AN1144: Using the Device Management Service with Zigbee Gateways

This application note describes how to use a Zigbee gateway application with the Zentri Device Management Service (DMS) to manage, monitor, and update Zigbee end nodes.

This setup requires at least two EFR32MG12 radio boards attached to a wireless starter kit (WSTK) mainboard (SLWSTK6000B). This example relies on a Unix-based host computer acting as a Zigbee gateway. The other EFR32MG12/WSTK devices act as Zigbee end nodes. The Zigbee gateway is provisioned for the DMS so that the gateway itself and all devices can be managed and observed from the cloud. The provided binary images have been generated from the Gecko SDK 2.2.2 Release and the Zigbee 6.2.3 stack.

KEY POINTS

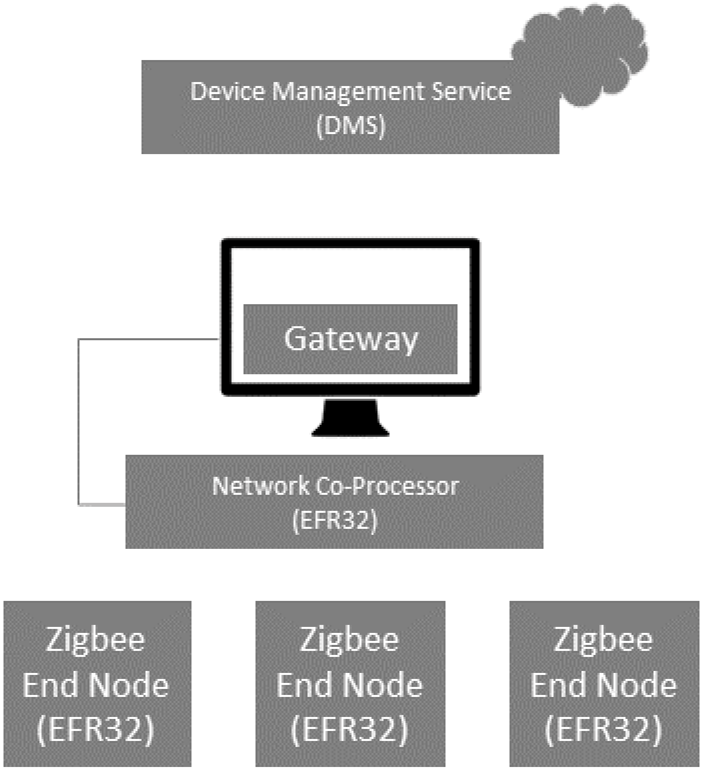
- Flash EFR32MG12 devices as a Zigbee NCP and end devices.
- Register for Device Management Service and provision a new Zigbee gateway device.
- Start and authenticate a Zigbee gateway with the DMS.
- Observe end nodes that join the gateways network and see their health and location characteristics on the DMS.

1. Introduction

In a realistic IoT deployment of Zigbee products three key elements are required for success: the Zigbee end nodes, a gateway, and a management service that assists in the development, deployment and monitoring of the health of the devices at scale. In this app note, we describe how a Zigbee developer can demonstrate a complete IoT system with no new development using a simple vertical stack.

A device management service is a critical element of any successful IoT product development and deployment. It helps design, develop, and manufacture the products at scale. Once the software for the end node or gateway is built, the device management service can distribute this software efficiently. Once the products have been deployed in market, the service can help keep track of the health of the product fleet, proactively fix any issues that occur, and monitor of security of the product fleet.

The Zentri Device Management Service (DMS) provides a range of services for managing devices, products, and product data. In this demonstration a Zigbee gateway will securely connect with the DMS and authorize it for management. Zigbee end nodes that attach to the gateway are reported to the DMS as end nodes belonging to the gateway. Once the DMS becomes aware of the end nodes it will support tracking those nodes.



1.1 Setup Requirements

- Host/Development computer running
  - macOS Sierra or later (x86_64)
  - Ubuntu 18.04 (amd64)

  Note: For Windows users it is recommended to use an Ubuntu 18.04 (amd64) Virtual Machine under Virtual Box.

- Two or more EFR32MG12 radio boards attached to a wireless starter kit (WSTK) mainboard with USB cables
  - These can be purchased as part of the EFR32 Mighty Gecko Wireless Starter Kit (SLWSTK6000B - https://www.silabs.com/products/development-tools/wireless/mesh-networking/mighty-gecko-starter-kit)
- Gateway and end node software located on the Silicon Labs github: https://github.com/SiliconLabs/managed-zigbee-gateway
- An account with the Zentri Device Management Service: https://dms.zentri.com/signup

1.2 Setup Steps

This document will walk the steps required to set up a managed Zigbee gateway with the DMS:

- Where to obtain Zigbee gateway and end node software.
- How to setup and flash devices with Simplicity Commander.
- How to create a DMS user account.
- How to launch a Zigbee gateway and register it with the DMS.
- How to join end-nodes to the Zigbee gateway.
- How to see the end-nodes in the DMS.

Note:

The steps in this application note assume knowledge and experience using Git and a Linux/Unix based environment. If Windows is being used, familiarity with virtualization (VirtualBox) is also helpful.

2. Obtain the Zigbee Gateway and End Node Software

A git repo containing all the binary images needed to run this example can be found on the Silicon Labs github at https://github.com/SiliconLabs/managed-zigbee-gateway. Clone this repo with the following:

```
$ git clone https://github.com/SiliconLabs/managed-zigbee-gateway
```

This repo contains the following items:

- managed-zigbee-gateway/bin/firmware/ncp-EFR32MG12P432F1024GL125
  - Bootloader and Application S37 images for the network co-processor (NCP)
- managed-zigbee-gateway/bin/firmware/ZigbeeEndNode-EFR32MG12P432F1024GL125
  - Bootloader and Application S37/OTA images for the sample Zigbee end node
- managed-zigbee-gateway/bin/gateway/macos
  - Zigbee gateway built for macOS (x86_64)
- managed-zigbee-gateway/bin/gateway/ubuntu
  - Zigbee gateway built for Ubuntu (amd64)

Note: If there are TLS errors seen when cloning the repo make sure to upgrade to the latest version of git.

3. Download Simplicity Commander and Set Up Devices

To set up a Zigbee mesh network with a gateway, devices must first be programmed. One device will act as the network co-processor for the gateway application. Any other devices will act as end nodes. This section describes how to program those images on to the devices.

3.1 Download Simplicity Commander

Simplicity Commander is required to flash the EFR32MG12P/WSTK devices. Simplicity Commander can be downloaded for macOS, Ubuntu and Windows. Download links are provided on this page summarizing programming options: https://www.silabs.com/products/mcu/programming-options. The commands to flash the devices for this example are provided in this document, but more information on Simplicity Commander can be found in *UG162: Simplicity Commander Reference Guide*: https://www.silabs.com/documents/public/user-guides/ug162-simplicity-commander-reference-guide.pdf.

3.2 Running Simplicity Commander at the Command Line

Simplicity Commander runs at the command line and is invoked differently depending on the host OS. Here are examples of how to run Commander after downloading it in macOS, Ubuntu, and Windows:

3.2.1 macOS

Unzip SimplicityCommander-Mac.zip, launch the DMG and drag the Commander application bundle to your desired location. To run Commander at the command line use this command:

```
$ <path/to>/Commander.app/Contents/MacOS/commander <args>
```

3.2.2 Ubuntu

Unzip SimplicityCommander-Linux.zip using the following commands:

```
$ sudo apt-get install unzip libqt5widgets5
$ unzip SimplicityCommander-Linux.zip
$ tar -xjf Commander_linux_<platform>_<version>.tar.bz
```

To run Commander at the command line use this command:

```
$ ./commander/commander <args>
```

3.2.3 Windows

Unzip SimplicityCommander-Windows.zip, then unzip Commander_win32_<version>.zip to your desired location. To run Commander at the command line use this command:

```
C:\> “<path/to>/Simplicity Commander/commander.exe” <args>
```

3.3 Flashing Devices with Simplicity Commander

One device should be flashed as an NCP, and remaining devices can be flashed as end devices using Simplicity Commander at the command line as described above.

Devices can be easily programmed if connected to the computer one at a time. If multiple devices are connected at once a "--serialno" parameter can be used to designate the device to program. Refer to UG162 for more information.

In this case, make sure you only have one device connected when running each programming step below.

3.3.1 Flashing a Connected Device as an NCP

To flash a device as an NCP use the following command:

```
$ <path/to>/commander flash --masserase \
<path/to>/managed-zigbee-gateway/bin/firmware/ncp-EFR32MG12P432F1024GL125/bootloader-uart-xmodem-combined.s37 \
<path/to>/managed-zigbee-gateway/bin/firmware/ncp-EFR32MG12P432F1024GL125/ncp-uart-hw.s37
```

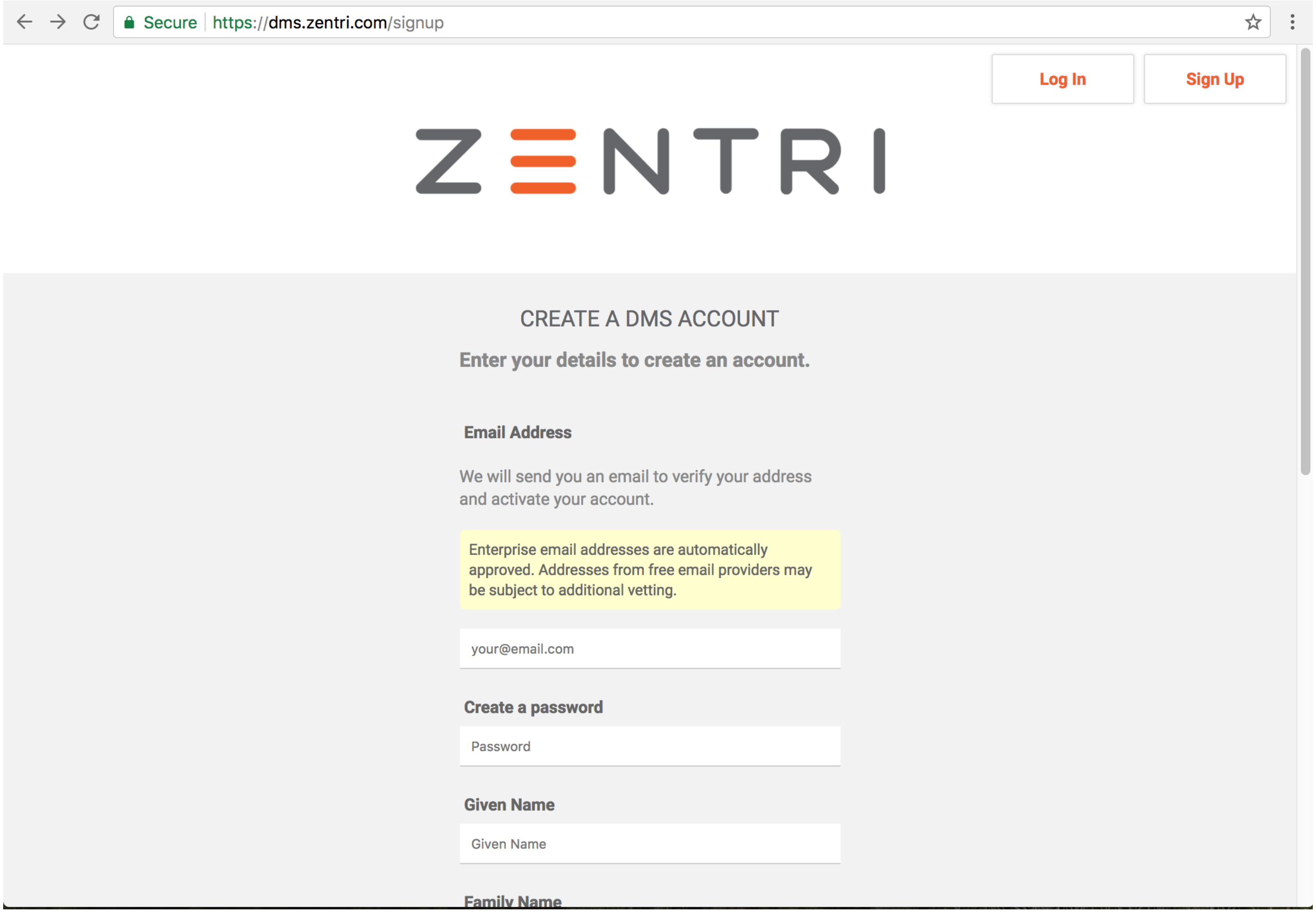
3.3.2 Flashing a Connected Device as an End Node

To flash a device as an end node use the following command:

```
$ <path/to>/commander flash --masserase \
<path/to>/managed-zigbee-gateway/bin/firmware/ZigbeeEndNode-EFR32MG12P432F1024GL125\
/bootloader-storage-internal-combined.s37 \
<path/to>/managed-zigbee-gateway/bin/firmware/ZigbeeEndNode-EFR32MG12P432F1024GL125/ZigbeeEndNode.s37
```

4. Create a DMS User Account and Create a Gateway Device

To create a DMS user account, visit https://dms.zentri.com/signup and follow the prompts:



CREATE A DMS ACCOUNT

Enter your details to create an account.

Email Address

We will send you an email to verify your address and activate your account.

Enterprise email addresses are automatically approved. Addresses from free email providers may be subject to additional vetting.

your@email.com

Create a password

Password

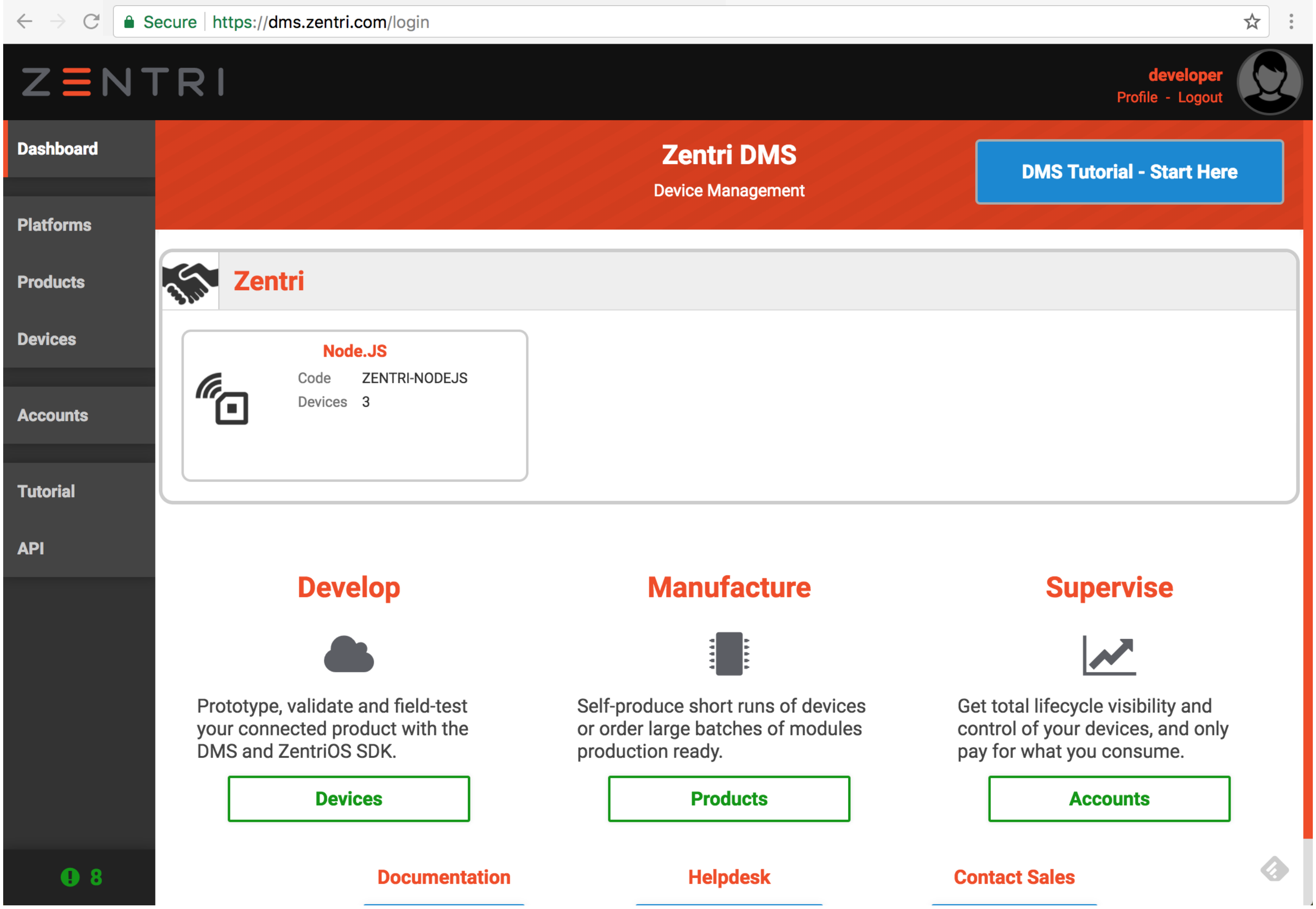
Given Name

Given Name

Family Name

Figure 4.1. DMS Account Sign Up Page

Once an account is created, log in to view the DMS dashboard:



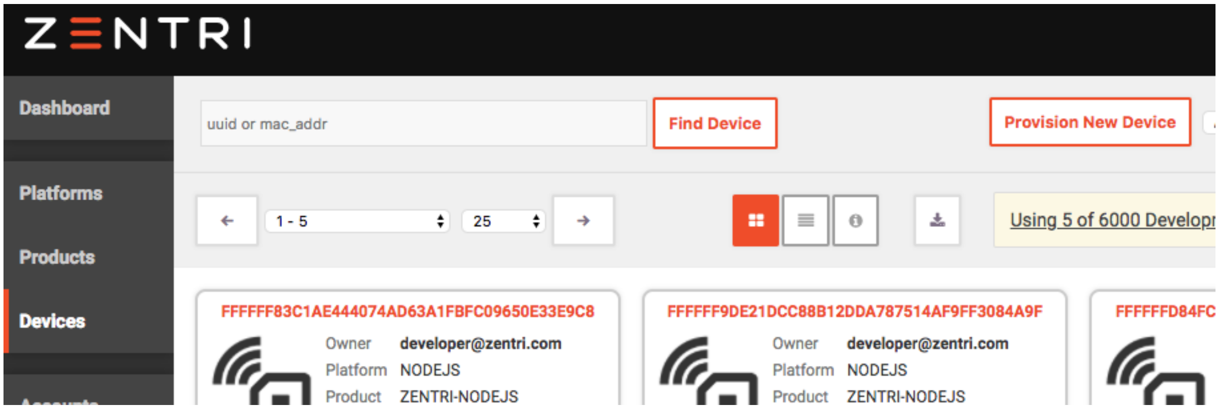
Prototype, validate and field-test your connected product with the DMS and ZentriOS SDK.

Self-produce short runs of devices or order large batches of modules production ready.

Get total lifecycle visibility and control of your devices, and only pay for what you consume.

Figure 4.2. DMS Account Dashboard

To create a gateway device in the DMS, click [**Devices**] on the left tool bar, then click [**Provision New Device**] at the top right:



FFFFFF83C1AE444074AD63A1FBFC09650E33E9C8
Owner developer@zentri.com
Platform NODEJS
Product ZENTRI-NODEJS

FFFFFF9DE21DCC88B12DDA787514AF9FF3084A9F
Owner developer@zentri.com
Platform NODEJS
Product ZENTRI-NODEJS

FFFFFFD84FC

Figure 4.3. Devices View

Select SOFT Platform to designate a software device., then click [**Provision Device**] below:

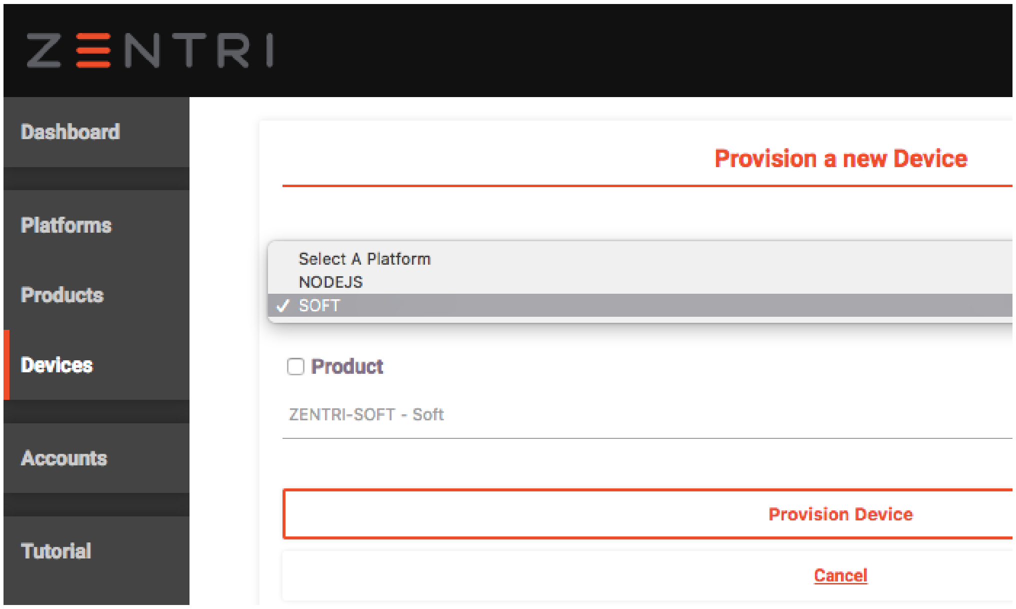
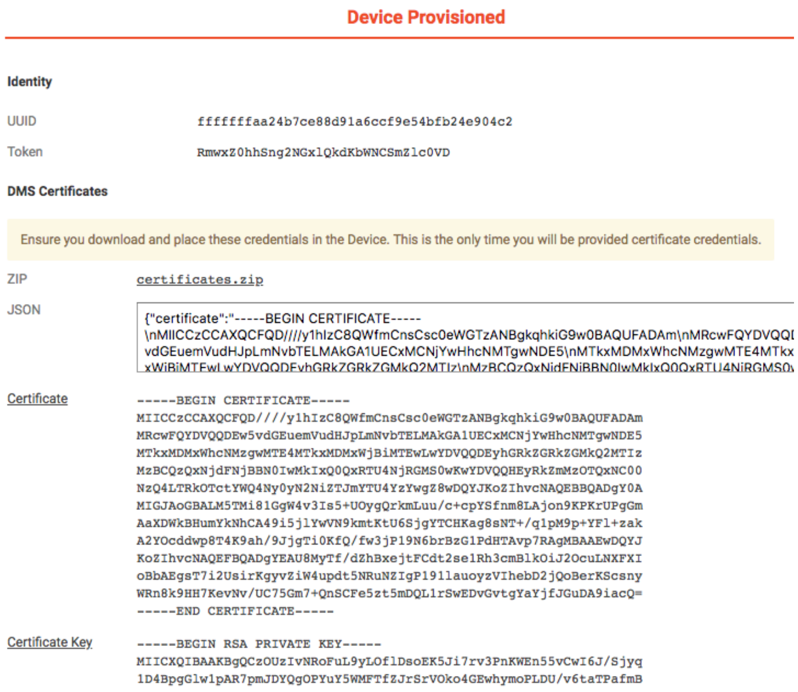


Figure 4.4. Create New Gateway Product

Take note of the UUID and the Token that are created, they will be used later to authenticate the Zigbee gateway, then click [**OK**]:



Device Provisioned

Identity

UUID ffffffaa24b7ce88d91a6ccf9e54bfb24e904c2

Token RmwxZ0hhSng2NGxlQkdKbWNCSmZlc0VD

DMS Certificates

Ensure you download and place these credentials in the Device. This is the only time you will be provided certificate credentials.

ZIP certificates.zip

JSON

```
{"certificate":"-----BEGIN CERTIFICATE-----
\nMIICCzCCAXQCFQD////y1hIzC8QWfmCnsCsc0eWGTzANBgkqhkiG9w0BAQUFADAm\nMRcwFQYDVQQD
vdGEuemVudHJpLmNvbTELMAkGA1UECxMCNjYwHhcNMTgwNDE5\nMTkxMDMxWhcNMzgwMTE4MTkx
xWjBiMTEwLwYDVQQDEyhGRkZGRkZGMkQ2MTIz\nMzBCQzQxNjdFNjBBN0IwMkIxQ0QxRTU4NjRGMS0
```

Certificate

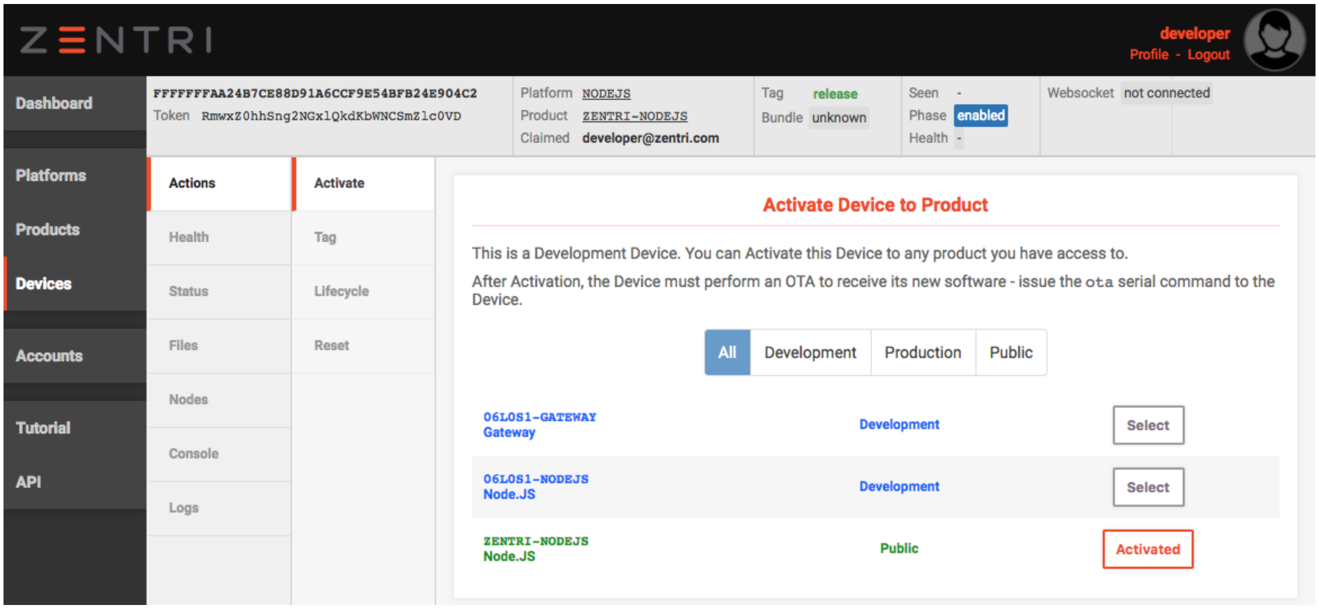
```
-----BEGIN CERTIFICATE-----
MIICCzCCAXQCFQD////y1hIzC8QWfmCnsCsc0eWGTzANBgkqhkiG9w0BAQUFADAm
MRcwFQYDVQQDEw5vdGEuemVudHJpLmNvbTELMAkGA1UECxMCNjYwHhcNMTgwNDE5
MTkxMDMxWhcNMzgwMTE4MTkxMDMxWjBiMTEwLwYDVQQDEyhGRkZGRkZGMkQ2MTIz
MzBCQzQxNjdFNjBBN0IwMkIxQ0QxRTU4NjRGMS0wKwYDVQQHEyRkZmMzOTQxNC00
NzQ4LTRkOTctYWQ4Ny0yN2NiZTJmYTU4YzYwgZ8wDQYJKoZIhvcNAQEBBQADgY0A
MIGJAoGBALM5TMi81GgW4v3Is5+UOygQrkmLuu/c+cpYSfnm8LAjon9KPKrUPgGm
AaXDWkBHumYkNhCA49i5jlYwVN9kmtKtU6SjgYTCHKag8sNT+/q1pM9p+YFl+zak
A2YOcddwp8T4K9ah/9JjgTi0KfQ/fw3jP19N6brBzG1PdHTAvp7RAgMBAAEwDQYJ
KoZIhvcNAQEFBQADgYEAU8MyTf/dZhBxejtFCdt2se1Rh3cmBlkOiJ2OcuLNXFXI
oBbAEgsT7i2UsirKgyvZiW4updt5NRuNZIgP191lauoyzVIhebD2jQoBerKScsny
WRn8k9HH7KevNv/UC75Gm7+QnSCFe5zt5mDQL1rSwEDvGvtgYaYjfJGuDA9iacQ=
-----END CERTIFICATE-----
```

Certificate Key

```
-----BEGIN RSA PRIVATE KEY-----
MIICXQIBAAKBgQCzOUzIvNRoFuL9yLOflDsoEK5Ji7rv3PnKWEn55vCwI6J/Sjyq
1D4BpgGlw1pAR7pmJDYQgOPYuY5WMFTfZJrSrVOko4GEwhymoPLDU/v6taTPafmB
```

Figure 4.5. New Device Credentials

This is the device view of the newly created device. Note that it shows “enabled” in the Phase. Once the Zigbee gateway attaches and authenticates with the DMS, this will change to “active”:



FFFFFFAA24B7CE88D91A6CCF9E54BFB24E904C2
Token RmwxZ0hhSng2NGxlQkdKbWNCSmZlc0VD

Platform NODEJS
Product ZENTRI-NODEJS
Claimed developer@zentri.com

Tag release
Bundle unknown

Seen -
Phase enabled
Health -

Websocket not connected

Activate Device to Product

This is a Development Device. You can Activate this Device to any product you have access to.

After Activation, the Device must perform an OTA to receive its new software - issue the `ota` serial command to the Device.

06LOS1-GATEWAY Gateway	Development	Select
06LOS1-NODEJS Node.JS	Development	Select
ZENTRI-NODEJS Node.JS	Public	Activated

Figure 4.6. Device View

5. Setup a Zigbee Gateway and Register with the DMS

Now that the DMS account is active a gateway device can be provisioned using the key created in the previous step.

Note:

The gateway software is a Unix-based project, and the binary in this example is only supported on macOS (x86_64) and Ubuntu 18.04 (amd64). If your native development machine does not fit these requirements a virtual machine running Ubuntu 18.04 (amd64) is provided for use with Virtual Box that can run on Windows, macOS and most current Linux distributions as a host.

Virtual Box can be downloaded from https://www.virtualbox.org/.

A preloaded version of Ubuntu can be downloaded from https://zigbee-managed-gateway.zentri.com/SiliconLabsZigbeeGatewayDMS.ova.

Use the Virtual Box File->Import Appliance menu item to import the OVA file for usage. The user name and password for this instance is developer/developer. Make sure to do another "git clone" on the repo to have access to the binaries on the VM instance.

5.1 Set Up a Zigbee Gateway

5.1.1 Deploy the Certificates

The certificates.zip downloaded in Setup a Zigbee Gateway and Register with the DMS needs to be unzipped into a folder named "certificates" that sits next to the Z3GatewayHost Zigbee gateway executable:

```
$ unzip <path/to>/certificates.zip -d <path/to>/managed-zigbee-gateway/gateway/<os>/certificates
```

5.1.2 Start the Zigbee Gateway

Start the gateway in the provided software repo using the command:

```
$ cd <path/to>/managed-zigbee-gateway/bin/gateway/<os>
$ ./Z3GatewayHost -p <ncp serial port>
```

Note: The serial port name varies depending on the operating system and other devices present. The command above might need to be run on each device until a successful gateway bringup occurs.

On macOS use this command to list possible choices:

```
$ ls /dev | grep tty.usbmodem
tty.usbmodem1411
tty.usbmodem1421
```

On Ubuntu use this command to list possible choices:

```
$ ls /dev | grep ttyACM
ttyACM0
```

On Ubuntu use this command to allow tty access once on the system before attempting to run the gateway:

```
$ sudo adduser <current user> dialout
```

Additionally, the `Z3GatewayHost` command above, should be prepended with "sudo":

```
$ sudo <path/to>/managed-zigbee-gateway/gateway/<os>/Z3GatewayHost -p <ncp serial port>
```

Once the gateway is up, it will show something similar to the following output:

```
$ ./Z3GatewayHost -p /dev/tty.usbmodem1421
Reset info: 11 (SOFTWARE)
ezsp ver 0x06 stack type 0x02 stack ver. [6.2.0 GA build 147]
Ezsp Config: set source route table size to 0x00FA:Success: set
Ezsp Config: set security level to 0x0005:Success: set
Ezsp Config: set address table size to 0x0002:Success: set
Ezsp Config: set TC addr cache to 0x0002:Success: set
Ezsp Config: set stack profile to 0x0002:Success: set
Ezsp Config: set MAC indirect TX timeout to 0x1E00:Success: set
Ezsp Config: set max hops to 0x001E:Success: set
Ezsp Config: set tx power mode to 0x8000:Success: set
Ezsp Config: set supported networks to 0x0001:Success: set
Ezsp Policy: set binding modify to "allow for valid endpoints & clusters only":Success: set
Ezsp Policy: set message content in msgSent to "return":Success: set
Ezsp Value : set maximum incoming transfer size to 0x00000052:Success: set
Ezsp Value : set maximum outgoing transfer size to 0x00000052:Success: set
Ezsp Config: set binding table size to 0x0010:Success: set
Ezsp Config: set key table size to 0x0000:Success: set
Ezsp Config: set max end device children to 0x0020:Success: set
NCP supports maxing out packet buffers
Ezsp Config: set packet buffers to 255
Ezsp Config: set end device poll timeout to 0x0005:Success: set
Ezsp Config: set end device poll timeout shift to 0x0006:Success: set
Ezsp Config: set zll group addresses to 0x0000:Success: set
Ezsp Config: set zll rssi threshold to 0xFF80:Success: set
Ezsp Config: set transient key timeout to 0x00B4:Success: set
Ezsp Endpoint 1 added, profile 0x0104, in clusters: 8, out clusters 19
Ezsp Endpoint 242 added, profile 0xA1E0, in clusters: 0, out clusters 1
Found 0 files

Z3GatewayHost>
```

This gateway is now ready for interaction.

5.1.3 Form a New Zigbee Network

The first thing to do once the gateway is started is to leave any network that has been already formed using the "network leave" command:

```
Z3GatewayHost> network leave
```

Next, form a new network using the network-creator:

```
Z3GatewayHost> plugin network-creator start 1
NWK Creator: Form: 0x00
Z3GatewayHost>NWK Creator Security: Start: 0x00
NWK Creator: Form. Channel: 15. Status: 0x00
NWK Creator: Stop. Status: 0x00. State: 0x00
EMBER_NETWORK_UP 0x0000
```

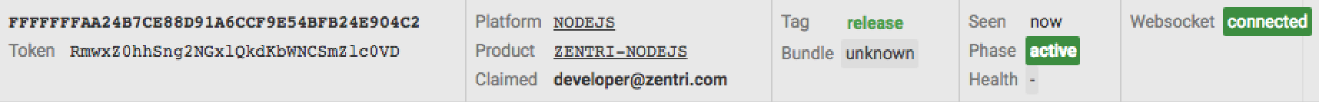
When the "EMBER_NETWORK_UP 0x0000" shows, a new network is formed, and the gateway is ready.

5.1.4 Authenticate the Gateway with the Device Management Service

Now that the gateway is up it can be authenticated to the DMS using the `dms-connect` command and the token acquired from the provisioning step:

```
Z3GatewayHost> custom dms-connect "RmwxZ0hhSng2NGxlQkdKbWNCSmZlc0VD"
Connecting to DMS
```

Now go back to the Device view on the DMS to see that the gateway has moved from “enabled” to “active”:



FFFFFFFAA24B7CE88D91A6CCF9E54BFB24E904C2
Token RmwxZ0hhSng2NGxlQkdKbWNCSmZlc0VD

Platform NODEJS
Product ZENTRI-NODEJS
Claimed developer@zentri.com

Tag release
Bundle unknown

Seen now
Phase active
Health -

Websocket connected

Figure 5.1. Websocket Connected and Device Active

6. Join Zigbee End Nodes to the Gateway for DMS Management

Once the gateway has been provisioned and connected to the DMS, end nodes that join the network will show up in the Nodes section of the Device view.

To join a node the network must be open for joining. To do this use the `plugin network-creator-security` command:

```
Z3GatewayHost> plugin network-creator-security open-network
NWK Creator Security: Open network: 0x00
Z3GatewayHost>Ezsp Policy: set Trust Center Policy to "Allow preconfigured key joins":Success: set
pJoin for 180 sec: 0x00
NWK Creator Security: Open network: 0x00
```

Next press PB0 on the end node to trigger a network leave and rejoin. When the device joins the gateway, something similar to the following output will be displayed:

```
Z3GatewayHost>TC Join Callback 8C6A , decision: 0, status: 1
000B57FFFE491BCD STANDARD_SECURITY_UNSECURED_JOIN
new device line 595
(>)000B57FFFE491BCD8C6A:
new device line 658
8C6A 82 6A 8C CD 1B 49 FE FF 57 0B 00 8C
(>)000B57FFFE491BCD8C6A:
Active Endpoints Response
number of ep: 2
ep: 1
ep: 2
(>)000B57FFFE491BCD8C6A:
Active Endpoints Response
(>)000B57FFFE491BCD8C6A:
Simple Descriptor Response
Device Joined 0x000B57FFFE491BCD
8C6A:
Simple Descriptor Response
Device Joined 0x000B57FFFE491BCD
```

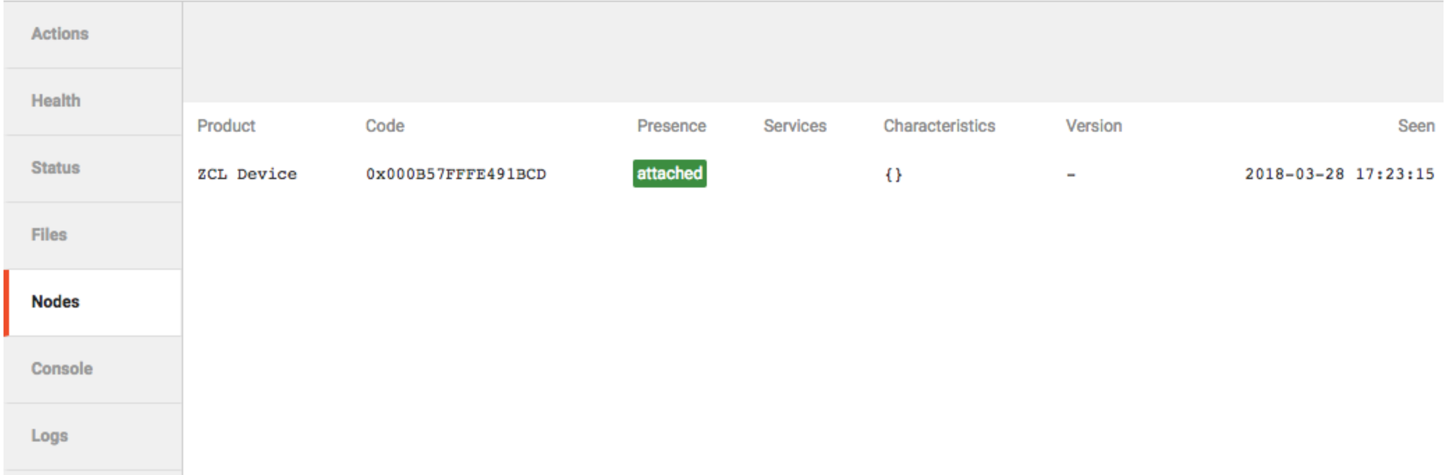
When the gateway gets the join message close the network for joining using the command:

```
Z3GatewayHost> plugin network-creator-security close-network
Ezsp Policy: set Trust Center Policy to "Disallow all joins and rejoins":Success: set
pJoin for 0 sec: 0x00
NWK Creator Security: Close network: 0x00
```

Note: If there are issues joining a device to the gateway, attach the end node to Ethernet and use "telnet <ip> 4901" to attach to its debug console. Use the "network leave" command to reset the network. Then try pressing PB0 again to check the output of the network scan.

7. View Zigbee End Nodes Associated with a Gateway in the DMS

Now go back to the Device view on the DMS and click the Nodes section to see the node has attached.



Product	Code	Presence	Services	Characteristics	Version	Seen
ZCL Device	0x000B57FFFE491BCD	attached		{}	-	2018-03-28 17:23:15

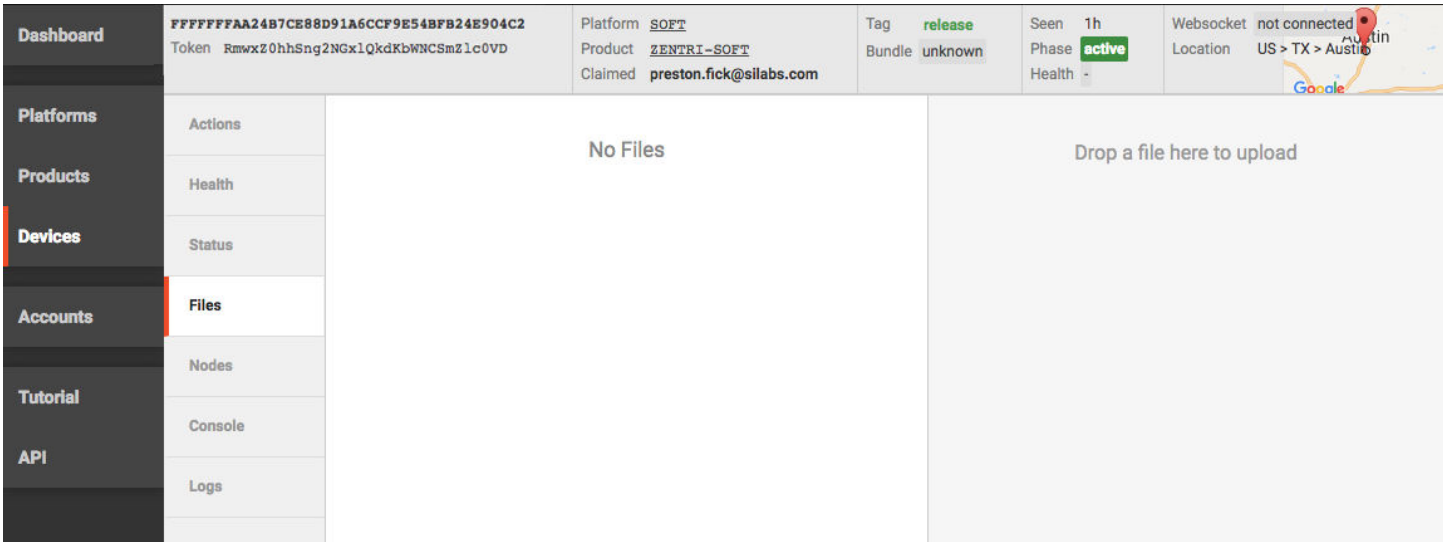
Figure 7.1. Zigbee Device Attached

8. OTA Update Zigbee End Nodes Attached to a Gateway from the DMS

The DMS has a Cloud Filesystem feature, which can be used to host OTA update files for the gateway or device on its network. In this solution an "ota-manifest.json" file describes the available OTA image files. The following steps describe how to use this feature in the DMS, and how to initiate an OTA update for the end node that was just connected to the network.

8.1 Upload ota-manifest.json and OTA Image Files to the DMS

Log in to the DMS and browse back to the devices section to view the device that was created for the gateway. Then go to the Files section as shown in the figure below.



FFFFFFFAA24B7CE88D91A6CCF9E54BFB24E904C2
Token RmwxZ0hhSng2NGxlQkdKbWNCSmZlc0VD

Platform SOFT
Product ZENTRI-SOFT
Claimed preston.fick@silabs.com

Tag release
Bundle unknown

Seen 1h
Phase active
Health -

Websocket not connected
Location US > TX > Austin

No Files

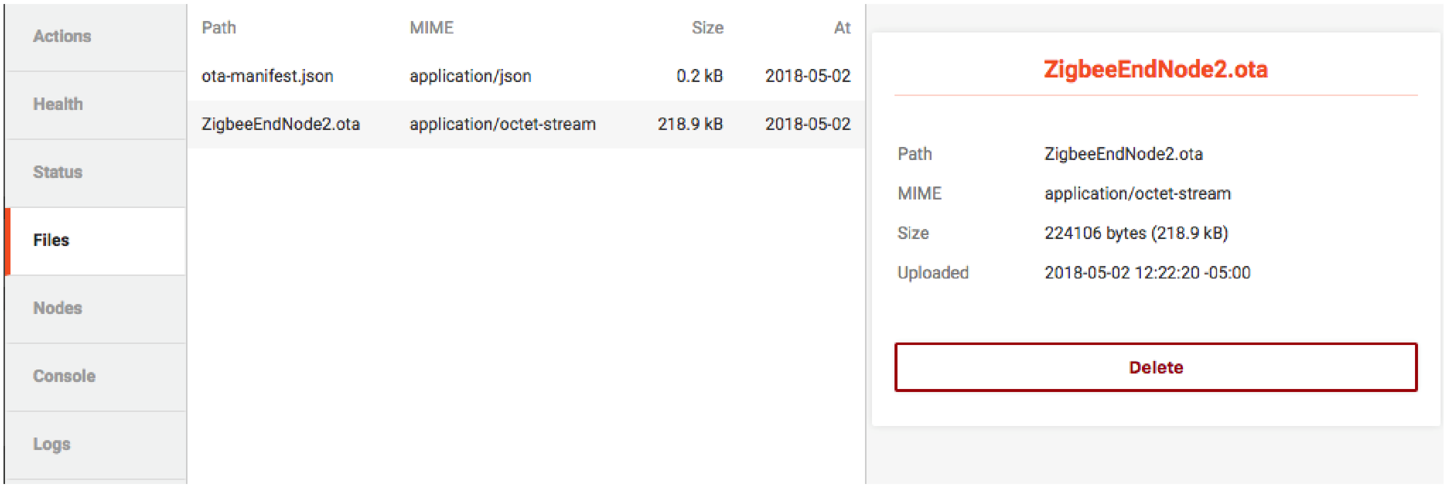
Drop a file here to upload

Figure 8.1. Cloud Filesystem for Gateway Device

Locate the following two files and drag them to the "Drop a file here to upload" section of the DMS shown on the right. The two files are located in the repo at the following locations:

- <path/to>/managed-zigbee-gateway/bin/firmware/ota-manifest.json
- <path/to>/managed-zigbee-gateway/bin/firmware/ZigbeeEndNode-EFR32MG12P432F1024GL125/ZigbeeEndNode2.ota

When the DMS accepts a file, it will present an Upload button; click this to complete the upload. Once both files have been uploaded they will be visible as shown in the figure below:



Path	MIME	Size	At
ota-manifest.json	application/json	0.2 kB	2018-05-02
ZigbeeEndNode2.ota	application/octet-stream	218.9 kB	2018-05-02

ZigbeeEndNode2.ota

Path ZigbeeEndNode2.ota
MIME application/octet-stream
Size 224106 bytes (218.9 kB)
Uploaded 2018-05-02 12:22:20 -05:00

Figure 8.2. New Files Added to the Cloud Filesystem

8.2 Issue an OTA Update for the Zigbee End Node

Once the files have been deployed to the DMS Cloud Filesystem, start up the gateway and connect to the DMS according to 5.1 Set Up a Zigbee Gateway, if it is not started and connected already. Issue the following command to list available OTA update files:

```
Z3GatewayHost>custom dms-ota-list
Manifest last updated: 2018-04-30T17:27:52.950Z
Available images:
ZigbeeEndNode2.ota
```

Then issue the following command to download the OTA update to the gateway:

```
Z3GatewayHost>custom dms-ota-get "ZigbeeEndNode2.ota"
Downloading OTA image ZigbeeEndNode2.ota
Completed write of "./ota-files/ZigbeeEndNode2.ota" with size 224106
Done with OTA image download
```

Now that the OTA file is on the gateway, the end node will find this request to update as part of the standard OTA update behavior. It could take up to 10 minutes to complete. To check the version of the end node, attach it to Ethernet and use "telnet <ip> 4901" to attach to its debug console. The following command can be used to get the current firmware version:

```
ZigbeeEndNode>plugin ota-client info
Client image query info
Manuf ID:         0x1002
Image Type ID:    0x0000
Current Version:  0x00000001
Hardware Version: NA
Query Delay ms:            300000
Server Discovery Delay ms: 600000
Download Delay ms:         0
Run Upgrade Delay ms:      600000
Verify Delay ms:           10
Download Error Threshold:  10
Upgrade Wait Threshold:    10
```

In the case above, the device has not been updated. The following command can be used on the device to force an OTA update:

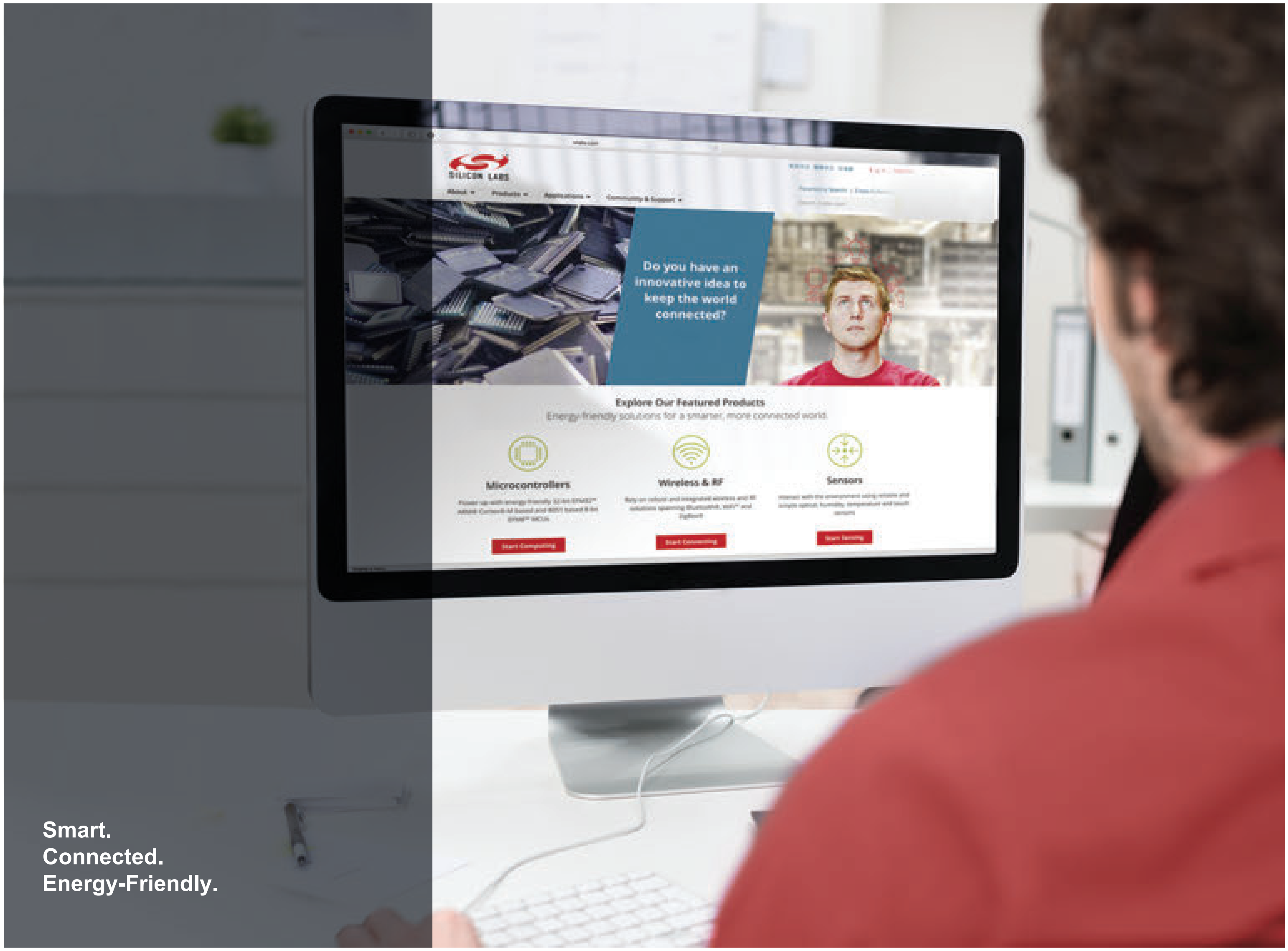
```
ZigbeeEndNode>plugin ota-client start
starting OTA client state machine
Bootload state: Discovering OTA Server
ZigbeeEndNode>Processing message: len=6 profile=0000 cluster=8006
Setting OTA Server to 0x0000
Bootload state: Get OTA Server EUI
OTA Cluster: setting IEEE address of OTA cluster
Last offset downloaded: 0x0003703E
Found fully downloaded file in storage (version 0x00000002).
Found file in storage with different version (0x00000002) than current version (0x00000001)
Last offset downloaded: 0x0003703E
No signature verification support, assuming image is okay.
Starting EBL verification
EBL passed verification.
Custom verification passed: 0x00
Bootload state: Waiting for Upgrade message
Sending Upgrade End request.
Processing message: len=19 profile=0104 cluster=0019

T00000000:RX len 19, ep 01, clus 0x0019 (Over the Air Bootloading) FC 19 seq 00 cmd 07 payload[02 10 00 00 02 0
0 00 00 00 00 00 00 00 00 00 00 ]
OTA Cluster: wait for 0 s
RXed timeOut 0x00000000 s, MAX timeOut 0x00000DBB s
Adding 3000 ms. delay for immediate upgrade.
Countdown to upgrade: 3000 ms
Bootload state: Countdown to Upgrade
Applying upgrade
Executing bootload callback.
```

The device will reset after the bootload, then the following command can be issued to verify the update occurred:

```
Z3SwitchSoc>plugin ota-client info
Client image query info
Manuf ID:         0x1002
Image Type ID:    0x0000
Current Version:  0x00000002
Hardware Version: NA
Query Delay ms:            300000
Server Discovery Delay ms: 600000
Download Delay ms:         0
Run Upgrade Delay ms:      600000
Verify Delay ms:           10
Download Error Threshold:  10
Upgrade Wait Threshold:    10
```

The update was successful. The device was updated from version 0x01 to 0x02. More information on OTA updates for Zigbee can be found in *AN728: Over-the-Air Bootload Server and Client Setup*.


SILICON LABS
Do you have an innovative idea to keep the world connected?
Explore Our Featured Products
Energy-friendly solutions for a smarter, more connected world.
Microcontrollers
Wireless & RF
Sensors

Smart.
Connected.
Energy-Friendly.

Products
www.silabs.com/products

Quality
www.silabs.com/quality

Support and Community
community.silabs.com

Disclaimer
Silicon Labs intends to provide customers with the latest, accurate, and in-depth documentation of all peripherals and modules available for system and software implementers using or intending to use the Silicon Labs products. Characterization data, available modules and peripherals, memory sizes and memory addresses refer to each specific device, and "Typical" parameters provided can and do vary in different applications. Application examples described herein are for illustrative purposes only. Silicon Labs reserves the right to make changes without further notice and limitation to product information, specifications, and descriptions herein, and does not give warranties as to the accuracy or completeness of the included information. Silicon Labs shall have no liability for the consequences of use of the information supplied herein. This document does not imply or express copyright licenses granted hereunder to design or fabricate any integrated circuits. The products are not designed or authorized to be used within any Life Support System without the specific written consent of Silicon Labs. A "Life Support System" is any product or system intended to support or sustain life and/or health, which, if it fails, can be reasonably expected to result in significant personal injury or death. Silicon Labs products are not designed or authorized for military applications. Silicon Labs products shall under no circumstances be used in weapons of mass destruction including (but not limited to) nuclear, biological or chemical weapons, or missiles capable of delivering such weapons.

Trademark Information

SILICON LABS

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

http://www.silabs.com