Smart-M3 Platform Installation to DD-WRT-Based Wi-Fi Router

Sergei Mikhailov  
ITMO University  
St. Petersburg, Russia  
saboteurincave@gmail.com

Abstract—The paper proposes an approach to Smart-M3 platform installation for DD-WRT-Based Wi-Fi Router.

I. INTRODUCTION

The open source Smart-M3 platform [1] is used for organization of smart space infrastructure. The key idea of this platform is that the formed smart space is device-, domain- and vendor-independent. Smart-M3 assumes that devices and software entities can publish their embedded information for other devices and software entities through simple, shared information brokers.

The Smart-M3 platform consists of two parts: information agents and kernel [2]. This paper describes way of installation platform’s kernel on router. This method allows us to improve scenarios with Lego® Mindstorms robots collaboration. The key idea is compiling and installing Smart-M3 package on a router — it is needed to access to router’s memory for storage compiler and packages for building chosen system.

II. MAIN PART

A. Router choosing and firmware upgrading

For Smart-M3 installation it is needed router USB support. Author used Asus RT-N16 [3], but it is possible to choose another vendor or model. USB flash drive is needed to store kernel and information storage because there is a lack of memory on router for Smart-M3 platform work (RT-N16 has 32Mb of RAM and 32K of NVRAM).

Also, it is needed to change a default router’s firmware. This action allows us to have full access to the router’s hardware for further packages installation. DD-WRT project is a Linux based alternative OpenSource firmware suitable for a great variety of WLAN routers and embedded systems [4]. There are a project’s wiki page, which describes router and firmware installation [5].

B. USB flash drive preparing

Next step after installing DD-WRT is a preparing USB flash drive for further data storage. First of all, it is needed to format flash drive and make partitions on it [6]. It is necessary to create three partitions on flash drive in exact order:

- 1–2 Gb partition with ext3 file system, which would uses as a libraries’ and applications’ storage (“Optware” label or /opt).
- 64–256 Mb partition with linux-swap system, which would uses as a swap-file (“Swapfile” label).
- Rest of disk’s space partition with ext3 file system, which would uses as a main storage for packages (“Data” label or /mnt).

After creating the partitions, it is needed to enable USB support on router. That can be done through using router’s Web GUI: Services — USB and enabling next options: “Core USB Support”, “USB Storage Support”, “Automatic Drive Mount”, and choosing /opt partition as “Disk Mount Point” (don’t forget to apply settings). If USB flash drive is plugged and partitions are done correctly, information about flash drive and mounted /opt partition is available on USB menu.

After this steps, it is needed to create script startup.bash at /opt, which allows us to mount /mnt partition:

```
#!/opt/bash
mount /dev/diskX /part3 /mnt
```

It is able to get access to the router through SSH [7]. After getting access and creating the script it is needed to set script as “Run-on-mount Script Name” at Web GUI: Services — USB.

C. Software downloading

For packages compilation it is needed to install Optware packages [8] from NSLU2-Linux project [9]. First of all, follow instructions from [10] — section “Installing ipkg-opt and uclibc-opt”. If installation have gone well, you can install via ipkg-opt “buildroot” (gcc compiler) and “optware-devel” (developer packages):

- `ipkg-opt install buildroot`
- `ipkg-opt install optware-devel`

After all these steps it will possible to compile Smart-M3 platform.

D. Smart-M3 compilation and installation

Before compilation it is necessary to unset and change some environment variables:

```
unset LD_LIBRARY_PATH
export PATH=/opt/bin:/opt/sbin:/bin:
/sbin:/usr/sbin:/usr/bin
```

ISSN 2305-7254
Also, it is necessary to configure packages with option --prefix=/opt where all libraries and binaries will be located. Typically working scenario with a package is: ./configure --prefix=/opt && make && make install.

There is a list of packages what is needed for Smart-M3 platform. With italics will be shown packages from Smart-M3 tar file. If there are no additional comments after package, it is needed to just configure, make and install package:

1) Libxml, Raptor — packages installation using the typically working scenario.
2) Rasqal — uClibc (C library from DD-WRT firmware) doesn’t have round() function, so it is impossible to build unmodified package on router. It is needed to add own realization of round() function in src/rasqal_literal.c file. Also, it’s necessary to disable round() checking in configuration files. It can be done in two ways: modifying configure.ac by removing line: “AC_MSG_ERROR((Could not find ceil, floor, round in default libs or with -lm))” and after that doing autoreconf -i && ./configure --prefix=/opt && make && make install or editing configure script by removing that line too and making standard package configuring, making and installing.
3) bdb — Berkley DB for triplet storage. It can install via package manager: ipkg-opt install libdb.
4) Redland — it is necessary to specify the bdb path directly and enable threads: ./configure --with-bdb=/opt --with-threads and make sure, that configuration file choose bdb at /opt directories.
5) libffi-3.2.1, gettext-0.19.2, glib 2.28.2 (Important: newer versions is not allowed, because uClibc doesn’t support some thread features), dbus-1.10.6, dbus-glib-0.100, libuuid-1.0.3, Whiteboard, Sib-tcp, libtool 1.14, Redsibd — packages installation using the typically working scenario.

E. Smart-M3 launching

For the Smart-M3 platform launching a message bus opening is required. This is possible by using dbus-launch command. Redsibd and sib-tcp require DBUS_SESSION_BUS_ADDRESS and DBUS_SESSION_BUS_PID environment variables for working. There is a shell script example for Smart-M3 launching:

```
#!/opt/bin/bash
eval $(dbus-launch --sh-syntax)
export DBUS_SESSION_BUS_ADDRESS
export DBUS_SESSION_BUS_PID
redsbin &
redsbinPid=$!
sib-tcp &
sibtcpPid=$!
```

```
echo $redsbinPid $sibtcpPid
$DBUS_SESSION_BUS_PID > /tmp>smartM3Pid
```

It is necessary to store pids for further platform closing. There is an example of closing Smart-M3 platform:

```
#!/opt/bin/bash
input=`cat /tmp/smartM3Pid`
IFS=’ ’ read -a pids <<< "$input"
```

F. Smart-M3 Control Panel

For DD-WRT there is a possibility to install web server on router. Using this feature, users can interact with the Smart-M3 platform through a web interface. For simplifying working with the Smart-M3 platform, author has been developing this kind of service: “Smart-M3 Control Panel”. The following service provides user to:

- launch, stop or relaunch Smart-M3 (Fig. 1).
- watch current status of platform (running, stopped, service breakdown) (Fig. 1).

![Smart-M3 Control panel](image)

**Fig. 1.** Status of the Smart-M3 platform

- download Smart-M3 log files (Fig. 2);

![Logs](image)

**Fig. 2.** Smart-M3 logs

- get storage’s triplets (with dynamically updating), sort and filter triplets and add, remove or edit triplets in storage (Fig. 3).
- set up options (for example, launch redsibd with other storages than Breckley DB).
III. Conclusion

This paper presents an approach of Smart-M3 platform installation to DD-WRT-Based Wi-Fi Router and shows the “Smart-M3 Control Panel” — service which provides user to work with the Smart-M3 platform. The main drawback of compiling packages on router is a general slowness of process. It can be fixed with using cross-compilation on x86 platform.

REFERENCES