

MOVIAL

Scratchbox Apophis-r3Release Test Plan

Jussi Hakala

26th April 2006

Contents

1	Scratchbox Apophis-r3release test plan	1
1.1	Architecture and platforms	1
2	Packages included in Scratchbox Apophis-r3	1
2.1	Packages	1
2.1.1	Debian	1
2.1.2	Tarball	2
2.1.3	Source	2
2.2	Toolchains	2
2.3	Devkits	3
2.4	CPU Transparency methods (provided by devkits)	3
3	New features in Scratchbox Apophis-r3	3
3.1	Tests for new features in Scratchbox Apophis-r3	4
3.2	Custom tasks in the testing of new features in Apophis-r3	5
3.3	Installing Scratchbox	5

Revision history

Version	Author	Description
2006-03-27	Jussi Hakala	First draft
2006-03-28	Jussi Hakala	Few definitions and corrections
2006-04-05	Jussi Hakala	New naming convention
2006-04-18	Jussi Hakala	Section describing included packages in the release
2006-04-19	Jussi Hakala	Refined structure of the document
2006-04-19	Jussi Hakala	Scratchbox Apophis r3
2006-04-26	Timo Savola	Toolchain terminology disambiguated, correct deb_list task description
2006-04-26	Jussi Hakala	Correct task descriptions

1 Scratchbox Apophis-r3 release test plan

1.1 Architecture and platforms

Scratchbox Apophis-r3 will be tested using *x86* architecture. The primary platform will be *Debian sarge*, other platforms have limited support. All the tests described by the Scratchbox Release Test Suite and Scratchbox Release Test Plan will be executed in the primary platform manually. In other platforms, only a set of tests from Scratchbox Release Test suite will be executed using automated testing utilities.

List of supported platforms:

- Debian etch
- Debian sarge
- FedoraCore Heidelberg
- Gentoo 2005.1

2 Packages included in Scratchbox Apophis-r3

2.1 Packages

2.1.1 Debian

- scratchbox-core_1.0.5_i386.deb
- scratchbox-devkit-cputransp_1.0_i386.deb
- scratchbox-devkit-debian_1.0.3_i386.deb
- scratchbox-devkit-doctools_1.0.4_i386.deb
- scratchbox-devkit-perl_1.0.3_i386.deb
- scratchbox-libs_1.0.5_i386.deb
- scratchbox-toolchain-arm-gcc3.3-glibc2.3_1.0.1_i386.deb
- scratchbox-toolchain-arm-gcc3.4-glibc2.3_1.0.2_i386.deb
- scratchbox-toolchain-arm-gcc3.4-uclibc0.9.27_1.0.3_i386.deb
- scratchbox-toolchain-arm-gcc3.44csn-glibc2.3_1.0.3_i386.deb
- scratchbox-toolchain-arm-gcc4.01-glibc2.3_1.0.3_i386.deb
- scratchbox-toolchain-host-gcc_1.0.5_i386.deb
- scratchbox-toolchain-i386-gcc3.3.2-uclibc20040229_1.0.2_i386.deb
- scratchbox-toolchain-i686-gcc3.3-glibc2.3_1.0.3_i386.deb

2.1.2 Tarball

- `scratchbox-core-1.0.5-i386.tar.gz`
- `scratchbox-devkit-cputransp-1.0-i386.tar.gz`
- `scratchbox-devkit-debian-1.0.3-i386.tar.gz`
- `scratchbox-devkit-doctools-1.0.4-i386.tar.gz`
- `scratchbox-devkit-perl-1.0.3-i386.tar.gz`
- `scratchbox-libs-1.0.5-i386.tar.gz`
- `scratchbox-toolchain-arm-gcc3.3-glibc2.3-1.0.1-i386.tar.gz`
- `scratchbox-toolchain-arm-gcc3.4-glibc2.3-1.0.2-i386.tar.gz`
- `scratchbox-toolchain-arm-gcc3.4-uclibc0.9.27-1.0.3-i386.tar.gz`
- `scratchbox-toolchain-arm-gcc3.44csn-glibc2.3-1.0.3-i386.tar.gz`
- `scratchbox-toolchain-arm-gcc4.01-glibc2.3-1.0.3-i386.tar.gz`
- `scratchbox-toolchain-host-gcc-1.0.5-i386.tar.gz`
- `scratchbox-toolchain-i386-gcc3.3.2-uclibc20040229-1.0.2-i386.tar.gz`
- `scratchbox-toolchain-i686-gcc3.3-glibc2.3-1.0.3-i386.tar.gz`

2.1.3 Source

Source packages can be obtained from the repository at scratchbox.org.

2.2 Toolchains

- `arm-gcc-3.3.4-glibc-2.3.2`
- `arm-linux-cs344-2.3`
- `arm-linux-ct401-2.3`
- `arm-linux-gcc3.4.cs-glibc2.3`
- `arm-linux-gcc3.4.cs-uclibc0.9.27`
- `i386-gcc-3.3.2-uclibc-snapshot-20040229`
- `i686-linux-gcc3.3-glibc2.3`

2.3 Devkits

- cputransp
- debian
- doctools
- perl

2.4 CPU Transparency methods (provided by devkits)

- qemu-arm
- qemu-i386
- qemu-ppc
- qemu-sparc
- sbrsh

3 New features in Scratchbox Apophis-r3

In addition to the tests in the Scratchbox Release Test Suite, new features in the Scratchbox Apophis-r3 will be tested as follows

- **New GCC wrapper which allows us to use foreign (not Scratchbox specific) toolchains inside Scratchbox.**
Execute test 2.2.1 from Scratchbox Release Test Suite using toolchains arm-linux-cs344-2.3 and arm-linux-ct401-2.3.
- **Refactored libsb to make binary redirection for target binaries under QEMU behave correctly**
Execute test 2.1.1 from Scratchbox Release Test Suite using any ARM toolchain and QEMU as CPU transparency method.
- **Paths are handled correctly with Scratchbox installed to a custom location**
Execute test 2.1.3 from Scratchbox Release Test Suite using any toolchain and any CPU transparency method.
- **dlopen of host libraries now works with static host binaries**
Execute custom test 3.1.1 from Scratchbox Release Test Plan.
- **Support for symbolic links in toolchain and devkit deb_lists**
Execute custom test 3.1.2 from Scratchbox Release Test Plan.
- **It's now possible to execute target binaries through scripts with sbrsh**
Execute custom test 3.1.3 from Scratchbox Release Test Plan.
- **Support for custom provided dependencies that are target specific**
Execute custom test 3.1.4 from Scratchbox Release Test Plan.

3.1 Tests for new features in Scratchbox Apophis-r3

Test 3.1.1 *Custom test 1*

Explanation: Installation of Scratchbox, create a target using a native toolchain and compile netcat as a static binary

Test steps:

1. Install Scratchbox [task 3.1.1]
2. Create a new user [task 3.2.1]
3. Login to Scratchbox with newly created user [task 3.2.2]
4. Create a new target using a native toolchain [task 3.3.1]
5. Select the newly created target [task 3.3.2]
6. Compile netcat as a static binary [task 3.3.1]
7. Logout [task 3.2.3]

Test 3.1.2 *Custom test 2*

Explanation: Installation of Scratchbox, create a target using a non-native toolchain, move deb_list directory to a custom location, provide a symbolic link in the original location and verify that the provided dependencies are still seen by the dpkg

Test steps:

1. Install Scratchbox [task 3.1.1]
2. Create a new user [task 3.2.1]
3. Login to Scratchbox with newly created user [task 3.2.2]
4. Create a new target using a non-native toolchain [task 3.3.1]
5. Select the newly created target [task 3.3.2]
6. Move deb_list directory to a custom location and provide a symbolic link there in the original location [task 3.3.2]
7. Logout [task 3.2.3]

Test 3.1.3 *Custom test 3*

Explanation: Installation of Scratchbox, create a target using a non-native toolchain and sbrsh, create and execute a script which executed target binaries

Test steps:

1. Install Scratchbox [task 3.1.1]
2. Create a new user [task 3.2.1]

3. Login to Scratchbox with newly created user [task 3.2.2]
4. Create a new target using a non-native toolchain and sbrsh as CPU transparency method [task 3.3.1]
5. Select the newly created target [task 3.3.2]
6. Create and execute a script which executes target binaries [task 3.3.3]
7. Logout [task 3.2.3]

Test 3.1.4 *Custom test 4*

Explanation: Installation of Scratchbox, create a target using a non-native toolchain, create custom provided dependencies for the target and verify they are seen by the dpkg
Test steps:

1. Install Scratchbox [task 3.1.1]
2. Create a new user [task 3.2.1]
3. Login to Scratchbox with newly created user [task 3.2.2]
4. Create a new target using a non-native toolchain [task 3.3.1]
5. Select the newly created target [task 3.3.2]
6. Add target specific, Scratchbox-provided dependencies [task 3.3.4]
7. Logout [task 3.2.3]

3.2 Custom tasks in the testing of new features in Apophis-r3

3.3 Installing Scratchbox

Task 3.3.1 *Compile netcat*

Explanation: Compiling netcat

Requirements: 3.1.1 or 3.1.2 or 3.1.3 or 3.1.4 and 3.2.1 and 3.3.1

Task steps:

1. Download netcat source from <http://netcat.sourceforge.net/>
2. Run configure and make

How to determine if the task was passed:

- Configure and Make processed are completed successfully

Task 3.3.2 *Use symbolic link for deb_list*

Explanation: Moving `deb_list` directory to a custom location and providing a symlink in the original location

Requirements: 3.1.1 or 3.1.2 or 3.1.3 or 3.1.4 and 3.2.1 and 3.3.1

Task steps:

1. Move `/scratchbox/devkits/debian/deb_list` to `/host_usr/debian_deb_list`
2. Create symlink `/scratchbox/devkits/debian/deb_list` pointing to `/host_usr/debian_deb_list`

How to determine if the task was passed:

- Verify that `/scratchbox/devkits/debian/deb_list` exists, is a symlink and points to `/host_usr/debian_deb_list`

Task 3.3.3 *Execute target binaries from a script*

Explanation: Executing target binaries from within a script using `sbrsh`

Requirements: 3.1.1 or 3.1.2 or 3.1.3 or 3.1.4 and 3.2.1 and 3.3.1

Task steps:

1. Create a script which executes target binaries
2. Execute the script

How to determine if the task was passed:

- Binaries executed from within the script execute normally

Task 3.3.4 *Add a target specific deb_list*

Explanation: Adding target specific dependencies

Requirements: 3.1.1 or 3.1.2 or 3.1.3 or 3.1.4 and 3.2.1 and 3.3.1

Task steps:

1. Create a directory `MYTARGET_deb_list` in `/targets`
2. Add files in the directory which describe Scratchbox provided dependencies for the target (use `deb_list` from the Debian devkit as an example)

How to determine if the task was passed:

- Added dependencies are seen by the `dpkg`