Raspberry Pi

This guide explains how to build applications for Rapsberry Pi based on the Kaa C++ endpoint SDK (further, the C++ SDK).

For cross-compilation instructions, please refer to Raspberry Pi cross-compilation.

Configuring Rapsberry Pi board

If this is the first time you use the Rapsberry Pi technology, you have to start with configuring your board. For this purpose, refer to the official Raspberry Pi getting started guide.

Installing third-party components for C SDK

The further instructions are expected to be executed directly **on the Rapsberry Pi board** and **with Linux installation and root permissions**.

1. Connect to the board via an ssh client. Refer to the official Raspberry Pi getting started guide for any help required.
2. Upgrade C/C++ compiler.
3. Install common prerequisites:

```
sudo apt-get update
sudo apt-get upgrade
sudo apt-get install cmake libssl-dev
```

Installing third-party components for C++ SDK

The following third-party components must be installed on the Rapsberry Pi board before building the C++ SDK.

- **Mandatory:** Boost (1.54 min.v.), Avro (1.7.5 - 1.7.7 v.), and Botan (only v.1.10) libraries.
- **Optional:** SQLite library. It should be installed only if you are going to use a persistent log storage for the Kaa data collection feature.

NOTE: The further instructions are expected to be executed **in the order given**, directly **on the Rapsberry Pi board,** and **with Linux installation and root permissions**.

To install these libraries, proceed as follows:

1. Connect to the board via an ssh client. Refer to the official Raspberry Pi getting started guide for any help required.
2. Upgrade C/C++ compiler.
3. Install common prerequisites:

```
sudo apt-get update
sudo apt-get upgrade
sudo apt-get install cmake libssl-dev python-dev
```

4. Install Boost (1.54 or higher).

```
sudo apt-get install libbz2-dev libbz2-1.0 zlib1g zlib1g-dev
wget http://sourceforge.net/projects/boost/files/boost/1.58.0/boost_1_58_0.tar.gz
tar -zxf boost_1_58_0.tar.gz
cd boost_1_58_0/
./bootstrap.sh
sudo ./b2 install
```

5. Install Avro (1.7.5-1.7.7).

```
wget http://apache.ip-connect.vn.ua/avro/avro-1.7.7/cpp/avro-cpp-1.7.7.tar.gz
tar -zxf avro-cpp-1.7.7.tar.gz
cd avro-cpp-1.7.7/
cmake -G "Unix Makefiles"
sudo make install
```

6. Install Botan (1.10).

```
wget http://botan.randombit.net/releases/Botan-1.10.9.tgz
tar -zxf Botan-1.10.9.tgz
cd Botan-1.10.9/
./configure.py --cpu=arm
sudo make install
ln -s /usr/local/include/botan-1.10/botan /usr/local/include/botan
```

7. Install SQLite (latest).

```
wget https://www.sqlite.org/2015/sqlite-autoconf-3081002.tar.gz
tar -zxf sqlite-autoconf-3081002.tar.gz
cd sqlite-autoconf-3081002/
./configure
sudo make install
```

Creating applications based on C++ SDK

To create an application based on the C++ SDK, you need to build static/shared Kaa libraries from the generated SDK at first and then link your application to those libraries. You can either do it manually or create some build script to automate the building process (see the *Example* section).

The Rapsberry Pi platform allows building the source code directly on the board. The only thing you need to do prior to that is export your code onto the board. For this purpose you can use, for example, the *scp* utility.

Example

To quickly start with the Kaa IoT platform, you can download one of Kaa demo applications from the Kaa Sandbox and run it on the Raspberry Pi board.
For this example, you need to download the notification demo from the Kaa Sandbox to your host machine. After that, export the downloaded archive to the board and run the demo, as follows:

1. Find the IP address assigned to the Rapsberry Pi network interface:

Rapsberry Pi

```
ifconfig eth0 | grep 'inet addr:' | cut -d: -f2 | awk '{ print $1}'
```

2. Copy the demo application source package to Rapsberry Pi from your host machine:

Host machine

```
scp /path/to/downloaded/demo/notification_demo.tar.gz root@<put ip address here>:notification_demo.
tar.gz
```

3. Build and run the notification demo from your Rapsberry Pi board:

Rapsberry Pi

```
tar -zxf notification_demo.tar.gz
cd CppNotificationDemo/
./build.sh deploy
```

If the build is successful, you will see the following output on the Rapsberry Pi terminal:

```
Notification demo started
--= Press Enter to exit =--
Topic list is empty
Topic list was updated
Topic: id '603', name 'Sample mandatory topic', type 'MANDATORY_SUBSCRIPTION'
Topic: id '604', name 'Sample optional topic', type 'OPTIONAL_SUBSCRIPTION'
Subscribing to optional topic '604'
Notification for topic id '603' received
Notification body: 'This is sample notification for mandatory topic. Client automatically receive
notifications for manadatory topics. More details here: https://docs.kaaproject.org/display/KAA
/Using+notifications'
Notification for topic id '604' received
Notification body: 'This is sample notification for optional topic. Client should subscribe to optional
topics in order to receive notifications. This application subscribes to optional topics automatically.'
Notification demo stopped
```