CE Workgroup

Fuego
Status and Roadmap
BOF

Tim Bird
Architecture Group Chair
LF CE Workgroup
Fuego
Status and Roadmap
BOF

Tim Bird
Architecture Group Chair
LF CE Workgroup
Micro-Introduction

Fuego = (Jenkins + abstraction scripts + pre-packaged tests) inside a container
Architecture Diagram

Host machine:

Container build system

Docker container:

Jenkins
Test programs
Scripts

Volume Mount

Toolchains
Config
Builds
Logs

Web control interface

Target board

4/10/23/2014 PA1
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.
  - For no good reason – most QA doesn’t need to be proprietary
  - There are OSS frameworks and test programs but parts are missing to create a testing community.
- Promote the sharing of tests, test methods, and test results, the way code is shared now
  - Make it easy to create, share and discover tests
  - Make test results easy to share and evaluate
Goals

- Allow quick and easy setup
- Support a wide variety of configurations and build systems
  - Yocto Project/OE, Buildroot, etc.
- Support a wide variety of target types:
  - Transports: serial, ssh, adb, ttc
  - Distributions: Debian, Angstrom, Poky, custom
- Send data to centralized repository
- Make it possible to join a decentralized test network
  - Allow a developer to use my hardware for testing
  - Help solve the “developer can’t test on different hardware” problem
Outline

Status
Projects
Vision
Roadmap
Status

• 3 main forks
  • Sony fork
  • Toshiba fork
  • AGL fork
• What features does each have?
Sony Fork

- In ‘next’ branch
  - Ftc – Fuego test control – command line tool
  - Test package system
    - Introduction of test package format (yaml file)
  - Client side of test server system
    - Test requests, test runs, packaging, install, running
  - New transports:
    - Serial, ttc

- Work in progress
  - Need_xxx system (test dependencies)
Toshiba Fork

- Refactored Jenkins integration
  - Uses minimal Jenkins plugins
  - Runs on latest Jenkins
- Refactored directories
  - Puts fuego-core outside the docker container
    - Much easier for development
  - Reduce symlink confusion
- Output results to excel file
AGL Fork

- Focused on LAVA integration
- Using latest Jenkins
- Test categories?
- Some reporting features??
Feature list

- Jenkins integration
- Containerization
- Overlay generation
  - Boards, distros, specs, plans
- Script system
- Transports
- Test collection
- Results parsing and post-processing
- Fuego server
Jenkins integration

- Toshiba fork simplifies integration
- ftc CLI allows Fuego use independent of Jenkins
- Want to use latest Jenkins
- Using a simple text template now
  - Maybe use Jenkins Job Builder to handle job and node creation?
- Add fuego-install-xxx commands to ftc?
- Handling dynamic parameters?
  - Target_cleanup, Reboot, Rebuild, Testplan
Overview of test framework landscape

- **KernelCI**
  - kernel build and boot
  - Multi-lab, client/server, results query

- **LAVA**
  - Board management, test scheduling

- **Fuego**
  - Host/target testing, jenkins integration
  - Docker container, test building, collection of tests

- **Jenkins**
  - Triggers, user interface, jobs, nodes, scheduling

- **Avocado**
  - Results processing, test server, matrix testing
# Comparison of Fuego and Lava

<table>
<thead>
<tr>
<th>Assumption</th>
<th>Fuego</th>
<th>LAVA</th>
<th>Jenkins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Board starting status</td>
<td>Board is running</td>
<td>Board will be provisioned and booted</td>
<td>Node is running</td>
</tr>
<tr>
<td>Test initiated by:</td>
<td>Manual, Jenkins trigger</td>
<td>External job insertion?</td>
<td>Jenkins trigger</td>
</tr>
<tr>
<td>Test software availability:</td>
<td>Source included, test binary is built and deployed to target</td>
<td>Is in distro or on target, or is installed during test</td>
<td>Builds software – no built-in deploy - left as exercise for test developer</td>
</tr>
<tr>
<td>Test scheduling</td>
<td>By Jenkins, cli has none, no target reservation system</td>
<td>By LAVA</td>
<td>By Jenkins</td>
</tr>
<tr>
<td>Results processing</td>
<td>Log parsing, send results to server (prototype)</td>
<td>Collect results?</td>
<td>Visualization for common formats (TAP, junit, xunit)</td>
</tr>
</tbody>
</table>
Feature list

- Containerization
- Overlay generation
- Script system
- Transport abstraction
- Test collection
- Results parsing and post-processing
- Command line
- Fuego server
Containerization

- Docker container builds OK
- Encountered issues with permissions for serial, usb (adb) operations
  - Docker-create-usb-privileged-container.sh
- Use of Fuego without container??
  - I had considered running outside docker, especially for tests with no build step (eg. Functional.bc)
    - To make host software lighter-weight
  - But there’s lots of installation dependencies even without builds
  - Reproducible builds really benefits from a container
Overlay generation

- Boards, distros, specs, plans
- Need to adopt following:
  - Spec = test variant (test with different options)
  - Plan = set of tests
  - Distro should move to board file, IMHO
- We’re not using overlays to inherit from board types (like LAVA device types)
  - But we could
    - Eg: TimsPi:
      - Inherit RaspberryPi
      - IP_ADDR=10.10.1.8
Script system

- Core script system seems OK
- Overlays seems like overkill
  - Could this just be shell sourcing?
- Have been working to simplify specs and plans
  - Testplan_default is now implied
    - No need for each test to add something to a central plan on installation
Transports

• Added support for serial
  • About 80% there
  • Most tests run – there are a few remaining issues
    • Benchmark.reboot, target_reboot
    • ov_transport_connect, ov_transport_disconnect

• Added support for ttc

• Still want ‘adb’

• Does ‘lava2’ support go here?
  • I think this is where the LAVA integration is in AGL
Test collection

- This is the biggest value of the system eventually (IMHO)
- Not many new tests
- Currently have about 20 “useful” tests
- Want hundreds in lots of areas
  - Filesystem, networking, realtime, power, boot time, size, security, apis, utilities, specific sub-systems, hardware (drivers)
- Have been working on test package formatting and infrastructure first
- It is important not to delay too long actually making tests
Pre-packaged tests

- Comes with over 50 tests, already integrated
  - aim7, blobsalad, bonnie, cyclitest, dbench, dhrystone, ebizzy, ffsb, fio, GLMark, gtkperf, hackbench, himeno, Interbench, IOzone, iperf, Java, linpack, Imbench2, nbench, netperf, netpipe, OpenSSL, reboot, signaltest, Stream, tiobench, whetstone, x11perf, aiostress, arch_timer, bzip2, cmt, crashme, expat, fontconfig, glib, ipv6connect, jpeg, libpng, linus_stress, LTP, netperf, posixtestsuite, rmaptest, scifab, scrashme, sdhi_o, stress, synctest, zlib

- Includes functional, benchmark and stress tests
Results parsing and post-processing

- Log_compare, parser.py, flot charts
- FUNCTIONAL_LTP_POS,NEG
  - Number of expected successes and failures, for large tests
  - Kind of a cop-out
- Diff against reference log
  - This is a feature not being utilized
  - Needs easy tool to do reference log capture
    - Cogent had awk scripts
  - Needs smartdiff
    - To filter timestamps and other variations that don’t matter
Command line

- \texttt{ftc <verb>-<object> <args>}
  - list-targets, query-target, get, set <values>
  - list-requests, put-request, run-request
  - list-tests, package-test, put-test, install-test, run-test
  - list-runs, package-run, put-run

- Proposed:
  - install-target, put-target
  - query-test
  - query-run
Fuego server

- **Distributed test coordinator**
  - So developer can do following workflow:
    - Create a test, publish to server, wait, collect results
  - Client node would:
    - Poll for test requests, match to local hardware
    - Download request (and test, if needed) and execute it
    - Push results

- **Future/vision:**
  - Intended to be a repository of hundreds of tests
    - “Test store” – like an app store
    - Choose the tests you like, and install them locally
  - Allow individuals to access a wide variety of nodes with different features
    - “need_xxx” system is key to matching tests to targets.
Outline

Status
Projects
Vision
Roadmap
Projects underway

- Fuego command line tool
- Jenkins integration refactoring
- Directory/link cleanup
- Test packaging system
- Test dependency system
- LAVA integration
Requested/Desired features

- Allow fuego use with other CI tools (Siemens)
- Make development and release management easier
  - Refactor Jenkins integration
- Fix bugs
  - reboot test, container issues, etc.
- Execute individual test phases
- Create & submit LAVA v2 jobs and post-process results (Kevin Hilman)
- Better test scheduling
  - Board reservation feature?
To-Do from Daniel

- Cleanup unneeded stuff
  - Overrides: the pretest should be able to automatically select
    - Otherwise override on the board file instead of selecting the dist?
- Keep it simple
  - Minimum requirements (sh + serial or ssh + writable folder/tmpfs)
  - It's our main advantage over other test frameworks
  - Simplistic testing
    - Ref: https://validation.linaro.org/static/docs/v2/simple-admin.html#index-0
To-Do from Daniel (cont.2)

- Provide deploy and boot as in LAVA
  - Deploy: prepare nfs/tftp
  - Boot: poweron board/reboot/ssh
- Transports:
  - ADB support
  - Serial port support
    - Use pexpect through
- Updating/deploy the OS
  - 1) hawkbeat/ostre... also tests the updates
  - 2) u-boot serial port with pexpect
  - 3) TFPT/NFS or NBDroot
  - 4) Fastboot
To-Do from Daniel (cont.3)

- Common output format:
  - TAP, junit, xunit, ... (which one??)
  - Allows for custom reports (excel)
- Parallel testing on same device types
  - Use Jenkins labels
- Multi-node tests like in LAVA
- Auto-generate timeouts
- Ability to run tests already in the target
- Autodiscovery of binary path for the run step
  [is_on_target()?]?
- Automatically prepare TFTP/NBDroot before testing
Support matrix of boards/tests
  • Fuzz coverage combinations
Command-line fuego tool [ftc??]
  • Similar to "avocado"
Create an interface to download and install and list new tests [ftc get-test, install-test?]
  • Tests in GIT (no tarballs)
  • Ability to specify tag or commit per test in the testplan (by default latest)
  • Plugin system like avocado
Bisects
Kernel CI integration
To-Do from Daniel (cont.5)

- LAVA support
  - Just open a hacking shell?
  - Or submitting YAML jobs?
- REST API instead of master-slave model
- Test preChecks in the YAML file [need_xxx?]
- Support for read-only filesystems
  - Create a ramfs?
- Support for including strace output or running gdb remotely
- Ability to deploy standard distributions (for testing the kernel, hardware, or apps!)
  - Yocto based generic filesystem
  - Debian, others
To-Do from Daniel (cont. 6)

- Allow to enter easily into a developer shell
  - $ fuego shell
- Update filesystem on the SD card by using update software
  - 2 partitions
- Login
  - support user, root password, ssh key [?]
- Jenkins-based test framework
- Testing as a service
  - https://bugzilla.redhat.com/show_bug.cgi?id=334411
  - https://gist.github.com/pklaus/319367
  - http://downloadmirror.intel.com/20927/eng/e1000.htm
  - https://github.com/kernelci/lava-ci
To-Do from Daniel (cont. 7)

- Tests:
  - RT-tests
    - LTP
    - rt-tests
    - rt-eval (disturbance)
  - Kselftests support
  - Software update tests
- Disturbance loads
  - stress, hackbench, ...
  - Power cut tests
    - target_poweroff/poweron
  - Simulate application environment.
Test dependency system

- Modeled on 0-day system
- Uses declaration in base script:
  - NEED_KCONFIG_FOO=y
  - NEED_MEMORY=8M
  - NEED_WIRELESS_NETWORK=y
  - NEED_HARDWARE_FOO=y
  - NEED_PROGRAM_ETHTOOL=y
- Board provides items:
  - Some built into Fuego: kconfig, memory, sysfs
  - HAS_PROGRAM_ETHTOOL=/usr/bin/ethtool
- Probe tests can determine if a target has a requirement, and populate the board file with them
  - Using get/set
- Can use to filter tests which are appropriate for target
Outline

Status
Projects
Vision
Roadmap
Vision

- Already covered.
Outline

Status

Projects

Vision

Roadmap
Roadmap

- Process issues
- What technology to leverage
  - LAVA
  - Avacado
  - Jenkins (what plugins to require)?
- What are the priorities?
  - Tim’s view: anything affecting test API or test packaging
    - Unified output format – may affect parsing API
- What tests to tackle next?
  - Move past the generic tests
Process

- Need to merge efforts
  - unfork the forks
- Need more real-time communication
  - Monthly conference call?
  - Use the AGL-CIAT call?
  - Fuego mini-conference (again)?
Priorities

- Unfork the Sony and Toshiba forks
- Features needed for LAVA integration
Resources

- http://bird.org/fuego/FrontPage
- Fuegotest.com
  - No domain name yet:
    - use http://52.88.166.49/server/FrontPage
  - Need to confirm name
  - Prototype server – very rough at the moment
- Daniel’s server
- Repositories:
  - Sony: bitbucket.org/tbird20d/fuego,fuego-core
  - Toshiba: bitbucket.org/
  - AGL:
Notes from BOF

- **ADB support**
  - Run `adb` outside container (on host), and container doesn’t have to know about USB changes

- **Could use transport=local for host as DUT**
  - Now currently used for docker container as DUT

- **Bypassing build step**
  - It’s OK to have something as a build cache, but make sure not to lost ability to build from source
  - Don’t allow “magic binaries” that someone can’t rebuild
Notes from BOF (2)

- **Bisect**
  - Should be a tool outside Fuego to bisect based on Fuego test result
  - Ftc needs to return proper error code
  - Maybe provide an example for how to do it

- **Image Deploy, re-flash**
  - Since LAVA does these, and AGL already uses LAVA, these are not high priority at the moment
Fuego

It's hot!
Overlay processing

Base script
- test-script.sh
  - test_build()
  - test_deploy()
  - test_run()

Functional script
- functional.sh
- functions.sh
- common.sh
- overlays.sh
- reports.sh
- etc.

Extended script
- \(<target>\)_prolog.sh

Base script dependencies:
- <board>.conf
- tools.sh
- testplan
- test specs
Miscellaneous

- HealthCheck test
  - Ftc target-status
- Automatic board installation /wizard
  - ftc find-board
- Use ftc in Jenkins, instead of direct invocation of base-script for test
  - This is what avocado does
Next Steps (old)

- De-clutter the Jenkins front end
- Improve documentation
- Handle USB connections
  - For ADB-based targets
  - For Sony debug board
Next Steps (cont.)

- More tests
  - kselftest
  - kernelci ??
  - Look for a vertical to build out the test suite
- Send results to a centralized repository