



SAMSUNG
ARTIK™ Modules

Samsung Training Lab
ARTIK Cloud and 055s

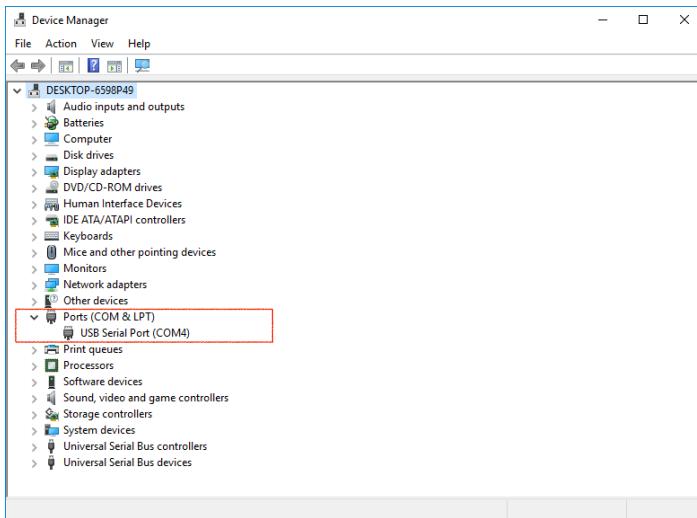
TABLE OF CONTENTS

Introduction	2
Hardware set up	3
Lab 1: Create a Device Type	4
Lab 2: Onboard an ARTIK 055s	8
2.1 Install ARTIK IDE/SDK	8
2.2 Build an Application for ARTIK055s	8
2.3 Prepare ARTIK 055s Board for Onboarding	11
2.4 Samsung ARTIK Onboarding Application	13
Lab 3: Steam Telemetry Data and Integrate Orchestration Engine	19
3.1 Extend Onboarding Application to Stream Telemetry Data	19
3.2 Define Rules Engine	23
3.3 Use Rules Engine to Trigger Device Actions	24
Lab 4: OTA solution	27
4.1 Enable Device Management and OTA Capabilities	27
4.2 Prepare an Update Package	29
4.3 Upload OTA Image to ARTIK Cloud Service	31
4.4 Initiate OTA Update	32
Lab 5: ARTIK SEE APIs	35
5.1 Extend existing app to include multiple TASH shell commands	35
5.2 Key Management APIs	35
5.3 Secure Storage APIs	38
Appendix A – Create your own Device Type	39
Appendix B – ARTIK Cloud Direct Onboarding Flow	40

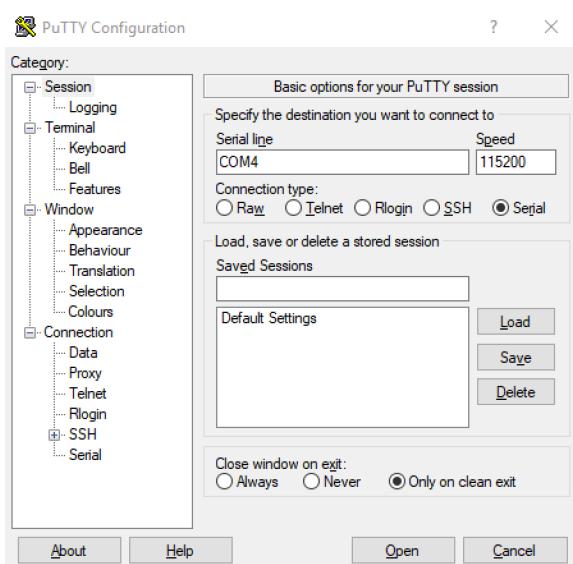
Introduction

This hands-on lab is intended to explain the process of onboarding an ARTIK 055s powered device. We will explain how to connect the ARTIK 055s to the ARTIK cloud service using ARTIK SDK example code and the ARTIK on-boarding mobile app. Once the ARTIK 055s is on-boarded to the ARTIK cloud service, the user can turn on/off LEDs on the ARTIK 055s board using ARTIK Cloud Orchestration Engine.

1. Connect ARTIK 055s development board to your host machine using an USB cable.
2. Find your serial device in your Device Manager / Ports and remember the COM port your ARTIK board is using. In our example, the board uses COM4 port.



3. On Windows host machine, please install and launch a serial console application PuTTY.. Select “Serial” as the Connection type, use the COM port number you find from Device Manager and Speed 115200 for serial access.



For Mac users, you can use the built-in `screen` utility to access the serial console. Normally the device name ends with 141B or 142B.

```
screen /dev/cu.usbserial-XXXXXXX 115200
```

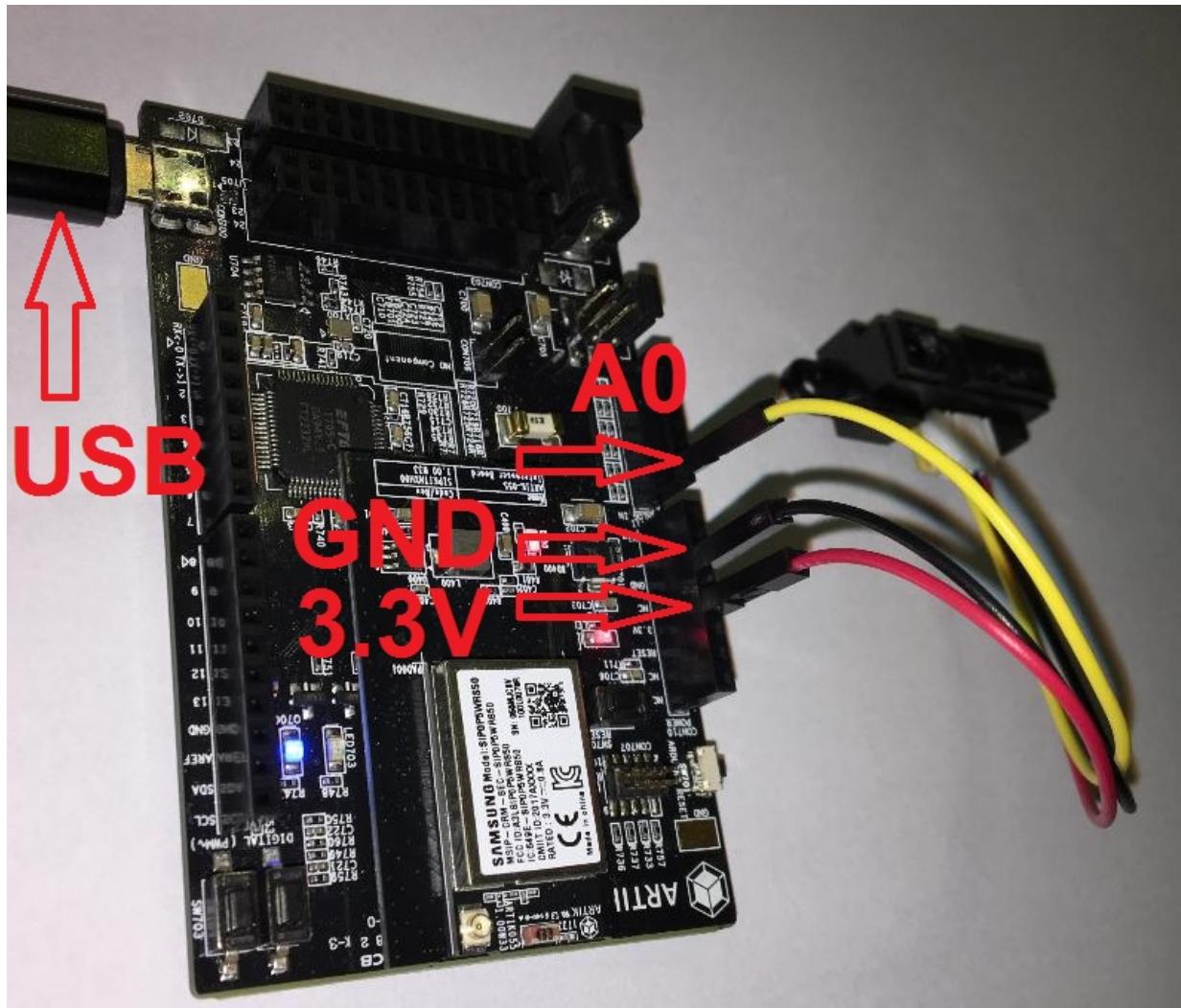
Hardware set up

1. Connect the USB cable to ARTIK 055S board and your host machine.
2. Connect the IR distance sensor as shown below:

Yellow wire: A0

Black wire: GND

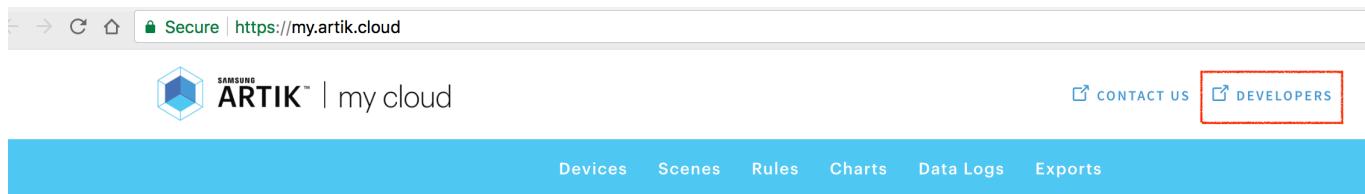
Red wire: 3.3V



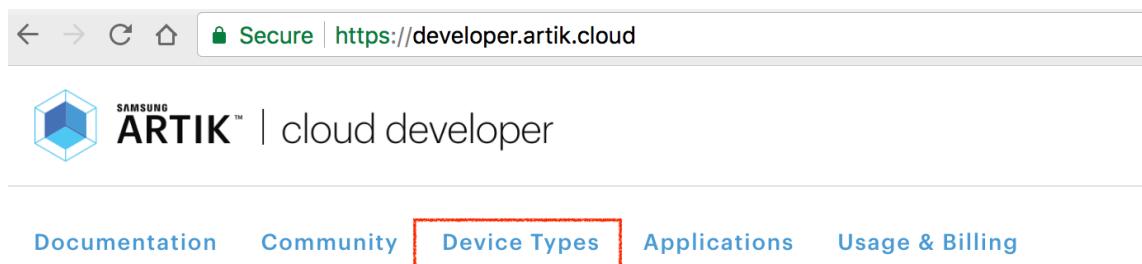
Lab 1: Create a Device Type

In this lab, we are going to create a device type in ARTIK Cloud. Once you have the device type, we will show you how to connect to this device type from your ARTIK devices in the following labs.

1. Open the Developer portal of ARTIK Cloud by using this URL <https://developer.artik.cloud/> or by clicking on “ARTIK CLOUD DEVELOPERS” from ARTIK Cloud user portal.



2. Click on Device Types



3. Click on “+NEW” button



4. Give your device type a Device Display Name such as “your_company_lab_device” (eg. ARTIK lab device) and give it an unique name such as com.your_company.your_project_device (eg. com.samsung.artik055) , then click on “CREATE DEVICE TYPE”. This will enable us to create a digital representation for our device type in ARTIK Cloud. (compare this to reserving a name space for a data type in programming languages)

DEVICE DISPLAY NAME	48
ARTIK lab device	
UNIQUE NAME	235
com.samsung.artik055	
CREATE DEVICE TYPE	CANCEL

- Now a name space is reserved for the device type. Actual definition of the device type needs to be done further. This is done using a **Manifest**. A Manifest is the representation of a device's data properties in ARTIK cloud. Any data, a device produces and sends to ARTIK cloud are represented as "Device Fields". Any data, the device can respond to is represented as "Actions". An action can have parameters as well, like an argument.

A Manifest can be created step-by-step using an intuitive user interface or by simply uploading from a JSON file. The steps below explain the creating a Manifest from a JSON file.

To create a manifest from a JSON file, select UPLOAD A MANIFEST.

The screenshot shows the ARTIK lab device configuration page. At the top, there is a blue icon of a device with a Wi-Fi signal, followed by the text "ARTIK lab device". Below this are two buttons: "EDIT INFO" and "... VIEW MORE DETAILS". Underneath the device icon, it says "DATA SOURCE Directly from device" with an "EDIT" link. To the right, it says "SECURE DEVICE REGISTRATION Not Enabled" with an "EDIT" link. The main content area has a section titled "Create Device Type" with a checked checkbox. Below it is another section titled "Create a Manifest" with a checked checkbox. A descriptive text explains that ARTIK cloud services is designed to communicate with any connected device. It encourages users to create a manifest today. There are two buttons at the bottom: "+ NEW MANIFEST" and "UPLOAD A MANIFEST", with the latter being highlighted with a red border. To the right of the "Create a Manifest" section, there is a "LEARN ABOUT MANIFESTS" section with links to "Dive into details" and "Follow a step by step guide".

- Upload the provided manifest file (com.samsung.artik055.json), for reference the contents of the manifest file is available in the appendix A

The screenshot shows a modal dialog box titled "Upload a new manifest". The text inside says "Create a manifest by uploading a JSON specification containing fields and actions." Below this is a large input field with the placeholder "Drag and drop JSON manifest here". At the bottom of the input field is a blue button labeled "OR BROWSE & UPLOAD...".

- Upon uploading the JSON file, make sure the Manifest is as shown below. This Manifest represents a device that can send data to ARTIK Cloud for LED state, distance sensor reading and a timestamp. And it can also receive "setOn" and "setOff" actions from ARTIK Cloud in order to toggle the LED on the board.

 ARTIK lab device

Simple Manifest

The active manifest describes the capabilities of your device type to other users and devices on the ARTIK cloud services platform. Use fields and actions to describe the data that this device type produces and accepts. [LEARN MORE »](#)

Device Fields

Describe fields for each piece of data produced by this device.

state	Boolean	
sensor	Integer	
timestamp	Double	SI.MILLI(SI.SECOND)

Device Actions

Describe actions that this device is capable of receiving.

setOn	Action
setOff	Action

Activate Manifest

Publish this device manifest on the ARTIK cloud services platform.

Your manifest is ready to be activated and does not require approval before going live. Activating this manifest will replace the current manifest. The device type will stay private.

Fields
Actions

ACTIVATE MANIFEST
CANCEL

6. Then activate the manifest, by clicking on the “ACTIVATE MANIFEST” button.
7. Make sure a device type is created. Access “VIEW MORE DETAILS” and copy the device type ID created

Documentation Community Device Types Applications Usage & Billing SEARCH 

DEVICE TYPES © NEW

Overview

ARTIK lab device

Set Up

Device Management

Monetization

Errors

 ARTIK lab device

DATA SOURCE
Directly from device [EDIT](#)

SECURE DEVICE REGISTRATION
Not Enabled [EDIT](#)

Private Device Type
Visible only to users that are part of your organization [CHANGE VISIBILITY](#)

[EDIT INFO](#) ... VIEW MORE DETAILS

8. Make a note of the device type ID created. The Device type ID is dt_____ <please fill in>

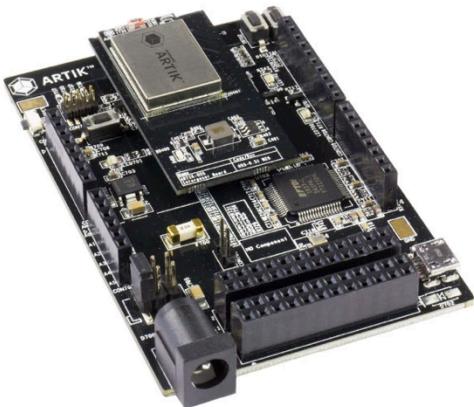
Device Type Details

UNIQUE NAME
com.artik.lab3_053_ota

DEVICE TYPE ID
dtad860bc01c794f2396d5c3e765f0009f

9. Congratulations you have successful represented your device in ARTIK Cloud.

Lab 2: Onboard an ARTIK 055s

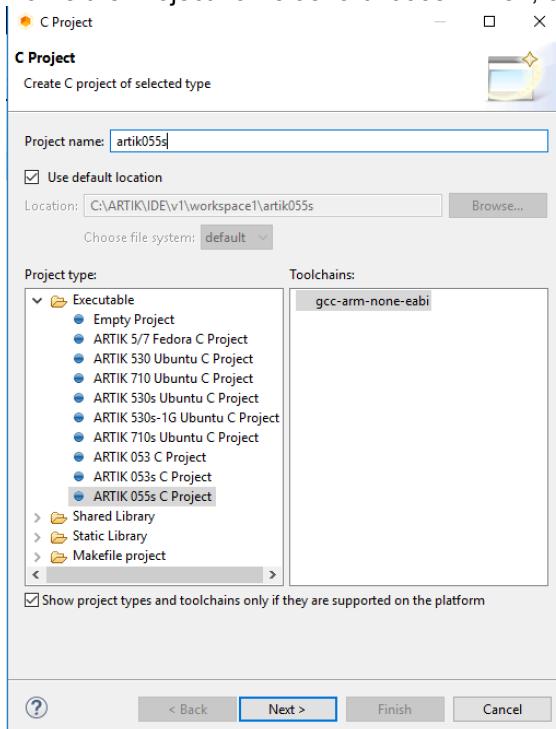


2.1 Install ARTIK IDE/SDK

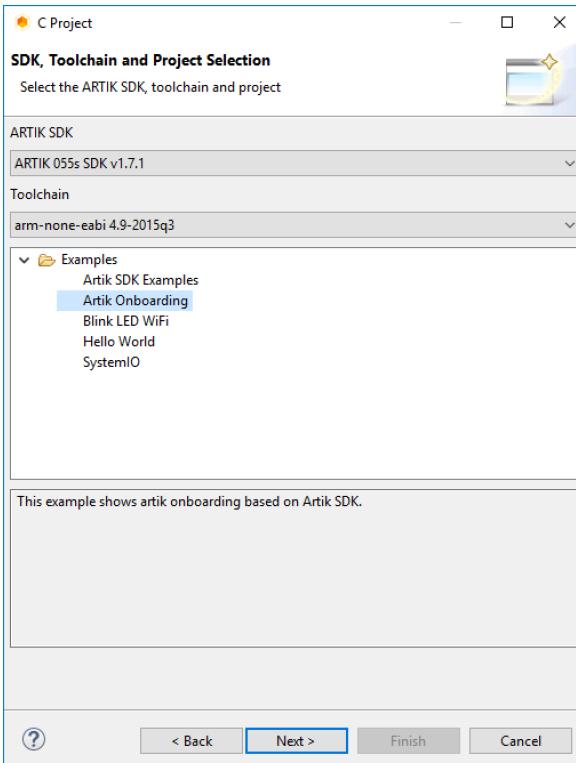
Please follow the instructions at <https://developer.artik.io/documentation/developer-guide/artik-ide/install-ide.html> to set up your development environment.

2.2 Build an Application for ARTIK055s

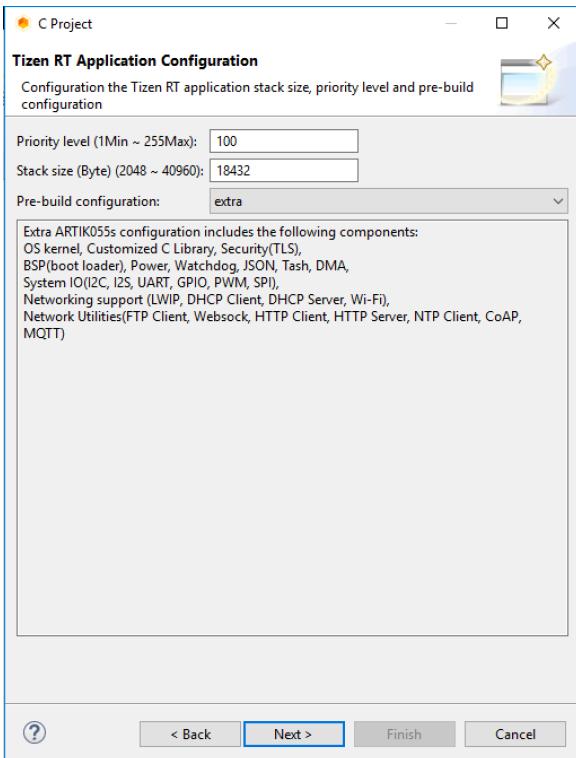
1. Launch your ARTIK IDE, go to File->New->C Project.
2. In the “C Project” dialog, select “ARTIK 055s C project”, and choose “gcc-arm-none-eabi” as the default toolchain. Name the Project name as “artik055s”. Then, click Next.



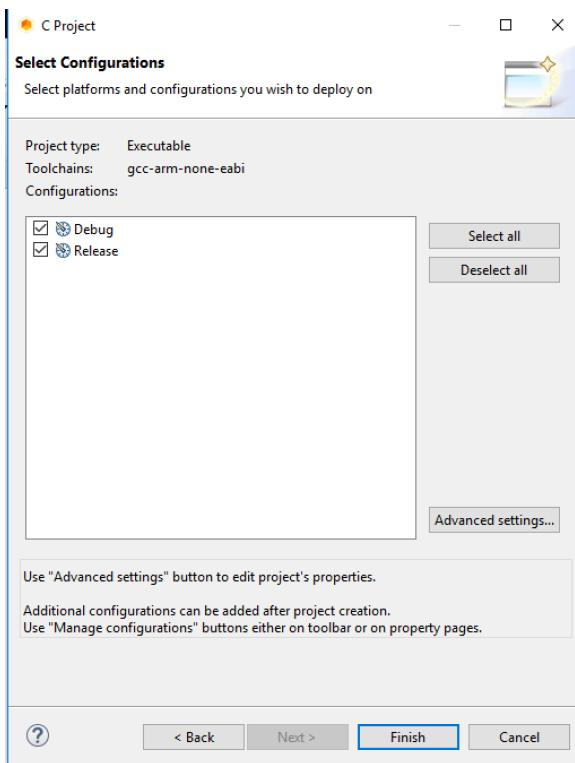
3. In “SDK, Toolchain and Project Selection” dialog, select “ARTIK 055s SDK v1.7.1” as the target SDK version. Click on “Artik Onboarding” example, we will use this example as our template to generate onboarding application. Then, click Next.



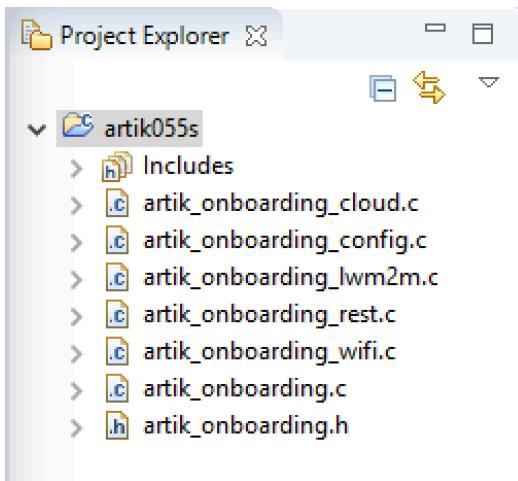
4. In “Tizen RT Application Configuration” dialog, use the default Priority level and Stack Size settings. Change the Pre-build configuration to “extra”, then click Next.



5. In “Select Configurations” dialog, keep the default Debug and Release builds settings, and click Finish.



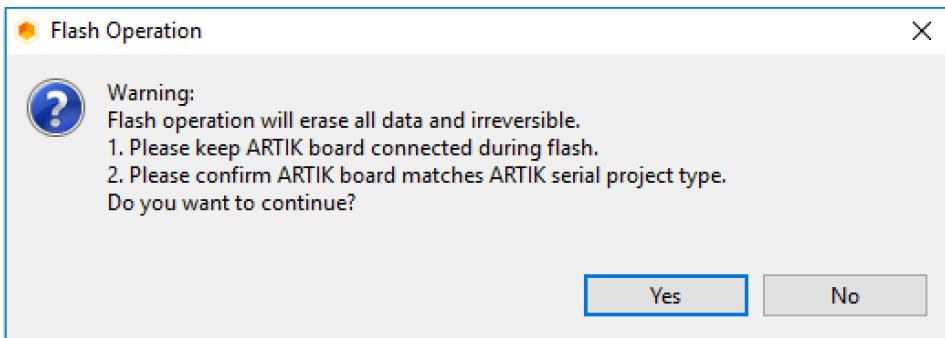
- Now you have an application artik055s created in your IDE Project Explorer.



- Build the application by selecting Project->Build Project. You can see the build log from the IDE Console tab.
- Once you have a successful build, connect your ARTIK 055s board to your laptop by using an USB cable, click the

"Flash System Image" button  to flash the newly built image to the board.

As a precaution, a warning dialog will pop up to confirm you want to flash the device and make sure your ARTIK board matches ARTIK serial project type. Click "Yes" to proceed.



9. From your IDE console, you can view the flash progress.

A screenshot of the Eclipse IDE's 'Console' tab titled 'Flash System Image Console'. The output window shows the following log entries:

```
[2018/02/03 10:20:05:095]target halted in ARM state due to debug-request, current mode: supervisor
[2018/02/03 16:20:05:693]cpsr: 0x0000001d3 pc: 0x04023a14
[2018/02/03 16:20:05:693]D-Cache: disabled, I-Cache: enabled
[2018/02/03 16:20:06:021]Flashing C:/ARTIK/IDE/v1/workspace1/artik055s/Debug/tinyara_head.bin-signed to '...
[2018/02/03 16:20:06:396]target halted in ARM state due to debug-request, current mode: Supervisor
[2018/02/03 16:20:06:396]cpsr: 0x8000001d3 pc: 0x040239f8
[2018/02/03 16:20:06:396]D-Cache: disabled, I-Cache: enabled
[2018/02/03 16:20:41:380]1208832 bytes written at address 0x040c8000
[2018/02/03 16:20:41:380]downloaded 1208832 bytes in 21.093744s (55.964 KiB/s)
Flash operation successful!
```

2.3 Prepare ARTIK 055s Board for Onboarding

1. Push the Reset button.



- From your serial console, you should be able to see the message below when the board boots up.

```

U-Boot 2017.01-00013-g814fa7d (Jan 08 2018 - 15:17:26 +0900)

CPU: Exynos200 @ 320 MHz
Model: ARTIK-05x based on Exynos T20
DRAM: 1.3 MiB
WARNING: Caches not enabled
BL1 released at 2017-3-13 15:00
SSS released at 2017-09-12
WLAN released at 2017-12-21
Flash: 8 MiB
*** Warning - bad CRC, using default environment

In:    serial@80180000
Out:   serial@80180000
Err:   serial@80180000
Autoboot in 50 milliseconds
gpio: pin gpg16 (gpio 46) value is 1
## Starting application at 0x040C8020 ...
s5j_sflash_init: FLASH Quad Enabled
i2c_uioregister: Registering /dev/i2c-0
i2c_uioregister: Registering /dev/i2c-1
System Information:
    Version: 1.0
    Commit Hash: 412a99dfeef802237d9463b9fb292a0f1fcacda6
    Build User: ARTIK@Samsung
    Build Time: 2018-01-16 20:24:06
    System Time: 01 Jan 2010, 00:00:00 [s] UTC Hardware RTC
Support
TASH>>Onboarding service version 1.7.1
Starting supplicant in foreground...
Starting AP ARTIK_286d974094ec
Web server started
ARTIK Onboarding Service started

```

- Press <Enter> to enter TASH shell, we will manually configure some onboarding parameters.

- From the ARTIK 055s TASH shell, reset the onboarding configuration. The device_type_id is associated with a default device type.

```

TASH>>onboard reset
Onboarding configuration was reset.
Reboot the board to return to onboarding mode

TASH>>onboard config

Wifi:
    ssid:
    passphrase:
    secure: false
    NTP server: 0.pool.ntp.org
Cloud:
    device_id: null
    device_token: null
    device_type_id: dt2d93bdb9c8fa446eb4a35544e66150f7
Lwm2m:
    is_ota_update: 0

```



5. Register ARTIK 055s to the device type you created in Lab 1. Replace the device id by using the device type id you saved. Reboot the board.

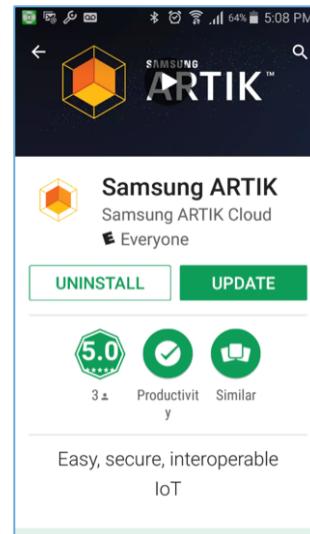
```
TASH>>onboard dtid dt36cebd13947147b5a5381e2310ce8264  
TASH>>reboot
```

6. Note that the ARTIK onboarding service has started.

```
TASH>>Onboarding service version 1.7.1  
Starting supplicant in foreground...  
Starting AP ARTIK_286d97409408  
Web server started  
ARTIK Onboarding Service started
```

2.4 Samsung ARTIK Onboarding Application

1. Download Samsung ARTIK App to your mobile device:

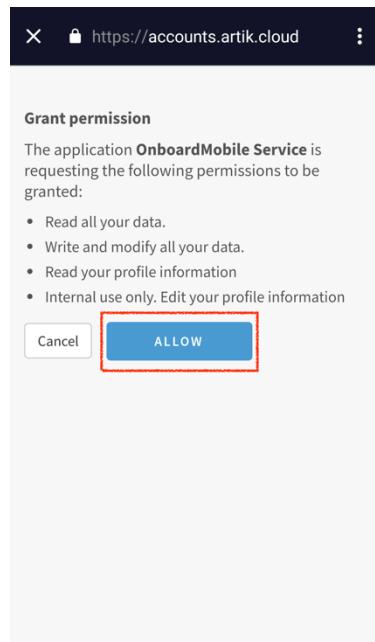


- For Android phones, the app can be downloaded from Google Play.
- For iPhone, the app can be found at the iTunes App Store (search for Samsung ARTIK). The app logo



looks like:

2. Launch ARTIK app on your mobile phone, sign in with your own account and grant permission to the onboarding app.



3. Click the “+” button to add a device.

No registered ARTIK module found

+

Never Connected

Mode Selection

Use QR code ➞

Manual input ➞

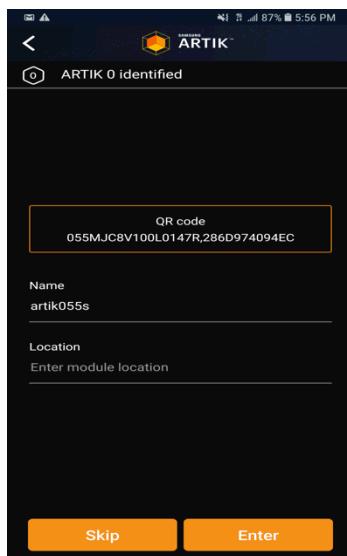
CANCEL

3. Press the arrows next to Use QR Code.

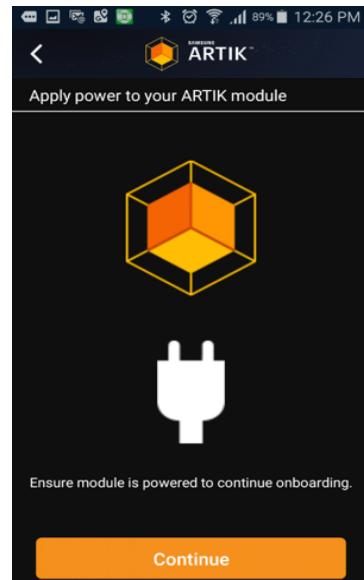


4. Scan the QR code.

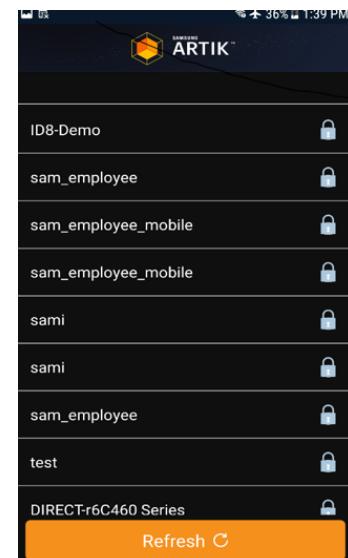
5. Enter a device name in the Name field and press Enter.



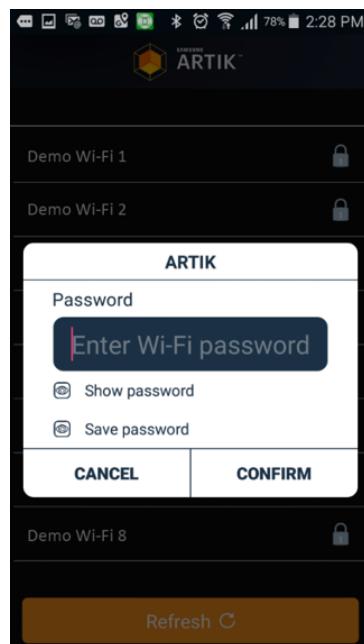
6. Press Continue



7. Select the WiFi SSID that you want your ARTIK 055s to connect to.



8.

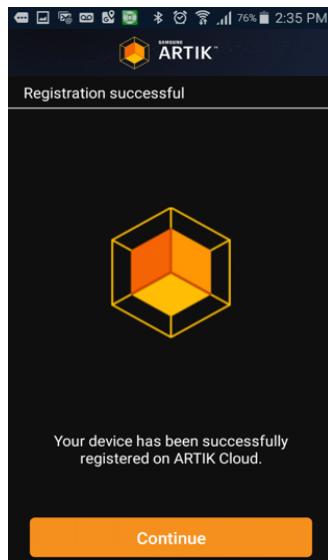


9. Enter the password and press Confirm.

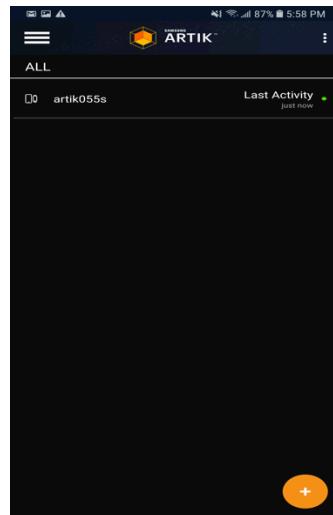
10. Now, from your ARTIK055s serial console, you should be able to see messages below:

```
TASH>>Onboarding service version 1.7.1
Starting supplicant in foreground...
Starting AP ARTIK_286d974094ec
Web server started
ARTIK Onboarding Service started
New Station connected
1 station connected
New Station connected
1 station connected
GET /v1.0/wifi/accesspoints
POST /v1.0/wifi/config
Web server stopped
Start station connection
Starting supplicant in foreground...
Connected to Access Point
Start DHCP client
Start NTP client
Start webserver cloud API
Web server started
Start MDNS service
GET /v1.0/artikcloud
POST /v1.0/artikcloud
Web server stopped
Start websocket to ARTIK Cloud
WebSocket successfully connected
ARTIK Cloud connection started
Start LWM2M connection to ARTIK Cloud
LWM2M connection started
```

11. After registration is successful, click Continue.



12. New device shows up.



The ARTIK 055s has been successfully on-boarded to the Cloud. From your ARTIK Cloud portal, you can see the newly onboarded ARTIK 055s device.

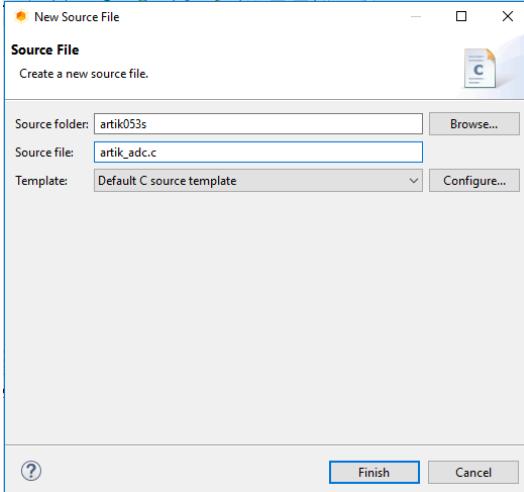
A screenshot of the ARTIK Cloud portal on a desktop browser. The top navigation bar includes the ARTIK logo, "my cloud", "CONTACT US", "DEVELOPERS", and a user profile. Below the navigation is a blue header with tabs: "Devices", "Rules", "Charts", "Data Logs", and "Exports". The main content area is titled "Devices". On the left, there are filters: "All", "Shared with me", and "My Devices". The main panel shows a list of devices with a "Devices" heading and a "Devices" icon. A button "+ Add Another Device..." is visible. To the right, a detailed view of the "artik055s" device is shown, including its name, ID "artik053", and status "Online". There are also icons for edit, delete, and more options.

Lab 3: Steam Telemetry Data and Integrate Orchestration Engine

In this lab, we will extend the application from lab 1. Add a new thread to collect sensor data, and stream sensor data to ARTIK cloud service. We will also use ARTIK cloud service orchestration engine to turn on or off LED onboard based on the streamed sensor reading.

3.1 Extend Onboarding Application to Stream Telemetry Data

1. From your ARTIK IDE, go to File->New->Source File, enter artik_adc.c as the source file name, and click "Finish".



2. Here is what you need to include in your artik_adc.c. In this file, we only have one function defined, which reads distance sensor data from the analog pin A0.

```

#include <stdio.h>
#include <string.h>

#include <artik_module.h>
#include <artik_error.h>
#include <artik_adc.h>

#include "artik_onboarding.h"

int ReadSensor(int pin)
{
    artik_adc_module *adc = NULL;
    artik_adc_handle handle = NULL;
    artik_adc_config config;
    artik_error err = S_OK;
    int val = 0;
    char name[16] = "";
    int ret = 0;

    adc = (artik_adc_module *)artik_request_api_module("adc");
    if (!adc) {
        fprintf(stderr, "ADC module is not available\n");
        return -1;
    }

    memset(&config, 0, sizeof(config));
    config.pin_num = pin;
    snprintf(name, 16, "adc%d", config.pin_num);
    config.name = name;

    err = adc->request(&handle, &config);
    if (err != S_OK) {
        fprintf(stderr, "Failed to request ADC %d (%d)\n", config.pin_num, err);
        ret = -1;
        goto exit;
    }

    err = adc->get_value(handle, &val);
    if (err != S_OK) {
        fprintf(stderr, "Failed to read ADC %d value (%d)\n", config.pin_num, err);
        adc->release(handle);
        ret = -1;
        goto exit;
    }

    fprintf(stdout, "ADC%d=%d\n", config.pin_num, val);
    adc->release(handle);

exit:
    artik_release_api_module(adc);
    return val;
}

```

3. Now, we will extend the onboarding flow to stream the sensor data. Go to `artik_onboarding_cloud.c`, add the code below before function `artik_error StartCloudWebsocket(bool start)`.



In these functions, we will start a new thread to collect sensor data at 10 seconds interval, construct the message into JSON format and post the data to ARTIK Cloud service. For the purpose of the exercise, we will only send sensor data 10 times.

```

static pthread_addr_t start_streaming(pthread_addr_t arg)
{
    int num_posts=0;
    int i=0;
    struct timespec timestamp;

    printf("Start streaming sensor data\n");
    for(i=0; i<num_posts;i++) {
        int val = ReadSensor(0);
        char message[100] = "";

        clock_gettime(CLOCK_REALTIME, &timestamp);
        snprintf(message, 100, "{\"sensor\":%d, \"timestamp\":%ld}",
val, timestamp.tv_sec);
        printf(message);
        SendMessageToCloud(message);
        usleep(10000 * 1000);
    }
}

artik_error StartStreaming()
{
    artik_error ret = S_OK;
    static pthread_t tid;
    pthread_attr_t attr;
    int status;

    pthread_attr_init(&attr);
    pthread_attr_setschedpolicy(&attr, SCHED_RR);
    pthread_attr_setstacksize(&attr, 16 * 1024);

    status = pthread_create(&tid, &attr, start_streaming, NULL);
    if (status) {
        printf("Failed to create thread for sensor streaming\n");
        ret = E_NO_MEM;
        goto exit;
    }
    pthread_setname_np(tid, "streaming start");
    pthread_join(tid, NULL);
    printf("Sensor Streaming successfully connected\n");

exit:
    return ret;
}

```

4. In artik_onboarding.h, add ReadSensor() function definition. Scroll to the bottom of the file, and add the lines below right after `#define ARRAY_SIZE(a) (sizeof(a)/sizeof(a)[0])`



```
/* ADC Reading */
int ReadSensor(int);
artik_error StartStreaming();
```

5. In artik_onboarding.c, look for line “if (StartLwm2m(true) == S_OK) {“ in function **static pthread_addr_t start_onboarding(pthread_addr_t arg)** , extend the logic to invoke StartStreaming().The bold lines are what we added to the existing flow. This piece of code is called the first time when you onboard your ARTIK 055s device.

```
if (StartLwm2m(true) == S_OK) {
    if (StartStreaming() == S_OK) {
        printf("ARTIK Cloud connection started\n");
        goto exit;
    } else {
        printf("Failed to stream sensor data\n");
    }
} else {
    printf("Failed to start DM connection, switching back to
onboarding mode\n");
}
```

6. In artik_onboarding_rest.c, look for **static pthread_addr_t start_websocket(pthread_addr_t arg)**, and invoke StartStreaming() as highlighted below. This piece of code is invoked every time you reboot your device after the device is onboarded.

```
static pthread_addr_t start_websocket(pthread_addr_t arg)
{
    StartMDNSService(false);
    StartWebServer(false, API_SET_CLOUD);
    if (StartCloudWebsocket(true) == S_OK)
        printf("ARTIK Cloud connection started\n");
    if (StartStreaming() == S_OK)
        printf("Start Streaming to ARTIK Cloud\n");
    if (StartLwm2m(true) == S_OK)
        printf("LWM2M connection started\n");

    return NULL;
}
```

Compile the code.



3.2 Define Rules Engine

1. Now, we are going to add Rules to the Cloud. Go to Rules from ARTIK Cloud portal, click “CREATE A NEW RULE” button if you don’t have any existing rules, or “+New Rule” button to add new rules.

Distance sensor reading increases when you move your hand close to the sensor. In the new rule we define, we will check if the sensor reading is more than or equal to 2000. If the value reaches the 2000 threshold, we will trigger the artik055 setOn action, which turns on the blue LED on the board(LED702).

Choose device activity to monitor

IF

Actual Value ▾ artik055s sensor X
ADC sensor reading

more than or equal to 2000

THEN

+ NEW CONDITION

This rule can run at any time on any day

Send actions to your devices

THEN

artik055s setOn X

Click “Save Rule” button once you are done.

2. We will create a corresponding rule to turn off the LED on the board once you move your hands away from the sensor.

Choose device activity to monitor

IF

Actual Value ▾ artik055s sensor X
ADC sensor reading

less than 2000

THEN

+ NEW CONDITION

This rule can run at any time on any day

Send actions to your devices

THEN

artik055s setOff X



3. Here are the rules we created:

The screenshot displays two rules in a software interface. Each rule has a title, conditions, actions, and status information.

Rule 1:

- Title:** SEND SETON TO ARTIK055S
- IF:** artik055s sensor is less than 2000
- THEN:** send to artik055s the action setOff
- Status:** CREATED Today, RUN 0 times

Rule 2:

- Title:** SEND SETON TO ARTIK055S
- IF:** artik055s sensor is more than or equal to 2000
- THEN:** send to artik055s the action setOn
- Status:** CREATED Today, RUN 0 times

3.3 Use Rules Engine to Trigger Device Actions

1. Now, go back to our IDE and reflash the board. Click “Flash System Image” button  to flash the built image to the board.
2. Reboot the board, from your serial console, you should be able to see messages like below.

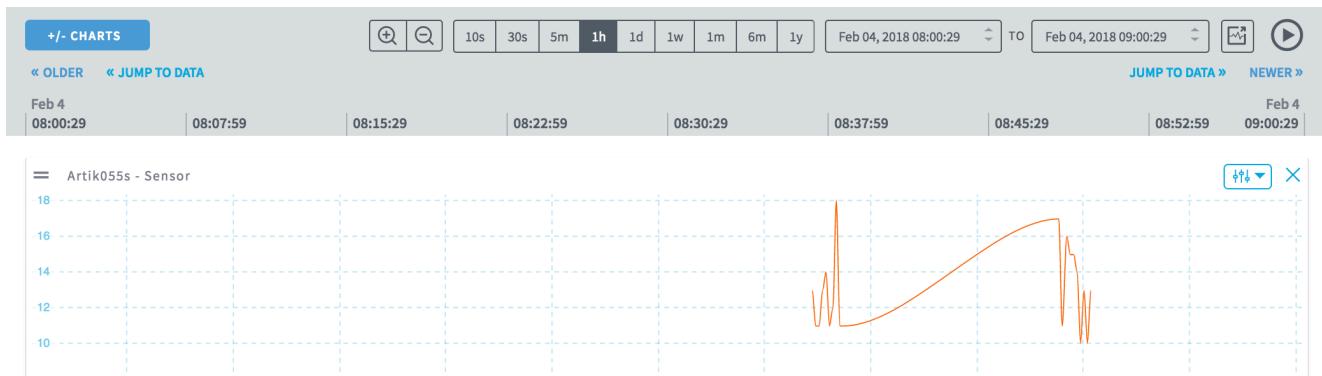
```
TASH>>Onboarding service version 1.7.1
Starting supplicant in foreground...
Connected to Access Point
Start DHCP client
Start NTP client
Start websocket to ARTIK Cloud
WebSocket successfully connected
Start LWM2M connection to ARTIK Cloud
Start streaming sensor data
ADC0=71
{"sensor":71, "timestamp":1521512079}ADC0=2752
{"sensor":2752, "timestamp":1521512089}ADC0=94
{"sensor":94, "timestamp":1521512099}ADC0=2765
{"sensor":2765, "timestamp":1521512109}ADC0=166
{"sensor":166, "timestamp":1521512119}ADC0=215
{"sensor":215, "timestamp":1521512129}ADC0=117
{"sensor":117, "timestamp":1521512139}ADC0=52
{"sensor":52, "timestamp":1521512149}ADC0=23
{"sensor":23, "timestamp":1521512159}ADC0=2748
{"sensor":2748, "timestamp":1521512169}
Sensor Streaming successfully connected
ARTIK Cloud connection started
```

Select 'sensor' option as below to enable data visualization.

The screenshot shows the ARTIK Cloud interface with the 'Charts' tab active. A red box highlights the 'Sensor' checkbox under the 'Artik055s' section. Other visible options include 'State', 'Timestamp', and various power-related metrics like 'AccumulatedConsumedEnergy', 'InstantPower', 'LastOnTime', and 'TimeWindowSpan'. The interface also includes a search bar, date/time filters, and navigation buttons.

From ARTIK Cloud portal, you can see the streamed sensor data. The LED on the board will be turned on and off based on the rules you defined.





Lab 4: OTA solution

4.1 Enable Device Management and OTA Capabilities

In this section, we will enable device management and Over-The-Air update capabilities for this device

1. Open the Developer portal of ARTIK cloud by using this URL <https://developer.artik.cloud/> or by clicking on “ARTIK CLOUD DEVELOPERS” and open the list of device types.
2. Under the new device type created, go to “Device Management -> Properties ” and click on “ENABLE DEVICE PROPERTIES” to enable Light Weight Machine 2 Machine (LWM2M) based device management

The screenshot shows the ARTIK Cloud Developer portal. On the left, there's a sidebar with 'DEVICE TYPES' and a list of device types: ARTIK Smart Parking LED, ARTIK Smart Parking System, ARTIK Smart Trash Cans, ARTIK Toggle, ARTIK Trigger, ARTIK Utility Pole Sound Recognition, artik053, and Philips Respironics. The 'Philips Respironics' item is selected and highlighted in blue. The main content area has a title 'Device Management / Properties' with a subtitle 'Organize and manage a fleet of devices with properties'. It contains two main sections: 'Server Properties' and 'Device Properties'. The 'Server Properties' section describes how server properties can be custom-defined and stored in ARTIK cloud services. The 'Device Properties' section describes how device properties are pre-defined by the LWM2M Standard and can be synchronized between the device and server. Both sections have a 'ADD SERVER PROPERTIES' button and an 'ENABLE DEVICE PROPERTIES' button respectively.

3. This provides an option for the device to connect with ARTIK Cloud at a determined frequency,

The screenshot shows a modal dialog titled 'Enable device properties'. It contains a message: 'Enabling device properties will add LWM2M-based fields to your device view.' Below this is a 'NOTIFICATION FREQUENCY' section. It includes two input fields: one for the minimum notification interval (set to 300) and another for the maximum notification interval (set to 21600). Both fields have placeholder text: 'Device must wait a minimum of [value] seconds between notifications.' and 'Device must wait a maximum of [value] seconds between notifications.' At the bottom of the dialog are two buttons: 'ENABLE' (in blue) and 'CANCEL'.

4. Upon successfully enabling the device properties, “OTA updates” will be available under device management for this device type and all the device properties will be available in ARTIK cloud. These properties are LWM2M device objects that are mirrored in ARTIK cloud from the device, based on the frequency defined earlier.

The screenshot shows the Philips Respironics Device Management / Properties interface. On the left, a sidebar lists device types: ARTIK Smart Parking LED, ARTIK Smart Parking System, ARTIK Smart Trash Cans, ARTIK Toggle, ARTIK Trigger, ARTIK Utility Pole Sound Recognition, artik053, Philips Respironics, Set Up, Device Management (selected), Properties, OTA Updates, Tasks, Monetization, Errors (with a red notification badge), and Sprinkler. The main content area has a header "Philips Respironics Device Management / Properties" with a sub-header "Organize and manage a fleet of devices with properties". It contains two sections: "Server Properties" (described as custom-defined values stored in ARTIK cloud services) and "Device Properties" (a table listing various device attributes like akc, update, device, etc., with their data types and default values). A green "Support" button is at the bottom right.

If you want to see an example of Device Properties, select Device Management of your device type on the left pane, click the Device ID of the connected device

The screenshot shows the Philips Respironics Device Management interface. The left sidebar is identical to the previous one. The main content area has a header "artik053 Device Management" with a search bar "SEARCH FOR KNOWN DEVICES". Below is a table titled "Selected 1 of 4 devices Select All" with columns: DEVICE ID, SERIAL NUMBER, CREATED, CONNECTION STATUS, and LAST CONNECTED. The table lists four devices, with the last one (3d96739259dd4bb5b2f8dfc32817fe39) highlighted with a red box. Buttons for EDIT, EXECUTE, ROWS, CARDS, and REFRESH LIST are above the table. Navigation buttons <<, <, 1, >, >> are at the bottom. A footer message says "Showing 1 to 4 of 4 Devices".

Edit Properties

X

Device	<input type="button" value="UPDATE ALL"/>
Firmware	
AVAILABLEPOWERSOURCES	BATTERYLEVEL
0, 0	100 %
BATTERYSTATUS	CURRENTTIME
0 %	1517802737000
DEVICETYPE	ERRORCODE
A05x	0
FIRMWAREVERSION	HARDWAREVERSION
1.7.1	1.0
<input type="button" value="SAVE"/>	

4.2 Prepare an Update Package

Go back to ARTIK IDE, and open artik_onboarding.h. Change the ONBOARDING_VERSION from 1.7.1 to 1.7.2

```
#ifndef _ARTIK_ONBOARDING_H_
#define _ARTIK_ONBOARDING_H_

#define ONBOARDING_VERSION "1.7.2"

/*
 * Service states
 */
enum ServiceState {
STATE_IDLE,
STATE_ONBOARDING,
STATE_CONNECTING,
STATE_CONNECTED
};
```

Re-compile to generate the **tinyara_head.bin** binary.

Create a **script gen-ota-header.sh** like below on your host machine:

```
#!/bin/bash

INPUT=$1
OUTPUT=$2

if [ $# -ne 2 ]; then
    echo -n -e "Wrong number of parameters\n"
    echo -n -e "Usage: ${BASH_SOURCE} <input file> <output file>\n"
    exit 1;
fi

rm -f ${INPUT}.gz
rm -f ${OUTPUT}

gzip -k ${INPUT}

CRC32=$(crc32 ${INPUT}.gz)
SIZE=$(printf %08x $(wc -c ${INPUT}.gz | awk '{print $1}'))


dd if=${INPUT}.gz of=${OUTPUT} bs=1024 seek=4
echo -n -
e "\x${CRC32:6:2}\x${CRC32:4:2}\x${CRC32:2:2}\x${CRC32:0:2}" | dd of=${OUTPUT} bs=1
seek=4 conv=notrunc
echo -n -e "\x${SIZE:6:2}\x${SIZE:4:2}\x${SIZE:2:2}\x${SIZE:0:2}" | dd of=${OUTPUT}
bs=1 seek=0 conv=notrunc
```

Use this script (tested in Ubuntu and macOS) against the generated **tinyara_head.bin** to create a compressed OTA package with a proper header.

```
$ chmod +x gen-ota-header.sh
$ ./gen-ota-header.sh tinyara_head.bin ota-a055s-firmware-1.7.2.bin
596+1 records in
596+1 records out
611147 bytes transferred in 0.002236 secs (273305929 bytes/sec)
4+0 records in
4+0 records out
4 bytes transferred in 0.000153 secs (26133 bytes/sec)
4+0 records in
4+0 records out
4 bytes transferred in 0.000016 secs (250406 bytes/sec)
```

4.3 Upload OTA Image to ARTIK Cloud Service

1. Upload the generated package to ARTIK cloud. In ARTIK Cloud's developer portal, go to the Device Management tab of the relevant device type, then click the OTA Updates sub-tab, then click the UPLOAD NEW IMAGE button.

2. In the upload window, select Edge Node as the image type then enter the version for the package, this string must exactly match the one you set earlier with ONBOARDING_VERSION. Use the browse button to select the update package generated in step 3.2

Upload a New OTA Update Image X



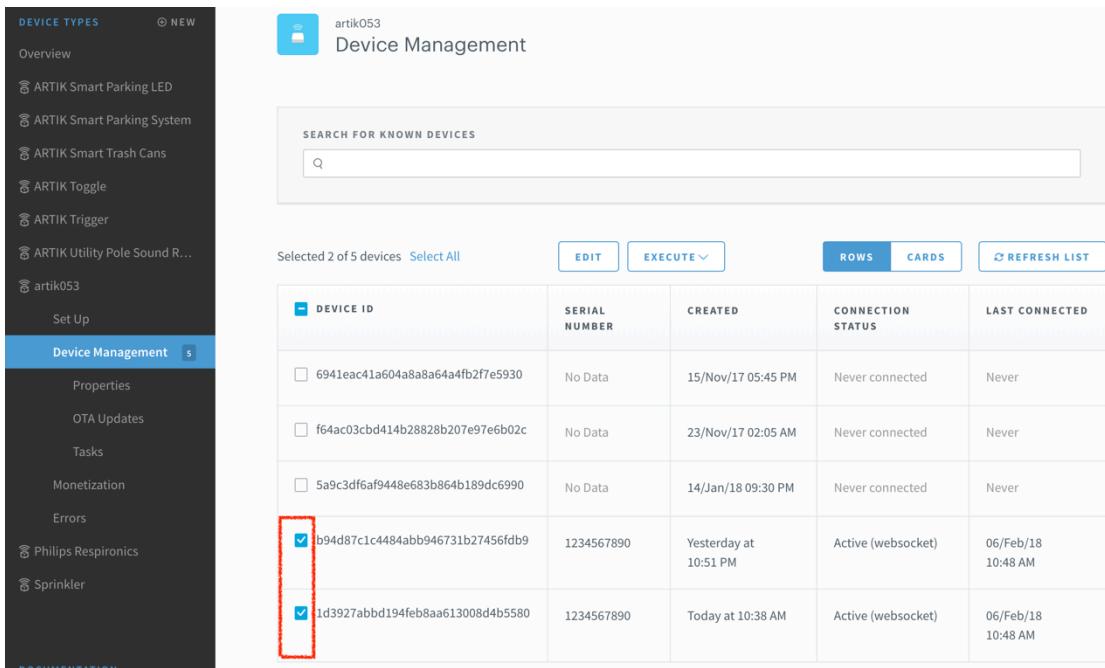
ota-a055s-firmware-1.7.2.bin
(600.82 KB)
[Delete](#)

TYPE <small>?</small>	<input type="text" value="Edge Node"/>
VERSION	<input type="text" value="1.7.2"/>
DESCRIPTION	<input type="text" value="ARTIK 055s OTA Image"/>
UPLOAD	

3. Click the Upload button to upload the image. You will see the new image in the list when upload is done.

4.4 Initiate OTA Update

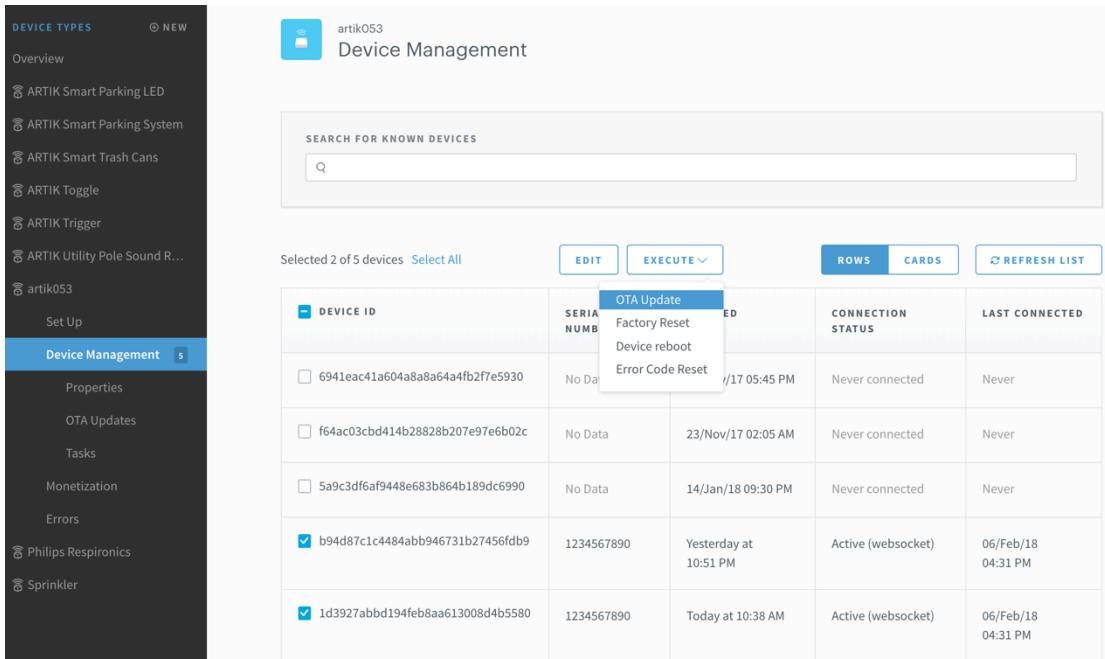
1. Go to Device Management, select the devices that will be upgraded.



The screenshot shows the Device Management interface. On the left, a sidebar lists various device types and management options. Under 'Device Management', 'OTA Updates' is selected. The main area displays a table of devices with columns for Device ID, Serial Number, Created, Connection Status, and Last Connected. Two devices are selected, indicated by checked checkboxes in the first column. A red box highlights these two selected rows.

DEVICE ID	SERIAL NUMBER	CREATED	CONNECTION STATUS	LAST CONNECTED
<input type="checkbox"/> 6941eac41a604a8a8a64a4fb2f7e5930	No Data	15/Nov/17 05:45 PM	Never connected	Never
<input type="checkbox"/> f64ac03cbd414b28828b207e97e6b02c	No Data	23/Nov/17 02:05 AM	Never connected	Never
<input type="checkbox"/> 5a9c3df6af9448e683b864b189dc6990	No Data	14/Jan/18 09:30 PM	Never connected	Never
<input checked="" type="checkbox"/> b94d87c1c4484abb946731b27456fdb9	1234567890	Yesterday at 10:51 PM	Active (websocket)	06/Feb/18 10:48 AM
<input checked="" type="checkbox"/> 1d3927abbd194feb8aa613008d4b5580	1234567890	Today at 10:38 AM	Active (websocket)	06/Feb/18 10:48 AM

2. Click Execute and select OTA update option



The screenshot shows the Device Management interface with the 'EXECUTE' button highlighted. A dropdown menu is open, showing the 'OTA Update' option, which is also highlighted with a blue box. The rest of the interface is identical to the previous screenshot, showing the list of devices and their status.

3. Select the package you want to update to, then click Next.

Step 1 of 2: Choose Image File

select Image Upload New Image

SEARCH FOR KNOWN OTA UPDATE IMAGES

FILE	UPDATE TYPE	VERSION	SIZE	UPLOAD TIME
ota-a055s-firmware-1.7.2.bin ARTIK 055s OTA Image	Edge Node	v1.7.2	600.82 KB	05/Feb/18 11:46 PM
tinyara.bin	System/OS	v1.7	1.12 MB	15/Nov/17 02:28 PM
tinyara_head.bin	System/OS	v1.5	1.15 MB	31/Oct/17 11:51 PM

Showing 1 to 3 of 3 OTA Update Images

<< < **1** > >>

NEXT **CANCEL**

4. In step 2, select a date and time to schedule the update, or leave it empty to perform the update right away. Click PERFORM OTA UPDATE to start.

Step 2 of 2: Perform/Schedule Update

Selected Image

ota-a055s-firmware-1.7.2.bin (600.82 KB)	CHANGE	
UPDATE TYPE	VERSION	UPLOAD TIME
Edge Node	V1.7.2	05/Feb/18 11:46 PM

Device [2 Devices in artik055](#)

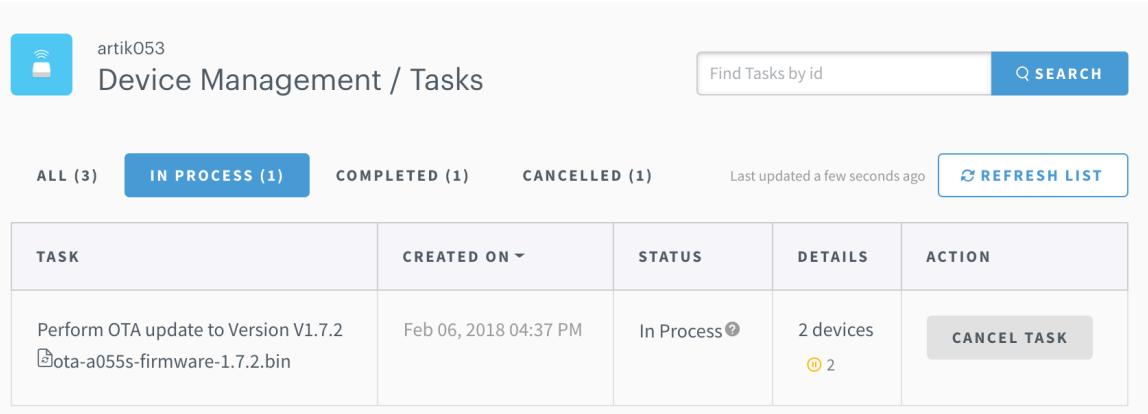
Schedule (Optional) Perform this OTA update with the selected image immediately, or specify a time to schedule for a time in the future.

DATE & TIME

TIMEZONE

PERFORM OTA UPDATE **CANCEL**

5. You can go to Device Management / Tasks to monitor the progress of your update.



The screenshot shows the Device Management / Tasks interface for an ARTIK 053 device. The device name is 'artik053'. There is one task listed:

Task	Created On	Status	Details	Action
Perform OTA update to Version V1.7.2 ota-a053s-firmware-1.7.2.bin	Feb 06, 2018 04:37 PM	In Process	2 devices	CANCEL TASK

At the top right, there is a search bar with placeholder 'Find Tasks by id' and a 'SEARCH' button. Below the search bar, it says 'Last updated a few seconds ago' and has a 'REFRESH LIST' button.

Do NOT hit the 'CANCEL TASK' button when OTA is In Progress.

6. You will see the messages below on the device side. It indicates the LWM2M connection receives the resource change notification. A TLS connection is established with ARTIK Cloud and the image is downloaded automatically.

```
Downloading firmware from
https://api.artik.cloud/v1.1/updates/urls/a999da50b59e4e5792bd22d31171be88?tk=
KpFPXi2xEswMFR08hdhf6DbRIE%2BaoOkwbDfaqpaV%2BS%2B0dZDH0%2F82YAjloVyXqv0eeHC7zJ
KWRCpjQtuog4jQuMAB%2FybCHd%2FMhn6xizuDGA4%3D
with_tls='1', hostname='api.artik.cloud'
filename='/v1.1/updates/urls/a999da50b59e4e5792bd22d31171be88?tk=KpFPXi2xEswMF
R08hdhf6DbRIE%2BaoOkwbDfaqpaV%2BS%2B0dZDH0%2F82YAjloVyXqv0eeHC7zJKWRCpjQtuog4j
QuMAB%2FybCHd%2FMhn6xizuDGA4%3D'
New hostname='ota-updates.artik.cloud' filename='/artik-griffin-ota-images-
prod/prod/dt74e78682ad2c45db93ae3e2dfde76bcd/baefa541055b4f9ca9fab1564c15091?
Expires=1511395118&Signature=MHB5VzcweLhFnvc~S8hM-
2~rHS8LNY8ZPnwlPpwZW9XmPMqQUfucfODtQWK9RTYTuObPq2vquY-
bQkwSorVSwZQVTkDCULVSeTzCdkp3opRfAxY5~ExcvFFt0LJ9VDjBf0p84JHGdz1YArdos8cBeeLtQ
sUb8Wz4qSAlcmrEjvZBiwXyB3CKBstIn9Lt1xfk-UZw5Q-
qVaqXlVEmTTHDKjQwEVnNJ0u~PSRha~tGElc9-
QjxzOfBorkVWzOwJ44pkTrmyXYJAIk8eMuDeEr8PUssbbic8-a-
6m7hWw4FxaxE58244rj0ouL160Tjpq6fNKftDs1EAfs15cd0fe5cXJQ__&Key-Pair-
Id=APKAIX7CUHNYD7CJPXOA'
Skip OTA header (header_size 0, len 0, remaining_header_size 4096)
Skip OTA header (header_size 4096, len 0, remaining_header_size 0)
```

7. Device will reboot automatically after the image update. And you can see the firmware image has been upgraded to 1.7.2

```
TASH>>Onboarding service version 1.7.2
Starting supplicant in foreground...
i2c_uioregister: Registering /dev/i2c-0
i2c_uioregister: Registering /dev/i2c-1
sssro_verify: PASS
Connected to Access Point
Start DHCP client
Start NTP client
Start websocket to ARTIK Cloud
Websocket successfully connected
Start LWM2M connection to ARTIK Cloud
ARTIK Cloud connection started
```

Lab 5: ARTIK SEE APIs

In this exercise, we will extend our app to support ARTIK SEE APIs. We will add a new TASH command for this.

5.1 Extend existing app to include multiple TASH shell commands

1. Add the 4 attached files to artik055s project. The 4 files are:

- examples-api.c
- command.c
- command.h
- security-api.c

You can simply drag and drop the files to your artik055s project.

2. Modify artik_onboarding.c to allow the application to create multiple TASH shell commands. Look for artik_onboarding_main() function, and added the bold lines below before and after.

```
#ifdef CONFIG_BUILD_KERNEL
int main(int argc, FAR char *argv[])
#else
int artik_onboarding_main(int argc, char *argv[ ])
#endif
```

3. Comment out the original int main(int argc, char *argv[]) function, which is at the bottom of artik_onboarding.c.

```
/*
int main(int argc, char *argv[ ])
{
#ifdef CONFIG_TASH
    // add tash command
tash_cmd_install("onboard", artik_onboarding_main, TASH_EXECMD_SYNC);
#endif

    artik_onboarding_main(argc, argv);

    return 0;
}
*/
```

Now, if you flash your ARTIK 055s device again, you should be able to see both onboard and security commands in TASH.

5.2 Key Management APIs

We can use Security Key Management APIs to retrieve factory injected certificates and keys, or generate your own certificate/key-pair.

Here is a function to retrieve factory default certificate.

```
int security_factory_getcert(int argc, char **argv)
{
    int ret = 0, i = 0;
    struct fac_info cert[SEE_CERTNUM] = { {FACTORYKEY_ARTIK_CERT, "ARTIK
CERT"} };

    unsigned char buf[4096];
    unsigned int buf_len = 4096;

    while (i < SEE_CERTNUM) {
        printf("[%d] %s ... ", i, cert[i].name);
        buf_len = 4096;

        ret = see_get_certificate(buf, &buf_len, cert[i].id, CERT_PEM);
        if (ret != 0) {
            printf("get failed : %d\n", ret);
            i++;
            continue;
        }

        ret = print_crt(1, buf, buf_len);

        if (ret) {
            printf("parse failed : %x\n", -ret);
            i++;
            continue;
        }

        set_existence(cert[i].id);
        i++;
        printf("success\n");
    }

    return 0;
}
```

```
TASH>>security factorycert
```

```
[0] ARTIK CERT ...
- cert. version      : 3
- serial number     : 59:42:F1:8F:AF:47:09:5A:0E:90:99:97:EB:32:E0:4B
- issuer name       : C=KR, O=Samsung Semiconductor ARTIK, OU=ARTIK
Root CA, CN=ARTIK Root CA
- subject name      : C=KR, O=Samsung Semiconductor ARTIK, OU=ARTIK
Root CA, CN=ARTIK Root CA
- issued on          : 2017-06-15 20:43:59
- expires on         : 2067-06-15 20:43:59
- signed using       : ECDSA with SHA256
- EC key size        : 256 bits
- basic constraints  : CA=true
- key usage          : Key Cert Sign, CRL Sign

- cert. version      : 3
- serial number     : 59:42:F5:7C:AC:C1:06:31:11:5F:E4:B0:C8:CC:12:2D
- issuer name       : C=KR, O=Samsung Semiconductor ARTIK, OU=ARTIK
Root CA, CN=ARTIK Root CA
- subject name      : C=KR, O=Samsung Semiconductor ARTIK, OU=ARTIK
High Security Device CA, CN=ARTIK High Security Device CA
- issued on          : 2017-06-15 21:00:44
- expires on         : 2067-06-15 21:00:44
- signed using       : ECDSA with SHA256
- EC key size        : 256 bits
- basic constraints  : CA=true, max_pathlen=0
- key usage          : Key Cert Sign, CRL Sign

- cert. version      : 3
- serial number     : 01:01:17:07:24:00:00:0A:D3
- issuer name       : C=KR, O=Samsung Semiconductor ARTIK, OU=ARTIK
High Security Device CA, CN=ARTIK High Security Device CA
- subject name      : C=KR, O=Samsung Semiconductor ARTIK, OU=ARTIK
High Security Device, CN=SIP-0P5WRS30 (01011707-2400-000a-d37e-
d4a58feef97)
- issued on          : 2017-07-24 00:30:30
- expires on         : 2047-07-24 00:30:30
- signed using       : ECDSA with SHA256
- EC key size        : 256 bits
- key usage          : Digital Signature, Non Repudiation
- ext key usage      : TLS Web Client Authentication, TLS Web Server
Authentication

success
```



5.3 Secure Storage APIs

Secure Storage APIs help us read/write data to data slots in secure storage. Here is the function that writes a string to secure storage data slot 0, then reads it back.

```
int security_read_write_secure_storage(int argc, char **argv)
{
    unsigned char input[30] = "Samsung ARTIK Partner Training";
    unsigned int len = sizeof(input);
    unsigned char output[30];

    printf(". SEE Write to Secure Storage...\n");
    if (see_write_secure_storage(input, len, 0) != SEE_OK) {
        printf("Fail\n! see_write_secure_storage\n");
        return 1;
    }
    printf("Write %s to secure storage data slot 0\n", input);

    printf(". SEE Read from Secure Storage...\n");
    if (see_read_secure_storage(output, &len, 0) != SEE_OK) {
        printf("Fail\n");
        return 1;
    }
    printf("Read secure storage data slot 0 : %s\n", output);
    return 0;
}
```

Run `security_api` with `securestorage`, and you should see the results below.

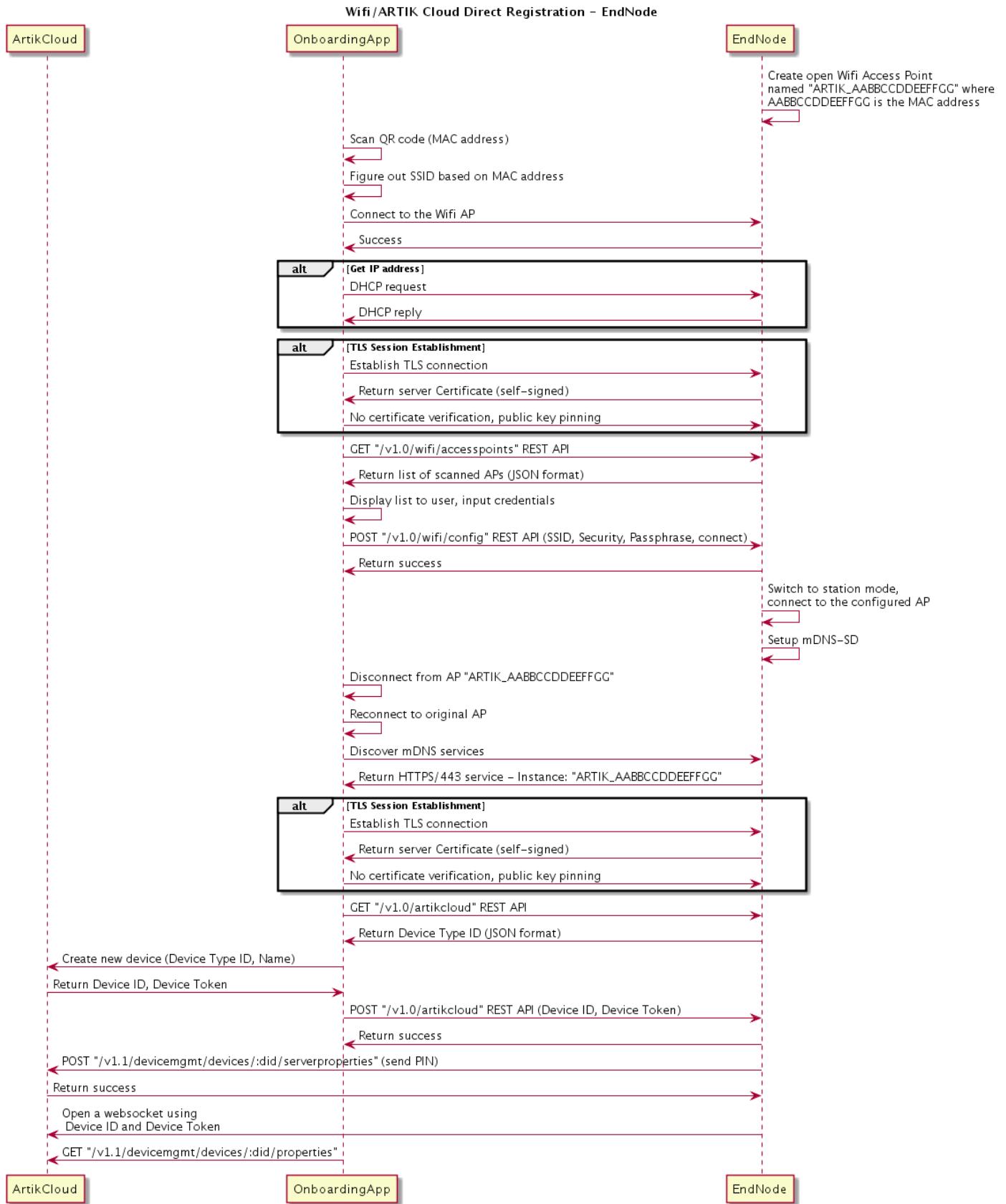
```
TASH >> security securestorage
. Write to Secure Storage...
Write Samsung ARTIK Partner Training to secure storage data slot 0
. Read from Secure Storage...
Read secure storage data slot 0 : Samsung ARTIK Partner Training
```

Appendix A – Create your own Device Type

This is the Manifest file for Lab 1 in JSON format.

```
{  
  "fields": [  
    {  
      "name": "state",  
      "type": "CUSTOM",  
      "valueClass": "Boolean",  
      "isCollection": false,  
      "description": "LED state",  
      "tags": []  
    },  
    {  
      "name": "sensor",  
      "type": "CUSTOM",  
      "valueClass": "Integer",  
      "isCollection": false,  
      "description": "ADC sensor reading",  
      "tags": []  
    },  
    {  
      "name": "timestamp",  
      "type": "TIMESTAMP",  
      "valueClass": "Double",  
      "isCollection": false,  
      "tags": [],  
      "unit": "SI.MILLI(SI.SECOND)"  
    }  
,  
  "actions": [  
    {  
      "name": "setOn",  
      "description": "Set state to On",  
      "parameters": [],  
      "isStandard": true,  
      "type": "CUSTOM"  
    },  
    {  
      "name": "setOff",  
      "description": "Set state to Off",  
      "parameters": [],  
      "isStandard": true,  
      "type": "CUSTOM"  
    }  
,  
  "messageFormat": "json"  
}
```

Appendix B – ARTIK Cloud Direct Onboarding Flow



Wifi/ARTIK Cloud Direct Registration – EndNode

