OPTIGA™ Trust M2 ID2

Product Version: V2

About this document

Scope and purpose

The purpose of this document is to guide a beginner to demonstrate mqttapp using AliOS-Things software package with the OPTIGA™ Trust M2 ID2 ESP32-DevKitC V4. The scope is limited to OPTIGA™ Trust M2 ID2 ESP32-DevKitC V4 and its hardware and software components.

Intended audience

This document addresses: customers, solution providers, porting guide and system integrators.

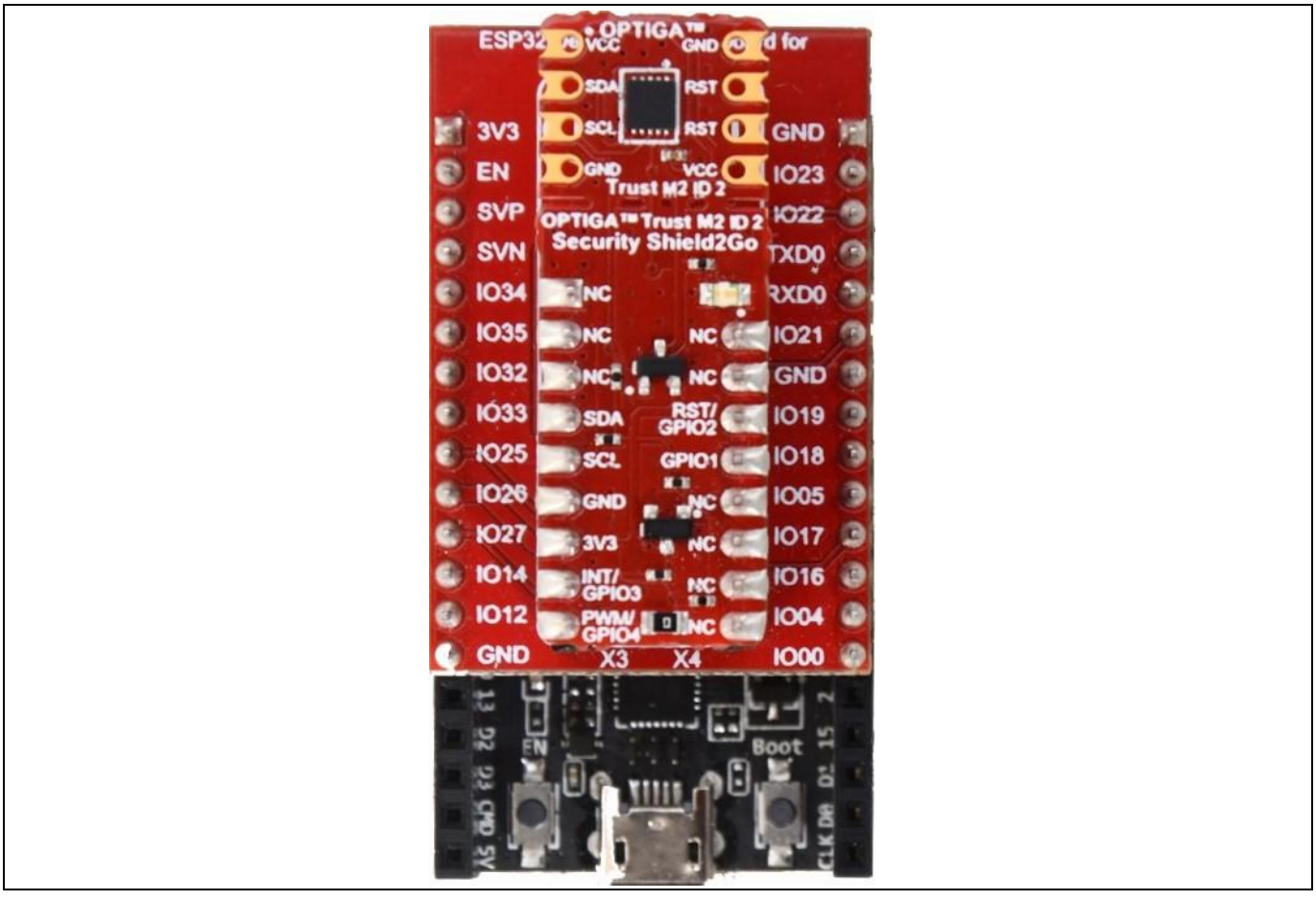

Table of Contents

1 Introduction

This document describes how to setup the environment to demonstrate mqttapp using AliOS-Things software package with the OPTIGA™ Trust M2 ID2 ESP32-DevKitC V4.

1.1 References

Table 1 References

Definition	Source
[1] ESP32-DevKitC V4_usermanual	espressif
[2] Infineon_I2C_Protocol	Infineon
[3] ESP32-WROOM32D-F	ESP32-WROOM32D-F

1.2 Abbreviations

Table 2 Abbreviations

Abbreviation	Definition
API	Application Programming Interface
ESP32-DevKitC	ESP32-DevKitC V4 with ESP32-WROOM32D-F
HW	Hardware
I2C	Inter Integrated Circuit
IoT	Internet of Things
OS	Operating System
PAL	Platform Abstraction Layer
RSA	Rivest-Shamir-Adleman
PC	Personal Computer
RST	Reset
SCL	Serial Clock
SDA	Serial Data
SW	Software
TTL	Transistor Transistor Logic
USB	Universal Serial Bus

OPTIGA™ Trust M2 ID2

2 OPTIGA™ Trust M2 ID2

OPTIGA™ Trust M2 ID2 is a security solution with a pre-programmed security controller with wide range of security features.

It supports secure data, key and metadata object update, hibernate and cryptographic toolbox functionalities, secure communication, platform integrity, data store protection and lifecycle management for Connected Device Security.

This document describe the porting guide of OPTIGA™ host library for other platforms supported by AliOS-Things. Repository link for the same is https://github.com/Infineon/alios-things-optiga-trust-m.

2.1 OPTIGA™ Trust M2 ID2 with ESP32-DevKitC V4

OPTIGA™ Trust M2 ID2 ESP32-DevKitC V4 is designed to provide all the components required to setup the environment to demonstrate the features of the OPTIGA™ Trust M2 ID2.

2.1.1 Evaluation Kit Components

Table 3 Evaluation Kit contents

No.	Item	Description
1	ESP32-DevKitC V4	Hardware Evaluation board for ESP32 microcontroller.
2	ESP32 DevKitC Adapter for Shield2Go	ESP32-DevKitC V4 compatible connector to add Shield2Go board on ESP32-DevKitC V4.
3	OPTIGA™ Trust M2 ID2 Security Shield2Go	Shield2Go board contains OPTIGA™ Trust M2 ID2 chip. It is compatible with Infineon's ESP32 DevKitC Adapter for Shield2Go.
4	Micro USB to USB cable	The cable provides DC supply to ESP32-DevKitC V4 and to flash software.

3 System Setup

This section explains the basic components required for system setup.

3.1 System Overview

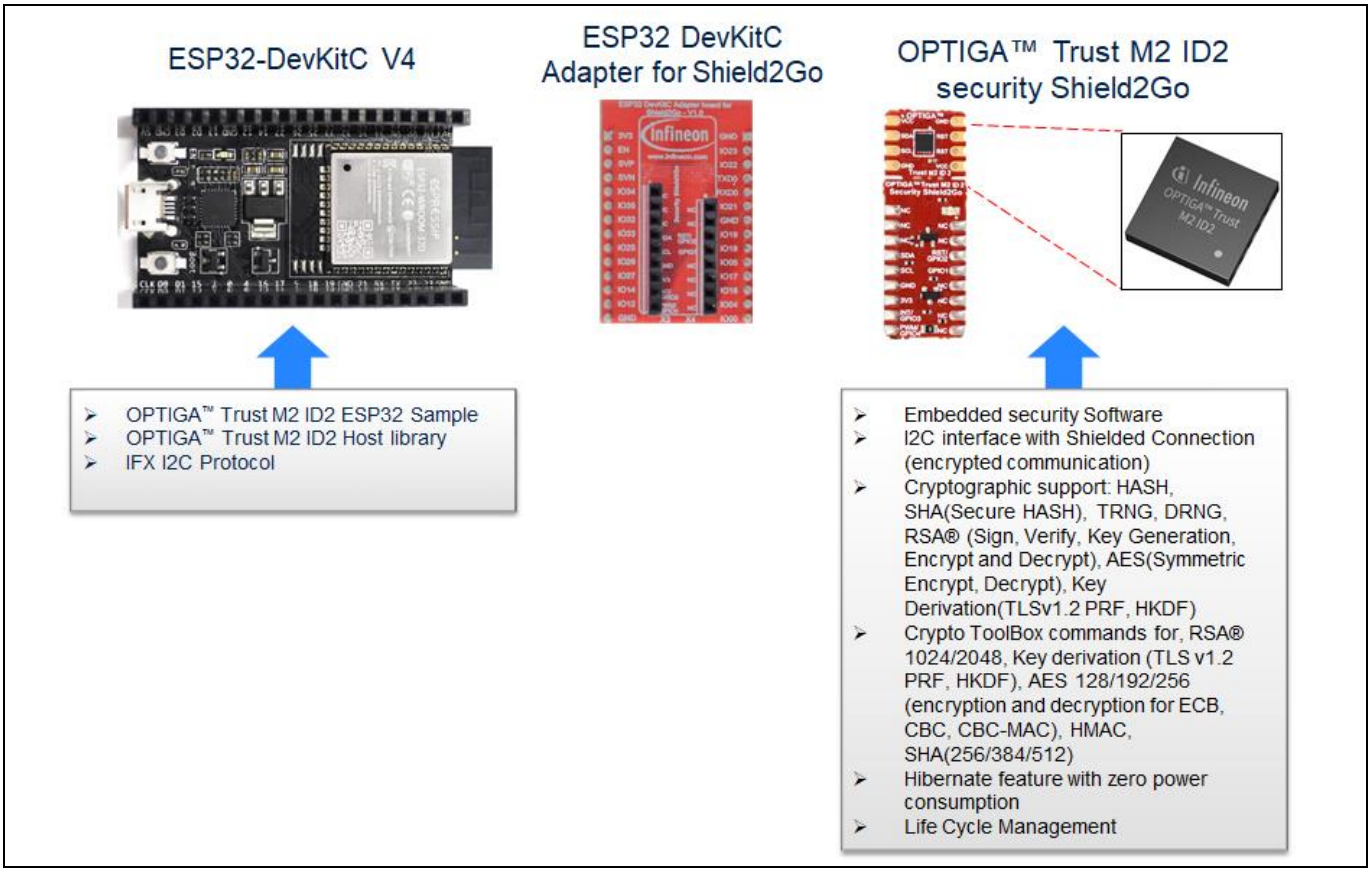

Figure 1 System Overview

This system consists of the following components:

1. ESP32-DevKitC V4
   - The ESP32-DevKitC V4 is an evaluation board with ESP32 Microcontroller from espressif. For more information refer document [1].
   - It is used as a reference platform to simulate the Host.
   - It interacts via I2C.
2. ESP32 DevKitC Adapter for Shield2Go
   - It acts as a gateway to add Shield2Go boards onto ESP32-DevKitC V4.
3. OPTIGA™ Trust M2 ID2 Security Shield2Go
   - Shield2Go board contains OPTIGA™ Trust M2 ID2 chip.

The following interface/connection is done among the above components:

- Micro USB data cable (with Data line) from PC is connected to ESP32-DevKitC V4 to supply power.

3.2 Hardware Setup

The hardware required to run OPTIGA™ Trust M2 ID2 setup is described in this section.

3.2.1 ESP32-DevKitC V4

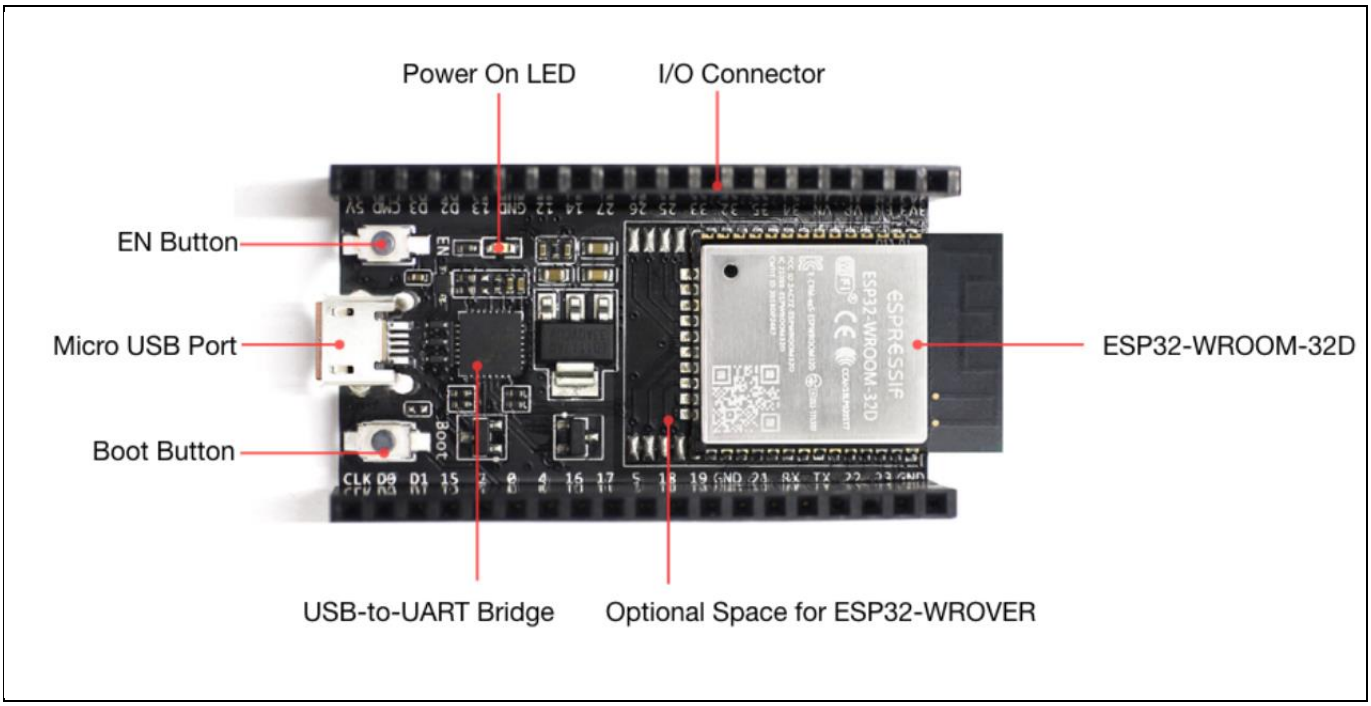

Figure 2 ESP32-DevKitC V4

Connector supports I2C, reset pin and power supply interfaces among others.

Table 4 ESP32-DevKitC V4 Pin Information

No.	Description	Pin
1	I2C SCL	GPIO 22
2	I2C SDA	GPIO 21
3	RST	GPIO 25
4	VCC	GPIO 26
5	GND	GND

For more information about the ESP32 Specification, Architecture and Design/Schematic, refer document [1]

3.2.2 ESP32 DevKitC Adapter for Shield2Go

The ESP32 DevKitC adapter is an evaluation board that allows users to easily combine different Shield2Go boards to ESP compliant ecosystem, for fast evaluation of IoT systems. With its solderless connectors, it allows users to easily stack Shield2Go boards instead of soldering it. The adapter design is derived from ESP32-DevKitC V4 evaluation board.

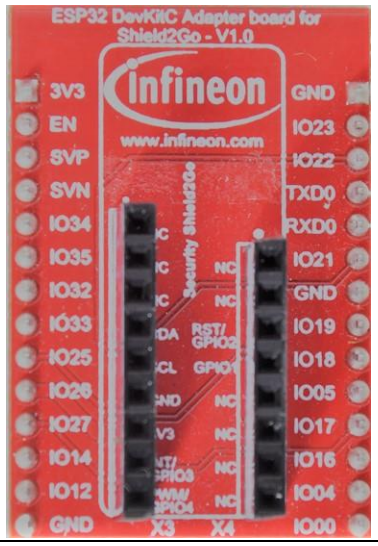

Figure 3 ESP32 DevKitC Adapter for Shield2Go

ESP32 DevKitC adapter features are as follows:

- Provide power supply and connectivity for Shield2Go boards.

3.2.3 Shield2Go Security OPTIGA™ Trust M2 ID2

Shield2Go boards are equipped with featured Infineon ICs and provide a standardized form factor and pin layout, allowing a 'plug and play' approach for easy prototyping.

Figure 4 OPTIGA™ Trust M2 ID2 Shield2Go

The OPTIGA™ Trust M2 ID2 Shield2Go is equipped with OPTIGA™ Trust M2 ID2 security chip. It allows users to develop system solutions by combining Shield2Go with ESP32 DevKitC Adapter for Shield2Go and ESP32.

Note: Ensure no voltage supplied to any of the pins exceeds the absolute maximum rating of V_{cc} + 0.3 V.

3.3 Software Setup

This section describes the software used in ESP32 to run the AliOS-Things OPTIGA™ Trust M2 ID2 setup.

3.3.1 Software Components

All the software components required on AliOS-Things for ESP32 are explained in the following sections.

3.3.1.1 ESP32-DevKitC V4

1. OPTIGA™ Trust M2 ID2 Host Library consists of the following:
   - Service Layer
     The layers (Util and Crypt) provide APIs to interact with OPTIGA™ for various use-case functionalities.
   - Access Layer
     This layer manages the access to the command interface of OPTIGA™ security chip. It also provides the communication interface to the OPTIGA™.
   - Platform Abstraction Layer
     This layer provides platform agnostic interfaces for the underlying HW and SW platform functionalities used by OPTIGA™ libraries.
   - Platform Layer
     This layer provides the platform specific components and libraries for the supported platforms.
2. IFX I2C Protocol
   This is an implementation as per document [2].
3. ESP32 I2C Driver
   These are low level I2C device driver for I2C communication from ESP32 to OPTIGA™ Trust M2 ID2 Security chip.

3.3.2 PC Requirements and Configurations

3.3.2.1 PC Requirement

A 32-bit or 64-bit PC with Windows 10 Operating System with the below requirements need to be used for setting up ESP32 to run the AliOS-Things using OPTIGA™ Trust M2 ID2 setup:

1. One USB port.
2. Python 2.7.14 version to install AliOS-Thing dependency packages
   Link to download Python 2.7.14: Download link
3. Visual Studio Code for development environment.
   Link to download Visual Studio Code: Download link
4. Git for downloading source code.
   Link to download git: Download link
5. FTD driver to access ESP32 via COM port.
   Link to download FTD driver: Download link
6. ASN1 editor require to extract OPTIGA™ supported RSA fields from key provided by Ali key distribution center user is free to use any editor which supports ASN format.

Note: Add C:\Python27 and C:\Python27\Scripts path to environment variable in the beginning of the environment variable list.

4 AliOS-Things environment setup Using OPTIGA™ Trust M2 ID2

4.1 Quick Setup

1. Setup Visual Studio Code as describe from the link
2. Follow the steps till (**安装 AliOS Studio 插件**) as per the git repo version from October 11, 2019.
3. Create a folder in < workspace > (here "workspace" is the folder where AliOS-Things repository will be cloned)
4. Open command prompt and go to the directory < workspace >
5. Download aos-2.1-esp32-with-optiga-se.patch file from the link (Refer section for the patch contain).
6. Execute below commands to download AliOS-Things source package
   - git clone https://github.com/alibaba/AliOS-Things.git
   - cd AliOS-Things
   - git checkout rel_2.1.0
   - git pull origin rel_2.1.0
   - git apply <patch file path> /aos-2.1-esp32-with-optiga-se.patch
     Note: Ignore below warnings while applying the patch

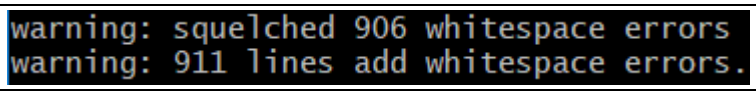

Figure 5 Warning while applying patch

7. Add project to Visual studio code (e.g. Go to File->Add Folder to Workspace->select top level directory of AliOS-Things repository).
8. Open new Terminal in visual studio code (go to Terminal ->New Terminal)
9. To upgrade aos-cube, follow the below steps in Visual Studio Code Terminal
   - pip install --upgrade setuptools
   - pip install --upgrade wheel
   - pip install --upgrade aos-cube

4.1.1 Configure and build mqttapp use case for ESP32-DevKitC V4

This section describes how to configure and build mqttapp example in AliOS-Things source code for ESP32.

Note: To use customize Device name and secret, please refer this section.

4.1.1.1 Configuration

1. Execute below command to configure the mqttapp example
   - aos make mqttapp@esp32devkitc -c config
     Note: while execution above step if below error occurs

```
Can't reach url: https://gitee.com/alios-things/kconfig-frontends-win32.git
Please check your network and download it manually:

  $ git clone https://gitee.com/alios-things/kconfig-frontends-win32.git ./build/kconfig/Win32/

make: *** [build/build_rules/aos_kconfig.mk:94: .\build\kconfig\Win32\kconfig-conf.bat] Error 1
```

Figure 6 Error while configuring the setup

- Execute below command in the terminal
- git clone https://gitee.com/alios-things/kconfig-frontends-win32.git ./build/kconfig/Win32/
- Repeat step 1

2. Open < workspace >\AliOS-Things\build\build_rules\toolchain\ aos_toolchain_xtensa.mk file and check the variable assigned with the below specified value.

```
COMPILER_SPECIFIC_OPTIMIZED_CFLAGS      := -O0
```

3. Execute below command to enable OPTIGA™ host library and iTLS
   - aos make menuconfig
4. Below options need to be selected for ID2(press space bar to select and deselect options)
   - Security -> Link Security ID2
   - Security -> Root of trust, SE-KM
   - Security -> Root of trust, OPTIGA

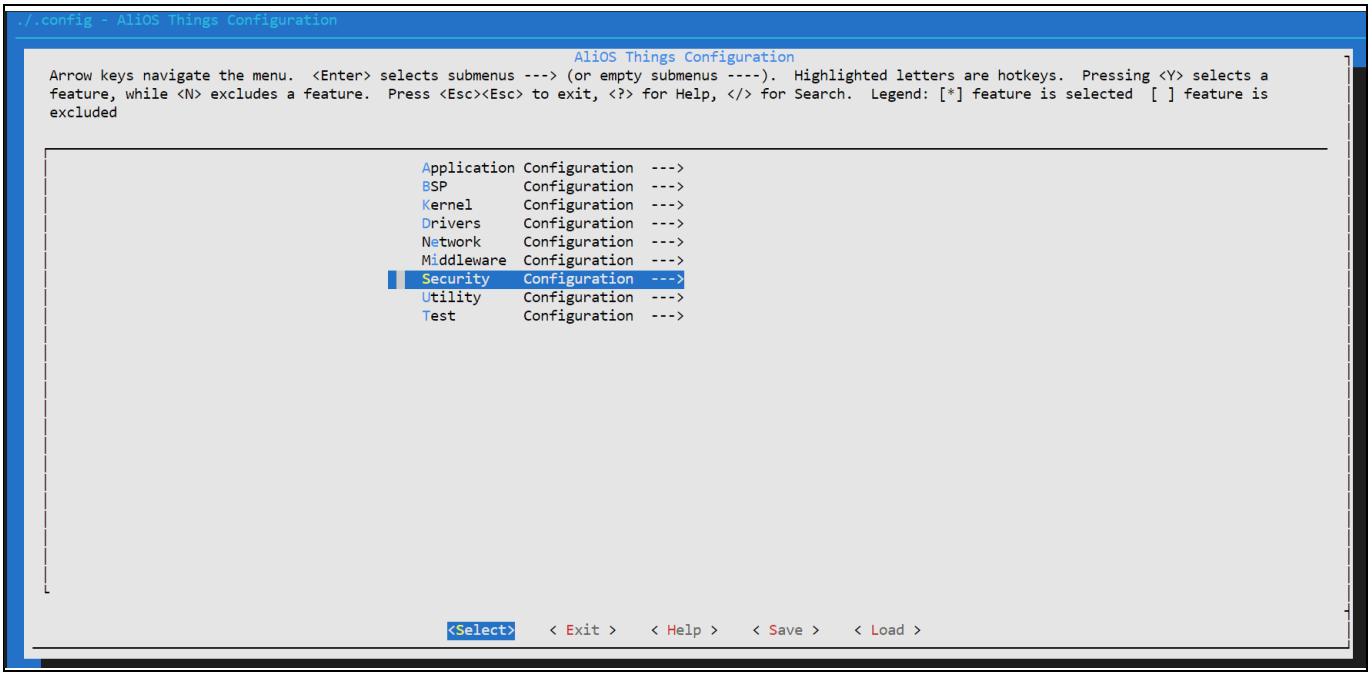

Figure 7 Menuconfig option to Security section

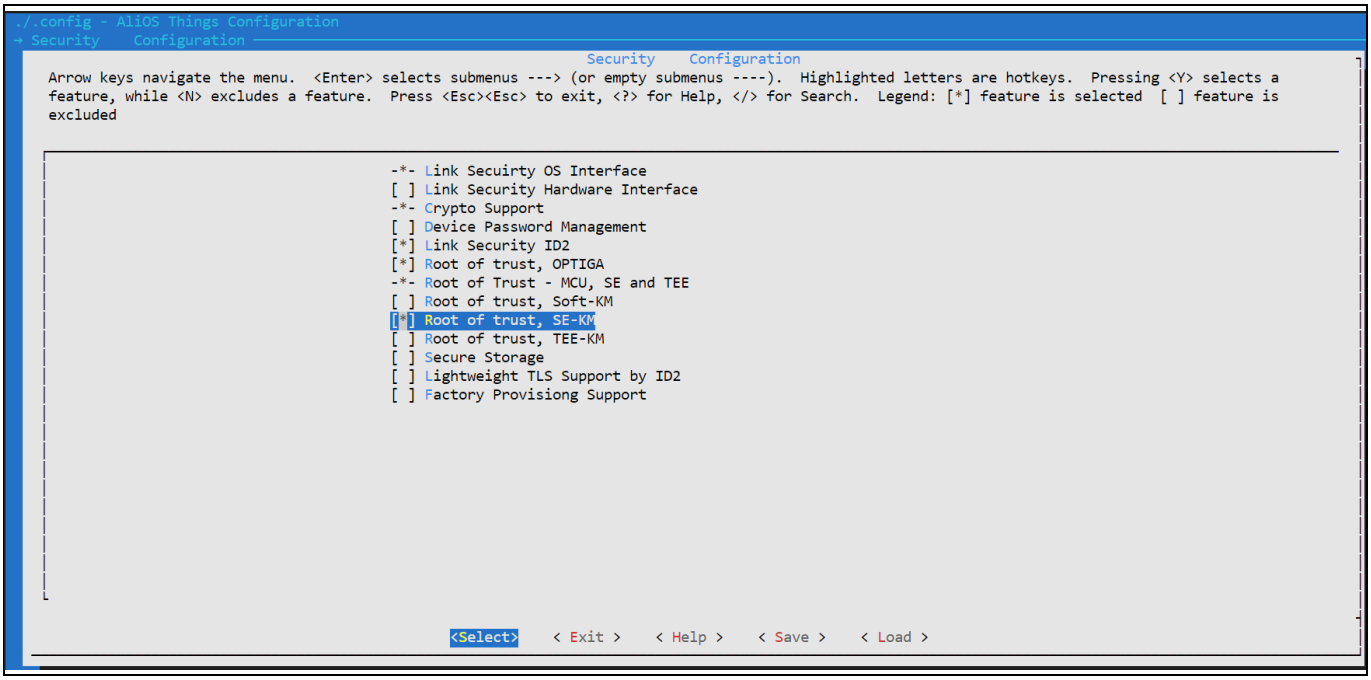

Figure 8 Menuconfig option to select ID2, SE-KM, OPTIGA

5. Change below options to change TLS to iTLS
   - Deselect Middleware -> Linkkit Configuration -> Linkkit HAL Config ->support TLS
   - Select Middleware -> Linkkit Configuration -> Linkkit HAL Config ->support ITLS

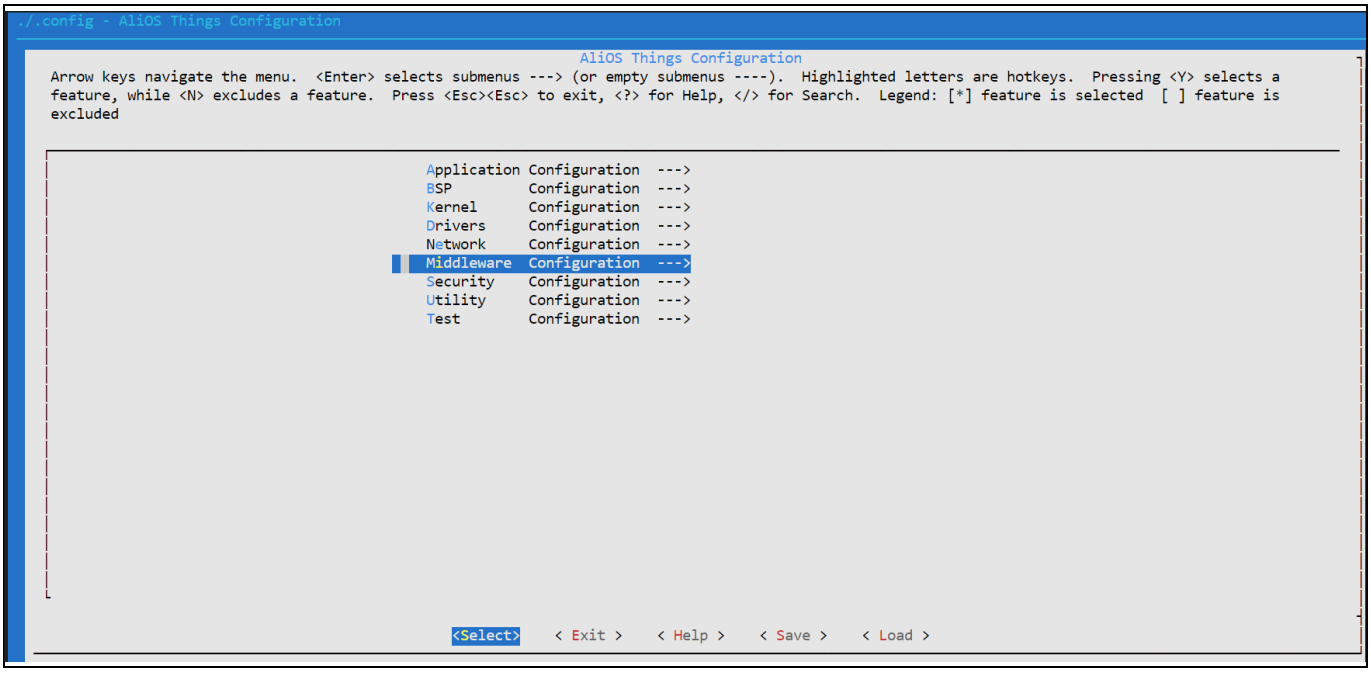

Figure 9 Menuconfig option to Middleware

AliOS-Things environment setup Using OPTIGA™ Trust M2 ID2

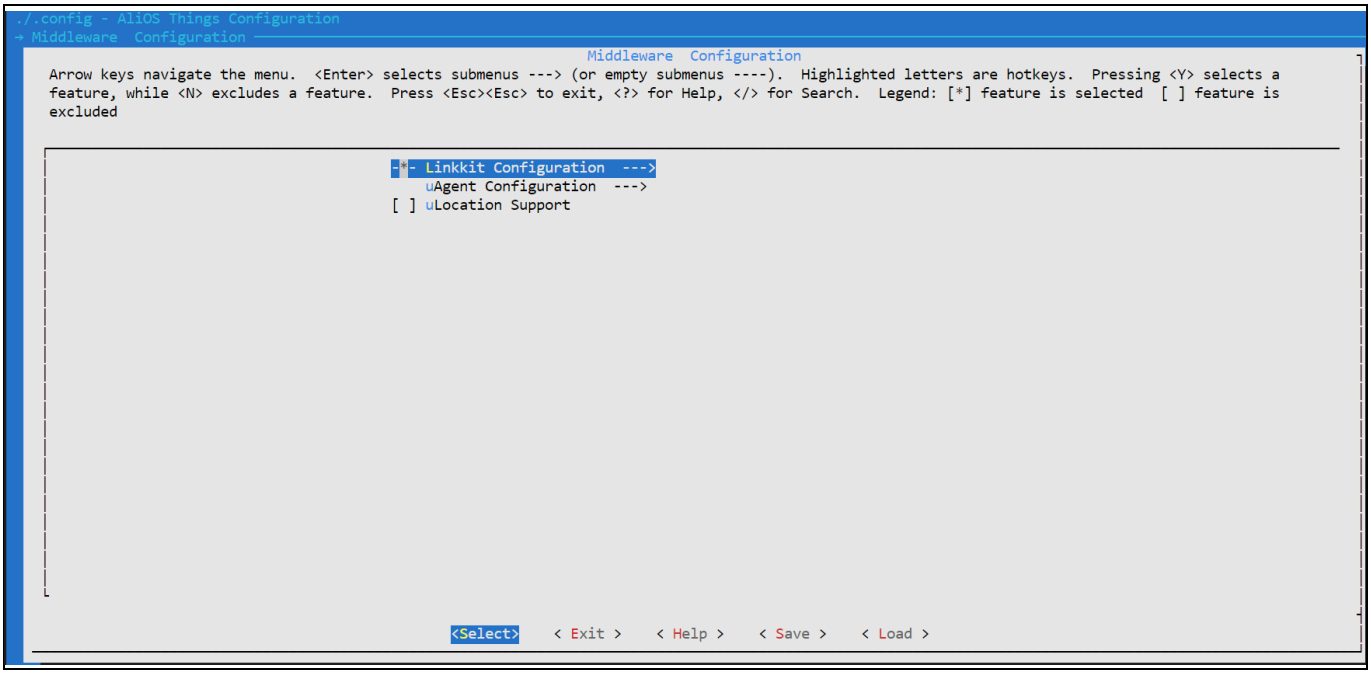

Figure 10 Menuconfig option to Linkkit Configuration

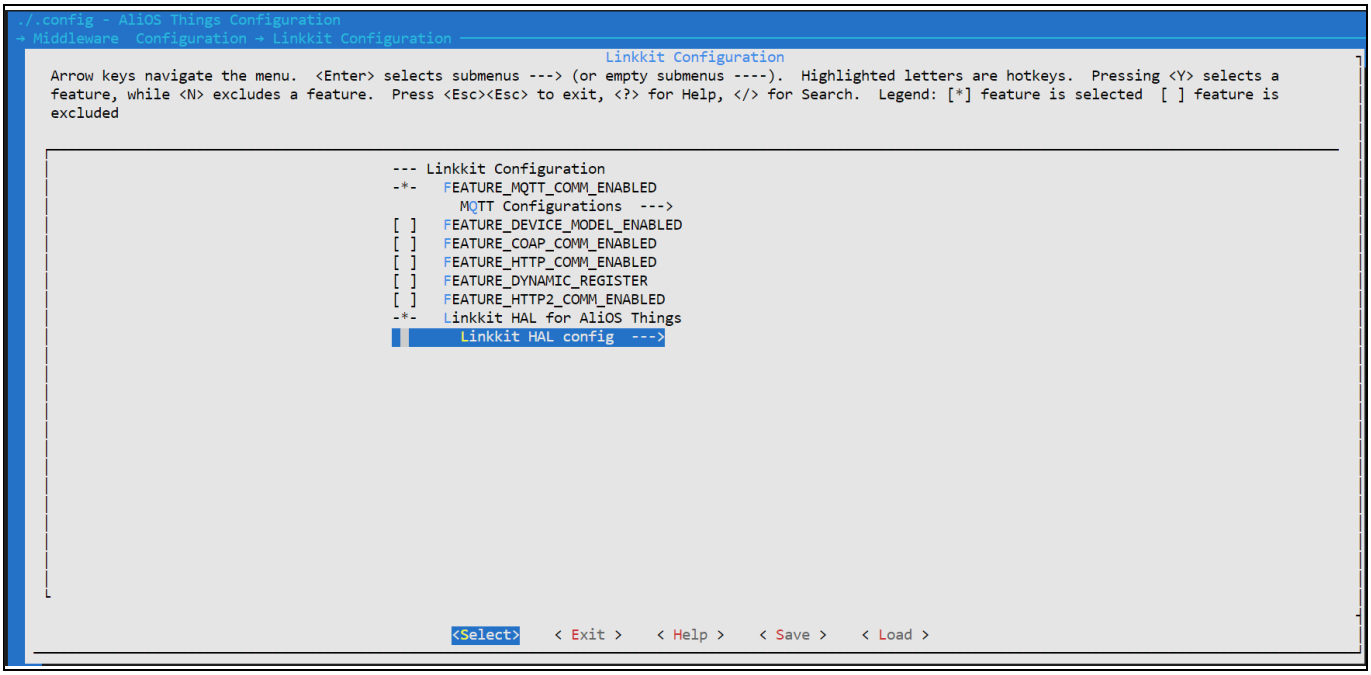

Figure 11 Menuconfig option to Linkkit HAL config

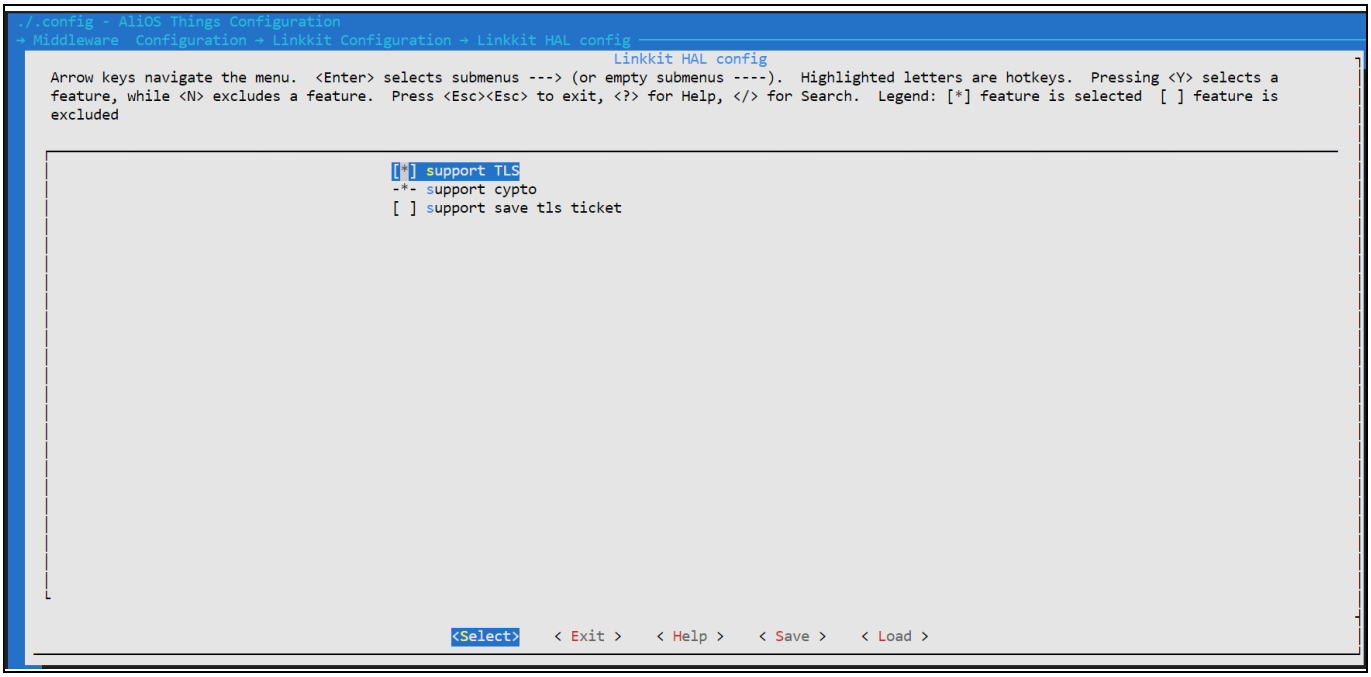

Figure 12 Menuconfig option to deselect support TLS

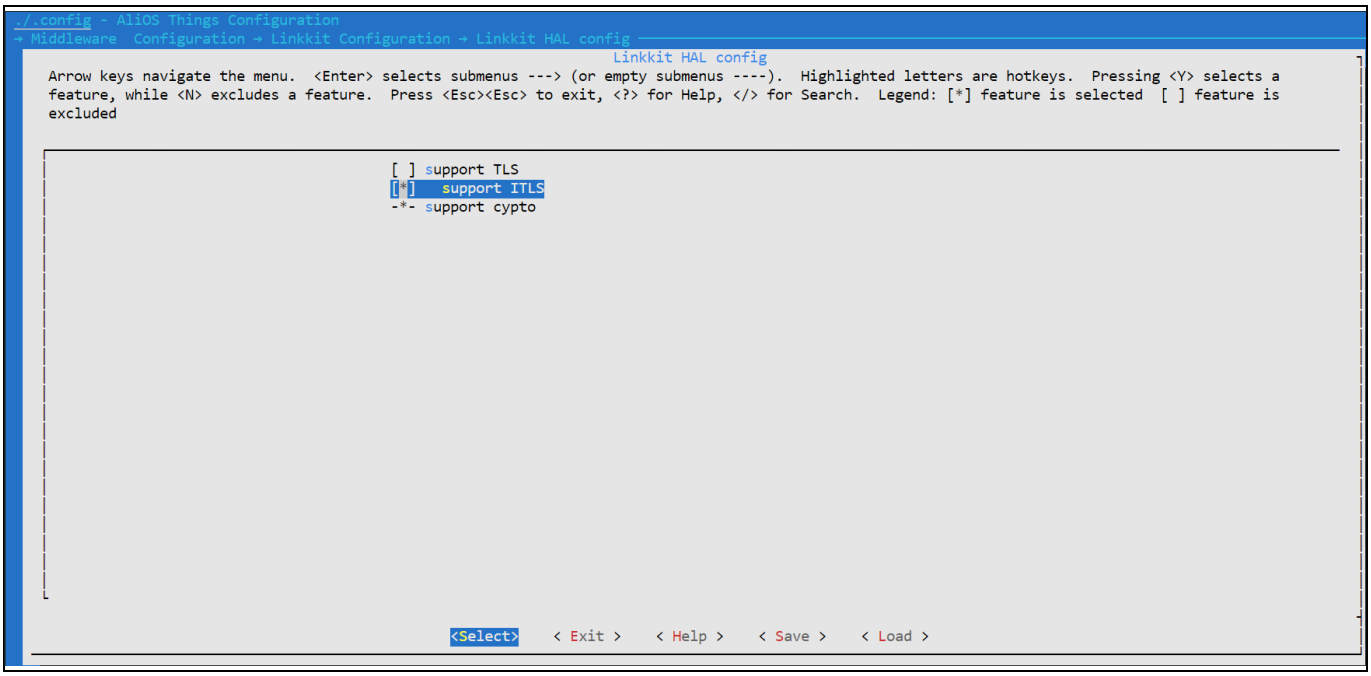

Figure 13 Menuconfig option to select support ITLS

6. Save and exit from menuconfig

4.1.1.2 Build source code

1. To build source code execute below command
   - aos make

4.1.2 Steps to download example hex file to ESP32-DevKitC V4

1. Execute below command to flash the generated HEX file (Check the COM port number from device manager which is connected with your ESP32-DevKitC V4)
   - aos upload mqttapp@esp32devkitc

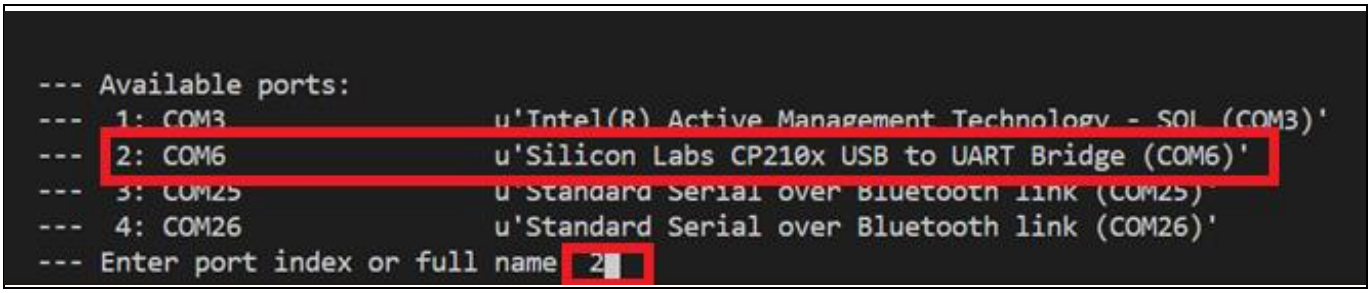

Figure 14 Selecting COM port

4.1.3 Steps to execute mqttapp

1. Execute below command to run mqttapp (Check the COM port number from device manager which is connected with your ESP32-DevKitC V4)
   - aos monitor COMn 115200('n' is the port number assigned to ESP32-DevKitC V4)
2. Press reset button
3. To configure Wi-Fi execute below command (after restart press enter in serial port console)
   - netmgr connect wifi_name wifi_password
4. Below is the example log of successful cloud connection

```
a1FCMDh4ypx.IFXDeviceMqttTest|securemode=8,timestamp=2524608000000,signmethod=
hmacsha256,gw=0,ext=0,partner_id=example.demo.partner-id,module_id=example.dem
o.module-id,authtype=id2,a=aos-r-2.1.0|
[002632]<I> ----------------------------------------
[002637]<I>     MQTT init success!
[002640]<I>     Connecting to /a1FCMDh4ypx.itls.cn-shanghai.aliyuncs.com/1883...
[004936]<I>     ok
[004937]<I>     . Setting up the SSL/TLS structure...
[004937]<I>     ok
ID2 Client Version: 0x00010104
ID2 Client Build Time: Aug 20 2019 12:12:52
=======================================
ID2_DEBUG                   : 0
ID2_OTP_LOCAL_TEST          : 0
ID2_LOCAL_TEST              : 0
ID2_SET_ID_KEY_SUPPORTED    : 0
ID2_KM_API_EMU              : 0
ID2_ITLS_SUPPORTED          : 1
ID2_OTP_SUPPORTED           : 0
ID2_USE_ALI_CRYPTO          : 1
ID2_CRYPTO_TYPE_CONFIG      : ID2_CRYPTO_TYPE_AES
=======================================
[005045]<I>     Performing the SSL/TLS handshake...
 ID2=> id2====> encrypt len=48 !!!
 ID2=> id2====> encrypt len=48 !!!
 mbedtls_parse_auth_code_ext 413:    . Verify iTLS Server authCode OK!
 ID2=> id2====> decrypt len=64 !!!
 ID2=> id2====> irot_hal_cleanup entry!!!
 ID2=> id2====> irot_hal_cleanup exit!!!
[039970]<I> ============================ iTLS handshake used time(usec):
34924719
[039971]<I>     ok
[039978]<I>     iTLS send data(307 bytes) used time(usec): 5455
[040521]<I> ============================ iTLS receive data(1 bytes) used
time(usec): 541832
[040521]<I> ============================ iTLS receive data(1 bytes) used
time(usec): 29
[040526]<I> ============================ iTLS receive data(2 bytes) used
time(usec): 29
[040534]<I>     mqtt connect success!
[040543]<I>     iTLS send data(201 bytes) used time(usec): 4737
[040545]<E>     the network interface info set failed or not set, writen len is 0
[040555]<I>     iTLS send data(479 bytes) used time(usec): 4470
[040559]<I>     iTLS send data(111 bytes) used time(usec): 2538
[040563]<I>     iTLS send data(64 bytes) used time(usec): 2222
mqtt_client|262 ::
 publish message:
 topic: /a1FCMDh4ypx/IFXDeviceMqttTest/update
 payload: update: hello! start!
 rc = 4
[040581]<I>     iTLS send data(42 bytes) used time(usec): 2359
[040583]<I>     mqtt subscribe packet sent,topic =
/a1FCMDh4ypx/IFXDeviceMqttTest/data!
[042795]<I>     iTLS send data(60 bytes) used time(usec): 3603
mqtt_client|287 ::
 publish message:
 topic: /a1FCMDh4ypx/IFXDeviceMqttTest/data
 payload: data: hello! start!
 rc = 6
[042804]<I> ============================ iTLS receive data(1 bytes) used
time(usec): 2958
[042809]<I> ============================ iTLS receive data(1 bytes) used
time(usec): 22
[042817]<I> ============================ iTLS receive data(2 bytes) used
time(usec): 18
event_handle|084 :: publish success, packet-id=1
mqtt_client|313 :: -----cnt = 1 ------
[043006]<I> iTLS send data(85 bytes) used time(usec): 3323
mqtt_client|318 :: packet-id=7, publish topic      msg=
{"attr_name":"temperature","attr_value":"1"}
mqtt_client|328 :: -----cnt 1 = 1 ------
mqtt_client|313 :: -----cnt = 2 ------
[045215]<I> iTLS send data(85 bytes) used time(usec): 3165
```

Figure 15 Successful client server authentication log of mqttapp

Note: Logs on server can be shown as below for connected device node.

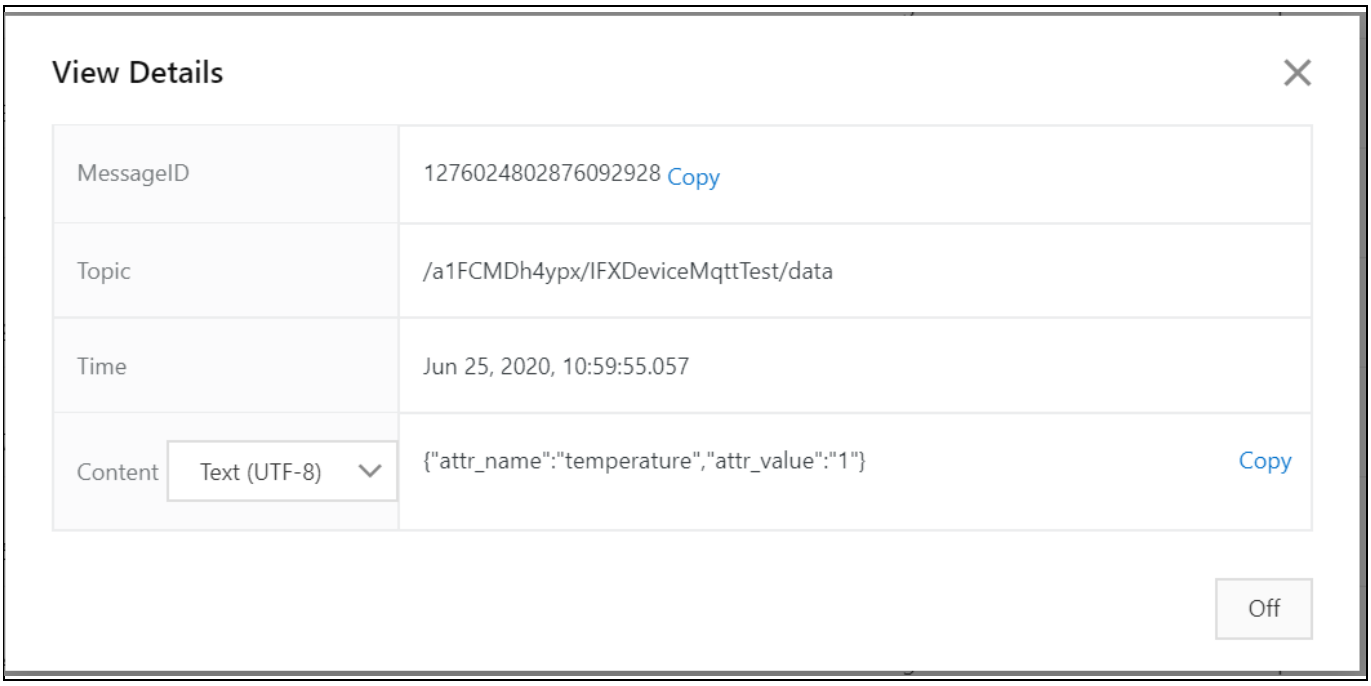

Figure 16 Server side hosted log

5 FAQs

5.1 How to check connectivity in Ali cloud

1. Login to https://cn.aliyun.com/ using personal credential
2. Go to show device log section

5.2 How to update key in OPTIGA™

1. Use protected key update feature to write AES and RSA key
2. Use the tool in the below path to generate manifest and fragments
   AliOS-Things\3rdparty*experimental**optiga*\example*tools**protected_update_data_set*\

5.2.1 Update AES key in OPTIGA™

1. Open ***AliOS-Things\3rdparty\experimental\optiga\example\tools\protected_update_data_set\samples\payload\key\aes_key_128.txt*** file.
2. Write 16 bytes AES key provided by ali ID2 key distribution center in aes_key_128.txt file.
3. Open **samples\gen_key_update_data_set.bat** file and copy the below batch command.

   %PATH%\protected_update_data_set.exe payload_version=3 trust_anchor_oid=E0E3 target_oid=E200 sign_algo=RSA-SSA-PKCS1-V1_5-SHA-256 priv_key=..\samples\integrity\sample_rsa_1024_priv.pem payload_type=key key_algo=129 key_usage=02 key_data=..\samples\payload\key\aes_key_128.txt
4. Execute the ***gen_key_update_data_set.bat*** from the below path
   AliOS-Things\3rdparty\experimental*optiga**example**tools**protected_update_data_set**samples*\
   - Example output is shown as below

FAQs

```
        Payload version      : 3
        Trust anchor oid     : E0E3
        Target oid           : E200
        Signature Algorithm  : RSA-SSA-PKCS1-V1_5-SHA-256
        Private key          : ..\samples\integrity\sample_rsa_1024_priv.pem
        Type of Payload      : key
        Key Usage            : 02
        Key algorithm        : 129
        Key Data             : ..\samples\payload\key\aes_key_128.txt

Info : Setting value for data formatter
        Payload version      : 3
        Trust anchor oid     : E0E3
        Target oid           : E200
        Digest algorithm     : 29
        Signature Algorithm  : FFFEFF5C
        Type of Payload      : FFFFFFFD
Manifest Data , size : [206]
        uint8_t manifest_data[] =
        {
        0x84, 0x47, 0xA1, 0x01, 0x3A, 0x00, 0x01, 0x00, 0xA3, 0xA1, 0x04, 0x42, 0xE0, 0xE3, 0x58, 0x3C,
        0x86, 0x01, 0xF6, 0xF6, 0x84, 0x22, 0x13, 0x03, 0x82, 0x18, 0x81, 0x02, 0x82, 0x82, 0x20, 0x58,
        0x25, 0x82, 0x18, 0x29, 0x58, 0x20, 0x37, 0x5E, 0x3D, 0xD1, 0x7A, 0xA8, 0x59, 0x88, 0xA2, 0x4A,
        0xB9, 0xC2, 0x9A, 0xBA, 0xCD, 0x2A, 0xD8, 0x8B, 0x06, 0x33, 0xD1, 0x84, 0x5B, 0x31, 0x4F, 0xDC,
        0x5D, 0xCD, 0x36, 0x58, 0x8C, 0xE7, 0xF6, 0x82, 0x40, 0x42, 0xE2, 0x00, 0x58, 0x80, 0x60, 0x4D,
        0xA6, 0x5F, 0x81, 0x40, 0x32, 0x49, 0x39, 0xC4, 0x0B, 0xD6, 0x5D, 0xE5, 0xBA, 0xBC, 0xAA, 0x4A,
        0xEB, 0xAF, 0xC4, 0x0A, 0x08, 0xDC, 0x0B, 0xED, 0x10, 0x63, 0x66, 0x08, 0xFA, 0x4C, 0x3C, 0x1F,
        0x41, 0x50, 0x1A, 0x13, 0x93, 0xAF, 0x36, 0x2E, 0x86, 0x46, 0x98, 0x36, 0x5C, 0xD5, 0x41, 0xE2,
        0x15, 0x1E, 0xF4, 0x37, 0x7E, 0x5D, 0x43, 0xB6, 0x1E, 0x49, 0xB6, 0xA8, 0x43, 0xAC, 0x66, 0x20,
        0xFE, 0xEE, 0x4C, 0x7C, 0x76, 0x66, 0xDA, 0xFF, 0xAC, 0x01, 0x89, 0xD9, 0x63, 0x18, 0x76, 0x3C,
        0xB0, 0x0C, 0x54, 0x75, 0x2C, 0x17, 0xB2, 0xE6, 0xE7, 0x6E, 0x22, 0xC4, 0x89, 0xE7, 0x68, 0xD9,
        0x37, 0x2F, 0xE8, 0xAD, 0x31, 0x76, 0x7A, 0xB5, 0x5C, 0x5A, 0x7B, 0xD0, 0x1F, 0x75, 0x2A, 0x66,
        0x37, 0x86, 0x96, 0xF1, 0x34, 0x01, 0x28, 0xE1, 0x11, 0x20, 0xCF, 0x69, 0x20, 0x56,
        };

Fragment number:[01], size:[019]
        uint8_t fragment_01[] =
        {
        0x01, 0x00, 0x10, 0x22, 0x38, 0x5E, 0xA7, 0xEB, 0x61, 0x94, 0xB9, 0xD7, 0xFF, 0xD4, 0x1C, 0xA5,
        0x68, 0x34, 0xD3,
        };
```

Figure 17 Example log of manifest and fragment data for AES key

5. Make the following changes in AliOS-Things\3rdparty\experimental\optiga\example\optiga\usecases\example_ali_id2_key_update.c file
    a. Copy "manifest_data[]" and replace it in "manifest_aes_key[]"
    b. Copy "fragment_01[]" and replace it in "aes_key_final_fragment_array[]"
    c. Copy 12 bytes unique device ID(provided by ali ID2 distribution center) and replace it in "device_id[]"
6. Invoke function example_optiga_util_ali_id2_aes_key_update() in the beginning of application_start() present in AliOS-Things\app\example\mqttapp\app_entry.c
7. Go back to root folder and build source code using below command
    - aos make
8. Flash and execute application

5.2.2 Update RSA 1024 key in OPTIGA™

1. Open **AliOS-Things\3rdparty\experimental\optiga\example\tools\protected_update_data_set\samples\payload\key\rsa_1024_test.pem** file.
2. Copy the RSA key generated from the key provided by Ali ID2 key distribution center in rsa_1024_test.pem file (Refer section ***How to extract RSA key***)
3. Open **samples\gen_key_update_data_set.bat** file and copy the below batch command
   %PATH%\protected_update_data_set.exe payload_version=3 trust_anchor_oid=E0E3 target_oid=E0FC sign_algo=RSA-SSA-PKCS1-V1_5-SHA-256 priv_key=..\samples\integrity\sample_rsa_1024_priv.pem payload_type=key key_algo=65 key_usage=12 key_data=..\samples\payload\key\rsa_1024_test.pem
4. Execute the **gen_key_update_data_set.bat** from the below path
   AliOS-Things\3rdparty\experimental\optiga\example\tools\protected_update_data_set\samples\
5. Example output is shown as below

FAQs

```
Tool Version : 2.00.2449
Info : Default values are set
Info : User provided inputs
        Payload version      : 3
        Trust anchor oid     : E0E3
        Target oid           : E0FC
        Signature Algorithm  : RSA-SSA-PKCS1-V1_5-SHA-256
        Private key          : ..\samples\integrity\sample_rsa_1024_priv.pem
        Type of Payload      : key
        Key Usage            : 12
        Key algorithm        : 65
        Key Data             : ..\samples\payload\key\rsa_1024_test.pem

Info : Setting value for data formatter
        Payload version      : 3
        Trust anchor oid     : E0E3
        Target oid           : E0FC
        Digest algorithm     : 29
        Signature Algorithm  : FFFEFF5C
        Type of Payload      : FFFFFFFD
Manifest Data , size : [208]
        uint8_t manifest_data[] =
        {
        0x84, 0x47, 0xA1, 0x01, 0x3A, 0x00, 0x01, 0x00, 0xA3, 0xA1, 0x04, 0x42, 0xE0, 0xE3, 0x58, 0x3E,
        0x86, 0x01, 0xF6, 0xF6, 0x84, 0x22, 0x19, 0x01, 0x0D, 0x03, 0x82, 0x18, 0x41, 0x12, 0x82, 0x82,
        0x20, 0x58, 0x25, 0x82, 0x18, 0x29, 0x58, 0x20, 0x5C, 0x05, 0x4B, 0xC2, 0x72, 0x73, 0xD2, 0xDC,
        0xC1, 0xFD, 0x29, 0x79, 0x29, 0x96, 0x1B, 0xC8, 0xB3, 0x86, 0xD6, 0xE5, 0x46, 0x1D, 0xD3, 0x58,
        0x22, 0xF6, 0xED, 0x12, 0x0F, 0x41, 0xDB, 0x04, 0xF6, 0x82, 0x40, 0x42, 0xE0, 0xFC, 0x58, 0x80,
        0x6B, 0xEA, 0x79, 0xB9, 0xE7, 0xB8, 0x81, 0x47, 0x18, 0x8F, 0x35, 0x9D, 0xD4, 0x3F, 0xF9, 0xFB,
        0x6F, 0x1C, 0x87, 0xA4, 0xD7, 0x21, 0x5F, 0xA3, 0x49, 0x1C, 0x2D, 0x90, 0xDA, 0xD6, 0x62, 0x5C,
        0x79, 0x11, 0x5B, 0xDA, 0x26, 0xB6, 0x5B, 0xCF, 0x26, 0x40, 0x4F, 0x9A, 0x01, 0x66, 0x44, 0x4D,
        0xFA, 0x43, 0x62, 0xD1, 0xA0, 0xAA, 0xCB, 0x18, 0x06, 0x73, 0x89, 0x8C, 0xFD, 0xF5, 0xDC, 0xCF,
        0xA0, 0xA8, 0x16, 0x4D, 0x4B, 0xAC, 0x52, 0xD0, 0x17, 0xBF, 0x66, 0x12, 0x87, 0xC2, 0x5A, 0x46,
        0x09, 0xBE, 0x9C, 0x1F, 0x86, 0x92, 0xE0, 0xEB, 0xAB, 0xBC, 0x06, 0x38, 0xB8, 0x94, 0xA3, 0x67,
        0x54, 0x97, 0xA0, 0xB7, 0xB1, 0xD2, 0xE9, 0xA8, 0xA1, 0x0A, 0x8A, 0xC9, 0x8F, 0xB2, 0xC0, 0x63,
        0x39, 0x57, 0xB9, 0xC3, 0xCC, 0x7C, 0x0E, 0x98, 0x4F, 0x8E, 0x7C, 0x93, 0x9A, 0x4A, 0x10, 0xBF,

        };

Fragment number:[01], size:[269]
        uint8_t fragment_01[] =
        {
        0x01, 0x00, 0x80, 0x47, 0x66, 0x91, 0x69, 0xF7, 0xA1, 0xAF, 0x25, 0x4F, 0xAA, 0x43, 0xF5, 0xCA,
        0x3D, 0x54, 0xB4, 0x1A, 0xDF, 0x08, 0xCC, 0xCC, 0xD1, 0xB0, 0xB1, 0x20, 0x02, 0xA9, 0x1D, 0xE0,
        0x55, 0x4D, 0x6A, 0x9A, 0x28, 0x94, 0x5A, 0x98, 0x67, 0x10, 0x03, 0x12, 0xDD, 0x46, 0x3E, 0x14,
        0x0F, 0x08, 0x06, 0x1A, 0x20, 0x3C, 0x42, 0x08, 0x17, 0x55, 0xA9, 0xED, 0x9A, 0x83, 0xB2, 0x99,
        0xA6, 0x37, 0x55, 0x79, 0x2D, 0x6F, 0xE2, 0xAC, 0x23, 0xCE, 0xAE, 0x43, 0x37, 0x12, 0xDF, 0x9B,
        0x7F, 0x1F, 0x92, 0x27, 0x14, 0x13, 0x29, 0xEB, 0x03, 0x02, 0x23, 0x35, 0x45, 0xF1, 0xB1, 0xCD,
        0x8C, 0x80, 0x9A, 0x46, 0x8E, 0x11, 0x46, 0xC8, 0xA6, 0xDA, 0xA1, 0xDC, 0xF6, 0xD9, 0x6A, 0x86,
        0x90, 0x2A, 0xD8, 0x11, 0xC8, 0x31, 0x5F, 0x4B, 0xBA, 0xBC, 0x29, 0xAA, 0xC7, 0x4E, 0x49, 0x1E,
        0x50, 0xE7, 0x21, 0x02, 0x00, 0x80, 0x8D, 0x8C, 0xFE, 0x31, 0x2A, 0xB2, 0x90, 0xFD, 0x7F, 0x4A,
        0xC1, 0xE0, 0xDA, 0xF6, 0x28, 0x2F, 0xD2, 0xCC, 0xAF, 0x99, 0xD2, 0xBB, 0x8A, 0x20, 0xAF, 0x5A,
        0x49, 0xB2, 0x1B, 0x60, 0x55, 0x3E, 0xAD, 0x26, 0x12, 0xC3, 0xD5, 0xD2, 0xDF, 0x12, 0xDE, 0xFF,
        0x77, 0x6C, 0x23, 0x99, 0x90, 0x3F, 0x74, 0x88, 0xA7, 0x29, 0xCD, 0xFC, 0x0C, 0x76, 0x4A, 0x88,
        0x06, 0x6B, 0xAE, 0xB9, 0xC5, 0x16, 0x5C, 0x15, 0xFB, 0x49, 0xC1, 0x10, 0xA9, 0x24, 0xD7, 0x8F,
        0x41, 0x83, 0x5C, 0xD7, 0x63, 0x47, 0x6A, 0xA0, 0x78, 0x81, 0xBB, 0xA4, 0x66, 0x94, 0x2B, 0x50,
        0xCA, 0x1F, 0xF7, 0x7E, 0x54, 0xB2, 0xAF, 0x43, 0x9B, 0x59, 0x2B, 0xAE, 0xAB, 0xF7, 0x79, 0x71,
        0xF2, 0x4B, 0x64, 0xFC, 0xBB, 0x86, 0x67, 0x70, 0x75, 0x44, 0x74, 0x0B, 0xCE, 0x5F, 0x50, 0x05,
        0xA5, 0x12, 0x87, 0xCD, 0x79, 0xE9, 0x03, 0x00, 0x04, 0x00, 0x01, 0x00, 0x01,
        };
```

Figure 18 Example log of manifest and fragment data for RSA key

6. Make the following changes in AliOS-Things\3rdparty\experimental\optiga\example\optiga\usecases\example_ali_id2_rsa_key_update.c file
    a. Copy "manifest_data[]" and replace it in "manifest_rsa_key[]"
    b. Copy "fragment_01[]" and replace it in "rsa_key_final_fragment_array[]"
    c. Copy 12 bytes unique device ID(provided by ali ID2 distribution center) and replace it in "rsa_device_id[]"
7. Invoke function example_optiga_util_ali_id2_rsa_key_update() in the beginning of application start() present in AliOS-Things\app\example\mqttapp\app_entry.c
8. Go back to root folder and build source code using below command
    - aos make

5.3 How to change the crypto configuration in AliOS-Things source code

This section describes the modification require to use key type as per needs.

5.3.1 RSA

1. Below Modification is required in AliOS-Things\security\irot\se\chipset\chip_template\chip_config.h
   #define CHIP_CRYPTO_TYPE_CONFIG CHIP_CRYPTO_TYPE_RSA
2. Below Modification is required in AliOS-Things\security\id2\aos.mk
   ifeq ($(CONFIG_LS_KM_SE), y)
   $(NAME)_DEFINES += ID2_CRYPTO_TYPE_CONFIG=ID2_CRYPTO_TYPE_RSA

5.3.2 AES

1. Below Modification is required in AliOS-Things\security\irot\se\chipset\chip_template\chip_config.h
   #define CHIP_CRYPTO_TYPE_CONFIG CHIP_CRYPTO_TYPE_AES
2. Below Modification is required in AliOS-Things\security\id2\aos.mk
   ifeq ($(CONFIG_LS_KM_SE), y)
   $(NAME)_DEFINES += ID2_CRYPTO_TYPE_CONFIG=ID2_CRYPTO_TYPE_AES

5.4 How to extract RSA key

1. Open the RSA key data provided by Ali in ASN.1 editor and copy the highlighted section as shown below (from the highlighted section its only contain the key part)

FAQs

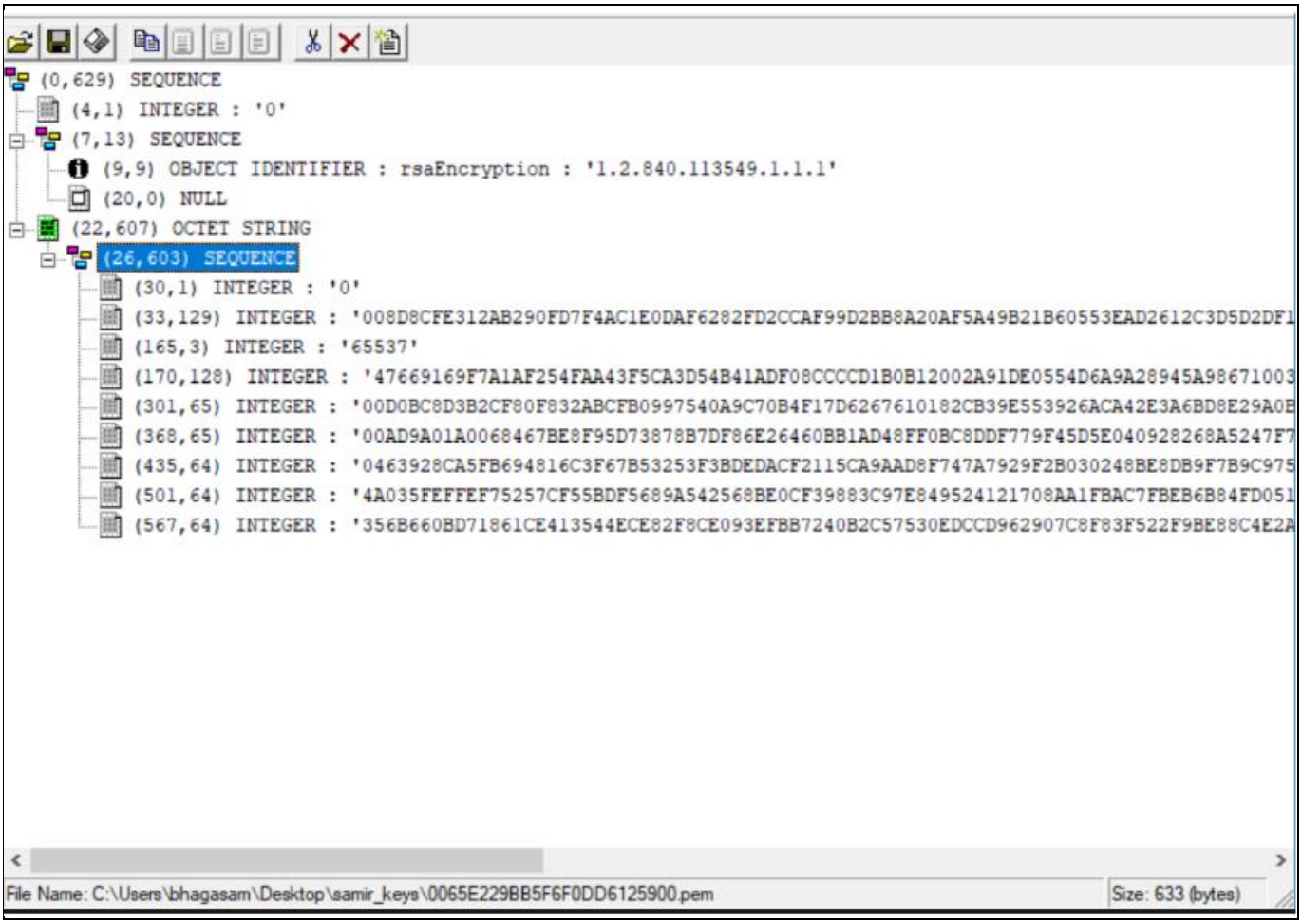

Figure 19 Extraction of only key part

2. Go to Tools-> Data Converter and paste the copied hexadecimal data as shown below

FAQs

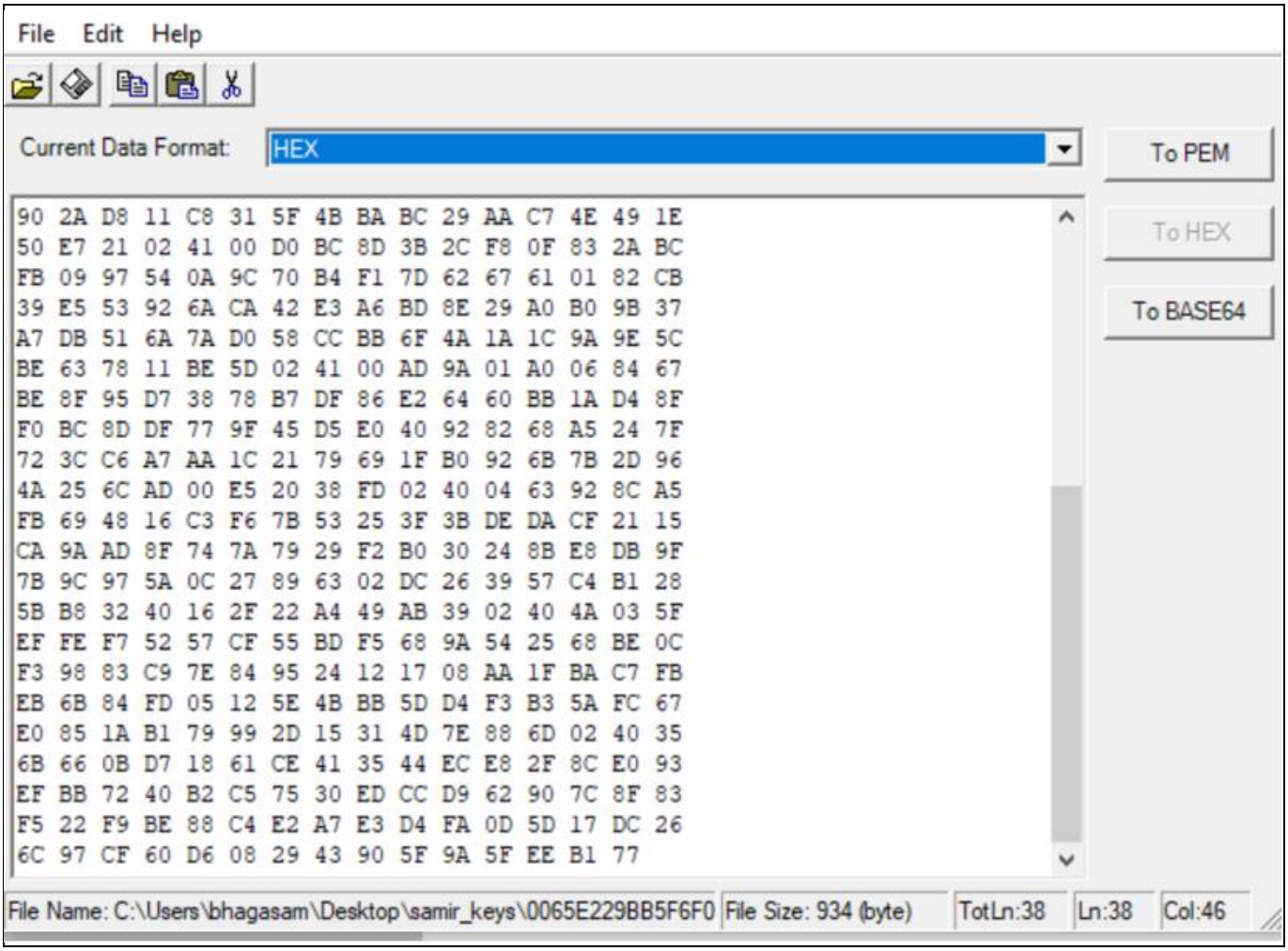

Figure 20 Converting to HEX format

3. Click button "To PEM" to convert hexadecimal data to .pem format

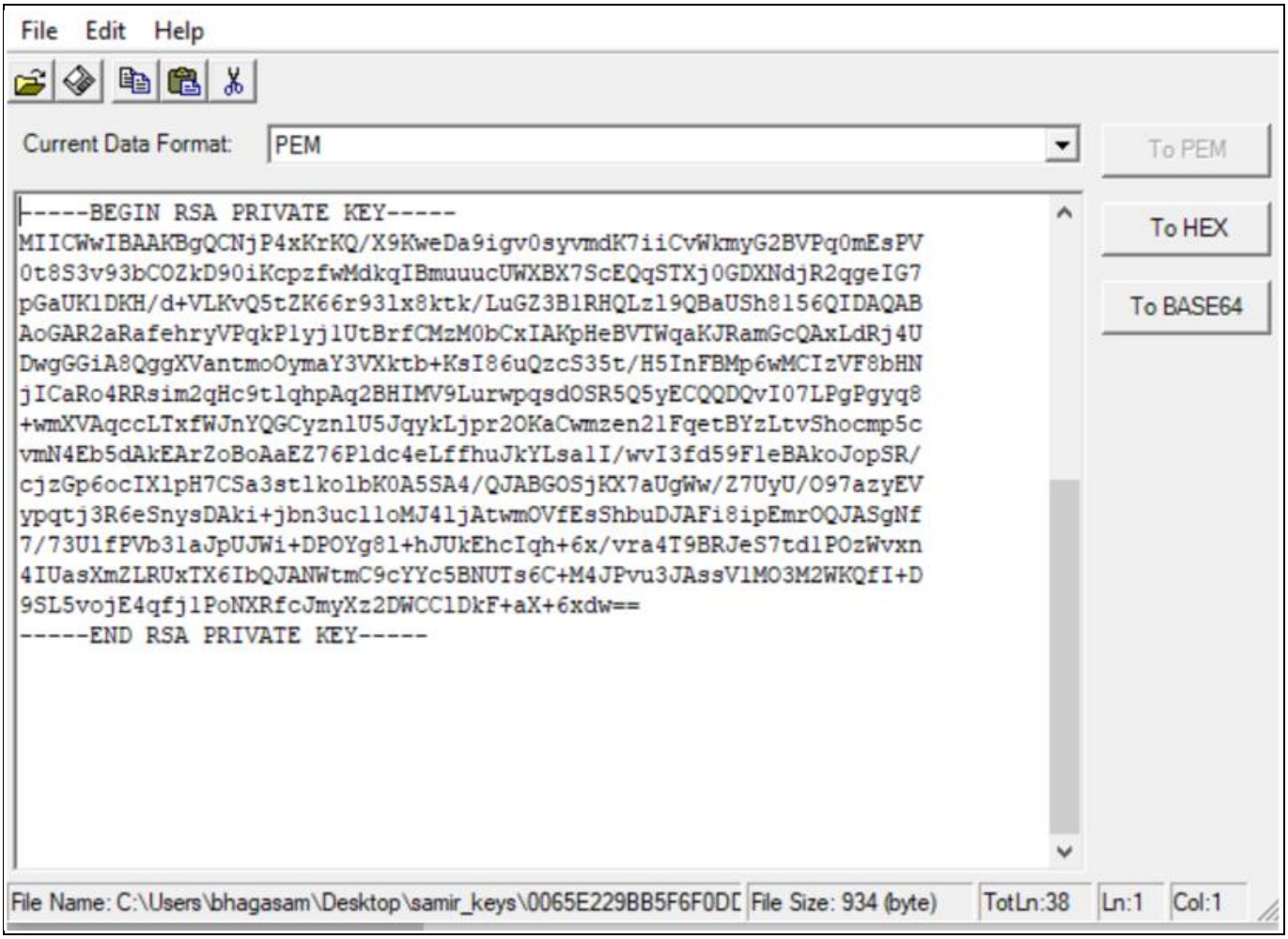

Figure 21 Converting HEX to PEM format

4. Save the file in AliOS-Things\3rdparty\experimental\optiga\example\tools\protected_update_data_set\samples\payload\key\rsa_1024_key.pem

5.5 How to create and update new ID2 device node

1. Create an ID2 device node in https://cn.aliyun.com/
2. Replace the device name and device secret in the below section present in AliOS-Things\app\example\mqttapp\mqtt_example.c file

```
#define DEVICE_NAME     "your device name"
#define DEVICE_SECRET   "your device secret"
#define PRODUCT_KEY     "your product key"
#define PRODUCT_SECRET   "your product secret"
```

5.6 How to enable power off option to OPTIGA™ chip

1. Before doing an OPTIGA™ chip power off, it is recommended to wait until the security event counter on OPTIGA™ reaches zero. This can lead to certain time delays which leads to connection timeout on the server side.

2. The above code flow is implemented in irot_hal_cleanup but it is disabled by default using macro OPTIGA_SE_ENABLE_POWER_DOWN.
3. To enable the code flow, uncomment the macro definition present in AliOS-Things\security\irot\se\src\core\optiga_se_adapter.c file

   #define OPTIGA_SE_ENABLE_POWER_DOWN

5.7 How to port OPTIGA™ host library to different platform

The host library present in AliOS-Things\3rdparty\experimental\optiga location can be port to different platform supported by the AliOS-Things framework.

1. Platform abstraction layer for platform low level drivers like I2C, Timer located in AliOS-Things\3rdparty\experimental\optiga\pal can be modified as described here.
2. User need to use platform specific libitls.a library which should be present in AliOS-Things\security\itls\lib\<platform specific folder>.

5.8 What Infineon patch file contain

Below are the modification present in the patch.

1. OPTIGA™ host library including platform dependent file for ESP32 specific.
2. Modified i2c driver to support read and write operation for maximum 20bytes of data
3. Shielded connection option is disabled due to the limitation of the i2c driver.
4. ESP32 platform supported libitls.a library.

Revision History

Table 5

Document version	Date of release	Description of changes
2.00	2020-09-21	Release to production release

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2020-09-21

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about this document?

Email:
CSSCustomerService@infineon.com

Document reference

IMPORTANT NOTICE

The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics (“Beschaffenheitsgarantie”) .

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party.

In addition, any information given in this document is subject to customer’s compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer’s products and any use of the product of Infineon Technologies in customer’s applications.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer’s technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

For further information on the product, technology, delivery terms and conditions and prices please contact your nearest Infineon Technologies office (**www.infineon.com**).

WARNINGS

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies’ products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.