LLDB in FreeBSD

Ed Maste
BSDCam 2013

Need a new debugger

- GDB 6.1.1 in FreeBSD, June 2004
- Major shortcomings
  - C++
  - thread support
  - scripting
  - performance
  - ...
- FreeBSD project policy - no GPLv3
- Last GPLv2 GDB is 6.6, December 2006
  - not much better

LLDB History

- Debugger in LLVM family of projects
- Originated within Apple
- Open sourced in June 2010
- ~ 600 KLOC (GDB is ~3M)
- 28 contributors last 12 months
  - up from 15, previous 12 months
- 15 contributors last month
  - 3 new
- Apple (14), Intel (5), FreeBSD (1), Debian (1), Valve Software (1), Individuals and unknown (17)

LLDB Benefits

- Speed
  - Multi-threaded, leverages performant LLVM classes
- Efficiency
  - Minimize memory footprint - lazy and partial evaluation
- Accuracy
  - Improved ability to set breakpoints, expression parsing
  - Breakpoints are always symbolic - reparsed after .so loading

LLDB Extensibility and Reusability

- Classes for process, thread, dynamic loader, object files, object containers, symbols, disassembly
- lldb commandline, XCode, Python front ends
  ```
  >>> import lldb
  ```
- built-in python interpreter for scripting
  - can easily be other languages too

LLDB Syntax

GDB

```
% gdb a.out
(gdb) break main
Breakpoint 1 at 0x100000f33: file main.c, line 4
(gdb) run
```

LLDB

```
% lldb a.out
(lldb) breakpoint set --name main
Breakpoint created: 1: name = 'main', locations = 1
(lldb) process launch
```

LLDB Syntax

GDB	LLDB
(gdb) `run`	**(lldb)** `process launch`
(gdb) `r`	**(lldb)** `run`
	(lldb) `r`
(gdb) `step`	**(lldb)** `thread step-in`
(gdb) `s`	**(lldb)** `step`
	(lldb) `s`
(gdb) `info break`	**(lldb)** `breakpoint list`
	(lldb) `br l`
(gdb) `info args`	**(lldb)** `frame variable`
and	**(lldb)** `fr v`
(gdb) `info locals`	

Demo

LLDB Status in FreeBSD

- Testsuite
  - 260 tests run, 30 fail and do not have a PR
- Targets
  - amd64
  - i386 code in tree
  - ARM supported in LLDB core, not Linux / FreeBSD
  - MIPS not yet
  - Others no plans

LLDB Status in FreeBSD

- Userland core files
  - "Just works" on 9.2+ and HEAD
  - for some value of "Just works"
  - further testing needed
- Userland live debugging (ptrace)
  - Process launch, process attach by pid
  - Process attach by name
  - Breakpoints
  - Watchpoints
  - Threads

LLDB Status in FreeBSD

- Kernel core files
  - Unimplemented
  - straightforward - see source/Plugins/Process/elf-core/
- Kernel live debugging
  - Unimplemented
  - gdb remote
  - /dev/mem

LLDB Status in FreeBSD

- Remote debugging - GDB protocol
  - Need to enable for Linux & FreeBSD
- Remote debugging - debugserver
  - Unimplemented
- Cross debugging
  - Cross-arch and cross-OS
  - Should "just work"
  - Fails due to some assumptions in source, but not difficult

Short term

- Source in contrib/llvm/tools/lldb
- FreeBSD build infrastructure to come
- Source in 10.0, likely not built by default
- WITH_LLDB=
- Testing

Medium term

- amd64 thread support for ptrace
- watchpoints
- MIPS host and target
- test suite failures

Longer term

- ARM support
- Kernel debug
- Remote debug