Real-time testing with Fuego

24 Oct 2018
Hirotaka MOTAI
OUTLINE

• Who I am
• Overview
• Related Tools
  – Automated Test Framework / Fuego
  – Real-time latency tool / cyclictest
  – Tracing kernel feature / ftrace
• Issue
• Approach
  – Our Use Cases
• Conclusion and Future work
WHO I AM

• Hirotaka MOTAI
  – Software Researcher for embedded systems of Mitsubishi Electric Corp.

• We have collaborated with LF projects.
  – LTSI: Long Term Support Initiative
  – AGL : Automotive Grade Linux
  – FUEGO: Test framework
    • Specifically designed for testing Embedded Linux
• Who I am
• Overview
• Related Tools
  – Automated Test Framework / Fuego
  – Real-time latency tool / cyclictest
  – Tracing kernel feature / ftrace
• Issue
• Approach
  – Our Use Cases
• Conclusion and Future work
OVERVIEW

• Linux can be adapted to various embedded devices, even though they need a hard real-time response.

• We need tons of time to ensure adequate real-time performance.
  – Real-time applications need to satisfy timing constraints.
  – We have to avoid kernel changes which might cause long delays.
OVERVIEW

- I have developed a new test script on Fuego.  
  - measure the real-time performance, plus get tracing.  
  - get clues to isolate the problem whether it was caused by our changes or not.
• Who I am
• Overview
• Related Tools
  – Automated Test Framework / Fuego
  – Real-time latency tool / cyclictest
  – Tracing kernel feature / ftrace
• Issue
• Approach
  – Our Use Cases
• Conclusion and Future work
FUEGO

- Fuego is the automated test framework
  - created by LTSI project, based on Jenkins.
  - OSS: anyone can use and contribute!
  - AGL-JTA: AGL chose Fuego as standard test environment.
FUEGO

• Fuego = "test distribution + Jenkins + host scripts + pre-packaged tests" on container
• Fuego can do specific tests automatically that is triggered by software update.
• You can click to start manually and monitor tests on Jenkins.
• You can also check test results on Jenkins.
What is CyclicTest?
- Benchmark tool for interval timer latency.
- Refer to: https://wiki.linuxfoundation.org/realtime/documentation/howto/tools/cyclictest
• What is Cyclictest?
  – Benchmark tool for interval timer latency.
  – Refer to: https://wiki.linuxfoundation.org/realtime/documentation/howto/tools/cyclictest
FTRACE

- Ftrace have been in the kernel since Linux v2.6.27.
- Traces kernel without recompiling.
- Useful for data gathering, debugging, and performance tuning.

- Detailed in Documentation/trace/index.rst
# tracer: function
#
# _-----=> irqs-off
# / _-----=> need-resched
# | / _----=> hardirq/softirq
# || / _--=> preempt-depth
# ||| / delay
#
# TASK-PID   CPU# | ||| TIMESTAMP FUNCTION
# | | | | | | | |
# stress-ng-shm-s-1257 [000] .... 7523.267555: vmacache_find <-find_vma
# stress-ng_1h_pl-1194 [001] .... 7523.267556: mutex_unlock <-rb_simple_write
# stress-ng-shm-s-1257 [000] .... 7523.267559: handle_mm_fault <-__do_page_fault

[snip]
• Who I am
• Overview
• Related Tools
  – Automated Test Framework / Fuego
  – Real-time latency tool / cyclictest
  – Tracing kernel feature / ftrace
• Issue
• Approach
  – Our Use Cases
• Conclusion and Future work
What changes have occurred the delay?

- Our changes?
- Potential performance issues?

It is necessary to save a trace log to isolate the problem.
• Who I am
• Overview
• Related Tools
  – Automated Test Framework / Fuego
  – Real-time latency tool / cyclictest
  – Tracing kernel feature / ftrace
• Issue
• Approach
  – Our Use Cases
• Conclusion and Future work
Our Approach

• Do in a simple way.
  – Script-based Test driver program
  – 1: setup ftrace configurations
  – 2: run stress program
  – 3: run cyclictest
  – 4: save a log with a ftrace data

• Cyclictest option
  – "--breaktrace breaktime[us]" can stop tracing when latency > breaktime
STOP TRACING

• But, --breaktrace option will stop not only tracing but also testing, when diff > breaktime.

```c
873       if (!stopped && test && (diff > tracelimit)) {
874          stopped++;
875          tracemark("hit latency threshold (%llu > %d)",
876              (unsigned long long) diff, tracelimit);
877          shutdown++;
878          pthread_mutex_lock(&break_thread_id_lock);
879          if (break_thread_id == 0)
880              break_thread_id = stat->tid;
881          break thread_value = diff;
```

The record may not be the worst case...
Improved Approach

• Using a ftrace feature "Snapshot"
  – Ciclicitest is modified to take a snapshot when maximum-latency is updated.
OUR ACTUAL CASE
WITH AN IMPROVED APPROACH…
Detail of Our Evaluation Environment

• Our test case
  – Latency of real-time priority process under some kinds of stress by non-realtime process.

• Cyclic test with realtime priority
• Stress-ng with non-realtime priority
  – making stressful
Hardware and OS

• Target board: Minnowboard Turbot Dual-core
  – Intel Atom E3826
    • #Cores / Threads: 2/2
    • Freq / Cache: 1.46GHz / 1MB
  – 2GB DDR3L 1067MT/s
  – Storage: microSD
  – Ethernet: Intel i211
• Debian Gnu/Linux 9.5.0
  – Linux 4.14 LTSI
Stress-ng

• stress-ng has stressors for a lot of components
  – cpu, fork, timer, sync, dentry, flock, udp, pipe, semaphore…
  • Beware, extreme scenarios seldom happen in real-life.

• $ ./stress-ng –stressors | wc –l
  207
Result

- Stress-ng
  - Create&delete directory entry (dentry), 8 instances without CPUSET, with non-realtime priority.
  - 1 hour

![Cyclic test histogram](image)

- 76.463msec

- Test script
  - LTSI Kernel
  - Debian
  - Minnowboard
  - FUEGO
  - cyclic test
  - snapshot logs
  - test logs
  - stress-ng
  - PC
• Snapshot log in the worst case.

cyclictest-17361 [001] .... 55024.413733: mutex_lock <-__fdget_pos
cyclictest-17361 [001] .... 55024.413733: vfs_write <-SyS_write
cyclictest-17361 [001] .... 55024.413734: rw_verify_area <-vfs_write
cyclictest-17361 [001] .... 55024.413735: security_file_permission <-

rw_verify_area
cyclictest-17361 [001] .... 55024.413736: __sb_start_write <-vfs_write
cyclictest-17361 [001] .... 55024.413737: preempt_count_add <-

vfs_write
cyclictest-17361 [001] .... 1 55024.413738: preempt_count_sub <-

__sb_start_write
cyclictest-17361 [001] .... 55024.413739: __vfs_write <-vfs_write
cyclictest-17361 [001] .... 55024.413741: tracing_mark_write: hit latency

snapshot threshold (76463 > 1000)
cyclictest-17361 [001] .... 55024.413745: __fsnotify_parent <-vfs_write
cyclictest-17361 [001] .... 55024.413746: fsnotify <-vfs_write
Evaluation

• We got clues to detect the factor by doing a test and tracing at the same time.
• Fuego helped repeat execution of both.
• Therefore, we can effectively figure out the reason of the delay with using the snapshot log.
• Who I am
• Overview
• Related Tools
  – Automated Test Framework / Fuego
  – Real-time latency tool / cyclic test
  – Tracing kernel feature / ftrace
• Issue
• Approach
  – Our Use Cases
• Conclusion and Future work
CONCLUSION

• Summary
  – It is important to ensure adequate performance before releasing products.
  – It is necessary to repeat tests for reproducing the rare case which does not meet real-time performance requirements.
  – Fuego is useful to us for not only measuring but also tracing at the same time.

• Future Works
  – Discussion with Fuego community for adding our test script which I have shown.
Resources

- Test Framework: FUEGO
  - http://fuegotest.org/
  - https://elinux.org/Fuego
- LTSI Project
  - https://ltsi.linuxfoundation.org/
- AGL Test framework: AGL-JTA
  - https://wiki.automotivelinux.org/agl-jta
- rt-tests
  - https://wiki.linuxfoundation.org/realtime/documentation/howto/tools/rt-tests
- stress-ng
THANK YOU!

Any Questions?
APPENDIX
Use Case: CyclicTest Options

- 1 msec latency, 10 msec interval, in 1 hour.
  - $ cyclicTest
    - --histogram=10000 --interval=10000 --duration=3600s
    - --smp --quiet --mlockall --priority=50
    - --snapshot=1000 (instead of --breaktime)
Use Case: Test Script

• Test script runs on target.
  – 1: chrt itself
  – 2: run stress-ng program as non-rt process
  – 3: maximize /proc/sys/kernel/sched_rt_runtime_us
  – 4: setup ftrace configurations
    • echo 0 > $ftracedir/tracing_on
    • echo 0 > $ftracedir/snapshot
    • echo function > $ftracedir/current_tracer
    • echo 1 > $ftracedir/tracing_on
  – 5: run cyclictest
  – 6: save the log and a snapshot data (if recorded) to Fuego
  – 7: normalize /proc/sys/kernel/sched_rt_runtime_us