JPHACKS 2022

FastMile 5G GW API Usage document

|  |  |
| --- | --- |
| Author | [daniel.francis@nokia.com](mailto:daniel.francis@nokia.com) |
| Owner | [chris.mcaloney@nokia.com](mailto:chris.mcaloney@nokia.com), jamie.hingley@nokia.com |
| Organization | Nokia Solutions and Networks Japan G.K. |
| Document Type | Developer Guide (Confidential) |
| Document version | 1.0 |
| Document location | https://jphacks.github.io/2022-guideline/sponsor-products/ |

Contents

[1 What is FastMile 5G GW 4](#_Toc116656679)

[1.1 Scope 4](#_Toc116656680)

[1.2 Device Technical Specifications 4](#_Toc116656681)

[1.3 LXC Containers 5](#_Toc116656682)

[2 Development overview 6](#_Toc116656683)

[2.1 Development Workflow 6](#_Toc116656684)

[2.1.1 Building the file system with necessary packages 6](#_Toc116656685)

[2.1.2 Custom App Development 6](#_Toc116656686)

[2.1.3 Merge the filesystem 7](#_Toc116656687)

[2.1.4 Create LXC container 7](#_Toc116656688)

[2.1.5 Deploy to FastMile 7](#_Toc116656689)

[2.1.6 Run & Debug 7](#_Toc116656690)

[2.2 Requirements 7](#_Toc116656691)

[2.2.1 Developer PC (personal) 7](#_Toc116656692)

[2.2.2 FastMile 5G GW (Nokia – remote access) 8](#_Toc116656693)

[2.2.3 Git environment 8](#_Toc116656694)

[3 BuildRoot 9](#_Toc116656695)

[3.1 Preparation 9](#_Toc116656696)

[3.2 Download BuildRoot tools 9](#_Toc116656697)

[3.3 Configure the file system 10](#_Toc116656698)

[3.4 Make 12](#_Toc116656699)

[4 App Development 12](#_Toc116656700)

[5 Merge Files 13](#_Toc116656701)

[6 Create LXC Container 14](#_Toc116656702)

[7 Deploy to FastMile 5G GW 15](#_Toc116656703)

[8 Run & Debug 18](#_Toc116656704)

[9 Sample container – Bcap 18](#_Toc116656705)

[10 Resources & Links 18](#_Toc116656706)

[11 Document change history 19](#_Toc116656707)

# What is FastMile 5G GW

The Nokia FastMile 5G Gateway 3.2 is a Fixed Wireless Access solution which uses 5G wireless networks to deliver internet access to homes and businesses.

The FastMile 3.2 supports containerized third-party applications. This opens a broad spectrum of use-cases to develop customized applications that can be run from inside the device giving a ultra-edge computing power.

## Scope

This document explains the steps and procedures to deploy the custom developed app and run on FastMile device.

Note that this document does not explain how to develop the app itself.

## Device Technical Specifications

Below is the technical specifications for FastMile 5G GW device.



Figure 1 FastMile 5G GW technical specification

## LXC Containers

The most popular LXC alternative is Docker. This platform, based on Linux Containers, has been continuously developed over the past few years and can now also be run on Windows systems.

LXC (a Linux container) is a virtualization solution on the operating system-level that enables the creation and operation of many isolated Linux virtual environments (VE). These can spun up on a single centralized host. Containers, which are isolation levels, can isolate certain apps or simulate a fully different host.

LXC mitigates VM disadvantages. Linux containers enable the host CPU to effectively divide memory allocations into confinement levels known as namespaces. In comparison, a VM contains the whole operating system and machine setup/emulators, such as the hard disk, virtual CPUs, and network interfaces. The entire (enormous) virtualized environment typically takes some time to boot and consumes a large amount of RAM and CPU.

Kernel

Container Engine/LXC

Bins/Libs

Bins/Libs

App 1

App 2

Nokia Fastmile

Figure 2 LXC container environment

# Development overview

## Development Workflow

A typical development process happens in 6 steps,

Configure BuildRoot

Add additional Packages

Build

Create custom app

Merge rootfs

Deploy in FastMile

Run & Debug

Figure 3 Development flow

### Building the file system with necessary packages

This step involves using BuildRoot tools to create the Linux file system specific to the hardware architecture of FastMile.

More info on BuildRoot is available here <https://buildroot.org/>

A comprehensive manual is available here <https://buildroot.org/downloads/manual/manual.html>

### Custom App Development

Developers are encouraged to use a wide range of tools and platforms to create their app to be run as a container. This includes but not limited to web-server applications using popular frameworks like Django, Ruby on Rails, Angular, Flask etc., standalone applications in JavaScript, Python, C# and much more. If there is a specific tool that you would like to use and would like to know about the compatibility, please reach out to our support team via slack.

### Merge the filesystem

This step basically merges your custom files like init.d/ service files, applications source code etc., with the built filesystem from BuildRoot.

Merging can be seamlessly done via git to maintain version control.

### Create LXC container

For the FastMile to recognize your app package and install it as a LXC container, additional files are required. This document is called the ADF file.

More information on how to create this file and variables used are in the document “Nokia ADF Document Usage Guide.pdf”

### Deploy to FastMile

Remote access to FastMile (SSH from ATC remote PC) and send the package to FastMile

For more info on remote access to ATC environment refer to the document “Nokia ATC Remote Access Usage Document.pdf”

### Run & Debug

Installation of the container package is managed by a TR-069 script. For the flexibility of developer, we have prepared a custom script that will easily install, start/stop and delete the containers.

More info on how to use this script, refer to “Nokia TRCLI script Usage Guide.pdf”

## Requirements

The following is the basic requirements to start building apps for FastMile 5G GW. Although if there is a need to add more devices or customize access control please reach out to our technical team.

### Developer PC (personal)

This will be the main platform for coding and managing personal repositories. We recommend using a Ubuntu environment either as a standalone OS or Windows PC with WSL.

The following guide will use windows PC with WSL2 installed.

More details on how to install WSL2 on your PC,

<https://learn.microsoft.com/en-us/windows/wsl/install>

### FastMile 5G GW (Nokia – remote access)

Nokia will be offering one FastMile 5G GW device per team for development of apps. This has been setup in Nokia Advanced Technology Center at Nokia Roppongi office.

More details on remote access will be shared to team in the future.

Please ask one of our technical support member via slack support channel.

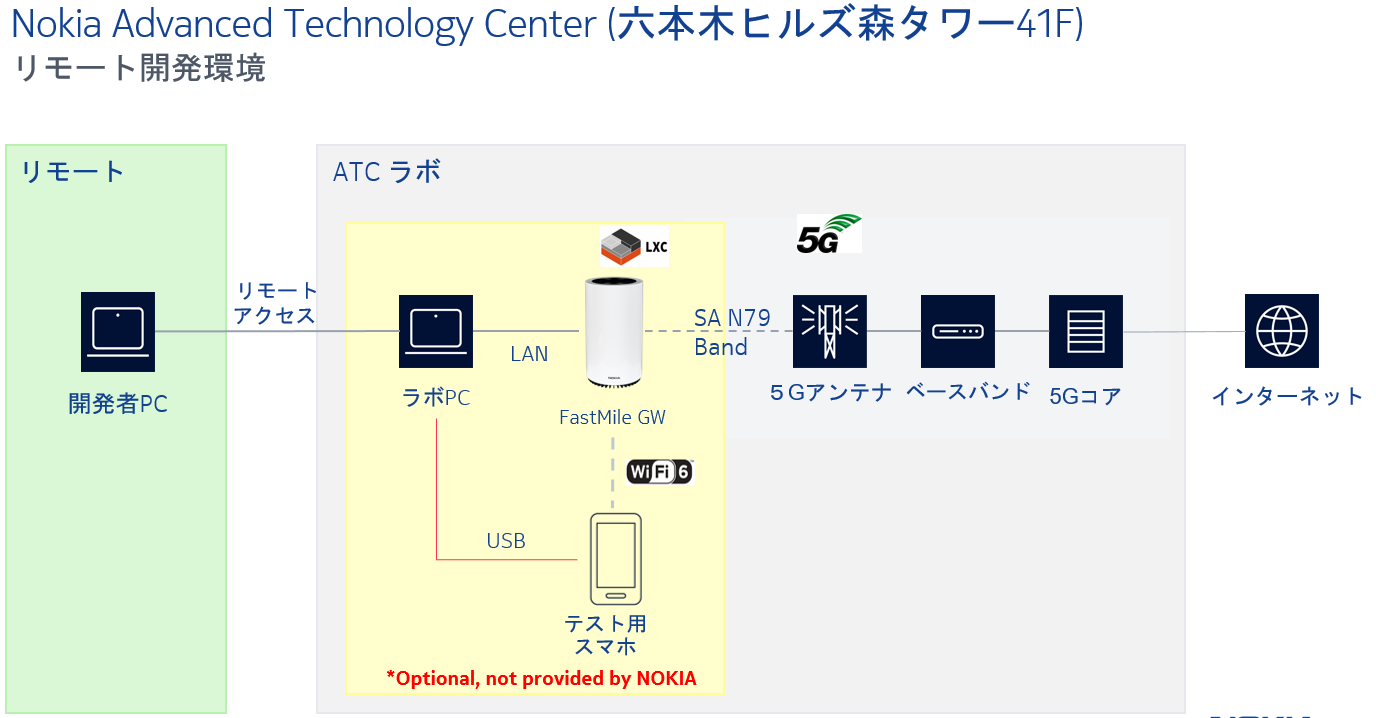


Figure 4 Development setup in Nokia ATC, Roppongi

### Git environment

We highly recommend using a git workspace to manage source code and version control. Client could be of user’s choice github, gitlab etc.,

|  |  |
| --- | --- |
| Sample Git folder - /home/user/fastmile/ | |
| /build |  |
| /build/BuildRoot-2022.02.5/ | BuildRoot config files to build filesystem |
| /root/ | Root filesystem output from BuildRoot (rootfs.tar) |
| /custom/ | This is where all your custom app source code goes. |
| /output/ | Merged files (rootfs + custom app) |
| /temp/ | Temporary location for script outputs |
| /ADF | Nokia ADF document |
| /testapp-1.0.tar.gz | Final container document |
| /make\_contianer.sh | Script to automatically merge and create final container. |

Table 1 sample git repository structure

# BuildRoot

We will be suing BuildRoot to create the Linux filesystem for our custom hardware of FastMile 5G GW.

More Info on BuildRoot - <https://buildroot.org/>

## Preparation

Install the necessary dependency packages

Install ncurses by the command

$ sudo apt install libncurses5-dev libncursesw5-dev

Install gcc, g++ and other build tools by the command

$ sudo apt install build-essential

Install unzip by the command

$ sudo apt install unzip

In single line

$ sudo apt install libncurses5-dev libncursesw5-dev build-essential unzip

## Download BuildRoot tools

Download the tools from [www.buildroot.org](http://www.buildroot.org). We recommend downloading the LTS 2022.02.5 version which have been tested in our environment.

Download to directory fastmile/build/

$ tar -xvf buildroot-2022.02.5.tar

$ cd buildroot-2022.02.5

## Configure the file system

Configure the file system using the menuconfig.

More info here <https://quick-advisors.com/what-is-menuconfig-in-linux/>

Then create config file using command

$ make menuconfig

This opens the GUI in console that lets you to configure the target system and package selection.

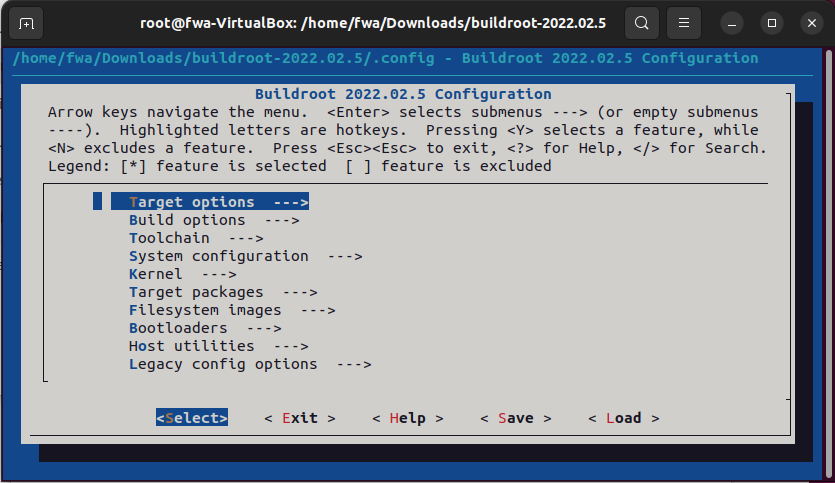


Figure 5 menuconfig selection screen

Use the below table as reference to choose the target options.

|  |  |  |  |
| --- | --- | --- | --- |
| **Config Options** | **Sub** | **Settings** |  |
| Target Options |  |  | See below |
| Target architecture - AARCH64 (little Indian)  Target Binary Format – ELF  Target architecture variant - Cortex-A53  Target ABI – EABI (EABLhf will not work)  Floating point strategy - FP-ARMv8 | | | |
| Build Options | - | - | Default |
| Toolchain | C library | musl | Everything else to Default |
| System Configuration | - | - | Default |
| Kernel | - | - | See below |
| Default. No need to bother with this; the containers will use the host device’s kernel | | | |
| Target Packages | - | - | See below |
| *This section is where additional packages can be added to the root filesystem to satisfy application-specific dependencies. Choices made here are necessarily very specific to the type of container being built. For example, the reference packet capture container (bcap) includes Lua 5.1 and some required Lua libraries, the cmdhosting the container’s interface and REST API.* | | | |
| Filesystem images | - | - | Default |
| Bootloaders | - | - | Default |
| Host utilities | - | - | See below |
| *For some container builds, there might be additional utilities that must be present in the build environment; these can be included in the “Host utilities” section.* | | | |
| Legacy config options | - | - | Default |

Table 2 menuconfig options

Once the selection is done, “exit” to save the configuration to a file.

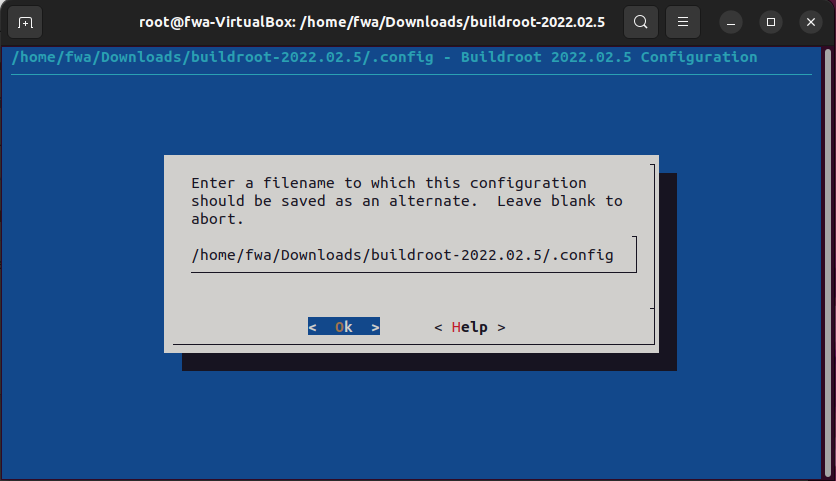


Table 3 Save configuration

## Make

Once the file is saved entering ‘make’ will start the build

$ make

The build approximately takes about 25-30 mins and the output ‘rootfs.tar’ file will be stored in

/buildroot-2022.02.5/

# App Development

This stage is used to develop your custom application.

More details will be provided based on developer’s requirements and the nature of app that they want to develop.



# Merge Files

Since we have two filesystems, one from BuildRoot and other from custom app, these files are merged to a single filesystem.

For eg.,

Adding a new service file to /etc/init.d/custom\_service

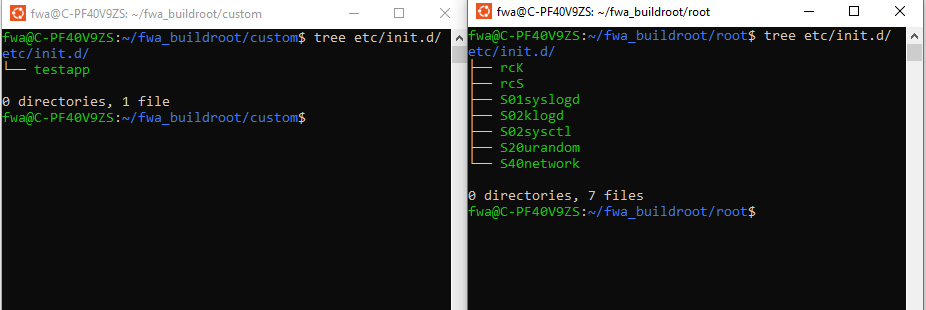


Figure 6 Merging root filesystem with custom files

This step has been added as a part of the execution inside the “prepare\_contianer.sh”

# Create LXC Container

There are several ways to create a LXC container. The most widely used method is to use ‘lxc-create’ directly on the target device. This method unfortunately cannot be used in FastMile due to resource access restrictions.

To overcome this, we will manually create the container package that will be later sent to FastMile and Installed via script.

The two pieces that we need to build our container is **ADF + rootfs.tar**

More information on how to create this file and variables used are in the document “Nokia ADF Document Usage Guide.pdf”



Figure 7 Sample ADF Document

The final container looks like below

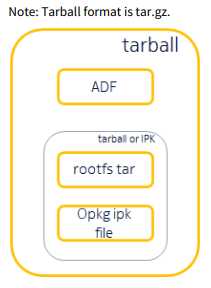


Figure 8 Final container package

# Deploy to FastMile 5G GW

To install the created We use TR-069 protocol to issue commands to the FastMile. This is made possible via the TRCLI script available in the remote PC.

To know more about TR-069 can be found here <https://en.wikipedia.org/wiki/TR-069>

The TRCLI script itself is available along with the “Quick Start Package”

Step 1: Login to Remote PC

Login using the provided AnyDesk ID and Password

Step 2: Transfer the container file

Transfer the applications files to /home/fwa/Desktop/source

Contents checklist

Application.tar.gz has

ADF

Rootf.tar.gz

Step 3: Creating a local HTTP server using python

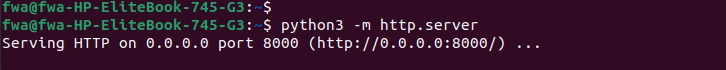
Open terminal and cd to working directory

$ cd /home/fwa/Desktop/source

Start HTTP server using python

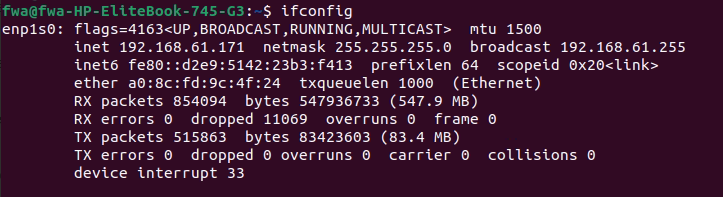
$ pythom3 -m http.server

This will start a http server in port 8000



Step 4: Find Local IP-Address

$ ifconfig



Step 5: Login to FastMile GW console

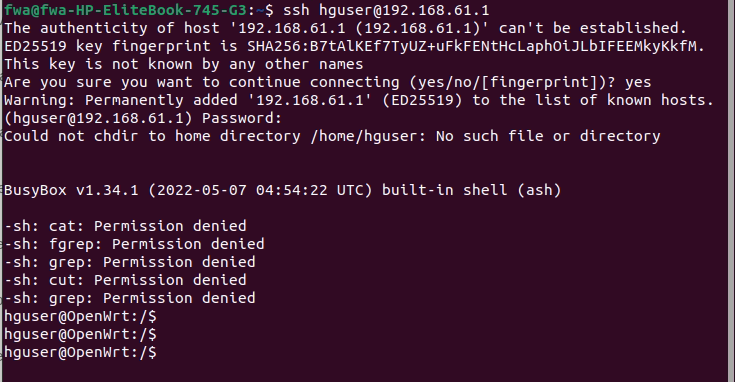
From terminal ssh to FastMile GW

IP: 192.168.61.1, port: 22

user: xxxxxx

pswd: xxxxxxxxxxx

$ ssh [hguser@192.168.61.1](mailto:hguser@192.168.61.1)



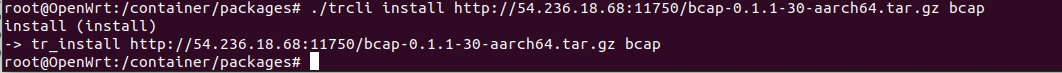
Step 6: Install container

Change to script location

$ cd /container/packages

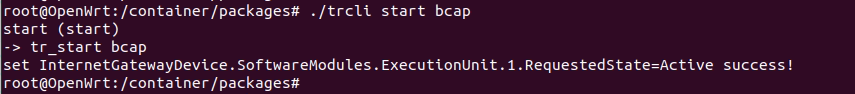
Install using command

$ ./trcli install http://192.168.61.171:8000/testapp-1.0.tar.gz testapp



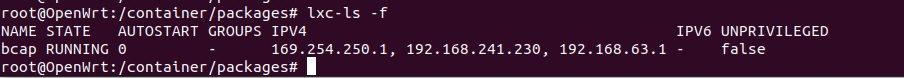
Start the service

$ ./trcli start testapp



Check if the container is running by using LXC commands

$ lxc-ls -f



You will not notice that the container is successfully running inside the FastMile device.

# Run & Debug

To debug a running container and perform testing, use the below LXC commands.

FastMile supports the below common commands of LXC

|  |  |
| --- | --- |
| # Note: all commands take -n as parameter to specify the container | |
| lxc-ls | Lists containers |
| lxc-start | Start and attach |
| lxc-start -d | Start in background |
| lxc-console | Attach to running container |
| lxc-stop | Stop a container |
| lxc-destroy | Terminates a container |
| lxc-attach -n <name> -- <command> | Run command in running container |
| lxc-monitor # | Monitor containers for state changes |
| lxc-wait | Wait for a state change |
| lxc-info | Give details on a container |
| lxc-netstat | View connections to container |
| lxc-ps | List the processes belonging to a specific container |

Table 4 List of supported lxc commands

# Sample container – Bcap

We have prepared a sample container BCAP – Bcapture which captures the network traffic on the FastMile network interface.

This can also be used for debugging in cases where data analysis of network traffic is necessary.

Refer to the file “bcap-0.1.1-30-aarch64.tar.gz” for more details.

# Resources & Links

# Document change history

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Version | Status | Date | Author | Approver | Description of changes |
| 1.1 | Draft | 2022/10/14 | Daniel Sampras |  | Initial draft |