Debian development with scratchbox

Lauri Arimo

Debian development with scratchbox

by Lauri Arimo

This document describes how Scratchbox can help developing software for debian

Revision history

Version:	Author:	Description:
2004-05-3	Arimo	Initial revision

Table of Contents

Chapter 1. Introduction

1.1. Purpose of this document

This document describes how Scratchbox can be used for Debian development. This document is not complete "how-to", instead one should get familiar with all the references for deeper knowledge. Most of the topics in this document are well addressed in other documentation, so it is no use to rewrite them.

1.2. About Debian

Debian GNU/Linux distribution[2] is dedicated to free and open source software; the main goal of the distribution is to ensure that one can download and install a fully-functional operating system that is completely adherent to the Debian Free Software Guidelines (DFSG)[3].

1.3. Scratchbox

Scratchbox[1] is a cross-compilation toolkit designed to make embedded Linux application development easier.

Scratchbox also provides a full set of tools to integrate and cross-compile an entire Linux distribution.

Chapter 2. Debian development

2.1. In general

Debian GNU/Linux is developed and maintained by collaborative work of thousands of individuals around the globe. For distributed effort of this magnitude to be successful, quite tight rules are needed. Some of the most important rules of Debian development are described in the Debian Policy Manual[4]. These includes the structure and contents of the Debian archive, several design issues of the operating system, as well as technical requirements that each package must satisfy to be included in the distribution.

2.2. Packaging

The Debian GNU/Linux system is maintained and distributed as a collection of *packages*. Every package must have a *maintainer*. The *maintainer* is responsible for ensuring that the package is placed in the appropriate distributions.

Each package has control information that states various aspects of how package will behavbe. Most important is the information about the package's dependencies. With this information debian package management system *dpkg* and it's front-end *apt* can ensure that system is usable after installing some new software.

2.3. Where to get more info

Debian website[2] is a good place to start searching. Especially Debian Developers' Corner[5] is full of very good documents about debian development. One of the very first things that new developer should read is Debian New Maintainers' Guide[6]. Most of the appropiate documents can be found in debian Debian Documentation Project's Developers manuals section[7].

Chapter 3. Debian and scratchbox

3.1. Rootstrap

Normally when one compiles Debian packages on a host natively Debian is already installed on that machine providing needed libraries, programs and a (hopefully) intact dpkg package database.

Rootstrap provides large number of programs and libraries for the target architecture that are needed for building software and resolve dependency issues. Rootstrap is provided as a separate tarball, that user should extract under target file hierarchy.

Rootstrap needs to be configured properly before taken into use. This can be done easily by issuing **dpkg configure -a** command inside scratchbox after the tarball has been extracted. The rootstrap can be kept up to date using **apt-get update** and **apt-get upgrade** commands (inside scratchbox), just like a normal Debian system.

Rootstrap usage is thoroughly documented at scratchbox website[1]. One should also read release notes for rootstrap version to be used. Concrete example of rootstrap usage is presented later in this document at case study chapter.

3.2. Devikits

Development kits provide Linux distribution specific set of tools that are installed to ease building of the distribution. Devkits run on host system, so faster compilation times can be achieved by using them.

The Debian Devkit consists of a bunch of additional tools for scratchbox (mainly dpkg related things) and a rootstrap tar-ball, which provides a large number of programs and libraries needed to compile software for Debian.

Devkit usage is thoroughly documented at scratchbox website[1]. Concrete example of devkit usage is presented later in this document at case study chapter.

3.3. Fakeroot

Fakeroot is a Debian utility used for building packages without real root privileges. It uses a daemon that keeps track of changes made to filesystems within the fake root environment. The communication is implemented with message queues and semaphores.

Fakeroot-net is a version of fakeroot developed for Scratchbox. It uses TCP sockets for communication. It is available as an source package or as an pre-built deb package.

Source package can be found inside scratchbox environment at */scratchbox/packages*. Pre-built deb binary package can be dowloaded and installed from crocodile repository. This can be done by adding **deb http://scratchbox.org/download/files/crocodile-repository/debian crocodile main** line to the **/etc/apt/sources.list** and then issuing following commmands:

```
[sbox-ARM: ~] > fakeroot apt-get update
Hit http://scratchbox.org crocodile/main Packages
Hit http://scratchbox.org crocodile/main Release
Hit http://scratchbox.org crocodile/main Sources
Hit http://scratchbox.org crocodile/main Release
Reading Package Lists... Done
[sbox-ARM: ~] > fakeroot apt-get install fakeroot-net
Reading Package Lists... Done
Building Dependency Tree... Done
The following NEW packages will be installed:
  fakeroot-net
0 upgraded, 1 newly installed, 0 to remove and 22 not upgraded.
Need to get 0B/60.2kB of archives.
After unpacking 221kB of additional disk space will be used.
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously deselected package fakeroot-net.
(Reading database ... 9124 files and directories currently installed.)
Unpacking fakeroot-net (from .../fakeroot-net_0.9.6-1_arm.deb) ...
Setting up fakeroot-net (0.9.6-1) ...
[sbox-ARM: ~] >
```

3.4. Perl

Using dynamic modules in Perl causes problems as a i386 executable may try to use ARM libraries dynamically. This has been addressed so that perl is configured to search **/host_usr/perl** directory first.

CPAN module can be used to install perl modules to **/host_usr/perl** from CPAN repository[8]. If module to be installed needs some external libraries or headers, they should be placed into **/host_usr/perl/lib** and **/host_usr/perl/include** before compilation.

XML::Parser::Expat has been the most problematic module so far, and that's why files needed by it are provided as a separate package. If XML::Parser::Expat module is needed, following instructions inside Scratchbox will get it in the use:

```
[sbox-ARM: ~] > cd /host_usr
[sbox-ARM: ~] > tar -xzvf /scratchbox/packages/scratchbox-perl-dev.tar.gz
```

Module can be tested with simple oneliner, which should exit with no further output if everything is ok:

```
[sbox-ARM: ~] > perl -e 'use XML::Parser::Expat'
```

Read more about scratchbox's perl from */scratchbox/doc/perl.txt*

Chapter 4. Case study

4.1. scenario

In this chapter we'll see step by step how one can compile debian packages in scratchbox environment. As a pre-requirement we will assume a fresh installation of scratchbox with user account with working cpu-transparency setup[10]. GTK[9] will act as an example package, that we will try to cross-compile and install for ARM target.

- Create ARM target:

```
[sbox-HOST: ~] > sbox-config --list-compilers
arm-gcc-3.3_3.3.2ds5
host-gcc
[sbox-HOST: ~] > sbox-config --create-target=ARM --compiler-name=arm-linux-gcc-3.3_3.3.2ds
Using CPU-transparency method: qemu-arm
Completed writing configuration to: /targets/ARM.config
[sbox-HOST: ~] > sbox-config --select-target=ARM
[sbox-ARM: ~] >
```

- Take debian devkit into use:

```
[sbox-ARM: ~] > sbox-config --list-devkits
none
debian
[sbox-ARM: ~] > sbox-config --select-devkit=debian
Restarting Scratchbox shell...
Hangup
Shell restart...
Appending to /etc/passwd: daemon bin sys sync games man lp mail news uucp proxy
www-data backup operator list irc gnats nobody
Appending to /etc/group: daemon bin sys adm tty disk lp mail news uucp man proxy
kmem dialout fax voice cdrom floppy tape sudo audio dip www-data backup operator
list irc src gnats shadow utmp video sasl staff games users nogroup
[sbox-ARM: ~] >
```

- Extract rootstrap (Note: as of writing this, there is no valid rootstrap version for current scratchbox version):

```
[sbox-ARM: ~] > cd /targets/ARM
[sbox-ARM: /targets/ARM] > tar -xzf ~/rootstrap_arm_0.9.7.2-1.tar.gz
[sbox-ARM: /targets/ARM] > sbox-config --copy-libfakeroot
Copying libfakeroot from
/scratchbox/device_tools/fakeroot-net-1.0.3/arm-linux-gcc-3.3_3.3.2ds5-glibc-2.3.2.ds1/li
to /usr/lib
[sbox-ARM: ~] > fakeroot dpkg --configure -a
Setting up base-passwd (3.5.5) ...
qemu: Unsupported syscall: 149

Setting up libtool (1.5.2) ...
Installing binaries
```

```
[sbox-ARM: ~] >
```

- Update system from crocodile repository:

```
[sbox-ARM: ~] > fakeroot apt-get update
Get:1 http://scratchbox.org crocodile/main Packages [116kB]
Hit http://scratchbox.org crocodile/main Release
Get:2 http://scratchbox.org crocodile/main Sources [31.2kB]
Hit http://scratchbox.org crocodile/main Release
Fetched 147kB in 0s (171kB/s)
Reading Package Lists... Done
[sbox-ARM: ~] > fakeroot apt-get upgrade
Reading Package Lists... Done
Building Dependency Tree... Done
The following packages will be upgraded
  libpopt-dev libpopt0 slang1-utf8-dev slang1a-utf8
4 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Need to get 710kB of archives.
After unpacking 0B of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://scratchbox.org crocodile/main slang1-utf8-dev 1.4.9-2.sbox1 [350kB]
Get:2 http://scratchbox.org crocodile/main slang1a-utf8 1.4.9-2.sbox1 [293kB]
Get:3 http://scratchbox.org crocodile/main libpopt-dev 1.7-4.sbox1 [38.1kB]
Get:4 http://scratchbox.org crocodile/main libpopt0 1.7-4.sbox1 [29.0kB]
Fetched 710kB in 3s (217kB/s)
debconf: delaying package configuration, since apt-utils is not installed
(Reading database ... 34740 files and directories currently installed.)
Preparing to replace slang1-utf8-dev 1.4.9-2 (using .../slang1-utf8-dev_1.4.9-2.sbox1_arm
Unpacking replacement slang1-utf8-dev ...
Preparing to replace slang1a-utf8 1.4.9-2 (using .../slang1a-utf8_1.4.9-2.sbox1_arm.deb)
Unpacking replacement slang1a-utf8 ...
Setting up slang1a-utf8 (1.4.9-2.sbox1) ...

(Reading database ... 34740 files and directories currently installed.)
Preparing to replace libpopt-dev 1.7-4 (using .../libpopt-dev_1.7-4.sbox1_arm.deb) ...
Unpacking replacement libpopt-dev ...
Preparing to replace libpopt0 1.7-4 (using .../libpopt0_1.7-4.sbox1_arm.deb) ...
Unpacking replacement libpopt0 ...
Setting up slang1-utf8-dev (1.4.9-2.sbox1) ...
Setting up libpopt0 (1.7-4.sbox1) ...

Setting up libpopt-dev (1.7-4.sbox1) ...
[sbox-ARM: ~] >
```

- Install fakeroot-net:

```
[sbox-ARM: ~] > fakeroot apt-get install fakeroot-net
Reading Package Lists... Done
Building Dependency Tree... Done
The following NEW packages will be installed:
  fakeroot-net
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 60.2kB of archives.
After unpacking 221kB of additional disk space will be used.
```

```
Get:1 http://scratchbox.org crocodile/main fakeroot-net 0.9.6-1 [60.2kB]
Fetched 60.2kB in 0s (123kB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously deselected package fakeroot-net.
(Reading database ... 34718 files and directories currently installed.)
Unpacking fakeroot-net (from .../fakeroot-net_0.9.6-1_arm.deb) ...
Setting up fakeroot-net (0.9.6-1) ...
[sbox-ARM: ~] >
```

- Initialize perl and verify that XML::Parser::Expat works:

```
[sbox-ARM: ~] > cd /host_usr/
[sbox-ARM: /host_usr] > tar -xzf /scratchbox/packages/scratchbox-perl-dev.tar.gz
[sbox-ARM: /host_usr] > perl -e 'use XML::Parser::Expat'
[sbox-ARM: /host_usr] >
```

- Satisfy build dependencies:

```
[sbox-ARM: ~/temp] > fakeroot apt-get build-dep libgtk2.0-0
Reading Package Lists... Done
Building Dependency Tree... Done
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
[sbox-ARM: ~/temp] >
```

- Compile:

```
[sbox-ARM: ~/temp] > fakeroot apt-get source --compile libgtk2.0-0
Reading Package Lists... Done
Building Dependency Tree... Done
Need to get 10.7MB of source archives.
Get:1 http://scratchbox.org crocodile/main gtk+2.0 2.2.4-2 (dsc) [1226B]
Get:2 http://scratchbox.org crocodile/main gtk+2.0 2.2.4-2 (tar) [10.6MB]
Get:3 http://scratchbox.org crocodile/main gtk+2.0 2.2.4-2 (diff) [91.2kB]
Fetched 10.7MB in 49s (217kB/s)
dpkg-source: extracting gtk+2.0 in gtk+2.0-2.2.4
dpkg-buildpackage: source package is gtk+2.0
dpkg-buildpackage: source version is 2.2.4-2
dpkg-buildpackage: source maintainer is Sebastien Bacher <seb128@debian.org>
dpkg-buildpackage: host architecture is arm
 debian/rules clean

<clip...>
...one coffee cup later...
</clip...>

dpkg-deb: building package 'libgtk2.0-0' in '../libgtk2.0-0_2.2.4-2_arm.deb'.
dpkg-deb: building package 'libgtk2.0-common' in '../libgtk2.0-common_2.2.4-2_arm.deb'.
dpkg-deb: building package 'libgtk2.0-dev' in '../libgtk2.0-dev_2.2.4-2_arm.deb'.
dpkg-deb: building package 'libgtk2.0-dbg' in '../libgtk2.0-dbg_2.2.4-2_arm.deb'.
dpkg-deb: building package 'gtk2.0-examples' in '../gtk2.0-examples_2.2.4-2_arm.deb'.
 dpkg-genchanges -b
dpkg-genchanges: binary-only upload - not including any source code
dpkg-buildpackage: binary only upload (no source included)
[sbox-ARM: ~/temp] >
```

- Install freshly-baked packages:

```
[sbox-ARM: ~/temp] > ls
gtk+2.0-2.2.4                   libgtk2.0-0_2.2.4-2_arm.deb
gtk+2.0_2.2.4-2.diff.gz         libgtk2.0-common_2.2.4-2_arm.deb
gtk+2.0_2.2.4-2.dsc             libgtk2.0-dbg_2.2.4-2_arm.deb
gtk+2.0_2.2.4-2_arm.changes     libgtk2.0-dev_2.2.4-2_arm.deb
gtk+2.0_2.2.4.orig.tar.gz       libgtk2.0-doc_2.2.4-2_all.deb
gtk2.0-examples_2.2.4-2_arm.deb
[sbox-ARM: ~/temp] > fakeroot dpkg -i *.deb
Selecting previously deselected package gtk2.0-examples.
(Reading database ... 34740 files and directories currently installed.)
Unpacking gtk2.0-examples (from gtk2.0-examples_2.2.4-2_arm.deb) ...
Preparing to replace libgtk2.0-0 2.2.4-2 (using libgtk2.0-0_2.2.4-2_arm.deb) ...
Unpacking replacement libgtk2.0-0 ...
Preparing to replace libgtk2.0-common 2.2.4-2 (using libgtk2.0-common_2.2.4-2_arm.deb) ..
Unpacking replacement libgtk2.0-common ...
Selecting previously deselected package libgtk2.0-dbg.
Unpacking libgtk2.0-dbg (from libgtk2.0-dbg_2.2.4-2_arm.deb) ...
Preparing to replace libgtk2.0-dev 2.2.4-2 (using libgtk2.0-dev_2.2.4-2_arm.deb) ...
Unpacking replacement libgtk2.0-dev ...
Selecting previously deselected package libgtk2.0-doc.
Unpacking libgtk2.0-doc (from libgtk2.0-doc_2.2.4-2_all.deb) ...
Setting up libgtk2.0-doc (2.2.4-2) ...

Setting up libgtk2.0-0 (2.2.4-2) ...

Setting up libgtk2.0-common (2.2.4-2) ...
Updating the IM modules list for GTK+-2.2.0...done.
Updating the gdk-pixbuf loaders list for GTK+-2.2.0...done.

Setting up libgtk2.0-dbg (2.2.4-2) ...
Setting up libgtk2.0-dev (2.2.4-2) ...
Setting up gtk2.0-examples (2.2.4-2) ...
[sbox-ARM: ~/temp] >
```

Chapter 5. Conclusion

5.1. Conclusion

With debian devkit, rootstrap, fakeroot and perl development package scratchbox offers easy to use build environment for debian development. With scratchbox, one can achieve faster compile times and possibility to test packages on multiple architectures. As a result, development pace should be quicker.

Some common Linux knowledge is required, but after some trial and error scratchbox should be development enviroment of choice.

Careful and thorough study of debian documentation[7] is strongly advised.

References

[1] *Scratchbox cross-compilation toolkit website (http://www.scratchbox.org/)* .

[2] *Debian GNU/Linux website (http://www.debian.org/)* .

[3] *The Debian Free Software Guidelines (DFSG) (http://www.debian.org/social_contract#guidelines)* .

[4] *Debian Policy Manual (http://www.debian.org/doc/debian-policy/)* .

[5] *Debian Developers' Corner (http://www.debian.org/devel/)* .

[6] *Debian New Maintainers' Guide (http://www.debian.org/doc/maint-guide/)* .

[7] *DDP Developers' Manuals (http://www.debian.org/doc/devel-manuals)* .

[8] *Comprehensive Perl Archive Network (http://www.cpan.org)* .

[9] *GTK+ homepage (http://www.gtk.org/)* .

[10] *Installing Scratchbox (http://www.scratchbox.org/documentation/docbook/installdoc.html)* , Rahkonen Valtteri.