MOVIAL

Scratchbox ApophisRelease Test Plan

Jussi Hakala

5th April 2006

Contents

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

1 Scratchbox Apophisrelease test plan

1.1 Architecture and distributions

Scratchbox Apophiswill be tested using x86 architecture as the host architecture. The primary distribution will be *Debian Sarge*. All the tests described by the Scratchbox Release Test Suite, will be executed in the primary distribution manually.

Additionally, the following distributions will have limited support:

- Debian Etch
- Fedora FC 3
- Gentoo
- RedHat Enterprise Linux 3.0 WS
- Ubuntu 5.10 Breezy Badger

In these distributions, the test will be done using automated testing utilities that perform the test functions specifed in the Scratchbox Release Test Suite. Only a limited set of all the tests will be executed in the distributions with limited support.

1.2 New features in Scratchbox Apophis

In addition to the tests in the Scratchbox Release Test Suite, new features in the Scratchbox Apophiswill be tested as follows

- **New GCC wrapper which allows us to use foreign (not Scratchbox specific) toolchains inside Scratchbox.**
  Create a target using a foreign toolchain and compile packages essential for a Debian environment using Crocodile.
- **Refactored libsb to make binary redirection for target binaries under QEMU behave correctly**
  Create a target using an ARM toolchain and QEMU. Run environment test suite.
- **dlopen of host libraries now works with static host binaries**
  Compile ping, netcat, etc. as a static host binary to test the dlopen of host libraries.
- **Paths are handled correctly with Scratchbox installed to a custom location**
  Install Scratchbox to a custom location. Verify that the installation is working normally.
- **Support for symbolic links in toolchain and devkit deb_lists**
  Move deb_list directories to a different location and provide a symlink in the original location. Check that the dependencies are found correctly.
- **It's now possible to execute target binaries through scripts with sbrsh**
  Create a script which executes target binaries and check if it works correctly.

- **Support for custom provided dependencies that are target specific**
  Add dependecies to `/targets/MY_TARGET_deb_list/` and test if APT sees them.