Porting LTTng to Android for Kernel-space and User-space tracing

Charles Brière <c.briere@samsung.com>
Samsung Research America
Why?

- Unified tracing within
  - Kernel
  - Native
  - Java
- Consuming traces through network instead of storing locally on device
What is different on Android

- Build system
  - Android.mk
- Shared memory
  - Android have ashmem (Anonymous SH MEMory)
- Pthread
  - Included within Bionic (Android’s Libc)
  - Not full implementation
Build system

- Android.mk works within
  - Android NDK (Native Development Kit)
  - Android source tree
- But setting up manually environment to build with autotools
  - Can use Androgenizer to generate Android.mk afterward
- NDK doesn’t expose as much as complete source tree
- Building LLTng within AOSP
Missing pthread functions

• pthread_cancel
  • Used to kill consumer thread if it failed to initialize
  • Otherwise thread will be stopped with pipe
  • Now using pthread_kill
  • Bad idea? Not too bad
    • No cleanup methods anyway

• pthread_cond_timedwait
  • Every pthread_condattr_setclock uses MONOTONIC
  • Use pthread_cond_timedwait_monotonic_np
Shared memory

- LTTng unlinks shm path
- Shares most memory regions by passing file descriptor through UNIX socket
- Makes using Android’s ashmem instead of SYSV’s shm a simple substitution
Shared memory: Futex

- Daemon notification
  - Application can be started before daemon
  - Unix socket or pipes
  - Ashmem not suited as anonymous
  - Sharing Futex through file instead of shared memory
    - `shm_open` → `open`
Missing definitions

- HOST_NAME_MAX  
  - 64
- NAME_MAX  
  - 255
- PATH_MAX  
  - 4096
- SPLICE_*  
  - Even if __NR_splice exists
- in_port_t  
  - uint16_t

<limits.h>

<fcntl.h>

<netinet/in.h>
Missing functions with equivalents

- posix_fadvise
  - __NR_arm_fadvise64_64()
- splice
  - __NR splice()
- bswap_{16,32,64}
  - __bswap_{16,32,64}
Missing functions w/o equivalents

- `pthread_cancel`
  - `pthread_kill ( with SIGKILL )`
- `shm_open`
  - `ashmem_create – ftruncate`
- `pthread_cond_timedwait`
  - `pthread_cond_timedwait_monotonic_np`
- `getpwuid_r`
  - Set home to /sdcard
- `sigwaitinfo`
  - `sigtimedwait ( arbitrary 10000 seconds timeout)`
Includes

- signal.h → sys/signal.h
- wait.h → sys/wait.h
What’s next

• Make patches upstreamable and submit
• Add JAVA tracing capabilities
• Integrate in AOSP source tree
Questions ?