

In-Band Manageability Framework

User Guide – ThingsBoard*

August 2021

Revision 2.8.1

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 information provided here is subject to change without notice. Contact your Intel® representative to obtain the latest Intel® product specifications and roadmaps.

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.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or by visiting: <u>www.intel.com/design/literature.htm</u>.

Intel® technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at <u>intel.com</u> or from the OEM or retailer.

No computer system can be absolutely secure.

 \odot 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.



Contents

1.0	Intro	duction	5	
	1.1	Purpose	6	
	1.2	Audience		
	1.3	Terminology	7	
2.0	Thing	JsBoard* Overview		
	2.1	Getting Started with ThingsBoard*	8	
	2.2	Adding a Device	11	
	2.3	Obtaining Device Credentials	13	
	2.4	Creating a Device to Use X509 Auth	14	
	2.5	Provisioning a Device	15	
	2.6	Setting up the Dashboards	20	
	2.7	Getting Familiar with ThingsBoard	21	
	2.8	Interacting with Individual Devices	22	
	2.9	Interacting with Multiple Devices	23	
	2.10	Modifying and Working with Intel Manageability Widgets	24	
3.0	ΟΤΑ	Updates	26	
	3.1	Trusted Repositories	26	
	3.1 3.2	Trusted Repositories Preparing OTA Update Packages	26 26	
	3.1 3.2 3.3	Trusted Repositories Preparing OTA Update Packages OTA Commands	26 26 30	
	3.1 3.2 3.3 3.4	Trusted Repositories Preparing OTA Update Packages OTA Commands Configuration Update	26 26 30 52	
	3.1 3.2 3.3 3.4 3.5	Trusted Repositories Preparing OTA Update Packages OTA Commands Configuration Update Power Management	26 26 30 52 61	
	3.1 3.2 3.3 3.4 3.5 3.6	Trusted Repositories Preparing OTA Update Packages OTA Commands Configuration Update Power Management Decommission Command	26 	
4.0	3.1 3.2 3.3 3.4 3.5 3.6 Telen	Trusted Repositories Preparing OTA Update Packages OTA Commands Configuration Update Power Management Decommission Command	26 	
4.0	3.1 3.2 3.3 3.4 3.5 3.6 Telen 4.1	Trusted Repositories Preparing OTA Update Packages OTA Commands Configuration Update Power Management Decommission Command netry Data Static Telemetry	26 30 52 61 63 64	
4.0	3.1 3.2 3.3 3.4 3.5 3.6 Telen 4.1 4.2	Trusted Repositories Preparing OTA Update Packages OTA Commands Configuration Update Power Management Decommission Command netry Data Static Telemetry Dynamic Telemetry	26 	
4.0	3.1 3.2 3.3 3.4 3.5 3.6 Telen 4.1 4.2 4.3	Trusted Repositories Preparing OTA Update Packages OTA Commands Configuration Update Power Management Decommission Command netry Data Static Telemetry Dynamic Telemetry Data		
4.0	3.1 3.2 3.3 3.4 3.5 3.6 Telen 4.1 4.2 4.3 Issue	Trusted Repositories Preparing OTA Update Packages OTA Commands Configuration Update Power Management Decommission Command netry Data Static Telemetry Dynamic Telemetry Viewing Telemetry Data	26 	
4. 0 5.0	3.1 3.2 3.3 3.4 3.5 3.6 Telen 4.1 4.2 4.3 Issue 5.1	Trusted Repositories Preparing OTA Update Packages OTA Commands Configuration Update Power Management Decommission Command netry Data Static Telemetry Dynamic Telemetry Viewing Telemetry Data es and Troubleshooting OTA Error Status		
4.0 5.0	3.1 3.2 3.3 3.4 3.5 3.6 Telen 4.1 4.2 4.3 Issue 5.1 5.2	Trusted Repositories Preparing OTA Update Packages OTA Commands Configuration Update Power Management Decommission Command Decommission Command Static Telemetry Dynamic Telemetry Viewing Telemetry Data es and Troubleshooting OTA Error Status Provisioning Unsuccessful or Device Not Connected to Cloud		

Revision History

Date	Revision	Description
August 2021	2.8.1	Update the provision script in use.
June 2021	2.8	Added provision-tc parameters info Added clarity on AOTA package generation
May 2021	2.x	Include X509 authentication mechanism and X509 based OTA package verification.
April 2021	2.x	Add customer NOTE on Trusted repositories
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

In-Band Manageability Framework (a.k.a. INB) is a Software running on Edge IoT Device, which enables an administrator to perform critical Device Management operations over-the-air remotely from the cloud. It also facilitates 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, ThingsBoard.io*, and so on).

Some of the key advantages of Intel's 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 ThingsBoard.

The Device Management use-cases covered by the 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 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 ThingsBoard Cloud Service •
- Provision the Edge IoT device running In-Band Manageability Framework
- Perform OTA updates through ThingsBoard.

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.
- ٠ Independent Software Vendors (ISV) providing OS and Application update packages.
- System Integrators administrating devices running In-Band Manageability framework. ٠

Terminology 1.3

Term	Description
ΑΟΤΑ	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	In-Band Manageability Framework
loT	Internet of Things
OS	Operating System
ΟΤΑ	Over-the-air
SMBIOS	System Management BIOS
SOTA	Software Over the Air (OS update)

§

7



2.0 ThingsBoard* Overview

Getting Started with ThingsBoard* 2.1

Creating a ThingsBoard account and obtaining the connection tokens from ThingsBoard is required for provisioning/enabling In-Band Manageability Over-the-Air updates. For reference and quick setup, you will also need to import INB's ThingsBoard things definition. 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:

- Accessing ThingsBoard
- Setting up ThingsBoard
- Setting up ThingsBoard TLS
- Changing ThingsBoard Server Port
- Creating ThingsBoard Account

2.1.1 Accessing ThingsBoard

To set up a ThingsBoard installation, follow the steps below:

Accessing ThingsBoard

- If not already done, create a ThingsBoard installation through the following link: https://thingsboard.io/docs/installation/ Note: For a sandbox environment, choose the "Community" edition
- In order to run a ThingsBoard server instance on the same device as Intel Manageability, see Changing ThingsBoard server port see section **Changing ThingsBoard Server Port**

2.1.2 Setting up ThingsBoard TLS

To allow for a secure TLS connection to be established between a device with Intel Manageability and a self-hosted ThingsBoard server, some configuration must be done to the server. Information on that process can be found below, or at: https://thingsboard.io/docs/user-guide/mgtt-over-ssl/

- 1. Download the server.keygen.sh and keygen.properties files from the link above
- 2. Fill out the *keygen.properties* accordingly
 - a. The only necessary change is the DOMAIN_SUFFIX field, which should match the hostname of the ThingsBoard server

In-Band Manageability Framework User Guide



- b. Any other changes (e.g. the **SERVER_*_PASSWORD** fields) should be noted
- 3. Run the server.keygen.sh file with root privileges
- Copy the resulting *.jks file to the ThingsBoard configuration directory
 a. This may be under: /etc/thingsboard/conf/
- 5. The *.pub.pem file will be needed later to provision Intel Manageability devices

2.1.3 Changing ThingsBoard Server Port

Because both ThingsBoard and the Intel Manageability framework use MQTT protocol, it is necessary to change the ThingsBoard MQTT Broker port to a different number for both to coexist on the same device.

To do this:

- 1. Locate and open the thingsboard.yml file
 - a. On Yocto, this file is located in /etc/thingsboard/conf/
- 2. Change the property transport > mqtt > bind_port to any other number (e.g. 2883)
 - a. This property should be under a section labeled: "Local MQTT transport parameters"
 - b. Be sure to note the new port number, and enter it accordingly in the provisioning step
- 3. If the ThingsBoard service is currently running, restart it to apply the changes

For the Docker version of Thingsboard, change the binding for port 1883, for example, with 2883:

docker run -it -p 9090:9090 -p 2883:1883 -p 5683:5683/udp -v ~/.mytb-data:/data -v ~/.mytb-logs:/var/log/thingsboard --name mytb --restart always thingsboard/tb-postgres

2.1.4 Creating a ThingsBoard Account

If not done already, a ThingsBoard account will need to be created by a ThingsBoard System Administrator. Note that in order to provision devices and set up the dashboard, an account with the privileges of a "Tenant Administrator" is required:

 Log into a system administrator account; the default system administrator account details can be found here: <u>https://thingsboard.io/docs/samples/demo-account/#system-</u> administrator



2. Add a tenant by clicking on "Tenants" (A), then the plus button (B):

👸 ThingsBoard	💒 Tenants	۹	0	:
🔒 НОМЕ				
2 TENANTS				
WIDGETS LIBRARY				
🔅 SYSTEM SETTINGS 🛛 🗸				
	NO TENANTS FOUND			
	NO TENANTO FOOD			
	(B)-	→	+
		-		

- 3. Fill out the form that appears accordingly, then click "Add"
- 4. The tenant should appear as a new entry; click the Users icon:

🗆 Demo Users	
NO ADDRESS	Manage tenant admins
	Θ 🖬

- 5. On the page that appears, click the big plus icon, and fill out the form accordingly
- 6. After clicking add, the new user should be presented with an activation link
- 7. Clicking on the activation link will lead to a page where the account password is set
- 8. The new user should now be able to sign in with the account's associated email and password



2.2 Adding a Device

1. Add a device by clicking on "Devices" (A), then the plus button (B), then the add button (C):





2. The following window should appear; fill it out accordingly, then click "Add". Note that Device Type must be set to "Intel Manageability"

Add Device ?	×
Name* Demo Device	
Device type * Intel Manageability ×	
Label	
S gateway	
Description	
ADD CANCEL	

2.3 Obtaining Device Credentials

1. Click the shield icon on the newly created menu entry:



2. Note the Access Token in the window that appears:

Device Credentials	s X
Credentials type Access token	Ŧ
Access token *	
	20 / 20
SAVE	CANCEL

2.4 Creating a Device to Use X509 Auth

2.4.1 Generating Device Keys and Certificates

Prior to having the device authentication done using X509 mechanism, it is mandatory to have TLS set on the ThingsBoard server. Refer <u>section 2.1.2</u> on how to setup ThingsBoard TLS.

Once the TLS is set up on the server, the instructions on how to generate a client-side certificate can be found in the following link:

https://thingsboard.io/docs/user-guide/certificates/

- Enter and save the *keygen.properties* accordingly and download the *client.keygen.sh* script.
- Running the script will generate .jks, .nopass.pem, .pub.pem files.
- The .nopass.pem file is used during provisioning in <u>section 2.5</u>.
- The *.pub.pem* file content is used during the creation of a device on the ThingsBoard portal in <u>section 2.4.2</u>.

2.4.2 Enrolling Device Created with X509 Public Key

1. Once the device is added as shown in <u>section 2.2</u>, click the **shield** icon on the created device entry.



2. Select X.509 Certificate as the Credentials type.

Device Credentials \times			
Credentials type X.509 Certificate			
RSA public key *			
SAVE CANCEL			

3. Copy and paste the content of the client public key generated in the **RSA public key** field and click **Save**. Instructions on how to generate device certificates and keys are available in <u>section 2.4.1</u>.

2.5 Provisioning a Device

1. Locate and launch the Intel Manageability provisioning script (*provision-tc.sh*):

\$ sudo provision-tc



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



3. A prompt appears to choose the cloud service; press 3 and [ENTER] for ThingsBoard:



4. A prompt appears for the IP address and Port set up in section Accessing ThingsBoard

Please enter the server IP:

127.000.000.1

Please enter the server port (default 1883):

8883

Note that the server port entry can be left empty to use the default port



5. A prompt for **Device provision type** appears; select the type of device authentication preferred:

Please choose provision type.

- 1: Token authentication
- 2: X509 authentication

6. Choosing option 1. Token Authentication requires user to enter the token fetched in [B]

Please enter the device token:

AsfghbvfnaJI12

7. Choosing option **2. X509 Authentication** requires user to have device certificate and key generated as mention in <u>section 2.4.1</u>. The file path of the file with extension *nopass.pem* is entered in the prompt.





8. If user selects Token based authentication in step 6, an option for TLS will appear; press **Y** if the server was configured in <u>Setting up ThingsBoard TLS</u>. Otherwise, press **N** and skip to step 11.

Configure TLS? [Y/N]

If the user selects X509 authentication, it is mandatory to have TLS configured. By default, the application proceeds with the TLS configuration.

9. Choose an input method for the *.*pub.pem* file. The "Absolute file path" option requires a path to the file that does not include wildcards like ~. The "Console input" option will ask for the file to be input into the console; note that all lines preceding a line break cannot be edited:

```
Configuring TLS.
Input ThingsBoard CA from file? [Y/N] y
Please enter a filename to import
ThingsBoard CA file (*.pub.pem):
/home/abc/mqttserver.pub.pem
```

10. If the cloud provisioning is successful, the following message appears:

Successfully configured cloud service!

11. A Yes/No user prompt appears asking for a certificate verification on an OTA package. Choose 'Y' if FOTA/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]
```

12. In-Band Manageability Framework Services are Enabled and Started.

Enabling and starting agents...

Created symlink /etc/systemd/system/multi-user.target.wants/configuration.service → /etc/systemd/system/configuration.service.

Created symlink /etc/systemd/system/multi-user.target.wants/dispatcher.service → /etc/systemd/system/dispatcher.service.

Created symlink /etc/systemd/system/multi-user.target.wants/diagnostic.service → /etc/systemd/system/diagnostic.service.

Created symlink /etc/systemd/system/multi-user.target.wants/cloudadapter.service → /etc/systemd/system/cloudadapter.service.

Created symlink /etc/systemd/system/multi-user.target.wants/telemetry.service → /etc/systemd/system/telemetry.service.

Turtle Creek Provisioning Complete

The script will then start the Intel Manageability services; when the script finishes, the device should be able to interact with the ThingsBoard dashboard; see <u>Setting up the Dashboards.</u>

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

Note: If you cannot provision it successfully, refer to Provisioning Unsuccessful for Troubleshooting.

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 with detecting automatically TPM is present or not:

\$sudo PROVISION_TPM=auto provision-tc

To run without TPM security:

\$sudo PROVISION_TPM=disable provision-tc

2.6 Setting up the Dashboards

1. Click "Widgets Library" (A, then the plus button (B, then the import button (C):

Demo User Tenant administrator
A
± /
± /

2. The following window should appear. Choose the *intel_manageability_widgets.json* file; if INB has been installed, this file can be found at */usr/share/cloudadapter-agent/thingsboard/*

Import widgets bundle	×
Widgets bundle file	
Drop a JSON file or click to select a file to upload.	×
No file selected	
IMPORT CANC	EL

- 3. Click "Dashboards", then the plus button and the import button as before
- 4. A window similar to the one in step 4 should appear; this time, choose the *intel_manageability_devices.json* and *intel_manageability_batch.json* files, which can also be found in the same directory.

In-Band Manageability Framework User Guide 20

5. The dashboards should now appear as options in the menu.

2.7 Getting Familiar with ThingsBoard

More information on using ThingsBoard can be found at: https://thingsboard.io/docs/

Managing Devices

To manage devices, c	:lick "Devices" (A:	
مَرْضِي ThingsBoard	[₀ī Devices	Q [] I
🔒 НОМЕ		
↔ RULE CHAINS	Demo Device	
LUSTOMERS		
ASSETS		
ENTITY VIEWS		
WIDGETS LIBRARY	\bigcirc \bigcirc	
DASHBOARDS	č č	
💩 AUDIT LOGS		
• B Edit the de	etails for a device, click the device entry	
• C View the a	ccess token	

D Remove the device
 E Add a device; see <u>Adding a Device</u>

2.8 Interacting with Individual Devices

To access the dashboard, click "Dashboards" (A), then on the "Intel Manageability Devices" entry



A dashboard similar to the one below should appear:

👸 ThingsBoard	E Dashboards > E Intel Manageability Devices	A B C C ant administrator
🛧 НОМЕ	Intel Manageability	Intel Manageability Devices 👻 🖬 Demo Device 🔇 Realtime - last day 🛓 🚼
↔ RULE CHAINS		
	Status: Online	
ASSETS	Demo Device	Dynamic Telemetry
	Product: NUC7/3DNKTC	40 % _ 40 °C _ 200000000 bytes
ENTITY VIEWS	Manufacturer: Intel(R) Client Systems	42/0 20000000 bree
WIDGETS LIBRARY	OS: Linux tc-NUC7/3DNKTC 4.15.0-58-generic #64~16.04.1-Ubunt u SMP Wed Aug 7 14:10:35 UTC 2019 x86_64 x86_64	20 5 41°C
	CPU: Intel(R) Core(TM) i3-7100U CPU @ 2.40GHz	20% 20% 20000000 here
MUDITLOGS	Restort Device SHUTT Restort Device SHUTT	1 31° 200000000 (washington) 1/100 1/100 20000 2000 62/100 64/100 60/1
(TREGER AGTA TREG Event Log Realtime - last 30 days Image: State	State THICGER HOTA THICGER MOTA Q. 12:

(A) Change the dashboard (that is, to the Intel Manageability Batch; see <u>Interacting with</u> <u>Multiple Devices</u>

• B Change the device being viewed

In-Band Manageability Framework User Guide 22



- C Change the time interval of the dashboard, affecting the dynamic telemetry displayed
- D Display the online status of the device; click the bar to manually check the online status
- (E) Display the static and dynamic telemetry of the device
- (F) Trigger a remote procedure call by clicking on the corresponding button
- G View the event logs of the device

2.9 Interacting with Multiple Devices

To access the dashboard, click "Dashboards" (A), then on the "Intel Manageability Batch" entry (B):



🖏 ThingsBoard	📰 Dashboards > 🚦 Intel Manageability Batch	A Construction Construction	strator I
🟫 НОМЕ	Intel Manageability	Intel Manageability Batch 🔹 🗔 Entities 🕓 Realtime - last day	± 0
< → RULE CHAINS	(в)	(c)	
	Intel Manageability Devices		- 1
ASSETS	Q Search	SHUTDOWN DEVICE REBOUT DEVICE	- :
	i		
	Demo Device		}
WIDGETS LIBRARY	Test Device 1	MANIFEST UPDATE TRIGGER CONFIG UPDATE	- 1
DASHBOARDS	V Test Device 2		
W AUDIT LOGS	V Test Device 3		
	Test Device 4	TRIGGER MOTA TRIGGER FOTA	
	Test Device 5		- 1
	8 8 9		- 1
	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1		- 1
	<u> </u>		!
	L Event Log		Q (3
0.(D Realtime - last 30 days		
	DEMO DEVICE TEST DEVICE 1 TEST DEVICE 2 TEST DEVICE 3 1	EST DEVICE 4 TEST DEVICE 5	
	Timestamp V Message	headeast' will Padrass' "-1" "astroats' "III III III III III III III III III I	-
	2019-09-10 [[radfores: "134 143 152 65", refmail: "255 255 05", broadcast: "134 143 132 65", refmail: "255 255 05", broadcast: "134 144 132 65", refmail: "255 255 05", broadcast: "134 144 135 65", refmail: "255 255 05", broadcast: "134 144 135 65", refmail: "255 255 05", broadcast: "134 144 135 65", refmail: "255 255 05", broadcast: "134 144 135 65", refmail: "255 255 05", broadcast: "134 144 135 65", refmail: "255 255 05", broadcast: "134 134 135 65", refmail: "255 255 05", broadcast: "134 134 135 65", refmail: "255 255 05", broadcast: "134 134 135 65", refmail: "255 255 05", broadcast: "134 134 135 65", refmail: "255 255 05", broadcast: "134 134 134 135 65", refmail: "255 255 05", broadcast: "134 134 134 135 65", refmail: "255 255 05", broadcast: "134 134 134 135 65", refmail: "255 255 05", broadcast: "134 134 134 135 65", refmail: "255 255 05", broadcast: "134 134 134 135 65", refmail: "255 255 05", broadcast: "134 134 134 134 134 134 134 134 134 134	Areaces hair, sodres heines, mininkinninninninninninnin diezes hair aus sodres tooococococo entres, hair so 2.255 (236) addees fabo 558 032-032 fabora), fabora si fabora fabora fabora fabora fabora fabora fabora fabora Skart 172 1725255 (206) est 160.242 Officiel 554 (206) est 1757 est 1767 est 1767 est 1767 est 1767 est 1767 est U broadcast "Hittiffittiffit (160) est 1767 est	A
	2019-09-10 13:11:09 Container Health check timed out. Please check health of the diagnostic agent		
	2019-09-10 [Caddress '1:341:35.66", refumsk': '255.255.0", broadcast': '1341:341 13:08:59 'fffffffffff, 'j, 'docker0': [Caddress': '127.0.1", 'netmask': '255.255.00', broadcast': '1341:441	zoadcast", mil), "address", "1", "netmasis", "fffffffffffffffffffffffffffffffffff	10 AL
	2019-09-10 world"; "pupPercent", "Orania": "[name" "hello-world"; "puPercent", "O, name 13:03:58 world"; "puPercent", "O, name" "hello-world"; "puPercent", "O, name" world"; "puPercent", "O, name", "hello-world"; "puPercent", "O, name", "hello-world"; "puPercent", "O, name", "hello-world"; "puPercent", "O, name", "hello-world"; "puPercent", "D, name, "hello-world"; "puPercent, "puPercent, "puPercent, "puPercent, "puPer	helio-wold "spuPersent" 03, hame" (helio-wold "spuPersent" 04, hame "helio-wold "spuPersent" 04, hame "helio-wold "spuPersent" 04, hame "helio- wersent" 04, hame "helio-wold "spuPersent" 04, hame "helio-wold "s	
	2019-09-10 3:01:05 Container Health check timed out. Please check health of the diagnostic agent	Poweredby Ibing	asboard x2.4.0

- (A) Change the dashboard (that is, to the Intel Manageability Devices; see (ii))
- B Select the devices to send the batch remote procedure call to
- © Trigger a remote procedure call by clicking on the corresponding button
- D View the event logs for all devices

2.10 Modifying and Working with Intel Manageability Widgets

ThingsBoard widgets are coded in HTML/CSS/JavaScript + AngularJS through ThingsBoard's built-in Widget IDE. They can then be added to dashboards and exported for later use. Resources on how to edit and use the widgets can be found below:

- <u>Getting Started with AngularJS</u>
- About AngularJS Material
- ThingsBoard Widgets Development Guide

The Intel Manageability Widgets bundle consists of seven widgets:

- Device Information: Provides a well formatted device properties display
- Connectivity Status: Displays the device connectivity status
- Dynamic Telemetry: Extension of built-in time series display
- Event Log: Extension of built-in textual time series display
- **OTA Form**: Provides a flexible RPC request trigger with user input fields
- Device List: Provides a selection box for batch RPC calls



• Docker Stats Widget: Provides a human readable view of the latest Docker statistics

The Device Information is self-contained. However, the other widgets communicate with each other through <u>Custom JavaScript Events</u>. The relationships of the events are illustrated below:



The Events allow for the widgets to overcome their single API limitation:

- Connectivity Status is actually an RPC widget, hence it cannot access Time Series data. To overcome this, Dynamic Telemetry and Event Log widgets broadcast updates that Connectivity Status uses in its polling decisions.
- Device List is used to send a list of selected devices to OTA Form widgets for batch operations



3.0 OTA Updates

After the In-Band Manageability Framework running on the Edge IoT Device is provisioned, it will establish a secure session with the ThingsBoard portal, and the status of the device can is visible as 'Online' – refer as seen below:

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, In-band Manageability requires the Server URL(location) of the OTA update repository be included in a "trusted repository list", which is maintained internally. Hence, it is mandatory that the OTA URL be included in the "trusted repository list" prior to initiating an OTA command. This can be achieved via OTA configuration Append command to add a new Server URL the existing Trusted Repository list.

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: refer to <u>Configuration Append</u> for adding the Server URL in the trustedRepositories via 'Trigger Config Update'.

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.

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 <u>How to</u> <u>generate Signature</u> 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:



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 <u>How to generate Signature</u>

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 format below.

AOTA Command	AOTA Package Structure
AOTA Docker-Compose package	Container Tag == Container Image Name
(Same format for up/pull)	Example: The container Image name and the tar file name should be the same
	Container Tag =CPU
	Tar file = CPU.tar.gz
	<i>Note:</i> 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
	<pre>\$ cp docker-compose.yml CPU/.</pre>
	3.Tar the folder

Table 1. Creating AOTA Package

	\$ 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 <u>How to generate Signature</u>. 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 and just the *intel_manageability.conf* file alone. Archiving the *intel_manageability.conf* file with a *tar* archive tool can be performed with 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, use shar-256 or sha-384 based encryption mechanism.

3.3 OTA Commands

To trigger Over the Air (OTA) updates, Device Status should be online as seen in <u>Interacting with</u> <u>Multiple Devices</u>. Go to **DashBoards** tab and select the correct **DashBoard[1]** and under entities, select your **Edge Device[2]** and click any **OTA buttons[3]** as seen below.

🍇 ThingsBoard	📑 Dashboards 🕞 📮 Intel Manageability	Devices	1 J Turtle Creek
🔒 НОМЕ	Intel Manageability		Intel Manageability Devices 🔹 🗔 Teja-test 🕐 Realtime - last day 🛓 😭
↔ RULE CHAINS	Teja-test	8	Dynamic Telemetry
2 CUSTOMERS	Product: NUC8i5BEK		1% 1°C 1 bytes
ASSETS	Manufacturer: Intel(R) Client Systems	M 1-	1% - 1°C - 1bytes
	Ubuntu SMP Mon Jul 29 16:12:28 UTC 2 x86_64	019	0% - 0°C - 0bjtes
ENTITY VIEWS	CPU: Intel(R) Core(TM) i5-8259U CPU @ 2.30	GHz -	1% -1°C -1 bytes 10:00 12:00 14:00 16:00 18:00 22:00 02:00 02:00 04:00 06:00 08:00
WIDGETS LIBRARY	RAM: 8223293440 bytes BIOS Vendor: Intel Corp.		Available Memory
DASHBOARDS	BIOS Version: BECFL357.86A.0051.2018.1015.1513 BIOS Release 2018-10-15 00:00:00		Disk Usage 3
🕑 AUDIT LOGS	Date:	-	
	 Q. Docker Container Stats C. Last updated: 5/12/2020, 10:22:12 AM 	0	SHUTDOWN DEVICE REBOOT DEVICE DECOMMISSION DEVICE
	docker-bench-security cpuPercent: 0		TRIGGER AOTA MANIFEST UPDATE TRIGGER SOTA
10.23.221.141:8080/home	docker-bench-security cpuPercent: 0 docker-bench-security cpuPercent: 0		TRIGGER FOTA TRIGGER CONFIG UPDATE

Commands - Definitions 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 In-Band Manageability configurations
<u>Reboot</u>	Remotely reboot the Endpoint
<u>Shutdown</u>	Remotely shutdown the Endpoint
Manifest Update	Any OTA update type can be done via the Manifest Update, by entering XML text to update the Endpoint. (Refer Developer Guide)

3.3.1 AOTA Updates

Supported AOTA commands and their functionality:

'docker-compose' commands currently supported:

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

'docker' commands currently supported:

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

List of AOTA commands NOT supported

	Import
Docker-Compose	Load
	Stats
	Up
Docker	Down
	List

Field	Input description
App and it's command	Docker-Compose supports: Up, Down, Pull, List and Remove.
App and it's command	Docker supports: Load, Import, Pull, Remove and Stats
	Name tag for image/container.
Container Tag	Note: Conatiner Tag can have both the Name and Version in this
	format Image:Version
Docker Compose File	Field to specify the name of custom yaml file for docker-compose
	command. Example: custom.yml
	Server URL to download the AOTA container tar.gz file
Fetch	If the server requires username/password to download the file, you
	can provide in server username/ server password
	NOTE: Follow Creating AOTA Package



Server Username/ Server Password	If server where we host the package to download AOTA file needs credentials, we need to specify the username and password
	Specify Docker Registry if accessing any registry other than the default 'index.docker.io'.
Docker Registry Docker Registry	Example for docker Registry: amr-registry-pre.caas.intel.com
Username/Password	Optional fields Docker Registry Username/Password can be used to when using private images in AOTA through docker pull and docker- compose up, pull commands.

To trigger Application-over the Air updates click the 'Trigger AOTA' button as seen below.

🍇 ThingsBoard	📲 Dashboards > 📑 Intel Manageability Dev	ices : S Turtle Creek
🔒 НОМЕ	Intel Manageability	Intel Manageability Devices 🝷 🗔 Teja-test 🕓 Realtime - last day 👱 🛟
↔ NULE CHAINS	Teja-test C	2 Dynamic Telemetry
CUSTOMERS ASSETS CO DEVICES ENTITY VIEWS WIDGETS LIBRARY	Product: NUC8i5BEK Manufacturer: Intel(R) Client Systems OS: Linux teja123 5.0.0-23-generic #24~18.04.1- Ubuntu SMP Mon Jul 29 16:12:28 UTC 2019 x86_64 CPU: Intel(R) Core(TM) i5-8259U CPU @ 2.30GHz RAM: 8223293440 bytes BIOS Vendor: Intel Corp.	1% 1°C 1 bytes 1% 1°C 1 bytes 0% 0°C 0 bytes 0% 0°C 0 bytes 1% 1°C 1 bytes 1% 1°C 1 bytes 1% 1°C 1 bytes 1% 1°C 1 bytes 1000 1200 1400 1600 2200 0000 0200 0400 0600 0800 avg
 DASHBOARDS AUDIT LOGS 	BIOS Version: BECFL357.86A.0051.2018.1015.1513 BIOS Release 2018-10-15.00:00:00 Date: C Q Docker Container Stats C Last updated: 5/12/2020, 10:22:12 AM	Core Temperature Disk Usage CPU Usage SHUTDOWN DEVICE REBOOT DEVICE DECOMMISSION DEVICE
10.23.221.141:8080/home	docker-bench-security cpuPercent: 0 docker-bench-security cpuPercent: 0 docker-bench-security cpuPercent: 0	TRIGGER AOTA MANIFEST UPDATE TRIGGER SOTA TRIGGER FOTA TRIGGER CONFIG UPDATE Powered by Thingsboard v.2.4.1

Now, populate the AOTA pop-up window with the required parameters and then click "Send" to trigger the AOTA update.

The AOTA form has these fields:

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



The arrows indicat	tes		
Mandatory field		-	
Optional field		-	
Not required	\otimes		

For each of the AOTA functions, insert the correct parameters as described and click **'send' button. The results can be viewed** by clicking on the **Events** tab.

AOTA Docker-Compose Operations

Docker-Compose Up

Note: Follow <u>Creating AOTA Package</u> to create the AOTA Package.

Go through **<u>Fields and Description</u>** on when to use optional fields.

Docker-Compose *yaml* file should have the correct docker version.

	Trigger AOTA ×
	METHOD RESULTS
	App docker-compose ~
	Command up +
	Container Tag
	Fetch
	Docker Compose file
0	Signature
0	Version
>	Server Username
	Server Password
	Docker Registry
	Docker Username
	Docker Password
	SEND

Docker-Compose Down

	Trigger AOTA ×
_	METHOD RESULTS
	App docker-compose
	Command down *
	Container Tag
0	Fetch
0	Docker Compose file
0	Signature
\otimes	Version
0	Server Username
0	Server Password
0	Docker Registry
0	Docker Username
0	Docker Password
	SEND

Docker-Compose Pull

Note: Follow <u>Creating AOTA Package</u> to create the AOTA Package.

Go through <u>Fields and Description</u> on when to use optional fields.

Docker-Compose *yaml* file should have the correct docker version.

angger ito i		
METHOD	RESULTS	
App		
docker-cor	mpose	*
Command		
pull		*
Container 1	Tag	
Fetch		
Docker Cor	mpose file	
Signature		
Version		
Server Use	mame	
Server Pas	eword	
Docker Reg	gistry	
Docker Use	emame	
Docker Par	ssword	
		SEND

Docker-Compose List

_

37	Trigger AOTA ×
	METHOD RESULTS
	App docker-compose +
	Command list *
-	Container Tag
0	Fetch
0	Docker Compose file
0	Signature
0	Version
0	Server Username
0	Server Password
0	Docker Registry
0	Docker Username
0	Docker Password
	SEND

Docker-Compose Remove

	nic Telemetor
	Trigger AOTA ×
	METHOD RESULTS
-	App docker-compose ~
- 1	Command remove *
	Container Tag
0	Fetch
0	Docker Compose file
0-	Signature
0	Version
0	Server Username
0	Server Password
0	Docker Registry
\otimes	Docker Username
0	Docker Password
	SEND

Intel Confidential

39

AOTA Docker Operations

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

Follow Creating AOTA Package

	Trigger AOTA ×
	METHOD RESULTS
	App docker *
	Command import ~
-	Container Tag
-	Fetch
0	Docker Compose file
0	Signature
0	Version
	Server Username
	Server Password
0	Docker Registry
0	Docker Username
0	Docker Password
- 1	SEND

In-Band Manageability Framework User Guide 40



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

Follow Creating AOTA Package

ap	Trigger AOTA ×
- 6	METHOD RESULTS
	App docker ~
	Command load ~
	Container Tag
	Fetch
0	Docker Compose file
0	Signature
\otimes	Version
	Server Username
	Server Password
0	Docker Registry
\otimes	Docker Username
0	Docker Password
	SEND



Docker Pull

ſ	Trigger AOTA ×	
	METHOD RESULTS	
	App docker ~	
	Command pull *	_
	Container Tag	
0	Fetch	
0	Docker Compose file	
0	Signature	
0	Version	
0	Server Username	
0	Server Password	
	Docker Registry	
	Docker Username	
	Docker Password	
	SEND	

Docker Remove

	Trigger AOTA ×
	METHOD RESULTS
	App docker -
	Command remove *
-	Container Tag
0	Fetch
0	Docker Compose file
0	Signature
0	Version
0	Server Username
0	Server Password
0	Docker Registry
0	Docker Username
0	Docker Password
	SEND

Docker Stats

	Trigger AOTA ×
- [METHOD RESULTS
	App docker ~
	Command stats ~
0	Container Tag
0	Fetch
0	Docker Compose file
0	Signature
0	Version
0	Server Username
0	Server Password
0	Docker Registry
0	Docker Username
0	Docker Password
	SEND

3.3.2 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 In-Band Manageability Framework to initiate a Firmware update process.

Information	Field	Checks
FW	Vendor	Checks for string match between the user input and platform vendor
	Version	
	Release Date	Checks if the current firmware file release date is newer than release date on the platform
System	Manufacturer	Checks for a string match between the user input and platform manufacturer
	Product Name	Checks for string match between the user input and platform product name

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

Intel Confidential

Intel x86 UEFI-based products

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

\$ sudo dmidecode -t bios -t system

Parameter	Description
BIOSVersion	Verify with BIOS Vendor (IBV)
Fetch	Repository URL
	NOTE: Follow Creating FOTA Package
Manufacturer	Endpoint Manufacturer Name
Path	FOTA path created in repository
Product	Product name set by Manufacturer
Release Date	Specify the release date of the BIOS file you are applying
Release Date	Verify with BIOS Vendor (IBV)
Signature	Digital signature
ToolOptions	Any Tool options to be given for the Firmware Tool
Server Username/Password	If server where we host the package to download FOTA file needs credentials, we need to specify the username and password

FOTA Update via Button Click

In order to trigger Application-over the Air updates click the 'Trigger FOTA' button as seen below





Populate the FOTA pop-up window with the required parameters and click "send" to trigger the FOTA update.

	Trigger FOTA ×
	METHOD RESULTS
	BIOS Version
	Fetch
	Manufacturer
0	Path
	Product
	Release Date
\rightarrow	Signature
	Tool Options
	Vendor
	Server Username
	Server Password
	SEND

Indicators:



August 2021 Document Number: 626763-2.8.1

Intel Confidential

In-Band Manageability Framework User Guide 47



3.3.3 SOTA Updates

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

SOTA Update Via Button Click (Debian Package Manager and Ubuntu OS)

🙀 ThingsBoard	📑 Dashboards > 📑 Intel Manageability Devices	C O Turtle Creek
🔒 НОМЕ	Intel Manageability	Intel Manageability Devices 👻 🗔 Teja-test 🕓 Realtime - last day 👱 😭
< → RULE CHAINS	Teja-test 🖸	Dynamic Telemetry
💒 CUSTOMERS	Product: NUC8i5BEK	1% 1°C 1 bytes
ASSETS	Manufacturer: Intel(R) Client Systems	1% 1°C 1bytes
	US: Linux teja123 5.0.0-23-generic #24~18.04.1- Ubuntu SMP Mon Jul 29 16:12:28 UTC 2019 x86 64	0% 0°C 0bytes 0% 0°C 0bytes
	CPU: Intel(R) Core(TM) i5-8259U CPU @ 2.30GHz	-1% -1°C -1 bytes 10:00 12:00 14:00 16:00 18:00 20:00 00:00 02:00 04:00 06:00 08:00
WIDGETS LIBRARY	RAM: 8223293440 bytes BIOS Vendor: Intel Corp.	- Available Memory avg
DASHBOARDS	BIOS Version: BECFL357.86A.0051.2018.1015.1513	Core Temperature Disk Usage
O AUDIT LOGS	Date:	- CPU Usage
	Q Docker Container Stats	SHUTDOWN DEVICE REBOOT DEVICE DECOMMISSION DEVICE
	docker-bench-security cpuPercent: 0	TRIGGER AOTA MANIFEST UPDATE TRIGGER SOTA
	docker-bench-security cpuPercent: 0	
10.23.221.141:8080/home	docker-bench-security cpuPercent: 0	TRIGGER FOTA TRIGGER CONFIG UPDATE Powered by Thingsboard v2.4.7

In order to trigger Application-over the Air updates click the 'Trigger SOTA' button as seen below

Populate the SOTA pop-up screen with 'Log to File' as 'Yes' to have logs will be written to the file otherwise 'No' to have logs to be written to the cloud. SOTA log files can be located at the endpoint /var/cache/manageability/repository-tool/sota/.



Populate the SOTA pop-up window with the required parameters and click "send" to trigger the SOTA update.

	Trigger SOTA	×
	METHOD RESULTS	
	Command update	Ŧ
->	Log to File yes	•
0	Fetch	
0	Username	
0	Password	
		SEND
TI	he arrows indicates	
N	1andatory field	
0	ptional field —	

0

Not required

SOTA Update Via Button Click (Mender)

In order to trigger Application-over the Air updates click the 'Trigger SOTA' button as seen below

🕷 ThingsBoard	📲 Dashboards > 📲 Intel Manageability 🛛	Devices C Creek :
🟫 номе	Intel Manageability	Intel Manageability Devices 🔹 🗔 Teja-test 🕓 Realtime - last day 👱 []
<> RULE CHAINS	Teja-test	Dynamic Telemetry
CUSTOMERS ASSETS COD DEVICES ENTITY VIEWS WIDGETS LIBRARY CUSTOMERS	Product: NUC8I5BEK Manufacturer: Intel(R) Client Systems OS: Linux teja123 5.0.23-generic #24~18.04 Uburntu SMP Mon Jul 29 16:12:28 UTC 20'x86.64 CPU: Intel(R) Core(TM) i5-8259U CPU @ 2.30GF RAM: 8223293440 bytes BIOS Vendor: Intel Corp. BIOS Version: BECFL357.86A.0051.2018.1015.1513 BIOS Release 2018-10-15 00:000	1% 1% <td< th=""></td<>
(b) AUDIT LOGS	Q Docker Container Stats C Last updated: 5/12/2020, 10:22:12 AM docker-bench-security cpuPercent: 0 docker-bench-security cpuPercent: 0 docker-bench-security cpuPercent: 0	SHUTDOWN DEVICE REBOOT DEVICE DECOMMISSION DEVICE TRIGGER AOTA MANIFEST UPDATE TRIGGER SOTA TRIGGER FOTA TRIGGER CONFIG UPDATE Powerd by Thiroshourd 42.4.1

Populate the SOTA pop-up screen with 'Log to File' as 'Yes' to have logs will be written to the file otherwise 'No' to have logs to be written to the cloud. SOTA log files can be located at the endpoint /var/cache/manageability/repository-tool/sota/.



Populate the SOTA pop-up window with the required parameters and click "send" to trigger the SOTA update.

METHOD	RESULTS	
Command update		
Log to File yes		
Fetch		
Username		
Password		

The arrows indicates



3.4 Configuration Update

Configuration update is used to update, retrieve, append, and remove configuration parameter values from the Configuration file located at */etc/intel_manageability.conf*. Refer to table below to understand the configuration tags, its values and the description.

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 the system should have before or after an 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	Repository link	Location used to update Debian packages
Proceed Without Rollback	True	Whether SOTA update should go through even when rollback is not supported on the system.

In-Band Manageability Framework User Guide 52



Trigger Configs	Description of field
Command	Set: Command used to update the configuration value using key:value pair.
	Get : Command used to retrieve a specific configuration value using key:value pair
	Load : Command used to replace an entire configuration file.
	Append : Command used to append values to a configuration parameter.
	Remove : Command used to remove a specific values 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, set, append or remove in key:value format
Signature	Digital signature refers to [TBD]

Below are the configuration update commands and input field description

To trigger a configuration update, follow the steps below:

In order to trigger Application-over the Air updates click the 'Trigger Config Update' button as seen below



In-Band Manageability Framework User Guide

Intel Confidential

53

Populate the 'Trigger Config Update' pop-up window with the required parameters and click "send" to trigger the Config Update as shown below.

Configuration Update Via Button Click

Configuration Set

Required Fields: Command and Path

Not required Fields are crossed out in red

Examples:

To set a configuration value, input for path field in key:value format-> minStorageMB:10

To set multiple values at once use ;

Example to separate key:value pairs-> minStorageMB:10;minMemoryMB:250

Note: Field path takes in key:value pairs as an input with key as the configuration parameter tag and value as the value to be updated.

METHOD	RESULTS	
Command set		
Fetch		
Path minStorage	eMB:10	
Signature		

Results:

The configuration contents inside the file before and after the update are shown below

Before update	After update
<minstoragemb>100</minstoragemb> <minmemorymb>200</minmemorymb>	<minstoragemb>10</minstoragemb> <minmemorymb>250</minmemorymb>

Configuration Get

Required Fields: command and Path



Not required Fields are crossed out in red

Examples:

To get one configuration value, use configuration tag as input for path-> minStorageMB

To get multiple values at once use ; to separate tags-> minStorageMB;minMemoryMB

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

METHOD RES	ULTS	
Command get		*
Fetch		
Path minStorageMB		
Signature		

Results of Configuration Get update can be seen in the ThingsBoard Events Log in the dashboard below the OTA buttons.

Configuration Load

Required Fields: Command and Fetch

Optional Field: Signature

Not required Fields are crossed out in red

Note: Follow Creating Configuration Load Package

METHOD	RESULTS	
Command		
Fetch http://134.	134.155.60:8000/inte	el_manage
Fetch http://134. Path	134.155.60:8000/inte	el_manage
Fetch http://134. Path	134.155.60:8000/inte	el_manage

Configuration Append

Required Fields: Command and Path

Not required Fields are crossed out in red

- **Note:** 1. Append is only applicable to three configuration tags that is, trustedRepositories, sotaSW and ubuntuAptSource from the configuration file.
 - 2. Field path takes in key:value pair format

Example: If you need to add a new Server URL to download OTA package, path is trustedRepositories:https://af01p-igk.devtools.intel.com/artifactory/turtle-creek/test/

ŀ	METHOD	RESULTS	
	Command append		0
	Fetch		
	Path trustedRep	ositories:https://af01	p-igk.devto

End Results of Configuration Append:



Configuration Remove

Required Fields: Command and Path

Not required Fields are crossed out in red

Note: Remove is only applicable to three configuration tags that are, trustedRepositories, sotaSW, and ubuntuAptSource

Field path takes in key value pair format, example: trustedRepositories:https://af01pigk.devtools.intel.com/artifactory/turtle-creek/test/

METHOD	RESULTS
Command remove	
Fetch	
Path trustedRe	positories:https://af01p-igk.devto
Signatura	



End Results of Configuration Remove:

Before configuration remove trigger (Removing the highlighted value)

<trustedRepositories>
https://af0lp-igk.devtools.intel.com/artifactory/SID-Docker-local/bmp/test
https://ubit-artifactory-or.intel.com/artifactory/iotg-bmp-internal-local/
https://af0lp-igk.devtools.intel.com/artifactory/iotg-bmp-igk-local/
https://af0lp-igk.devtools.intel.com/artifactory/turtle-creek/test/
</trustedRepositories>

After configuration remove update, you can see the value removed

After configuration remove update, you can see the value removed

<



3.5 Power Management

Shutdown and Restart capabilities are supported via button click as seen below.

System Reboot Via Button Click

Click the 'Reboot Button' as seen below in the dashboard to trigger a Device Reboot

🍇 ThingsBoard	📲 Dashboar	rds 🔹 📲 Intel Manageability Devices	: Carlte Greek	
🛧 НОМЕ	Intel Managea	bility	Intel Manageability Devices 👻 🗔 Teja-test 🕓 Realtime - last day 👱 []	
< ↔ RULE CHAINS	Teja-test	0	Dynamic Telemetry	
CUSTOMERS Solution CONTRACT	Product: Manufacturer: OS:	NUC8/58EK Intel(R) Client Systems Linux teja123 5.0.0-23-generic #24~18.04.1- Ubuntu SMP Mon Jul 29 16:12:28 UTC 2019 x86_64	1% 1% 1bjts 1% 1% 1bjts 0% 0% 0bjts 0% 0% 0bjts	
ENTITY VIEWS WIDGETS LIBRARY DASHBOARDS	CPU: RAM: BIOS Vendor; BIOS Version: BIOS Release Date:	Intel(R) Core(TM) i5-8259U CPU @ 2.30GHz 8223293440 bytes Intel Corp. BECFL357.86A.0051.2018.1015.1513 2018-10-15 00:00.00	-1 10 -1 10 -1 10 1 100 <td< th=""><th></th></td<>	
	Q Docker I C Last upd docker-bench-se cpuPercent: 0 docker-bench-se cpuPercent: 0	Container Stats 22	SHUTDOWN DEVICE REBOOT DEVICE DECOMMISSION DEVICE TRIGGER AOTA MANIFEST UPDATE TRIGGER SOTA	
10.23.221.141:8080/home	docker-bench-se cpuPercent: 0	curity	TRIGGER FOTA TRIGGER CONFIG UPDATE Powered by Thingsboard v.2.4.1	

Now on the pop-up window shows up, click the 'Send' button on the box titled 'Reboot Device'.

Reboot Devic	e	×
METHOD	RESULTS	
	SEND	

System Shutdown Via Button Click

Click the 'Shutdown Button' as seen below in the dashboard to trigger a Device Reboot.

🕵 ThingsBoard	📲 Dashboa	rds 🕞 📑 Intel Manageability Devices	: O Turtle Creek :
🔒 НОМЕ	Intel Managea	bility	Intel Manageability Devices \star 🗔 Teja-test 🕓 Realtime - last day 🛓 🚦
<> RULE CHAINS	Teja-test	0	Dynamic Telemetry
22 CUSTOMERS	Product:	NUC8i5BEK	1% 1°C 1 bytes
ASSETS	Manufacturer:	Intel(R) Client Systems	1% - 1°C - 1 bytes
	OS:	Linux teja123 5.0.0-23-generic #24~18.04.1- Ubuntu SMP Mon Jul 29 16:12:28 UTC 2019 x86_64	0% 0°C 0bytes
🖬 ENTITY VIEWS	CPU:	Intel(R) Core(TM) i5-8259U CPU @ 2.30GHz	-1% -1°C -1 bytes
👫 WIDGETS LIBRARY	RAM: BIOS Vendor:	8223293440 bytes Intel Corp.	Available Memory
DASHBOARDS	BIOS Version: BIOS Release	BECFL357.86A.0051.2018.1015.1513	Core temperature Disk Usage
🕑 AUDIT LOGS	Date:	×	- CPU Usage
	Q Docker C Last upp	Container Stats	SHUTDOWN DEVICE REBOOT DEVICE DECOMMISSION DEVICE
	docker-bench-se cpuPercent: 0	curity	TRIGGER AOTA MANIFEST UPDATE TRIGGER SOTA
	docker-bench-se cpuPercent: 0	curity	
10.23.221.141:8080/home	docker-bench-se cpuPercent: 0	curity	TRIGGER FOTA TRIGGER CONFIG UPDATE Powered by Thingsboard v.2.4.1

Now on the pop-up window shows up, click the 'Send' button on the box titled **'Shutdown Device'.**





3.6 Decommission Command

In-band manageability provides a mechanism to handle the decommission request over the air. The Decommission command is used to remove all the credentials and then result in a device shutdown.

To trigger Decommission, click the 'Reboot Button' as seen below in the dashboard to trigger a Device Reboot.

🕵 ThingsBoard		rds 🕞 📲 Intel Manageability Device	s Ci 😗 Turtle Creek :
🔒 НОМЕ	Intel Managea	bility	Intel Manageability Devices 👻 🗔 Teja-test 🔇 Realtime - last day 🛓 🛟
<> RULE CHAINS	Teja-test	D	Dynamic Telemetry
22 CUSTOMERS	Product:	NUC8i5BEK	1% - 1°C - 1 bytes
ASSETS	Manufacturer:	Intel(R) Client Systems	1% 1°C 1 bytes
	OS:	Linux teja123 5.0.0-23-generic #24~18.04.1- Ubuntu SMP Mon Jul 29 16:12:28 UTC 2019 x86_64	0% 0/C 05/ee
ENTITY VIEWS	CPU:	Intel(R) Core(TM) i5-8259U CPU @ 2.30GHz	1% -1°C -1 bytes
	RAM:	8223293440 bytes	arg
WIDGETS LIBRARY	BIOS Vendor:	Intel Corp.	Available Memory Core Temperature
DASHBOARDS	BIOS Version:	BECFL357.86A.0051.2018.1015.1513	Disk Usage
🙆 AUDIT LOGS	Date:	2018-10-15 00:00:00	- CPU Usage
	Q Docker	Container Stats	SHUTDOWN DEVICE REBOOT DEVICE DECOMMISSION DEVICE
	docker-bench-se cpuPercent: 0	curity	TRIGGER AOTA MANIFEST UPDATE TRIGGER SOTA
	docker-bench-se cpuPercent: 0	curity	
10.23.221.141:8080/home	docker-bench-se cpuPercent: 0	curity	TRIGGER FOTA TRIGGER CONFIG UPDATE Powered by Thingsboard v.2.4.1

Now on the pop-up window shows up, click the 'Send' button on the box titled **'Decommission Device'.**

Decommission Device		
METHOD	RESULTS	
	SEND	

4.0 Telemetry Data

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

Telemetry can be viewed under DashBoard, 1 for Static Telemetry 2 for Dynamic Telemetry.



4.1 Static Telemetry

This contains the following information

- BIOS-release-date
- BIOS-vendor
- BIOS-version
- CPU-ID

- OS-information
- System-Manufacturer
- System-Product-Name
- Total-physical-memory

Static Telemetry can be viewed in the DashBoard when you maximize the Static Telemetry window.

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 ThingsBoard.

4.3.1 Static Telemetry

To view the device's static telemetry, click the Static Telemetry window.

\leftrightarrow > C \textcircled{a}	
tj-test	•
Product:	NUC6CAYS
Manufacturer:	Intel corporation
OS:	Linux teja-NUC6CAYS 5.3.0-53-generic #47~18.04.1-Ubuntu SMP Thu May 7 13:10:50 UTC 2020 x86_64
CPU:	Intel(R) Celeron(R) CPU J3455 @ 1.50GHz
RAM:	1923637248 bytes
BIOS Vendor:	Intel Corp.
BIOS Version:	AYAPLCEL.86A.0027.2016.1108.1529
BIOS Release Date:	2016-11-08 00:00:00
Disk Information:	[('NAME': 'loop0', 'SIZE': '978944', 'SSD': 'True'), ('NAME': 'loop1', 'SIZE': '93417472', 'SSD': 'True'), ('NAME': 'loop2', 'SIZE': '3825664', 'SSD': 'True'), ('NAME': 'loop3', 'SIZE': '1403200', 'SSD': 'True'), ('NAME': 'loop6', 'SSD': 'True'), ('NAME': 'loop7', 'SIZE': '167931904', 'SSD': 'True'), ('NAME': 'loop6', 'SIZE': '37053040', 'SSD': 'True'), ('NAME': 'loop7', 'SIZE': '57618432', 'SSD': 'True'), ('NAME': 'loop7', 'SIZE': '37053040', 'SSD': 'True'), ('NAME': 'loop7', 'SIZE': '57618432', 'SSD': 'True'), ('NAME': 'loop7', 'SIZE': '37053040', 'SSD': 'True'), ('NAME': 'loop7', 'SIZE': '57618432', 'SSD': 'True'), ('NAME': 'loop7', 'SIZE': '2723280', 'SSD': 'True'), ('NAME': 'loop7', 'SIZE': '57518432', 'SSD': 'True'), ('NAME': 'loop10', 'SIZE': '262544', 'SSD': 'True'), ('NAME': 'loop11', 'SIZE': '255590 0', 'True'), ('NAME': 'loop10', 'SIZE': '65105920', 'SSD': 'True'), ('NAME': 'loop16', 'SIZE': '267980800', 'SSD': 'True'), ('NAME': 'loop16', 'SIZE': '30752636928', 'SSD': 'True'), ('NAME': 'loop16', 'SIZE': '31268536320', 'SSD': 'True')]



4.3.2 Dynamic Telemetry

To view the device's Dynamic telemetry, click the Dynamic Telemetry to see the below



§

5.0 Issues and Troubleshooting

Error Message	Description
COMMAND_FAILURE	Diagnostic agent checks fail to run properly or if diagnostic agent/ config agent is not up when contacted. {'status': 301, 'message': 'COMMAND FAILURE'}
COMMAND_SUCCESS	Post and pre-install check go through. {'status': 200, 'message': 'COMMAND SUCCESS'}
FILE_NOT_FOUND	File to be fetched is not found. {'status': 404, 'message': 'FILE NOT FOUND'}
IMAGE_IMPORT_FAILURE	Image is already present when Image Import is triggered. {'status': 401, 'message': 'FAILED IMAGE IMPORT, IMAGE ALREADY PRESENT'}
INSTALL_FAILURE	Installation was not successful due to invalid package or one of the source file, signature or version checks failed. {'status': 400, 'message': 'FAILED TO INSTALL'}
OTA_FAILURE	Another OTA is in progress when OTA is triggered. {'status': 303, 'message': 'OTA IN PROGRESS, TRY LATER'}
UNABLE_TO_START_DOCKER_COMPOSE	Docker-composed container is not able to be started or spawned etc. {'status': 400, 'message': "Unable to start docker-compose container."}
UNABLE_TO_STOP_DOCKER_COMPOSE	Docker-composed down command was not successful. {'status': 400, 'message': "Unable to stop docker-compose container."}

5.1 OTA Error Status



UNABLE_TO_DOWNLOAD_DOCKER_COMPOSE	Docker-composed downloaded command failed. {'status': 400, 'message': "Unable to download docker-compose container."}
XML_FAILURE	Result of bad formatting, missing mandatory tag. {'status': 300, 'message': 'FAILED TO PARSE/VALIDATE MANIFEST'}

5.2 Provisioning Unsuccessful or Device Not Connected to Cloud

If the provisioning script is struck while creating *symlinks* at the end of provisioning or Device is not connected to the cloud, 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 system-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 following the steps in section 5.3

5.3 Acquiring Debug Messages from Agents

Refer Developer Guide Documentation.

§