Edge Device by clicking on **Dashboard** tab and by clicking on the **device name**.

Figure 36. Dashboard Tab



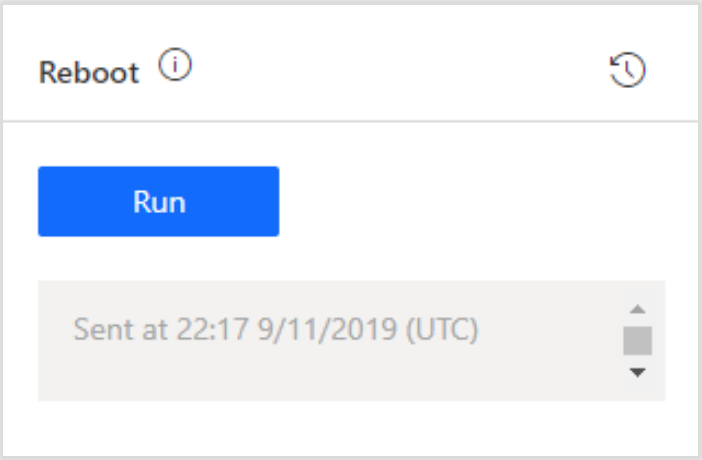
- Now select the '**Commands'** tab

Figure 37. Commands Tab



3.11.2 System Reboot

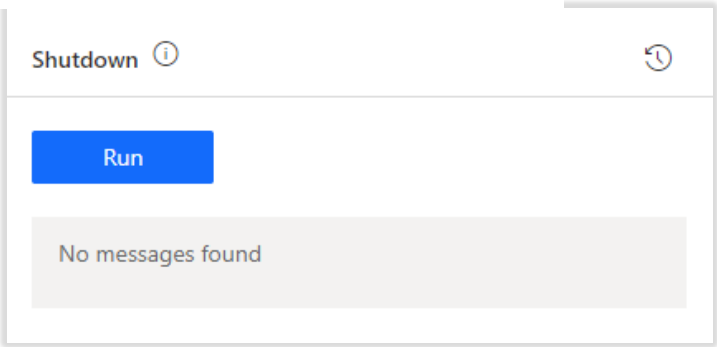
Figure 38. Reboot



- To reboot the device, click the **Run** button on the box titled *Reboot***.**

3.11.3 System Shutdown

Figure 39. Shutdown



- To shut down the device, click the **Run** button on the box titled *Shutdown.*

3.11.4 Power Management via Manifest

Refer to **Developer Guide Documentation**.

3.11.5 Decommission Command

The Intel® In-Band Manageability provides a mechanism to handle the decommission request over the air.

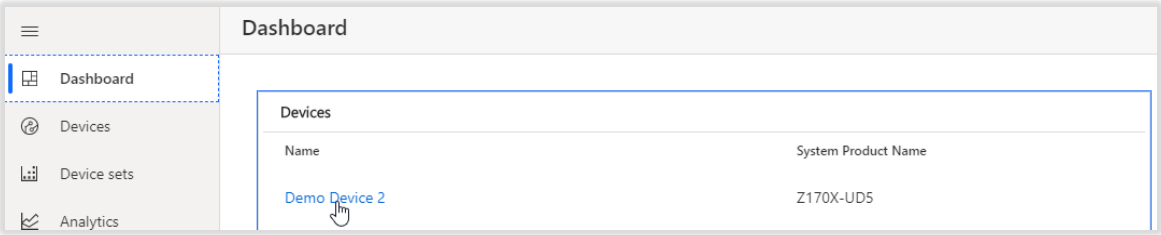
***NOTE*:** On receiving a Decommission cmd:

- The Intel® In-Band Manageability credentials (all user/device data which allows the device to identify and connect to cloud) will be deleted from the device.
- The device shutdowns.

In order to trigger Decommission:

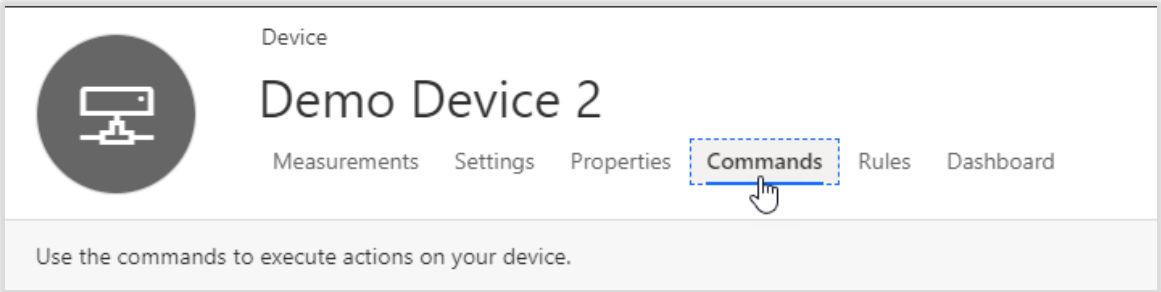
- Select Edge Device by clicking on **Dashboard** tab and by clicking on the **device name**.

Figure 40. Dashboard Tab



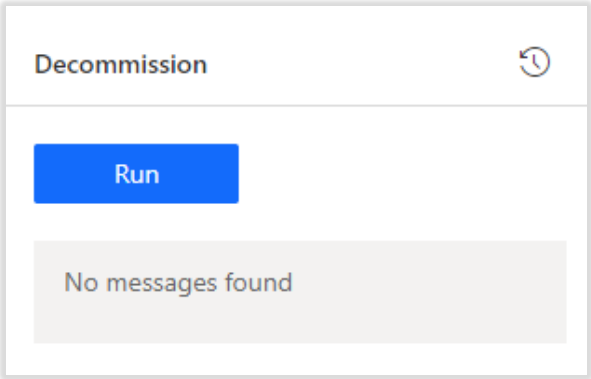
- Now select the **Commands** tab.

Figure 41. Commands Tab



- On the text area named *Decommission* and click **Run.**

Figure 42. Decommission



§

4.0 Telemetry Data

The Intel® In-Band Manageability provides two types of telemetry data, static telemetry and dynamic telemetry. The telemetry data will indicate the health of each endpoint.

4.1 Static Telemetry

This contains the following information and can be viewed under the **Properties** tab for a selected *Device*.

- BIOS-release-date
- BIOS-vendor
- BIOS-version
- CPU-ID
- OS-information
- System-Manufacturer
- System-Product-Name
- Total-physical-memory
- System-Product-Name

4.2 Dynamic Telemetry

Each endpoint publishes the following Dynamic Telemetry Data in 5-minute intervals.

- Available-memory
- Core-temp-Celsius
- Percent-disk-used
- System-cpu-percent
- Container-stats(cpu-usage)
- Network Information

4.3 Viewing Telemetry Data

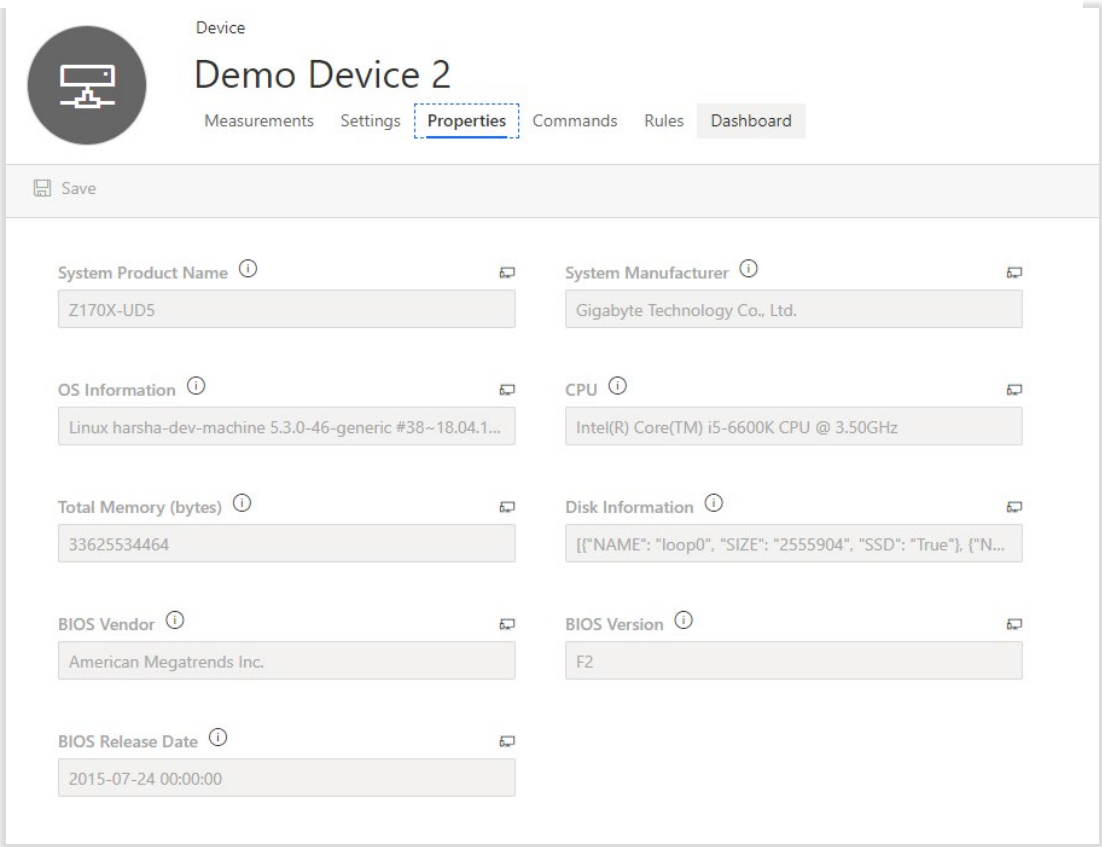
The device must be connected in order to view the telemetry information on the Azure* portal.

To view the telemetry data, navigate to the device item that is provisioned. Refer to Section 2.5.1.

4.3.1 Static Telemetry:

To view the device's static telemetry, click the **Properties** tab of the device item.

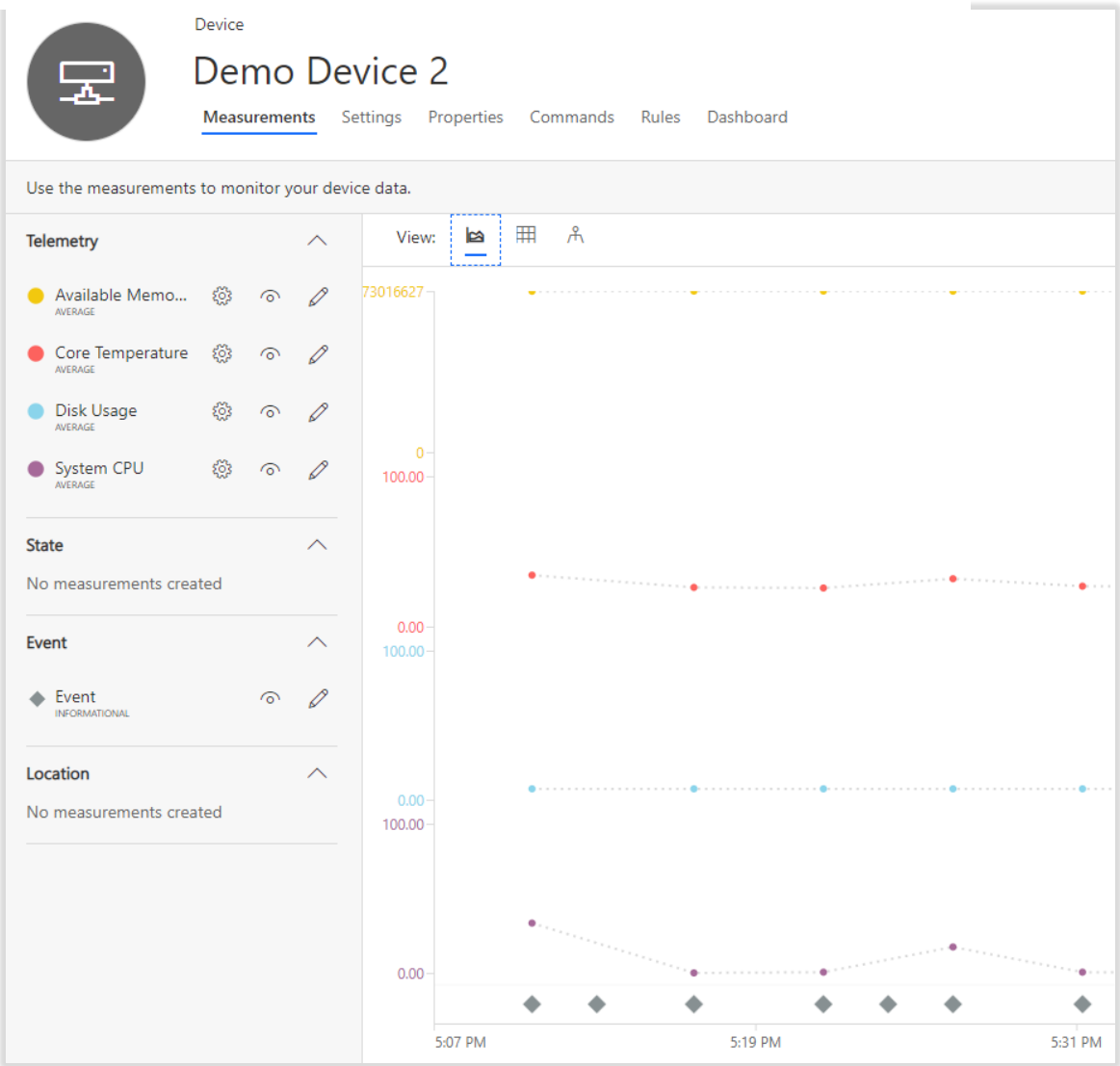
Figure 43. Properties Tab



4.3.2 Dynamic Telemetry:

To view the device's static telemetry, click the **Measurements** tab of the device item.

Figure 44. Measurements Tab



§

5.0 Issues and Troubleshooting

5.1 Error viewing Devices on Azure* Portal:

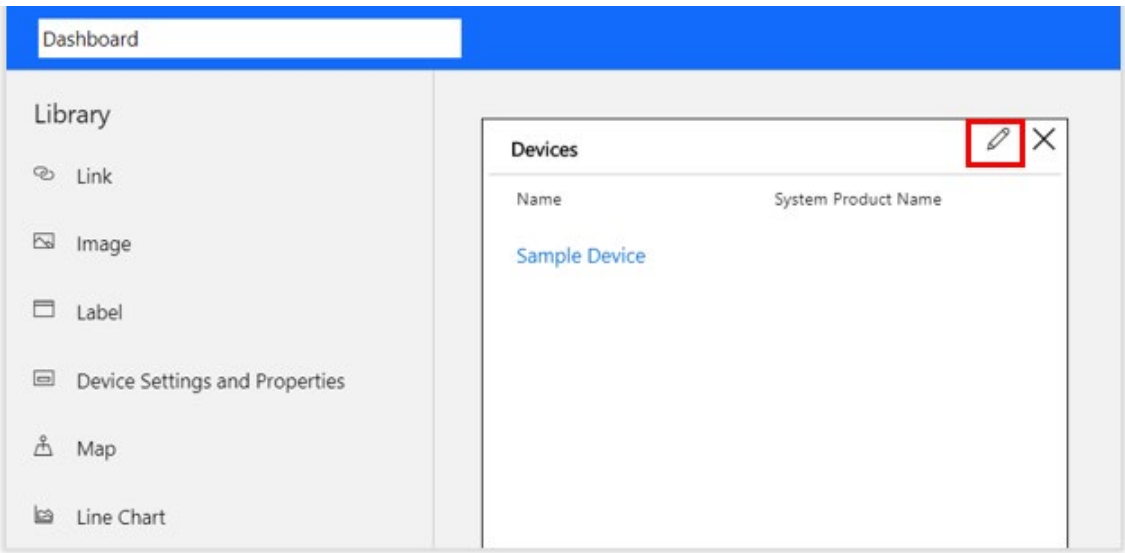
While following the steps in Section 2.5.1, if there is an error viewing device, do as below:

- Click **Edit** in the upper right-hand corner:

Figure 45. Edit

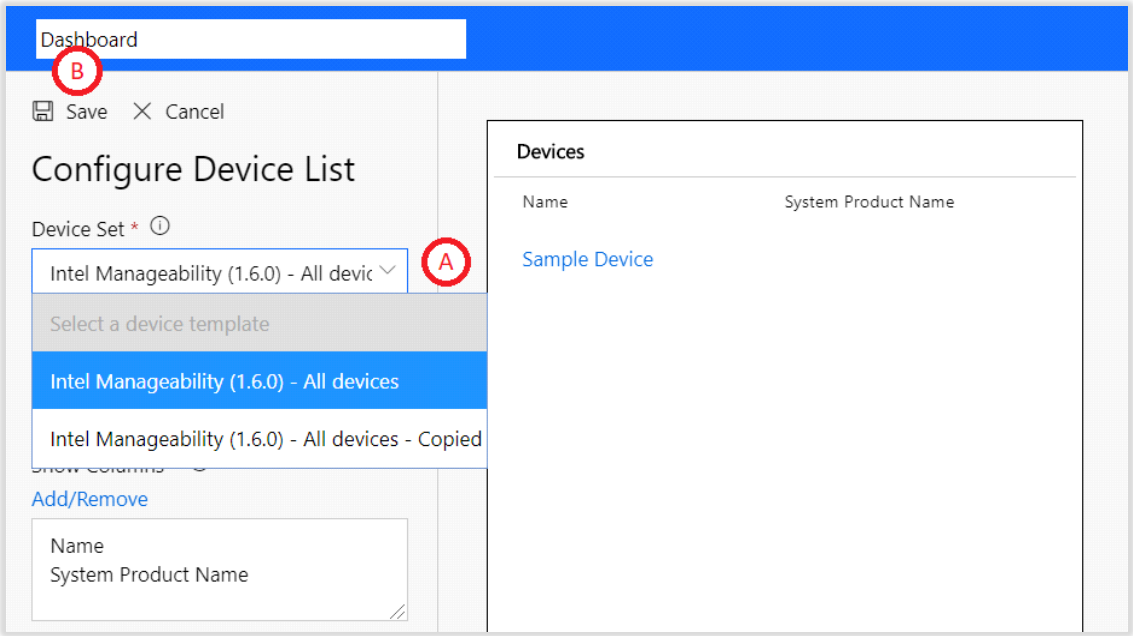
- Hover the cursor over the *Devices* panel, and click the icon:

Figure 46. Devices



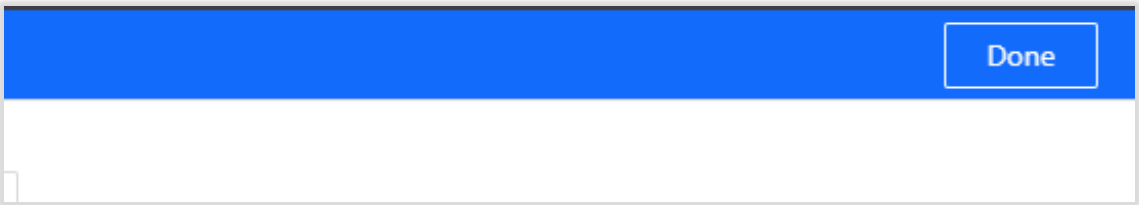
- On the left-hand panel, click **Device Set** and select the option without "Copied" appended to it Ⓐ, then click **Save** Ⓑ:

Figure 47. Device Set



- Finally, click **Done** in the upper right-hand- corner:

Figure 48. Click Done



5.2 Agents unable to Start After Provisioning

Incase if the agents are unable to start after provisioning and/or struck while creating symlinks at the end of provisioning there is a chance that other system services that are waiting might possibly blocked the INB services from starting. In order to fix this issue, follow the steps:

Check if bootup is complete or not using the command:

```
sudo systemd-analyze critical-chain
```

If the boot-up isn't complete, list all the jobs:

```
sudo systemctl list-jobs
```

Stop all the jobs that are under 'waiting' state:

```
sudo systemctl stop <job_unit_name>
```

And try provisioning the device again using 'provision-tc' command.

5.3 OTA Error Status

Refer to **Developer Guide Documentation.**

5.4 Dispatcher-Agent not Receiving Messages:

Refer to **Developer Guide Documentation**.

5.5 Acquiring Debug Messages from Agents

Refer to **Developer Guide Documentation.**

§