Fuego Test System Status Update
June 2018
Tim Bird
Fuego Maintainer
Sony Electronics
Outline

Introduction
Status
Vision
Using Fuego
Micro-Introduction

Fuego =
(Fuego Linux distribution +
host scripts +
pre-packaged tests +
Jenkins)
all inside a container
Architecture Diagram

Host machine:
- Container build system

Docker container:
- Fuego Linux distribution
- Jenkins
- Test programs
- Scripts

Volume Mount

Toolchains
Config
Builds
Logs

Web control interface

Target board
Core features

- Distribution of Linux for testing
- Build system
  - Architecture-neutral & inherently cross-platform
- Includes a collection of tests
  - Source for test program
  - Scripts for test orchestration, results parsing, analysis, and visualization
- Report generation
- Multiple transports (ssh, serial, ...)
- Jenkins front end
  - Also has a command line tool
Outline

Introduction
Status
Vision
Using Fuego
Release Status

- Lots of “big” features in 1.2 release
  - Released October 2017
  - Focus on consolidating the test APIs
- Recently released version 1.3.0 ("Draft")
  - Released May 2018
  - Incremental features
- Already working on 1.4 features
Version 1.2 Features

• Unified Output Format
• Test dependency system
• Complex pass criteria handling
• Dynamic board variables
• Charting improvements
• Get test program source from git repositories
• Test improvements
Version 1.2 Features

- Unified Output Format (*run.json*)
- Test dependency system
  - *(NEED_vars, and test_pre_check)*
- Complex pass criteria handling (*criteria.json*)
- Dynamic board variables (*<board>.vars*)
- Charting improvements (*chart_config.json*)
- Get test program source from git repositories
- Test improvements
Version 1.3 Features

- Report generation improvements
- Log splitting by testcase
- New tests
  - Including Fuego self-tests
- Web page and image comparison tools
- Infrastructure enhancements
  - Hardware board control
  - Individual test phases
  - ftc outside the docker container
Report generation improvements

- More output formats
  - html, rst, pdf, excel, csv
- Control of fields displayed
  - header_fields
  - report fields
- More filtering (--where options)
  - especially tguid:result
  - Try this:
    - ftc gen-report –where “tguid:result!=PASS”
Log splitting by testcase

- Ability to split the test log into pieces, according to testcase boundaries
- Only works for some logs
- Requires slight modification to test’s parser.py
- Addition to UI
  - Can click on testcase in Jenkins UI, and see section of log related to that testcase
- Very handy for examining results details
split log by
testcase
screenshot
UI
rm: cannot remove `./test_dir`: Permission denied
'ls' -> 'test_dir/bin/ls'
lib/x86_64-linux-gnu/libc.so.6
lib/x86_64-linux-gnu/libpcre.so.3
lib/x86_64-linux-gnu/libssl.so.2
lib/x86_64-linux-gnu/libthread.so.0
 copying lib/x86_64-linux-gnu/libc.so.6 to chroot area
 lib_dirname=/lib/x86_64-linux-gnu
 asking lib directory test_dir/lib/x86_64-linux-gnu
 '/lib/x86_64-linux-gnu/libc.so.6' -> 'test_dir/lib/x86_64-linux-gnu/libc.so.6'
 copying lib/x86_64-linux-gnu/libpcre.so.3 to chroot area
 lib_dirname=/lib/x86_64-linux-gnu
 asking lib directory test_dir/lib/x86_64-linux-gnu
 '/lib/x86_64-linux-gnu/libpcre.so.3' -> 'test_dir/lib/x86_64-linux-gnu/libpcre.so.3'
 copying lib/x86_64-linux-gnu/libssl.so.2 to chroot area
 lib_dirname=/lib/x86_64-linux-gnu
 asking lib directory test_dir/lib/x86_64-linux-gnu
 '/lib/x86_64-linux-gnu/libssl.so.2' -> 'test_dir/lib/x86_64-linux-gnu/libssl.so.2'
 copying lib/x86_64-linux-gnu/libthread.so.0 to chroot area
 lib_dirname=/lib/x86_64-linux-gnu
 asking lib directory test_dir/lib/x86_64-linux-gnu
 '/lib/x86_64-linux-gnu/libthread.so.0' -> 'test_dir/lib/x86_64-linux-gnu/libthread.so.0'
 chroot: can't change root directory to 'test_dir': Operation not permitted
 --> chroot: TEST-FAIL
New tests

- Realtime benchmarks:
  - backfire, deadlinetest, migratetest, pmqtest, ptsemaptest, sigwaittest, svsemaptest

- Other tests (updated or new):
  - dbench4, dd, iperf3, vuls, autopkgtest, year2038
  - LTP_one_test
  - fuego_board_status

- Fuego self-tests:
  - fuego_lint, fuego_tguid_check, fuego_ftc_check
  - fuego_release_test
Fuego release test

- Complicated test to do a full release test
  - Builds docker container
  - Runs docker container for “release under test”, alongside “test-runner” container
- Checks Jenkins web interface
  - Using HTML DOM element checks
  - Using comparisons of web page rendered images
- Adds capabilities to Fuego distribution for testing of other DUT web or image features
Web page and image comparison tools

• Added Selenium to Fuego distribution
  • For web page testing
• Added Chromium to Fuego distribution
  • For web page rendering automation
• Added tools for:
  • Comparison of returned HTML
  • Web page image capture
  • Image comparison
    • With support for masked regions
• Note: This is not generalized yet
  • Need to read Functional.fuego_release_test scripts and use as example
Hardware board control

- General feature is ability to control board under automation
  - Added in 1.3:
    - Hook for hardware board reboot
    - Shows method for adding board control hooks
  - Goal is to support provisioning and other hardware functions, as well
    - ex. off-DUT test hardware control and multiplexing
- Would rather re-use some other board control layer
  - e.g. LAVA, labgrid, Dryad (from SLAV), ttc, etc.
Individual test phases

• Ability to run test phases individually
  • Main purpose is to allow separation of:
    • Test program build (on host)
    • Test execution on target
• Can use ftc option:
  • ftc run-test –p ‘pcb’
    • Executes pre_test, pre_check and build phases, then stops
• Can use environment variable
  • FUEGO_TEST_PHASES=“pre_test pre_check build”
ftc outside the docker container

- docker adds unneeded overhead to some commands
- Some commands can now be run outside the docker container
  - New fuego.conf file to specify directory locations
  - list-runs, gen-report can be done directly on host
- Partial step towards use of low-level Fuego functionality with alternate UIs and frameworks
Missing from 1.3...

- Things that slipped from the 1.3 release:
  - Documentation conversion
  - LTS Provisioning support
  - Pre-built docker container
Documentation conversion

- Conversion of docs to reStructuredText
  - Replace PDF and wiki docs with rst
  - Move all docs under source repository
  - Use sphynx to create multiple formats
  - Publish on readthedocs.io

- Made some progress
  - Have sphynx templates in place

- Got stuck on markup conversion
  - Considered automation, but hit some hurdles

- See http://fuegotest.org/wiki/rst_docs
LTS Provisioning support

- Provisioning
  - Ability to provision board with new system software (particularly the kernel)
  - Fuego historically has left this as an exercise for the user
- Did some work on this in my lab
  - usb keyboard automation
    - teensy-usb – host-controlled keyboard for target
  - LTS download and build
  - Ubuntu kernel replacement
  - Haven’t generalized the feature
    - Some support was put into ttc
Pre-built docker image

- Ability to use Fuego without building the docker image
  - Create a pre-built Fuego docker image, and host it at docker.io
  - e.g. “docker run fuego”
- Requires automatic container customization
  - Network proxy
  - User and group
  - Volume mount customization
- Includes refactoring the Fuego directory layout
  - Turned out to be too intrusive for 1.3 release
Miscellaneous

• Contributor guidelines
• Events:
  • Fuego Jamboree #2
  • Automated Testing Summit
Contributor guidelines

- Recently added to wiki
- Coding style guide
  - Mostly indentation (4 spaces, no tabs)
  - See http://fuegotest.org/wiki/Coding_style
- License guide
- Patch submission tips
- See http://fuegotest.org/wiki/License_And_Contribution_Policy
Fuego Jamboree #2

• Saturday, June 23, 9:00 to 12:00
• Ariake, Tokyo, Japan
• Hosted by Panasonic
• Details at:
  • http://fuegotest.org/wiki/Fuego_Jamboree_2
  • Please add your name to attendee list, if you plan to come
    • May have to create wiki account, but it’s quick and free
Automated Testing Summit

- October 25, Edinburgh Scotland
  - See http://elinux.org/Automated_Testing_Summit
- Sponsored by Linux Foundation Core Embedded Linux Project
- Attempt to assemble wide variety of Linux test stakeholders and practitioners
  - Register interest by adding name to list on:
    - http://elinux.org/Automated_Testing_Summit
Outline

Introduction
Status
Vision
Using Fuego
Vision – super high level

Do for testing what open source has done for coding

• Significant parts of the test process are unshared, ad hoc, private, etc.
  • However, most QA doesn’t need to be proprietary
• Fuego Goal:
  • Promote the sharing of tests, test methods, and results, the way code is shared now
  • Make it easy to create, share and discover tests
  • Make test results easy to share and evaluate
Core principles

• Useful
  • Actually find bugs or prevent regressions

• Scales
  • Allow sharing
  • Usable by wide audience
    • Minimal requirements
    • Customizable
    • Easy to use

• Modular

• Applicable to embedded
Outline

Introduction
Status
Vision
Using Fuego
Using Fuego

- What can Fuego do?
- How to use it yourself
- Resources
What can Fuego do?

- Fuego includes about 110 tests
  - About 40 benchmarks
  - Includes LTP, which has thousands of testcases, in 3 main groups:
    - realtime
    - POSIX
    - kernel system call
  - Includes kselftest
- Provides Jenkins for continuous integration management
  - schedule jobs, view results, watch trends, notify of results
  - Benchmarks include automatic plotting and threshold checking
How to get started

- **Startup steps:**
  - Download Fuego
  - Build container
  - Configure your board and toolchain
  - Execute tests (using Jenkins web interface)

- **Detailed instructions at:**
Customizing Fuego

- **Configure**
  - Add your board and toolchain
  - Set up your board provisioning

- **Customize**
  - Create test plans (which tests to run)
  - Adjust test variations (specs)
    - or add your own
  - Adjust pass criteria for your board
  - Configure Jenkins job triggers and notifications

- **Add**
  - Add your own tests
What you provide

- Your device, with your software (Linux distro)
- Board configuration
  - How to communicate with board
  - How to control the board
    - (Thinking about support for semi-automation)
- Toolchain
  - SDK used to build software for the board
Want to convert QA to an Open Source (collaborative/shared) activity

Can share now in Fuego:
- board definitions
- tests
- pass criteria
- test plans
- test specs (test variations)

Intend to share in future:
- tests (via test store)
- test results
- your test hardware (via distributed lab)
Want more information about testing?

- More sessions today at ALS/OSSJ
- LTS/LTSI workshop
  - Friday, 10:30 – 12:00
  - Room 2
Resources

• Fuego web server:
  • http://fuegotest.org/>
  • wiki: http://fuegotest.org/wiki
• Mailing list:
  • https://lists.linuxfoundation.org/mailman/listinfo/fuego
• Repositories:
  • https://bitbucket.org/tbird20d/fuego
  • https://bitbucket.org/tbird20d/fuego-core
Fuego
Abstract

The Fuego test system is used for testing LTS and some aspects of AGL. In this presentation, Tim will describe the current status of the project, and how it can be used by independent developers to test their Linux systems. Fuego version 1.3 includes additional tests, and new features such as enhanced report generation, more detailed testcase results display, and board control features.

Come see Fuego's latest features, to see if it can be useful in your own Linux projects.