How to integrate Fuego automated testing tool in your CI loop

Daniel Sangorrin
Toshiba Corporation
Software Engineering and Technology Center
Open Source Technology Dept.
01

Background
“What will you do without freedom? Will you fight?!”, William Wallace (Braveheart)
Automated Testing Summit

- October 25, 2018 (Edinburgh)
  - https://elinux.org/Automated_Testing_Summit
- Glossary
  - https://elinux.org/Test_Glossary
- Test stack survey
  - https://elinux.org/Test_Stack_Survey

“Let’s try each others’ testing tools and figure out how to collaborate”, Michal Simek (Xilinx)
In this section, we will look at Fuego as a black box and we will learn how to combine it with other testing tools.
Host-target configuration

- **myboard.conf**
  - IP=192.168.2.3
  - LOGIN=root
  - PASS=3sd$_dfas$
  - TRANSPORT=ssh
  - TOOLCHAIN=armhf

- **ftc** → **Fuego** → **myboard**

- **PC**
ftc: Fuego command line tool
ftc: Fuego command line tool

$ ftc help
$ ftc list-boards
$ ftc query-board -b vm -n IPADDR
$ ftc list-tests
$ ftc run-test -b vm -t Functional.hello_world -s hello-fail
$ echo $?
$ ftc gen-report

Supports autocompletion
Integration 1: Jenkins + Fuego (default)

Jenkins role

- Web server interface including user accounts etc
- Test serialization (only one test at a time for each board)
- Display results (logs, plots, tables)
- Test triggers
- Notifications
- Plugins
- Board scheduling (using Jenkins labels)

Jenkins is installed by default with Fuego unless you use ./install.sh --no-Jenkins
TOSHIBA

DEMO

Fuego with Jenkins
Fuego with Jenkins

$ ftc list-boards
$ ftc add-nodes -b vm
$ ftc list-tests
$ ftc add-job -b vm -t Functional.hello_world
$ ftc list-specs -t Functional.hello_world
$ ftc add-job -b vm -t Functional.hello_world -s hello-fail
$ ftc add-job -b vm -t Benchmark.Dhrystone
$ ftc list-specs -t Benchmark.Dhrystone
$ ftc add-job -b vm -t Benchmark.Dhrystone -s 500M
$ ftc build-jobs "vm.*.Functional.hello_world"

- Trigger remaining jobs from Jenkins
- Show graphic tables, timecharts, testlog, run.json..
- Show that Jenkins calls ftc by clicking configure
Fuego & Ktest

- **Ktest role**
  - Build the Linux kernel
  - Deploy the kernel into the target board
  - Execute a test
    - Boot test
    - Custom test \( \Leftarrow \) Fuego test
  - Patchcheck
  - Bisect
Fuego with Ktest

$ ./init (only once)
$ ./up
$ less ktest/ktest.conf
$ vi /fuego-ro/boards/vm.board (ajust ip, sshkey to ktest.conf’s)
$ less examples/test/fuego-hello
$ ./test test examples/test/fuego-hello
$ ./halt
$ ./fini (only once to destroy the environment)
Fuego with Gitlab CI

Fuego

src

.gitlab-ci.yml

Gitlab Runner

Docker / VM

Source code repository

git push

src

.gitlab-ci.yml
TOSHIBA

DEMO

Fuego with Gitlab CI
Fuego with Gitlab CI

- **Show Fuego.zlib2 repository**
  - `.gitlab-ci.yml`: packages the test (tarball) and deploys it as an artifact

- **Show zlib2 repository**
  - `.gitlab-ci.yml`: show how it works

- **Ideal behavior**
  - `.gitlab-ci.yml`: downloads and installs the test package file into fuego
  - Runs the test against the latest source code
  - Inter-project triggers require premium account

- **Modify the code to introduce a bug**
  - Git push origin master
  - Triggers the CI loop
Fuego & LAVA (Using ssh hacking session)

• Implemented by Jan-Simon Moeller
• Fuego board files can have 2 variables
  • TARGET_SETUP_LINK=fuego-lava-target-setup
  • TARGET_TEARDOWN_LINK=fuego-lava-target-teardown
• fuego-lava-target-setup (bash script)
  • Prepares a LAVA job (yaml file)
    • Deploy: dtb, kernel,initrd,nbdroot
    • Boot: autologin
    • Test: hacking-session-oe.yaml
  • Submits the job and polls for boot complete
  • Checks that SSH is working and hands control over to Fuego
• fuego-lava-target-teardown (bash script)
  • Executes “stop_hacking” and “lava-tool cancel-job”
Fuego & LAVA (Using LXC/Docker on LAVA slave)

- Implemented Qi Chase
  - [https://github.com/Linaro/test-definitions/tree/master/automated/linux/fuego-multinode](https://github.com/Linaro/test-definitions/tree/master/automated/linux/fuego-multinode)
Fuego & LAVA (Using native installation)

- Prepare the target image with Fuego installed
  - Debian image build systems: ISAR, meta-eid, Debos, ...
- Use `ftc run-test -b local`

```
      LAVA master
         ↓
        LAVA slave
           ↓
          NFS rootfs
             ↓
          Fuego on Debian
               ↓
            myboard
```
Fuego & LAVA (native)
- Show how the image was created in Debos
  $ less iwg20m_debos/metadata/iwg20m.yaml
  $ aws s3 cp ..
- Run the job and in parallel show job yaml
  $ lavacli jobs
  submit ./linaro_debos_fuego_hello.yaml
  $ less linar_debos_fuego_hello.yaml
Fuego & PDUdaemon

- **PDU daemon role**
  - Power ON/OFF/Reboot the target board
  - Supports many commercially available PDUs
  - Has a client (pduclient) and a http interface (curl)

```
ftc power-on
ftc power-off
ftc reboot
```
Fuego & Squad

- Squad role
  - Visualization of results
  - Search results by keys
  - Compare results across boards and kernel versions

Fuego with Squad
- Show interface and previous jobs
- Run hello-fail from Jenkins
$ ftc list-run -q
$ ftc put-run xxx
03

Fuego as modules

Here we provide a modularized view of Fuego internals and explain how to link it with other testing tools
## Fuego modules

<table>
<thead>
<tr>
<th>Module</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command line (ftc)</td>
</tr>
<tr>
<td>Test and run packaging</td>
</tr>
<tr>
<td>Jenkins interaction</td>
</tr>
<tr>
<td>Squad interaction</td>
</tr>
<tr>
<td>Board definitions</td>
</tr>
<tr>
<td>Overlays</td>
</tr>
<tr>
<td>Charts and plots</td>
</tr>
<tr>
<td>Report generator</td>
</tr>
<tr>
<td>Tests definitions</td>
</tr>
<tr>
<td>Shell library</td>
</tr>
<tr>
<td>Test log Parser</td>
</tr>
<tr>
<td>Pass Criteria</td>
</tr>
<tr>
<td>Specs</td>
</tr>
<tr>
<td>Dynamic variables</td>
</tr>
<tr>
<td>Cross-build scripts</td>
</tr>
<tr>
<td>Dependency system</td>
</tr>
</tbody>
</table>
Fuego & LAVA (using native definitions)

- **Concept:**
  - Extract fuego test definitions
  - Add a wrapper for Linaro test definitions

- **Difficulties**
  - Some tests are not ready for local execution
  - Some tests have a python parser
  - Fuego tests that use log_compare script work fine though
  - Maintenance would be an issue

1 [https://github.com/sangorrin/test-definitions/tree/master/automated/linux/fuego](https://github.com/sangorrin/test-definitions/tree/master/automated/linux/fuego)
Fuego LAVA test definitions
Fuego & LAVA (test definitions)

$ lavacli jobs submit ./lava-cip-core-ethtool.yaml
$ show the yaml file on the repo
**Fuego & Test runners**

- **Example test suites supported:**
  - Autopkgtest (tests inside Debian packages)
  - Ptest (yocto/oe tests)
  - LTP
  - Kselftests
  - Linaro test definitions

- **Others we may support in the future**
  - 0day (lkp), Avocado, CKI

```
ftc run-test → Fuego Functional.Linaro → test-runner myboard
```
Fuego running Linaro test definitions
$ . ./automated/bin/setenv.sh
$ cd ./automated/linux/smoke/
$ ./smoke.sh -s true
$ cat ../output/result.txt
$ ftc run-test -b vm -t Functional.linaro -dynamic-vars "yaml=./automated/linux/smoke/smoke.yaml"
$ ftc run-test -b vm -t Functional.linaro -dynamic-vars "yaml=./automated/linux/lshw/lshw.yaml"
Fuego as a test log parser

- Fuego contains a lot of valuable parsing code
- **Started** creating a python library

```bash
$ iozone -a -i 0 > iozone.log
$ fuego-parser -l iozone.log -o output.json
```

- It should support most famous test frameworks and benchmarks
- It should also support most famous test output formats (xunit, tap, ..)
- **Difficulties**
  - Parsing often depends on the parameters supplied to the program
  - It is hard to understand all possible combinations
- **Development status**: initial, best effort
04

Conclusions
CI Loop components

- **Review** → **Build** → **Deploy** → **Review**

**Source Code Repos**
- Code Review (GitHub, Gerrit, etc)
- Test Definitions
  - test source
  - dependencies
  - run instructions

**Build / Test Management**
- build kernel, DTB, ramdisk
- build distro images
- build test software

**Build Artifacts**
- Build logs
- kernel, ramdisk, rootfs
- distro images
- Test images

**Run Artifacts**
- Boot logs
- Test logs
- Monitor logs
- Test artifacts

**Lab / Board Farm**
- Test Scheduler
  - scheduling
  - resource allocation
  - dependency checking

**DUT Control**
- Control Host
  - Console
  - Power
- Network
- Logging
- External

**External Equipment**
- Power Measurement, Analyzing

**CLI tools**
- Interact
- Analyze
- Publish
- Collect Results

© 2019 Toshiba Corporation
Conclusions

- 4th methods to run Fuego on LAVA!
- How to run Linaro tests on Fuego
  - Others: ptest, autopkgtest, LTP, kselftests..
- Gitlab integration
- Jenkins integration
- Ktest integration
- Squad integration