Debagovanje sa LLDB-om

Prezentacija seminarskog rada

Momir Adžemović, Miloš Miković, Marko Spasić,
Mladen Dobrašinović

momir.adzemovic@gmail.com, spaskeasm@gmail.com,
milos.mikovicpos@gmail.com, dobrasinovic.mladen@gmail.com

Matematički fakultet

Uvod

- Zašto koristiti debager?
- LLDB debager
- Platforme na kojima radi
- LLDB u poređenju sa drugim debagerima

Debagovanje i bagovi uopšteno

- Šta je debagovanje?
- Koji su koraci pri debagovanju?
- Šta je bag?

„Debagovanje je duplo teže od kodiranja, ako napišete kod na najlukaviji (odnosno najkomplikovaniji) način, po definiciji niste dovoljno pametni da ga debagujete.” (Brian W. Kernighan)

Debager

Šta je debager?

Zašto debager?

```
1 const char *str = "hello";
2 size_t length = strlen(str);
3 size_t i;
4 for(i = length - 1; i >= 0; i--)
5   putchar(str[i]);
```

Šta je LLDB?

- LLDB je napravljen sa ciljem da se stvori visoko-performantni debager sledeće generacije u okviru projekta LLVM.
- LLDB iskorištava funkcionalnosti Clang kompilatora i LLVM-a. (disassembly, parser, JIT kompilator, podrška za jezike)
  - Podržava C++, C, Objective-C
- LLDB pruža C++ i Python API-e i pruža podršku za skripting.

Osnove LLDB debagera komandne linije

Sintaksna struktura komandi:
`<noun> <verb> [-option [value]] [argument [argument...]]`

Nove komande se mogu definisati sa:
`command alias`

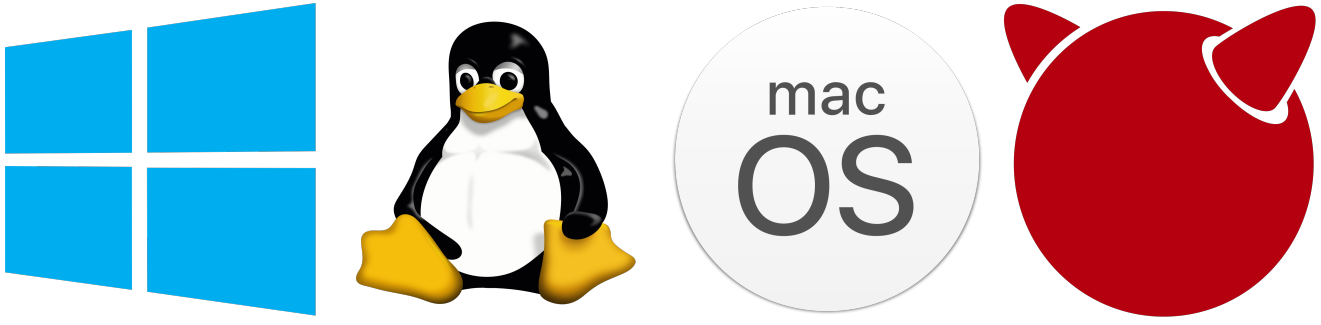
Za istraživanje komandi u alatu se mogu koristiti `help` i `apropos`.

Upoznavanje sa LLDB-om

Tabela: Neke od osnovnih komandi LLDB-a.

1.	`process launch -- <argumenti>`
2.	`thread step-in`
3.	`thread step-inst-over`
4.	`breakpoint set --file 1.c --line 42`
5.	`breakpoint list`
6.	`breakpoint disable 1`
7.	`watchpoint set variable global_var`
8.	`watchpoint modify -c '(global_var == 5)'`

Platforme koje podržava LLDB



Funkcionalnosti

Uglavnom svuda:

- Backtracking
- Breakpoints
- C++11
- Core file debugging
- Remote debugging
- Disassembly
- Expression evaluation Windows ima problema

Razvojna okruženja

(a) Deo okruženja

(b) Deo okruženja

(c) Plugin

(d) Plugin

Poređenje sa ostalim popularnim debagerima

Kako bi programer izabrao debager potrebno je da odgovori na sledeća pitanja poštujući redosled:

- Koje debagere podržava operativni sistem?
- Da li okruženje koje koristi ima specijalnu podršku za odgovarajući debager?
- Koji debager mu najviše odgovara?

Konkurencija:

- GDB
- Visual Studio Debugger

Poređenje sa GDB i VSD

LLDB i GDB

- Sličan skup komandi i opcija
- GDB nudi više mogućnosti za sada (primer: single thread breakpoint)
- LLDB je lakše formalno ubaciti u softver (licenca)
- LLDB je moderniji (moderne biblioteke)
- LLDB nema maskotu kao GDB:

LLDB i Visual Studio Debugger

- VSD nudi grafički interfejs
- LLDB ima bolju podršku za operativne sisteme

Zaključak

- Odličan uz MacOS
- Nema dodatnih podešavanja uz Clion i Xcode5
- Na Windows-u i dalje možda bolji Visual Studio Debager
- Na Linux-u i dalje možda bolji GDB
- Deo LLVM projekta
- Razvoj brzo napreduje

Literatura

LLVM.
The LLDB Debugger, 2020.
at: https://lldb.llvm.org.

Saša Malkov.
Debagovanje, 2019.
at: http://poincare.matf.bg.ac.rs/~smalkov/files/rs.r290.2019/public/Predavanja/Razvojsoftvera.08.2019-Debagovanje.p4.pdf.

Derek Selander.
Advanced Apple Debugging & Reverse Engineering, 2nd edition.
2017.