gdb

공기석

디버깅 및 디버거

- 일반적인 프로그램의 제작 과정
  - 스펙작성-코드설계-코딩-테스트-디버깅-배포-유지보수 및 업데이트

- 디버깅(debugging)
  - 디버깅이란 말 그대로 버그(벌레)를 제거하는 작업이다.
  - 디버깅에서 '버그' 정의
    - 버그란 컴파일러가 컴파일을 하거나, 링커가 링크 작업을 하면서 출력하는 syntax error, logical error, runtime error와는 달리 컴파일러, 링커가 찾아내지 못하는 에러를 뜻한다.

- 디버거(debugger)
  - 디버거란 디버깅을 위한 도구이다.
  - 디버거의 목적
    - 어떤 프로그램이 실행되는 동안, 또는 프로그램이 에러로 인하여 멈추는 순간에 내부에서는 어떤 일이 일어나고 있는지를 보여주는 것이다.

gdb 개요

- GDB는 FSF(Free Software Foundation)에서 제공하는 도구이다.

- GDB의 4가지 주요기능(더 많은 일을 할 수 있음)
  - 프로그램의 행동에 영향을 줄 수 있는 각종 조건을 설정한 후, 프로그램을 시작한다.
  - 특정 조건을 만나면 프로그램을 정지시킨다.
  - 프로그램이 정지됐을 때 무슨 일이 일어났는지 검사한다.
  - 프로그램 내부 설정을 바꾸어서 버그를 수정함으로써 다른 버그를 계속 찾아나간다.

- GDB는 프로그래머에게 몇 분만에 심각한 버그를 찾아서 정정할 수 있도록 도움을 주기에 이를 배우는데 드는 시간과 노력에 대해서 아까워한다거나 할 필요가 없다.

gdb 시작하기

- 디버깅 세션의 시작
  - 디버깅 옵션을 준 컴파일 : gcc –g –o debugme debugme.c
    - GDB 실행 : gdb progname [corefile]
      - progname : 디버깅하기 원하는 프로그램의 이름
      - corefile : 코어파일의 사용은 선택적이지만, gdb의 기능을 강화함
      - -q , –quiet : 라이센스 메시지를 원지 않을경우 제거해준다
      - -d : gdb에게 소스코드의 위치를 알려주는 디렉토리의 이름
        - gdb는 기본적으로 현재 작업 디렉토리에서 소스 코드를 찾는다.

에러 발견

- 소스코드 (imsi.c)

```
#include <stdio.h>

int main(void)
{
   char imsi[50];

   strcpy(imsi, "Advanced Programming 2016 - gdb test!!");
   printf("imsi = %s\n", imsi[0]);

    return 0;
}
```

- 실행

```
% gcc –o imsi imsi.c
% imsi
세그멘테이션 결함(Segmentation Fault)
%
```

디버깅 시작

```
% gcc -g -o imsi imsi.c
% imsi
세그멘테이션 결함(Segmentation Fault)
%
% gdb imsi
GNU gdb 6.8
Copyright (C) 2008 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "sparc-sun-solaris2.10"...
(gdb) run
Starting program: /u1/prof/kskong/advprog/imsi

Program received signal SIGSEGV, Segmentation fault.
0xff232d18 in strlen () from /usr/lib/libc.so.1
(gdb) backtrace
#0  0xff232d18 in strlen () from /usr/lib/libc.so.1
#1  0xff29f414 in _ndoprnt () from /usr/lib/libc.so.1
#2  0xff2a15fc in printf () from /usr/lib/libc.so.1
#3  0x00010920 in main () at imsi.c:8(gdb)
(gdb) list
8           printf("imsi = %s\n", imsi[0]);
9       }
(gdb)
```

gdb 명령어 (1)

- help [command]: 특정 명령어의 온라인 매뉴얼
- run : 디버거에서 프로그램을 실행시킨다.
- list : 지정된 행에 대한 코드를 보여준다.
- backtrace(역추적) : 프로그램 스택을 보여준다. (*where*)
  - 스택은 스택 프레임으로 구성되어진다. GDB는 스택 프레임에 번호를 지정한다. GDB는 가장 안쪽에 있는 (현재 실행중인) 프레임에 대해 0번부터 번호를 부여한다. 항상 GDB는 한 프레임만을 선택된 프레임으로 간주한다. 디버깅중인 프로그램이 정지될 때, gdb는 가장 안쪽에 있는 프레임을 선택한다.
- print : 수식, 변수, 배열의 값을 출력한다. (*p*)
- whatis : 수식, 변수, 배열의 형태를 출력한다.

```
(gdb) print imsi[0]
$1 = 65 'A'
(gdb) p/x imsi[0]
$2 = 0x41
(gdb) p imsi
$3 = "Advanced Programming 2016 - gdb test!!\000\000\000\000\000\003□웝?000"
(gdb) p/x imsi
$4 = {0x41, 0x64, 0x76, 0x61, 0x6e, 0x63, 0x65, 0x64, 0x20, 0x50, 0x72, 0x6f,
  0x67, 0x72, 0x61, 0x6d, 0x6d, 0x69, 0x6e, 0x67, 0x20, 0x32, 0x30, 0x31,
  0x36, 0x20, 0x2d, 0x20, 0x67, 0x64, 0x62, 0x20, 0x74, 0x65, 0x73, 0x74,
  0x21, 0x21, 0x0, 0x0, 0x0, 0x0, 0x0, 0x3, 0xff, 0xbf, 0xfb, 0xc4, 0x0, 0x0}
(gdb) whatis imsi
type = char [50]
(gdb)
```

- quit: gdb 실행 종료

gdb 명령어 (2)

- 중단지점(breakpoint) 설정
  - break : 특정 라인이나 함수에 중단점을 설정한다. (br)
- 중단지점 해제
  - delete [번호]
  - delete all
  - clear
  - clear [라인 번호]
- 실행재개
  - cont

```
(gdb) br 8
Breakpoint 1 at 0x106bc: file imsi.c, line 8.
(gdb) run
Starting program: /u1/prof/kskong/advprog/imsi

Breakpoint 1, main () at imsi.c:8
8           printf("imsi = %s\n", imsi[0]);
(gdb) cont
Continuing.

Program received signal SIGSEGV, Segmentation fault.
0xff232d18 in strlen () from /usr/lib/libc.so.1
(gdb)
```

gdb 명령어 (3)

- 라인단위 처리 명령어
  - next: 현재 라인을 실행하고 다음 라인으로 넘어감. 현재 라인이 함수이면 그 함수를 실행하고 넘어간다.
  - next [라인번호]: 해당 라인까지 실행
  - step: next 처럼 한 라인씩 실행하지만 현재 라인이 함수이면 그 함수 내부로 들어간다.
  - step [번호]: 지정한 라인 수 만큼 실행. 내부에 함수가 있으면 그 함수 내부로 들어간다.
- 조건부 처리 명령
  - break [라인번호] if [조건]
    - (gdb) break 24 if i==50
- 실행중인 코드의 변수 값의 변경: set variable

  set variable varname = value

  (gdb)break 25 if i==15

  (gdb)set variable i =10

  (gdb)p i

  $1 = 10

gdb 연습

```
/* gdb 테스트 프로그램 예제 */
/* 1부터 100까지 누적한 합과 총합을 구하고
   1부터 100까지 홀수만 다시 출력한다.
   이때 중간지점(50)에 이르면 중간에
    도달했다는 메시지를 출력한다.
*/

#include <stdio.h>

int main(void)
{
    int i, j=0;

    for (i=1; i<=100; i++)
    {
        j = j+i;
        printf("sub %d\n", j);
    }

    puts(" ");
    printf("sum %d\n", j);
```

```
for (i=1; i<=100; i++)
    {
        if (i % 50)
        {
            puts("arrived a half");
            i = 60;
        }
        if ( i%2 == 0) continue;
        printf("odd %d\n", i);
    }
    return 0;
}
```

참고자료

- gdb online manual
  - https://sourceware.org/gdb/current/onlinedocs/gdb/ (영문)
  - http://korea.gnu.org/manual/release/gdb/ (한글)

- 책

  백창우, **유닉스 리눅스 프로그래밍 필수 유틸리티 (개정판)**, 한빛미디어, 2010.