100个gdb小技巧

100 gdb

> hellogcc
>
> 100-gdb-tips

gdb 100

PULL REQUEST

1. src md markdown http://wowubuntu.com/markdown/ md https://www.zybuluo.com/mdeditor
2. index.md md
3. OK!

html

1. go md2min
2. build.sh
3. html html

-
- IRC, freenode, #hellogcc
- ( )

-

- GDB
- GDB
- GDB dashboard
- Gdbinit for OS X, iOS and others - x86, x86_64 and ARM
- dotgdb gdb

gdb

gdb gdb “ `show version` ” :

```
(gdb) show version
GNU gdb (GDB) 7.7.1
Copyright (C) 2014 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show c
and "show warranty" for details.
This GDB was configured as "x86_64-pc-solaris2.10".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word".
```

gdb

nanxiao

gdb

gdb gdb “ `show copying` ” :

```
(gdb) show copying
                    GNU GENERAL PUBLIC LICENSE
                       Version 3, 29 June 2007

 Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/
 Everyone is permitted to copy and distribute verbatim copies
 of this license document, but changing it is not allowed.

                            Preamble

  The GNU General Public License is a free, copyleft license for
software and other kinds of works.

  The licenses for most software and other practical works are desi
to take away your freedom to share and change the works.  By contra
the GNU General Public License is intended to guarantee your freedo
share and change all versions of a program--to make sure it remains
software for all its use
......
```

“ `show warranty` ”

```
(gdb) show warranty
  15. Disclaimer of Warranty.

  THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY
APPLICABLE LAW.  EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRI
HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WA
OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITE
THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICU
PURPOSE.  THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE
IS WITH YOU.  SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE CO
ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

  16. Limitation of Liability.

  IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRI
WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR C
THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLU
GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT O
USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO L
DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU O
PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGR
EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBIL
SUCH DAMAGES.

  17. Interpretation of Sections 15 and 16.

  If the disclaimer of warranty and limitation of liability provide
above cannot be given local legal effect according to their terms,
reviewing courts shall apply local law that most closely approximat
an absolute waiver of all civil liability in connection with the
Program, unless a warranty or assumption of liability accompanies a
copy of the Program in return for a fee.
```

gdb

nanxiao

```
$ gdb
GNU gdb (GDB) 7.7.50.20140228-cvs
Copyright (C) 2014 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show c
and "show warranty" for details.
This GDB was configured as "x86_64-unknown-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word".
```

gdb

`-q` :

```
$ gdb -q
(gdb)
```

~/.bashrc gdb

```
alias gdb="gdb -q"
```

gdb

xmj

gdb

gdb :

```
A debugging session is active.

    Inferior 1 [process 29686    ] will be killed.

Quit anyway? (y or n) n
```

gdb :

```
(gdb) set confirm off
```

.gdbinit

nanxiao

gdb gdb
“ `---Type <return> to continue, or q <return> to quit---` ”

```
 81 process 2639102      0xff04af84 in __lwp_park () from /usr/lib/
 80 process 2573566      0xff04af84 in __lwp_park () from /usr/lib/
---Type <return> to continue, or q <return> to quit---Quit
```

“ `set pagination off` ” “ `set height 0` ” gdb

gdb .

nanxiao

```
#include <stdio.h>
#include <pthread.h>
void *thread_func(void *p_arg)
{
        while (1)
        {
                sleep(10);
        }
}
int main(void)
{
        pthread_t t1, t2;

        pthread_create(&t1, NULL, thread_func, "Thread 1");
        pthread_create(&t2, NULL, thread_func, "Thread 2");

        sleep(1000);
        return;
}
```

gdb “`info functions`”

```
(gdb) info functions
All defined functions:

File a.c:
int main(void);
void *thread_func(void *);

Non-debugging symbols:
0x0805079c  _PROCEDURE_LINKAGE_TABLE_
0x080507ac  _cleanup@plt
0x080507bc  atexit
0x080507bc  atexit@plt
0x080507cc  __fpstart
0x080507cc  __fpstart@plt
0x080507dc  exit@plt
0x080507ec  __deregister_frame_info_bases@plt
0x080507fc  __register_frame_info_bases@plt
0x0805080c  _Jv_RegisterClasses@plt
0x0805081c  sleep
0x0805081c  sleep@plt
0x0805082c  pthread_create@plt
0x0805083c  _start
0x080508b4  _mcount
0x080508b8  __do_global_dtors_aux
0x08050914  frame_dummy
0x080509f4  __do_global_ctors_aux
0x08050a24  _init
0x08050a31  _fini
```

“ `info functions regex` ”

```
(gdb) info functions thre*
All functions matching regular expression "thre*":

File a.c:
void *thread_func(void *);

Non-debugging symbols:
0x0805082c  pthread_create@plt
```

gdb “ `thre` ”

gdb

nanxiao

```
#include <stdio.h>

int func(void)
{
    return 3;
}

int main(void)
{
    int a = 0;

    a = func();
    printf("%d\n", a);
    return 0;
}
```

gdb step s

```
(gdb) n
12              a = func();
(gdb) s
func () at a.c:5
5               return 3;
(gdb) n
6       }
(gdb)
main () at a.c:13
13              printf("%d\n", a);
```

gdb func

next n gdb

```
(gdb) n
12                a = func();
(gdb) n
13                printf("%d\n", a);
(gdb) n
3
14                return 0;
```

gdb func

gdb

nanxiao

```
#include <stdio.h>
#include <pthread.h>

typedef struct
{
        int a;
        int b;
        int c;
        int d;
        pthread_mutex_t mutex;
}ex_st;

int main(void) {
        ex_st st = {1, 2, 3, 4, PTHREAD_MUTEX_INITIALIZER};
        printf("%d,%d,%d,%d\n", st.a, st.b, st.c, st.d);
        return 0;
}
```

gdb

```
(gdb) n
15              printf("%d,%d,%d,%d\n", st.a, st.b, st.c, st.d);
(gdb) s
1,2,3,4
16              return 0;
```

printf “s” s “step” printf

“set step-mode on” gdb

```
(gdb) set step-mode on
(gdb) n
15                printf("%d,%d,%d,%d\n", st.a, st.b, st.c, st.d);
(gdb) s
0x00007ffff7a993b0 in printf () from /lib64/libc.so.6
(gdb) s
0x00007ffff7a993b7 in printf () from /lib64/libc.so.6
```

gdb printf

gdb

nanxiao

```
#include <stdio.h>

int func(void)
{
    int i = 0;

    i += 2;
    i *= 10;

    return i;
}

int main(void)
{
    int a = 0;

    a = func();
    printf("%d\n", a);
    return 0;
}
```

“ `finish` ”

```
(gdb) n
17            a = func();
(gdb) s
func () at a.c:5
5                int i = 0;
(gdb) n
7                i += 2;
(gdb) fin
find    finish
(gdb) finish
Run till exit from #0  func () at a.c:7
0x08050978 in main () at a.c:17
17            a = func();
Value returned is $1 = 20
```

`func` “ `finish` ” gdb
“ `20` ”

gdb

“ `return` ”
“ `return expression` ”

```
(gdb) n
17            a = func();
(gdb) s
func () at a.c:5
5                int i = 0;
(gdb) n
7                i += 2;
(gdb) n
8                i *= 10;
(gdb) re
record              remove-inferiors    return              reverse
refresh             remove-symbol-file  reverse-continue    reverse
remote              restore             reverse-finish      reverse
(gdb) return 40
Make func return now? (y or n) y
#0  0x08050978 in main () at a.c:17
17            a = func();
(gdb) n
18            printf("%d\n", a);
(gdb)
40
19            return 0;
```

“ `return` ”

gdb

nanxiao

```
#include <stdio.h>

int global = 1;

int func(void)
{
    return (++global);
}

int main(void)
{
    printf("%d\n", global);
    return 0;
}
```

gdb “ `call` ” “ `print` ”

```
(gdb) start
Temporary breakpoint 1 at 0x4004e3: file a.c, line 12.
Starting program: /data2/home/nanxiao/a

Temporary breakpoint 1, main () at a.c:12
12              printf("%d\n", global);
(gdb) call func()
$1 = 2
(gdb) print func()
$2 = 3
(gdb) n
3
13              return 0;
```

`func` `global` `3`

gdb .

nanxiao

```
#include <stdio.h>
int func(int a, int b)
{
    int c = a * b;
    printf("c is %d\n", c);
}

int main(void)
{
    func(1, 2);
    return 0;
}
```

gdb “ `i frame` ” `i` `info`

```
Breakpoint 1, func (a=1, b=2) at a.c:5
5               printf("c is %d\n", c);
(gdb) i frame
Stack level 0, frame at 0x7fffffffe590:
 rip = 0x40054e in func (a.c:5); saved rip = 0x400577
 called by frame at 0x7fffffffe5a0
 source language c.
 Arglist at 0x7fffffffe580, args: a=1, b=2
 Locals at 0x7fffffffe580, Previous frame's sp is 0x7fffffffe590
 Saved registers:
  rbp at 0x7fffffffe580, rip at 0x7fffffffe588
(gdb) i registers
rax            0x2      2
rbx            0x0      0
rcx            0x0      0
rdx            0x7fffffffe688   140737488348808
rsi            0x2      2
rdi            0x1      1
rbp            0x7fffffffe580   0x7fffffffe580
rsp            0x7fffffffe560   0x7fffffffe560
r8             0x7ffff7dd4e80   140737351863936
r9             0x7ffff7dea560   140737351951712
r10            0x7fffffffe420   140737488348192
```

```
r11            0x7ffff7a35dd0   140737348066768
r12            0x400440 4195392
r13            0x7fffffffe670   140737488348784
r14            0x0      0
r15            0x0      0
rip            0x40054e 0x40054e <func+24>
eflags         0x202    [ IF ]
cs             0x33     51
ss             0x2b     43
ds             0x0      0
es             0x0      0
fs             0x0      0
gs             0x0      0
(gdb) disassemble func
Dump of assembler code for function func:
   0x0000000000400536 <+0>:     push   %rbp
   0x0000000000400537 <+1>:     mov    %rsp,%rbp
   0x000000000040053a <+4>:     sub    $0x20,%rsp
   0x000000000040053e <+8>:     mov    %edi,-0x14(%rbp)
   0x0000000000400541 <+11>:    mov    %esi,-0x18(%rbp)
   0x0000000000400544 <+14>:    mov    -0x14(%rbp),%eax
   0x0000000000400547 <+17>:    imul   -0x18(%rbp),%eax
   0x000000000040054b <+21>:    mov    %eax,-0x4(%rbp)
=> 0x000000000040054e <+24>:    mov    -0x4(%rbp),%eax
   0x0000000000400551 <+27>:    mov    %eax,%esi
   0x0000000000400553 <+29>:    mov    $0x400604,%edi
   0x0000000000400558 <+34>:    mov    $0x0,%eax
   0x000000000040055d <+39>:    callq  0x400410 <printf@plt>
   0x0000000000400562 <+44>:    leaveq
   0x0000000000400563 <+45>:    retq
End of assembler dump.
```

“ `i frame` ”

gdb .

nanxiao

```
#include<stdio.h>
void a(void)
{
        printf("Tail call frame\n");
}

void b(void)
{
        a();
}

void c(void)
{
        b();
}

int main(void)
{
        c();
        return 0;
}
```

“ Tail call ”
‘-O’

```
gcc -g -O -o test test.c
```

`main`

```
(gdb) disassemble main
Dump of assembler code for function main:
0x0000000000400565 <+0>:     sub    $0x8,%rsp
0x0000000000400569 <+4>:     callq  0x400536 <a>
0x000000000040056e <+9>:     mov    $0x0,%eax
0x0000000000400573 <+14>:    add    $0x8,%rsp
0x0000000000400577 <+18>:    retq
```

`main` `a` `b` `c`

`a`

```
(gdb) i frame
Stack level 0, frame at 0x7fffffffe590:
 rip = 0x400536 in a (test.c:4); saved rip = 0x40056e
 called by frame at 0x7fffffffe5a0
 source language c.
 Arglist at 0x7fffffffe580, args:
 Locals at 0x7fffffffe580, Previous frame's sp is 0x7fffffffe590
 Saved registers:
  rip at 0x7fffffffe588
```

“ `debug entry-values` ” 0

```
(gdb) set debug entry-values 1
(gdb) b test.c:4
Breakpoint 1 at 0x400536: file test.c, line 4.
(gdb) r
Starting program: /home/nanxiao/test

Breakpoint 1, a () at test.c:4
4       {
(gdb) i frame
tailcall: initial:
Stack level 0, frame at 0x7fffffffe590:
 rip = 0x400536 in a (test.c:4); saved rip = 0x40056e
 called by frame at 0x7fffffffe5a0
 source language c.
 Arglist at 0x7fffffffe580, args:
 Locals at 0x7fffffffe580, Previous frame's sp is 0x7fffffffe590
 Saved registers:
  rip at 0x7fffffffe588
```

“ `tailcall: initial:` ”

gdb .

nanxiao

```
#include <stdio.h>

int func1(int a)
{
        return 2 * a;
}

int func2(int a)
{
        int c = 0;
        c = 2 * func1(a);
        return c;
}

int func3(int a)
{
        int c = 0;
        c = 2 * func2(a);
        return c;
}

int main(void)
{
        printf("%d\n", func3(10));
        return 0;
}
```

gdb “`frame n`” `n`

```
(gdb) b test.c:5
Breakpoint 1 at 0x40053d: file test.c, line 5.
(gdb) r
Starting program: /home/nanxiao/test

Breakpoint 1, func1 (a=10) at test.c:5
5               return 2 * a;
(gdb) bt
#0  func1 (a=10) at test.c:5
#1  0x0000000000400560 in func2 (a=10) at test.c:11
#2  0x0000000000400586 in func3 (a=10) at test.c:18
#3  0x000000000040059e in main () at test.c:24
(gdb) frame 2
#2  0x0000000000400586 in func3 (a=10) at test.c:18
18              c = 2 * func2(a);
```

`0` `frame 2`
`fun3`

“ `frame addr` ” `addr`

```
(gdb) frame 2
#2  0x0000000000400586 in func3 (a=10) at test.c:18
18              c = 2 * func2(a);
(gdb) i frame
Stack level 2, frame at 0x7fffffffe590:
 rip = 0x400586 in func3 (test.c:18); saved rip = 0x40059e
 called by frame at 0x7fffffffe5a0, caller of frame at 0x7fffffffe5
 source language c.
 Arglist at 0x7fffffffe580, args: a=10
 Locals at 0x7fffffffe580, Previous frame's sp is 0x7fffffffe590
 Saved registers:
  rbp at 0x7fffffffe580, rip at 0x7fffffffe588
(gdb) frame 0x7fffffffe568
#1  0x0000000000400560 in func2 (a=10) at test.c:11
11              c = 2 * func1(a);
```

“ `i frame` ” `0x7fffffffe568` `func2`
“ `frame 0x7fffffffe568` ” `func2`

gdb .

nanxiao

```
#include <stdio.h>

int func1(int a)
{
        return 2 * a;
}

int func2(int a)
{
        int c = 0;
        c = 2 * func1(a);
        return c;
}

int func3(int a)
{
        int c = 0;
        c = 2 * func2(a);
        return c;
}

int main(void)
{
        printf("%d\n", func3(10));
        return 0;
}
```

gdb “ `up n` ” “ `down n` ”
`n`

```
(gdb) b test.c:5
Breakpoint 1 at 0x40053d: file test.c, line 5.
(gdb) r
Starting program: /home/nanxiao/test

Breakpoint 1, func1 (a=10) at test.c:5
5               return 2 * a;
(gdb) bt
#0  func1 (a=10) at test.c:5
#1  0x0000000000400560 in func2 (a=10) at test.c:11
#2  0x0000000000400586 in func3 (a=10) at test.c:18
#3  0x000000000040059e in main () at test.c:24
(gdb) frame 2
#2  0x0000000000400586 in func3 (a=10) at test.c:18
18              c = 2 * func2(a);
(gdb) up 1
#3  0x000000000040059e in main () at test.c:24
24              printf("%d\n", func3(10));
(gdb) down 2
#1  0x0000000000400560 in func2 (a=10) at test.c:11
11              c = 2 * func1(a);
```

“ `frame 2` ” `fun3`
“ `up 1` ” `main`
“ `down 2` ” `n`
`n` `1` .

“ `up-silently n` ” “ `down-silently n` ”
“ `up n` ” “ `down n` ”

```
(gdb) up
#2  0x0000000000400586 in func3 (a=10) at test.c:18
18              c = 2 * func2(a);
(gdb) bt
#0  func1 (a=10) at test.c:5
#1  0x0000000000400560 in func2 (a=10) at test.c:11
#2  0x0000000000400586 in func3 (a=10) at test.c:18
#3  0x000000000040059e in main () at test.c:24
(gdb) up-silently
(gdb) i frame
Stack level 3, frame at 0x7fffffffe5a0:
 rip = 0x40059e in main (test.c:24); saved rip = 0x7ffff7a35ec5
 caller of frame at 0x7fffffffe590
 source language c.
 Arglist at 0x7fffffffe590, args:
 Locals at 0x7fffffffe590, Previous frame's sp is 0x7fffffffe5a0
 Saved registers:
  rbp at 0x7fffffffe590, rip at 0x7fffffffe598
```

`func3` `main`

gdb .

nanxiao

```
namespace Foo
{
  void foo()
  {
  }
}

namespace
{
  void bar()
  {
  }
}
```

gdb namespace Foo foo

```
(gdb) b Foo::foo
```

bar

```
(gdb) b (anonymous namespace)::bar
```

xmj

```
0000000000400522 <main>:
  400522:       55                      push   %rbp
  400523:       48 89 e5                mov    %rsp,%rbp
  400526:       8b 05 00 1b 00 00       mov    0x1b00(%rip),%eax
  40052c:       85 c0                   test   %eax,%eax
  40052e:       75 07                   jne    400537 <main+0x15>
  400530:       b8 7c 06 40 00          mov    $0x40067c,%eax
  400535:       eb 05                   jmp    40053c <main+0x1a>
```

`b *address`

```
(gdb) b *0x400522
```

gdb

xmj

```
$ strip a.out
$ readelf -h a.out
ELF Header:
  Magic:   7f 45 4c 46 02 01 01 00 00 00 00 00 00 00 00 00
  Class:                             ELF64
  Data:                              2's complement, little endian
  Version:                           1 (current)
  OS/ABI:                            UNIX - System V
  ABI Version:                       0
  Type:                              EXEC (Executable file)
  Machine:                           Advanced Micro Devices X86-64
  Version:                           0x1
  Entry point address:               0x400440
  Start of program headers:          64 (bytes into file)
  Start of section headers:          4496 (bytes into file)
  Flags:                             0x0
  Size of this header:               64 (bytes)
  Size of program headers:           56 (bytes)
  Number of program headers:         9
  Size of section headers:           64 (bytes)
  Number of section headers:         29
  Section header string table index: 28
```

```
$ gdb a.out
>>> info files
Symbols from "/home/me/a.out".
Local exec file:
    `/home/me/a.out', file type elf64-x86-64.
    Entry point: 0x400440
    0x0000000000400238 - 0x0000000000400254 is .interp
    0x0000000000400254 - 0x0000000000400274 is .note.ABI-tag
    0x0000000000400274 - 0x0000000000400298 is .note.gnu.build-id
    0x0000000000400298 - 0x00000000004002b4 is .gnu.hash
    0x00000000004002b8 - 0x0000000000400318 is .dynsym
    0x0000000000400318 - 0x0000000000400355 is .dynstr
    0x0000000000400356 - 0x000000000040035e is .gnu.version
    0x0000000000400360 - 0x0000000000400380 is .gnu.version_r
    0x0000000000400380 - 0x0000000000400398 is .rela.dyn
    0x0000000000400398 - 0x00000000004003e0 is .rela.plt
    0x00000000004003e0 - 0x00000000004003fa is .init
    0x0000000000400400 - 0x0000000000400440 is .plt
    0x0000000000400440 - 0x00000000004005c2 is .text
    0x00000000004005c4 - 0x00000000004005cd is .fini
    0x00000000004005d0 - 0x00000000004005e0 is .rodata
    0x00000000004005e0 - 0x0000000000400614 is .eh_frame_hdr
    0x0000000000400618 - 0x000000000040070c is .eh_frame
    0x0000000000600e10 - 0x0000000000600e18 is .init_array
    0x0000000000600e18 - 0x0000000000600e20 is .fini_array
    0x0000000000600e20 - 0x0000000000600e28 is .jcr
    0x0000000000600e28 - 0x0000000000600ff8 is .dynamic
    0x0000000000600ff8 - 0x0000000000601000 is .got
    0x0000000000601000 - 0x0000000000601030 is .got.plt
    0x0000000000601030 - 0x0000000000601040 is .data
    0x0000000000601040 - 0x0000000000601048 is .bss
```

`start`

```
(gdb) start
Function "main" not defined.
```

main `readelf`
gdb `info files`

```
(gdb) b *0x400440
(gdb) r
```

- xmj
- weekface

```
/* a/file.c */
#include <stdio.h>

void print_a (void)
{
  puts ("a");
}

/* b/file.c */
#include <stdio.h>

void print_b (void)
{
  puts ("b");
}

/* main.c */
extern void print_a(void);
extern void print_b(void);

int main(void)
{
  print_a();
  print_b();
  return 0;
}
```

`b linenum`

```
(gdb) b 7
```

`b file:linenum`

```
(gdb) b file.c:6
Breakpoint 1 at 0x40053b: file.c:6. (2 locations)
(gdb) i breakpoints
Num
```

```
#include <stdio.h>
#include <pthread.h>

typedef struct
{

        int b;
        int c;
        int d;
        pthread_mutex_t mutex;
}ex_st;

int main(void) {
        ex_st st = {1, 2, 3, 4, PTHREAD_MUTEX_INITIALIZER};
        printf("%d,%d,%d,%d\n", st.a, st.b, st.c, st.d);
        return 0;
}
```

gdb

nanxiao

```
#include <stdio.h>

int main(void)
{
        int i = 0;
        int sum = 0;

        for (i = 1; i <= 200; i++)
        {
            sum += i;
        }

        printf("%d\n", sum);
        return 0;
}
```

gdb

“ `break … if cond` ”

```
(gdb) start
Temporary breakpoint 1 at 0x4004cc: file a.c, line
Starting program: /data2/home/nanxiao/a

Temporary breakpoint 1, main () at a.c:5
5                       int i = 0;
(gdb) b 10 if i==101
Breakpoint 2 at 0x4004e3: file a.c, line 1
(gdb) r
Starting program: /data2/home/nanxiao/a

Breakpoint 2, main () at a.c:10
10
(gdb) p sum
$1 = 5050
```

`i` `101` `sum` . ³ µXŒŒó'U) –†' \

```
#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
int a = 0;

void *thread1_func(void *p_arg)
{
        while (1)
        {
                a++;
                sleep(10);
        }
}

int main(int argc, char* argv[])
{
        pthread_t t1;

        pthread_create(&t1, NULL, thread1_func, "Thread 1");

        sleep(1000);
        return 0;
}
```

gdb “ `watch` ”

:

```
(gdb) start
Temporary breakpoint 1 at 0x4005a8: file a.c, line 19.
Starting program: /data2/home/nanxiao/a
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".

Temporary breakpoint 1, main () at a.c:19
19              pthread_create(&t1, NULL, thread1_func, "Thread 1")
(gdb) watch a
Hardware watchpoint 2: a
(gdb) r
Starting program: /data2/home/nanxiao/a
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
[New Thread 0x7ffff782c700 (LWP 8813)]
[Switching to Thread 0x7ffff782c700 (LWP 8813)]
Hardware watchpoint 2: a

Old value = 0
New value = 1
thread1_func (p_arg=0x4006d8) at a.c:11
11                      sleep(10);
(gdb) c
Continuing.
Hardware watchpoint 2: a

Old value = 1
New value = 2
thread1_func (p_arg=0x4006d8) at a.c:11
11                      sleep(10);
```

“ `watch a` ” `a` `0` `1` `1` `2`

“ `watch *(data type*)address` ” :

```
(gdb) p &a
$1 = (int *) 0x6009c8 <a>
(gdb) watch *(int*)0x6009c8
Hardware watchpoint 2: *(int*)0x6009c8
(gdb) r
Starting program: /data2/home/nanxiao/a
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
[New Thread 0x7ffff782c700 (LWP 15431)]
[Switching to Thread 0x7ffff782c700 (LWP 15431)]
Hardware watchpoint 2: *(int*)0x6009c8

Old value = 0
New value = 1
thread1_func (p_arg=0x4006d8) at a.c:11
11                      sleep(10);
(gdb) c
Continuing.
Hardware watchpoint 2: *(int*)0x6009c8

Old value = 1
New value = 2
thread1_func (p_arg=0x4006d8) at a.c:11
11                      sleep(10);
```

`a` `0x6009c8` “ `watch *(int*)0x6009c8` ”
“ `watch a` ”

gdb .

Hardware
watchpoint num: expr

set can-use-hw-watchpoints

info watchpoints

watch disable enable delete

nanxiao

```
#include <stdio.h>
#include <pthread.h>

int a = 0;

void *thread1_func(void *p_arg)
{
        while (1)
        {
                a++;
                sleep(10);
        }
}

void *thread2_func(void *p_arg)
{
        while (1)
        {
                a++;
                sleep(10);
        }
}

int main(void)
{
        pthread_t t1, t2;

        pthread_create(&t1, NULL, thread1_func, "Thread 1");
        pthread_create(&t2, NULL, thread2_func, "Thread 2");

        sleep(1000);
        return;
}
```

gdb “ `watch expr thread threadnum` ” `threadnum` :

```
(gdb) start
Temporary breakpoint 1 at 0x4005d4: file a.c, line 28.
Starting program: /data2/home/nanxiao/a
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".

Temporary breakpoint 1, main () at a.c:28
28              pthread_create(&t1, NULL, thread1_func, "Thread 1"
(gdb) n
[New Thread 0x7ffff782c700 (LWP 25443)]
29              pthread_create(&t2, NULL, thread2_func, "Thread 2"
(gdb)
[New Thread 0x7ffff6e2b700 (LWP 25444)]
31              sleep(1000);
(gdb) i threads
  Id   Target Id         Frame
  3    Thread 0x7ffff6e2b700 (LWP 25444) 0x00007ffff7915911 in clo
  2    Thread 0x7ffff782c700 (LWP 25443) 0x00007ffff78d9bcd in nan
* 1    Thread 0x7ffff7fe9700 (LWP 25413) main () at a.c:31
(gdb) wa a thread 2
Hardware watchpoint 2: a
(gdb) c
Continuing.
[Switching to Thread 0x7ffff782c700 (LWP 25443)]
Hardware watchpoint 2: a

Old value = 1
New value = 3
thread1_func (p_arg=0x400718) at a.c:11
11                      sleep(10);
(gdb) c
Continuing.
Hardware watchpoint 2: a

Old value = 3
New value = 5
thread1_func (p_arg=0x400718) at a.c:11
11                      sleep(10);
(gdb) c
Continuing.
Hardware watchpoint 2: a

Old value = 5
New value = 7
thread1_func (p_arg=0x400718) at a.c:11
11                      sleep(10);
```

“`wa a thread 2`” `wa` `watch` `thread1_func` `a`

gdb

.

nanxiao

```
#include <stdio.h>
#include <pthread.h>

int a = 0;

void *thread1_func(void *p_arg)
{
        while (1)
        {
                printf("%d\n", a);
                sleep(10);
        }
}

int main(void)
{
        pthread_t t1;

        pthread_create(&t1, NULL, thread1_func, "Thread 1");

        sleep(1000);
        return;
}
```

gdb “ `rwatch` ”
:

```
(gdb) start
Temporary breakpoint 1 at 0x4005f3: file a.c, line 19.
Starting program: /data2/home/nanxiao/a
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".

Temporary breakpoint 1, main () at a.c:19
19              pthread_create(&t1, NULL, thread1_func, "Thread 1"
(gdb) rw a
Hardware read watchpoint 2: a
(gdb) c
Continuing.
[New Thread 0x7ffff782c700 (LWP 5540)]
[Switching to Thread 0x7ffff782c700 (LWP 5540)]
Hardware read watchpoint 2: a

Value = 0
0x00000000004005c6 in thread1_func (p_arg=0x40071c) at a.c:10
10                      printf("%d\n", a);
(gdb) c
Continuing.
0
Hardware read watchpoint 2: a

Value = 0
0x00000000004005c6 in thread1_func (p_arg=0x40071c) at a.c:10
10                      printf("%d\n", a);
(gdb) c
Continuing.
0
Hardware read watchpoint 2: a

Value = 0
0x00000000004005c6 in thread1_func (p_arg=0x40071c) at a.c:10
10                      printf("%d\n", a);
```

“`rw a`” `rw` `rwatch`
`a`
`rwatch` gdb .

nanxiao

```
#include <stdio.h>
#include <pthread.h>

int a = 0;

void *thread1_func(void *p_arg)
{
        while (1)
        {
                a++;
                sleep(10);
        }
}

void *thread2_func(void *p_arg)
{
        while (1)
        {
                printf("%d\n", a);;
                sleep(10);
        }
}

int main(void)
{
        pthread_t t1, t2;

        pthread_create(&t1, NULL, thread1_func, "Thread 1");
        pthread_create(&t2, NULL, thread2_func, "Thread 2");

        sleep(1000);
        return;
}
```

gdb “ `awatch` ”

:

```
(gdb) aw a
Hardware access (read/write) watchpoint 1: a
(gdb) r
Starting program: /data2/home/nanxiao/a
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
[New Thread 0x7ffff782c700 (LWP 16938)]
[Switching to Thread 0x7ffff782c700 (LWP 16938)]
Hardware access (read/write) watchpoint 1: a

Value = 0
0x00000000004005c6 in thread1_func (p_arg=0x40076c) at a.c:10
10                      a++;
(gdb) c
Continuing.
Hardware access (read/write) watchpoint 1: a

Old value = 0
New value = 1
thread1_func (p_arg=0x40076c) at a.c:11
11                      sleep(10);
(gdb) c
Continuing.
[New Thread 0x7ffff6e2b700 (LWP 16939)]
[Switching to Thread 0x7ffff6e2b700 (LWP 16939)]
Hardware access (read/write) watchpoint 1: a

Value = 1
0x00000000004005f2 in thread2_func (p_arg=0x400775) at a.c:19
19                      printf("%d\n", a);;
(gdb) c
Continuing.
1
[Switching to Thread 0x7ffff782c700 (LWP 16938)]
Hardware access (read/write) watchpoint 1: a

Value = 1
0x00000000004005c6 in thread1_func (p_arg=0x40076c) at a.c:10
10                      a++;
```

“`aw a`” `aw` `awatch`
`a`
`awatch` gdb .

nanxiao

Catchpoint

catchpoint

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>

int main(void) {
    pid_t pid;
    int i = 0;

    for (i = 0; i < 2; i++)
    {
        pid = fork();
        if (pid < 0)
        {
            exit(1);
        }
        else if (pid == 0)
        {
            exit(0);
        }
    }
    printf("hello world\n");
    return 0;
}
```

gdb “ `tcatch` ” `catchpoint`

```
(gdb) tcatch fork
Catchpoint 1 (fork)
(gdb) r
Starting program: /home/nan/a

Temporary catchpoint 1 (forked process 27377), 0x00000034e42acdbd i
(gdb) c
Continuing.
hello world
[Inferior 1 (process 27373) exited normally]
(gdb) q
```

`fork`

gdb .

nanxiao

fork catchpoint

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>

int main(void) {
    pid_t pid;

    pid = fork();
    if (pid < 0)
    {
        exit(1);
    }
    else if (pid > 0)
    {
        exit(0);
    }
    printf("hello world\n");
    return 0;
}
```

gdb “ `catch fork` ” `fork` `catchpoint`

```
(gdb) catch fork
Catchpoint 1 (fork)
(gdb) r
Starting program: /home/nan/a

Catchpoint 1 (forked process 33499), 0x00000034e42acdbd in fork ()
(gdb) bt
#0  0x00000034e42acdbd in fork () from /lib64/libc.so.6
#1  0x0000000000400561 in main () at a.c:9
```

`fork` gdb HP-UX GNU/Linux gdb .

nanxiao

vfork catchpoint

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>

int main(void) {
    pid_t pid;

    pid = vfork();
    if (pid < 0)
    {
        exit(1);
    }
    else if (pid > 0)
    {
        exit(0);
    }
    printf("hello world\n");
    return 0;
}
```

gdb “ `catch vfork` ” `vfork` `catchpoint`

```
(gdb) catch vfork
Catchpoint 1 (vfork)
(gdb) r
Starting program: /home/nan/a

Catchpoint 1 (vforked process 27312), 0x00000034e42acfc4 in vfork (
   from /lib64/libc.so.6
(gdb) bt
#0  0x00000034e42acfc4 in vfork () from /lib64/libc.so.6
#1  0x0000000000400561 in main () at a.c:9
```

`vfork` gdb
HP-UX GNU/Linux
gdb .

nanxiao

exec catchpoint

```
#include <unistd.h>

int main(void) {
    execl("/bin/ls", "ls", NULL);
    return 0;
}
```

gdb “ `catch exec` ” `exec` `catchpoint`

```
(gdb) catch exec
Catchpoint 1 (exec)
(gdb) r
Starting program: /home/nan/a
process 32927 is executing new program: /bin/ls

Catchpoint 1 (exec'd /bin/ls), 0x00000034e3a00b00 in _start () fror
(gdb) bt
#0  0x00000034e3a00b00 in _start () from /lib64/ld-linux-x86-64.so
#1  0x0000000000000001 in ?? ()
#2  0x00007fffffffe73d in ?? ()
#3  0x0000000000000000 in ?? ()
```

`execl` gdb HP-UX GNU/Linux

gdb .

nanxiao

catchpoint

```
#include <stdio.h>

int main(void)
{
    char p1[] = "Sam";
    char *p2 = "Bob";

    printf("p1 is %s, p2 is %s\n", p1, p2);
    return 0;
}
```

gdb `catch syscall [name | number]` `catchpoint`

```
(gdb) catch syscall mmap
Catchpoint 1 (syscall 'mmap' [9])
(gdb) r
Starting program: /home/nan/a

Catchpoint 1 (call to syscall mmap), 0x00000034e3a16f7a in mmap64 (
   from /lib64/ld-linux-x86-64.so.2
(gdb) c
Continuing.

Catchpoint 1 (returned from syscall mmap), 0x00000034e3a16f7a in mr
   from /lib64/ld-linux-x86-64.so.2
```

`mmap` gdb `catchpoint`

```
(gdb) catch syscall 9
Catchpoint 1 (syscall 'mmap' [9])
(gdb) r
Starting program: /home/nan/a

Catchpoint 1 (call to syscall mmap), 0x00000034e3a16f7a in mmap64 (
   from /lib64/ld-linux-x86-64.so.2
(gdb) c
Continuing.

Catchpoint 1 (returned from syscall mmap), 0x00000034e3a16f7a in mm
   from /lib64/ld-linux-x86-64.so.2
(gdb) c
Continuing.

Catchpoint 1 (call to syscall mmap), 0x00000034e3a16f7a in mmap64 (
   from /lib64/ld-linux-x86-64.so.2
```

`catch syscall mmap`
`xml`
`/usr/local/share/gdb/syscalls` `amd64-linux.xml`
`catchpoint`

```
(gdb) catch syscall
Catchpoint 1 (any syscall)
(gdb) r
Starting program: /home/nan/a

Catchpoint 1 (call to syscall brk), 0x00000034e3a1618a in brk ()
   from /lib64/ld-linux-x86-64.so.2
(gdb) c
Continuing.

Catchpoint 1 (returned from syscall brk), 0x00000034e3a1618a in brk
   from /lib64/ld-linux-x86-64.so.2
(gdb)
Continuing.

Catchpoint 1 (call to syscall mmap), 0x00000034e3a16f7a in mmap64 (
   from /lib64/ld-linux-x86-64.so.2
```

gdb .

nanxiao

ptrace catchpoint anti-debugging

```
#include <sys/ptrace.h>
#include <stdio.h>

int main()
{
        if (ptrace(PTRACE_TRACEME, 0, 0, 0) < 0 ) {
                printf("Gdb is debugging me, exit.\n");
                return 1;
        }
        printf("No debugger, continuing\n");
        return 0;
}
```

gdb “ `ptrace` ” gdb

```
(gdb) start
Temporary breakpoint 1 at 0x400508: file a.c, line 6.
Starting program: /data2/home/nanxiao/a

Temporary breakpoint 1, main () at a.c:6
6                       if (ptrace(PTRACE_TRACEME, 0, 0, 0) < 0 )
(gdb) n
7                               printf("Gdb is debugging me, exit.
(gdb)
Gdb is debugging me, exit.
8                               return 1;
```

`ptrace` `catchpoint` `ptrace`

```
(gdb) catch syscall ptrace
Catchpoint 2 (syscall 'ptrace' [101])
(gdb) r
Starting program: /data2/home/nanxiao/a

Catchpoint 2 (call to syscall ptrace), 0x00007ffff7b2be9c in ptrace
(gdb) c
Continuing.

Catchpoint 2 (returned from syscall ptrace), 0x00007ffff7b2be9c in
(gdb) set $rax = 0
(gdb) c
Continuing.
No debugger, continuing
[Inferior 1 (process 11491) exited normally]
```

`rax` gdb
“ `No debugger, continuing` ”
PTRACE_TRACME .

nanxiao

ASCII

```
#include <stdio.h>
#include <wchar.h>

int main(void)
{
        char str1[] = "abcd";
        wchar_t str2[] = L"abcd";

        return 0;
}
```

gdb “ `x/s` ” ASCII

```
Temporary breakpoint 1, main () at a.c:6
6               char str1[] = "abcd";
(gdb) n
7               wchar_t str2[] = L"abcd";
(gdb)
9               return 0;
(gdb) x/s str1
0x804779f:      "abcd"
```

`str1`

```
Temporary breakpoint 1, main () at a.c:6
6               char str1[] = "abcd";
(gdb) n
7               wchar_t str2[] = L"abcd";
(gdb)
9               return 0;
(gdb) p sizeof(wchar_t)
$1 = 4
(gdb) x/ws str2
0x8047788:      U"abcd"
```

4 “ `x/ws` ” 2 “ `x/hs` ”

gdb .

nanxiao

STL

```
#include <iostream>
#include <vector>

using namespace std;

int main ()
{
  vector<int> vec(10); // 10 zero-initialized elements

  for (int i = 0; i < vec.size(); i++)
    vec[i] = i;

  cout << "vec contains:";
  for (int i = 0; i < vec.size(); i++)
    cout << ' ' << vec[i];
  cout << '\n';

  return 0;
}
```

gdb C++ STL

```
(gdb) p vec
$1 = {<std::_Vector_base<int, std::allocator<int> >> = {
    _M_impl = {<std::allocator<int>> = {<__gnu_cxx::new_allocator<
          _M_end_of_storage = 0x404038}}, <No data fields>}
```

gdb 7.0 gcc python

```
(gdb) p vec
$1 = std::vector of length 10, capacity 10 = {0, 1, 2, 3, 4, 5, 6,
```

(Fedora 11+) gdb

```
(gdb) info pretty-printer
```

:

1. python gcc

   ```
   sudo find / -name "*libstdcxx*"
   ```

2. python gcc

   ```
   gcc-4.8.1/libstdc++-v3/python
   ```

3.

   ```
   svn co svn://gcc.gnu.org/svn/gcc/trunk/libstdc++-v3/python
   ```

4. .gdbinit python /home/maude/gdb_printers/

   ```
   python
   import sys
   sys.path.insert(0, '/home/maude/gdb_printers/python')
   from libstdcxx.v6.printers import register_libstdcxx_printers
   register_libstdcxx_printers (None)
   end
   ```

   https://sourceware.org/gdb/wiki/STLSupport

`p vec`

```
(gdb) p *(vec._M_impl._M_start)@vec.size()
$2 = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
```

dbinit_stl_views ,

```
cat dbinit_stl_views-1.03.txt >> ~/.gdbinit
```

```
std::vector  pvector stl_variable
std::list  plist stl_variable T
std::map  pmap stl_variable
std::multimap  pmap stl_variable
std::set  pset stl_variable T
std::multiset  pset stl_variable
std::deque  pdequeue stl_variable
std::stack  pstack stl_variable
std::queue  pqueue stl_variable
std::priority_queue  ppqueue stl_variable
std::bitset  pbitset stl_variable
std::string  pstring stl_variable
std::widestring  pwstring stl_variable
```

xmj

xanpeng

enjolras

```
int main()
{
  int array[201];
  int i;

  for (i = 0; i < 201; i++)
    array[i] = i;

  return 0;
}
```

gdb 200

```
(gdb) p array
$1 = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,
  48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 6
  95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108,
  133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145,
  170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182,
```

```
(gdb) set print elements number-of-elements
```

```
(gdb) set print elements 0
```

```
(gdb) set print elements unlimited
(gdb) p array
$2 = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,
  48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 6
  95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108,
  133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145,
  170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182,
```

gdb

xmj

```
int main(void)
{
  int array[201];
  int i;

  for (i = 0; i < 201; i++)
    array[i] = i;

  return 0;
}
```

gdb
“ `p array[index]@num` ” `p` `print` `index`
0 `num`

```
(gdb) p array
$8 = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,
  32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47,
  63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78,
  94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107,
  120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132,
  145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157,
  170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182,
  195, 196, 197, 198, 199...}
(gdb) p array[60]@10
$9 = {60, 61, 62, 63, 64, 65, 66, 67, 68, 69}
```

`array` 60~69
“ `p *array@num` ”

```
(gdb) p *array@10
$2 = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
```

gdb

nanxiao

```
#include <stdio.h>

int num[10] = {
  1 << 0,
  1 << 1,
  1 << 2,
  1 << 3,
  1 << 4,
  1 << 5,
  1 << 6,
  1 << 7,
  1 << 8,
  1 << 9
};

int main (void)
{
  int i;

  for (i = 0; i < 10; i++)
    printf ("num[%d] = %d\n", i, num[i]);

  return 0;
}
```

gdb

```
(gdb) p num
$1 = {1, 2, 4, 8, 16, 32, 64, 128, 256, 512}
```

```
(gdb) set print array-indexes on

(gdb) p num
$2 = {[0] = 1, [1] = 2, [2] = 4, [3] = 8, [4] = 16, [5] = 32, [6]
```

gdb

xmj

```
#include <stdio.h>

void fun_a(void)
{
    int a = 0;
    printf("%d\n", a);
}

void fun_b(void)
{
    int b = 1;
    fun_a();
    printf("%d\n", b);
}

void fun_c(void)
{
    int c = 2;
    fun_b();
    printf("%d\n", c);
}

void fun_d(void)
{
    int d = 3;
    fun_c();
    printf("%d\n", d);
}

int main(void)
{
    int var = -1;
    fun_d();
    return 0;
}
```

“bt full” bt backtrace
fun_a

```
(gdb) bt
#0  fun_a () at a.c:6
#1  0x000109b0 in fun_b () at a.c:12
#2  0x000109e4 in fun_c () at a.c:19
#3  0x00010a18 in fun_d () at a.c:26
#4  0x00010a4c in main () at a.c:33
```

“bt full”

```
(gdb) bt full
#0  fun_a () at a.c:6
        a = 0
#1  0x000109b0 in fun_b () at a.c:12
        b = 1
#2  0x000109e4 in fun_c () at a.c:19
        c = 2
#3  0x00010a18 in fun_d () at a.c:26
        d = 3
#4  0x00010a4c in main () at a.c:33
        var = -1
```

“bt full n” n

```
(gdb) bt full 2
#0  fun_a () at a.c:6
        a = 0
#1  0x000109b0 in fun_b () at a.c:12
        b = 1
(More stack frames follow...)
```

“bt full -n” n

```
(gdb) bt full -2
#3  0x00010a18 in fun_d () at a.c:26
        d = 3
#4  0x00010a4c in main () at a.c:33
        var = -1
```

gdb

```
(gdb) info locals
a = 0
```

gdb

nanxiao

xmj

gdb “i proc mappings”
i info :

```
(gdb) i proc mappings
process 27676 flags:
PR_STOPPED Process (LWP) is stopped
PR_ISTOP Stopped on an event of interest
PR_RLC Run-on-last-close is in effect
PR_MSACCT Microstate accounting enabled
PR_PCOMPAT Micro-state accounting inherited on fork
PR_FAULTED : Incurred a traced hardware fault FLTBPT: Breakpoint t

Mapped address spaces:

    Start Addr   End Addr       Size     Offset   Flags
     0x8046000  0x8047fff     0x2000 0xfffff000 -s--rwx
     0x8050000  0x8050fff     0x1000          0 ----r-x
     0x8060000  0x8060fff     0x1000          0 ----rwx
    0xfee40000 0xfef4efff   0x10f000          0 ----r-x
    0xfef50000 0xfef55fff     0x6000          0 ----rwx
    0xfef5f000 0xfef66fff     0x8000   0x10f000 ----rwx
    0xfef67000 0xfef68fff     0x2000          0 ----rwx
    0xfef70000 0xfef70fff     0x1000          0 ----rwx
    0xfef80000 0xfef80fff     0x1000          0 ---sr--
    0xfef90000 0xfef90fff     0x1000          0 ----rw-
    0xfefa0000 0xfefa0fff     0x1000          0 ----rw-
    0xfefb0000 0xfefb0fff     0x1000          0 ----rwx
    0xfefc0000 0xfefeafff    0x2b000          0 ----r-x
    0xfeff0000 0xfeff0fff     0x1000          0 ----rwx
    0xfeffb000 0xfeffcfff     0x2000    0x2b000 ----rwx
    0xfeffd000 0xfeffdfff     0x1000          0 ----rwx
```

flags
gdb .

"i files" “i target”

```
(gdb) i files
Symbols from "/data1/nan/a".
Unix /proc child process:
    Using the running image of child Thread 1 (LWP 1) via /proc.
    While running this, GDB does not access memory from...
Local exec file:
    `/data1/nan/a', file type elf32-i386-sol2.
    Entry point: 0x8050950
    0x080500f4 - 0x08050105 is .interp
    0x08050108 - 0x08050114 is .eh_frame_hdr
    0x08050114 - 0x08050218 is .hash
    0x08050218 - 0x08050418 is .dynsym
    0x08050418 - 0x080507e6 is .dynstr
    0x080507e8 - 0x08050818 is .SUNW_version
    0x08050818 - 0x08050858 is .SUNW_versym
    0x08050858 - 0x08050890 is .SUNW_reloc
    0x08050890 - 0x080508c8 is .rel.plt
    0x080508c8 - 0x08050948 is .plt
    ......
    0xfef5fb58 - 0xfef5fc48 is .dynamic in /usr/lib/libc.so.1
    0xfef5fc80 - 0xfef650e2 is .data in /usr/lib/libc.so.1
    0xfef650e2 - 0xfef650e2 is .bssf in /usr/lib/libc.so.1
    0xfef650e8 - 0xfef65be0 is .picdata in /usr/lib/libc.so.1
    0xfef65be0 - 0xfef666a7 is .data1 in /usr/lib/libc.so.1
    0xfef666a8 - 0xfef680dc is .bss in /usr/lib/libc.so.1
```

gdb

nanxiao

```
/* main.c */
extern void print_var_1(void);
extern void print_var_2(void);

int main(void)
{
  print_var_1();
  print_var_2();
  return 0;
}

/* static-1.c */
#include <stdio.h>

static int var = 1;

void print_var_1(void)
{
  printf("var = %d\n", var);
}

/* static-2.c */
#include <stdio.h>

static int var = 2;

void print_var_2(void)
{
  printf("var = %d\n", var);
}
```

gdb

```
$ gcc -g main.c static-1.c static-2.c
$ gdb -q ./a.out
(gdb) start
(gdb) p var
$1 = 2

$ gcc -g main.c static-2.c static-1.c
$ gdb -q ./a.out
(gdb) start
(gdb) p var
$1 = 1
```

```
(gdb) p 'static-1.c'::var
$1 = 1
(gdb) p 'static-2.c'::var
$2 = 2
```

gdb

xmj

```
#include <stdio.h>

struct child {
  char name[10];
  enum { boy, girl } gender;
};

struct child he = { "Tom", boy };

int main (void)
{
  static struct child she = { "Jerry", girl };
  printf ("Hello %s %s.\n", he.gender == boy ? "boy" : "girl", he.r
  printf ("Hello %s %s.\n", she.gender == boy ? "boy" : "girl", she
  return 0;
}
```

gdb

```
(gdb) whatis he
type = struct child
```

```
(gdb) ptype he
type = struct child {
    char name[10];
    enum {boy, girl} gender;
}
```

```
(gdb) i variables he
All variables matching regular expression "he":

File variable.c:
struct child he;

Non-debugging symbols:
0x0000000000402030  she
0x00007ffff7dd3380  __check_rhosts_file
```

gdb

```
(gdb) i variables ^he$
All variables matching regular expression "^he$":

File variable.c:
struct child he;
```

`info variables` static

gdb

xmj

```
#include <stdio.h>

int main(void)
{
        int i = 0;
        char a[100];

        for (i = 0; i < sizeof(a); i++)
        {
                a[i] = i;
        }

        return 0;
}
```

gdb “ `x` ” “ `x/nfu addr` ” `f` `addr` `n` `u`
a n
b f `x` 16 `o` 8 ,
c u `b` `byte` `h` `byte` halfword `w` `byte` word `g` `byte` giant word

1 16 `a` 16 byte

```
(gdb) x/16xb a
0x7fffffffe4a0: 0x00    0x01    0x02    0x03    0x04    0x05    0x
0x7fffffffe4a8: 0x08    0x09    0x0a    0x0b    0x0c    0x0d    0x
```

2 10 `a` 16 byte

```
(gdb) x/16ub a
0x7fffffffe4a0: 0       1       2       3       4       5       6
0x7fffffffe4a8: 8       9       10      11      12      13      14
```

3 2 16 `a` byte

```
(gdb) x/16tb a
0x7fffffffe4a0: 00000000        00000001        00000010        000
0x7fffffffe4a8: 00001000        00001001        00001010        000
```

4 16 `a` 16 word 4 byte

```
(gdb) x/16xw a
0x7fffffffe4a0: 0x03020100      0x07060504      0x0b0a0908      0x0
0x7fffffffe4b0: 0x13121110      0x17161514      0x1b1a1918      0x1
0x7fffffffe4c0: 0x23222120      0x27262524      0x2b2a2928      0x2
0x7fffffffe4d0: 0x33323130      0x37363534      0x3b3a3938      0x3
```

gdb .

nanxiao

```
$ gdb -q `which gdb`
(gdb) l
15
16        You should have received a copy of the GNU General Public
17        along with this program.  If not, see <http://www.gnu.org/
18
19     #include "defs.h"
20     #include "main.h"
21     #include <string.h>
22     #include "interps.h"
23
24     int
```

gdb `list` l
`list`

```
(gdb) l 24
(gdb) l main
```

```
(gdb) l -
(gdb) l +
```

```
(gdb) l 1,10
```

gdb

xmj

```
#include <stdio.h>
#include <pthread.h>

typedef struct
{
        int a;
        int b;
        int c;
        int d;
        pthread_mutex_t mutex;
}ex_st;

int main(void) {
        ex_st st = {1, 2, 3, 4, PTHREAD_MUTEX_INITIALIZER};
        printf("%d,%d,%d,%d\n", st.a, st.b, st.c, st.d);
        return 0;
}
```

gdb “ ”

```
(gdb) n
15              printf("%d,%d,%d,%d\n", st.a, st.b, st.c, st.d);
(gdb) p st
$1 = {a = 1, b = 2, c = 3, d = 4, mutex = {__data = {__lock = 0, __
      __spins = 0, __list = {__prev = 0x0, __next = 0x0}}, __size =
```

“set print pretty on”

```
(gdb) set print pretty on
(gdb) p st
$2 = {
  a = 1,
  b = 2,
  c = 3,
  d = 4,
  mutex = {
    __data = {
      __lock = 0,
      __count = 0,
      __owner = 0,
      __nusers = 0,
      __kind = 0,
      __spins = 0,
      __list = {
        __prev = 0x0,
        __next = 0x0
      }
    },
    __size = '\000' <repeats 39 times>,
    __align = 0
  }
}
```

gdb

nanxiao

```
#include <iostream>
using namespace std;

class Shape {
 public:
  virtual void draw () {}
};

class Circle : public Shape {
 int radius;
 public:
  Circle () { radius = 1; }
  void draw () { cout << "drawing a circle...\n"; }
};

class Square : public Shape {
 int height;
 public:
  Square () { height = 2; }
  void draw () { cout << "drawing a square...\n"; }
};

void drawShape (class Shape &p)
{
  p.draw ();
}

int main (void)
{
  Circle a;
  Square b;
  drawShape (a);
  drawShape (b);
  return 0;
}
```

gdb

```
(gdb) frame
#0  drawShape (p=...) at object.cxx:25
25      p.draw ();
(gdb) p p
$1 = (Shape &) @0x7fffffffde90: {_vptr.Shape = 0x400a80 <vtable for
```

p class Shape class Circle Square

```
(gdb) set print object on

(gdb) p p
$2 = (Circle &) @0x7fffffffde90: {<Shape> = {_vptr.Shape = 0x400a8
```

```
(gdb) whatis p
type = Shape &
(gdb) ptype p
type = class Shape {
  public:
    virtual void draw(void);
} &

(gdb) set print object on
(gdb) whatis p
type = /* real type = Circle & */
Shape &
(gdb) ptype p
type = /* real type = Circle & */
class Shape {
  public:
    virtual void draw(void);
} &
```

gdb

xmj

xanpeng

```
#include <stdio.h>

int main(void)
{
  int i;

  for (i = 0; i < 100; i++)
    {
      printf("i = %d\n", i);
    }

  return 0;
}
```

gdb gdb

```
$ tty
/dev/pts/2
```

```
$ gdb -tty /dev/pts/2 ./a.out
(gdb) r
```

gdb

```
(gdb) tty /dev/pts/2
```

gdb

xmj

“$_” “$__”

```
#include <stdio.h>

int main(void)
{
        int i = 0;
        char a[100];

        for (i = 0; i < sizeof(a); i++)
        {
                a[i] = i;
        }

        return 0;
}
```

" `x` " “ `$_` ” “convenience variable” “ `$__` ” “convenience variable”
:

```
(gdb) b a.c:13
Breakpoint 1 at 0x4004a0: file a.c, line 13.
(gdb) r
Starting program: /data2/home/nanxiao/a

Breakpoint 1, main () at a.c:13
13              return 0;
(gdb) x/16xb a
0x7fffffffe4a0: 0x00    0x01    0x02    0x03    0x04    0x05    0x0
0x7fffffffe4a8: 0x08    0x09    0x0a    0x0b    0x0c    0x0d    0x0
(gdb) p $_
$1 = (int8_t *) 0x7fffffffe4af
(gdb) p $__
$2 = 15
```

“ `$_` ” `0x7fffffffe4af` " `x` "
“ `$__` ” `15`
“ `info line` ” “ `info breakpoint` ”
" `x` " “ `$_` ”

```
(gdb) p $_
$5 = (int8_t *) 0x7fffffffe4af
(gdb) info breakpoint
Num     Type           Disp Enb Address            What
1       breakpoint     keep y   0x00000000004004a0 in main at a.c:1
        breakpoint already hit 1 time
(gdb) p $_
$6 = (void *) 0x4004a0 <main+44>
```

“ `info breakpoint` ” “ `$_` ” `0x4004a0`
gdb .

nanxiao

```
#include <stdio.h>
#include <malloc.h>

int main(void)
{
        char *p[10];
        int i = 0;

        for (i = 0; i < sizeof(p)/sizeof(p[0]); i++)
        {
                p[i] = malloc(100000);
        }
        return 0;
}
```

gdb

```
define mallocinfo
  set $__f = fopen("/dev/tty", "w")
  call malloc_info(0, $__f)
  call fclose($__f)
end
```

```
Temporary breakpoint 5, main () at a.c:7
7               int i = 0;
(gdb) mallocinfo
<malloc version="1">
<heap nr="0">
<sizes>
</sizes>
<total type="fast" count="0" size="0"/>
<total type="rest" count="0" size="0"/>
<system type="current" size="135168"/>
<system type="max" size="135168"/>
<aspace type="total" size="135168"/>
```

```
<aspace type="mprotect" size="135168"/>
</heap>
<total type="fast" count="0" size="0"/>
<total type="rest" count="0" size="0"/>
<system type="current" size="135168"/>
<system type="max" size="135168"/>
<aspace type="total" size="135168"/>
<aspace type="mprotect" size="135168"/>
</malloc>
$20 = 0
$21 = 0
(gdb) n
9               for (i = 0; i < sizeof(p)/sizeof(p[0]); i++)
(gdb)
11                      p[i] = malloc(100000);
(gdb)
9               for (i = 0; i < sizeof(p)/sizeof(p[0]); i++)
(gdb)
11                      p[i] = malloc(100000);
(gdb)
9               for (i = 0; i < sizeof(p)/sizeof(p[0]); i++)
(gdb)
11                      p[i] = malloc(100000);
(gdb)
9               for (i = 0; i < sizeof(p)/sizeof(p[0]); i++)
(gdb)
11                      p[i] = malloc(100000);
(gdb)
9               for (i = 0; i < sizeof(p)/sizeof(p[0]); i++)
(gdb)
11                      p[i] = malloc(100000);
(gdb) mallocinfo
<malloc version="1">
<heap nr="0">
<sizes>
</sizes>
<total type="fast" count="0" size="0"/>
<total type="rest" count="0" size="0"/>
<system type="current" size="532480"/>
<system type="max" size="532480"/>
<aspace type="total" size="532480"/>
<aspace type="mprotect" size="532480"/>
</heap>
<total type="fast" count="0" size="0"/>
<total type="rest" count="0" size="0"/>
<system type="current" size="532480"/>
<system type="max" size="532480"/>
<aspace type="total" size="532480"/>
<aspace type="mprotect" size="532480"/>
</malloc>
$22 = 0
$23 = 0
(gdb) n
```

```
9               for (i = 0; i < sizeof(p)/sizeof(p[0]); i++)
(gdb)
11                      p[i] = malloc(100000);
(gdb)
9               for (i = 0; i < sizeof(p)/sizeof(p[0]); i++)
(gdb)
11                      p[i] = malloc(100000);
(gdb)
9               for (i = 0; i < sizeof(p)/sizeof(p[0]); i++)
(gdb)
11                      p[i] = malloc(100000);
(gdb)
9               for (i = 0; i < sizeof(p)/sizeof(p[0]); i++)
(gdb)
11                      p[i] = malloc(100000);
(gdb)
9               for (i = 0; i < sizeof(p)/sizeof(p[0]); i++)
(gdb)
11                      p[i] = malloc(100000);
(gdb)
9               for (i = 0; i < sizeof(p)/sizeof(p[0]); i++)
(gdb) mallocinfo
<malloc version="1">
<heap nr="0">
<sizes>
</sizes>
<total type="fast" count="0" size="0"/>
<total type="rest" count="0" size="0"/>
<system type="current" size="1134592"/>
<system type="max" size="1134592"/>
<aspace type="total" size="1134592"/>
<aspace type="mprotect" size="1134592"/>
</heap>
<total type="fast" count="0" size="0"/>
<total type="rest" count="0" size="0"/>
<system type="current" size="1134592"/>
<system type="max" size="1134592"/>
<aspace type="total" size="1134592"/>
<aspace type="mprotect" size="1134592"/>
</malloc>
$24 = 0
$25 = 0
```

gdb
stackoverflow.

nanxiao

```
#include <stdio.h>

int func1(int a)
{
  int b = 1;
  return b * a;
}

int func2(int a)
{
  int b = 2;
  return b * func1(a);
}

int func3(int a)
{
  int b = 3;
  return b * func2(a);
}

int main(void)
{
  printf("%d\n", func3(10));
  return 0;
}
```

gdb

```
(gdb) b func1
(gdb) r
(gdb) bt
#0  func1 (a=10) at frame.c:5
#1  0x0000000000400560 in func2 (a=10) at frame.c:12
#2  0x0000000000400582 in func3 (a=10) at frame.c:18
#3  0x0000000000400596 in main () at frame.c:23
(gdb) f 1
(gdb) p b
(gdb) f 2
(gdb) p b
```

```
(gdb) p func2::b
$1 = 2
(gdb) p func3::b
$2 = 3
```

C++

```
(gdb) p '(anonymous namespace)::SSAA::handleStore'::n->pi->inst->du
```

gdb

xmj

/

```
#include <stdio.h>
#include <pthread.h>
void *thread_func(void *p_arg)
{
        while (1)
        {
                printf("%s\n", (char*)p_arg);
                sleep(10);
        }
}
int main(void)
{
        pthread_t t1, t2;

        pthread_create(&t1, NULL, thread_func, "Thread 1");
        pthread_create(&t2, NULL, thread_func, "Thread 2");

        sleep(1000);
        return;
}
```

gdb ID gdb program
processID -p --pid ID gdb program -p=10210
“ps” ID 10210

```
bash-3.2# gdb -q a 10210
Reading symbols from /data/nan/a...done.
Attaching to program `/data/nan/a', process 10210
[New process 10210]
Retry #1:
Retry #2:
Retry #3:
Retry #4:
Reading symbols from /usr/lib/libc.so.1...(no debugging symbols fou
[Thread debugging using libthread_db enabled]
[New LWP    3        ]
[New LWP    2        ]
[New Thread 1 (LWP 1)]
[New Thread 2 (LWP 2)]
[New Thread 3 (LWP 3)]
Loaded symbols for /usr/lib/libc.so.1
Reading symbols from /lib/ld.so.1...(no debugging symbols found)..
Loaded symbols for /lib/ld.so.1
[Switching to Thread 1 (LWP 1)]
0xfeeeae55 in ___nanosleep () from /usr/lib/libc.so.1
(gdb) bt
#0  0xfeeeae55 in ___nanosleep () from /usr/lib/libc.so.1
#1  0xfeedcae4 in sleep () from /usr/lib/libc.so.1
#2  0x080509ef in main () at a.c:17
```

ps

```
#      xgdb.sh
#     xgdb.sh a
prog_bin=$1
running_name=$(basename $prog_bin)
pid=$(/sbin/pidof $running_name)
gdb attach $pid
```

gdb “attach” “ ”

```
bash-3.2# gdb -q a
Reading symbols from /data/nan/a...done.
(gdb) attach 10210
Attaching to program `/data/nan/a', process 10210
[New process 10210]
Retry #1:
Retry #2:
Retry #3:
Retry #4:
Reading symbols from /usr/lib/libc.so.1...(no debugging symbols fou
[Thread debugging using libthread_db enabled]
[New LWP    3        ]
[New LWP    2        ]
[New Thread 1 (LWP 1)]
[New Thread 2 (LWP 2)]
[New Thread 3 (LWP 3)]
Loaded symbols for /usr/lib/libc.so.1
Reading symbols from /lib/ld.so.1...(no debugging symbols found)..
Loaded symbols for /lib/ld.so.1
[Switching to Thread 1 (LWP 1)]
0xfeeeae55 in ___nanosleep () from /usr/lib/libc.so.1
(gdb) bt
#0  0xfeeeae55 in ___nanosleep () from /usr/lib/libc.so.1
#1  0xfeedcae4 in sleep () from /usr/lib/libc.so.1
#2  0x080509ef in main () at a.c:17
```

“detach” “ ”

```
(gdb) detach
Detaching from program: /data/nan/a, process 10210
(gdb) bt
No stack.
```

gdb

nanxiao

```
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>

int main(void) {
    pid_t pid;

    pid = fork();
    if (pid < 0)
    {
        exit(1);
    }
    else if (pid > 0)
    {
        exit(0);
    }
    printf("hello world\n");
    return 0;
}
```

gdb

```
(gdb) start
Temporary breakpoint 1 at 0x40055c: file a.c, line 8.
Starting program: /data2/home/nanxiao/a

Temporary breakpoint 1, main () at a.c:8
8               pid = fork();
(gdb) n
9               if (pid < 0)
(gdb) hello world

13              else if (pid > 0)
(gdb)
15                      exit(0);
(gdb)
[Inferior 1 (process 12786) exited normally]
```

15

“set follow-fork-mode child”

```
(gdb) set follow-fork-mode child
(gdb) start
Temporary breakpoint 1 at 0x40055c: file a.c, line 8.
Starting program: /data2/home/nanxiao/a

Temporary breakpoint 1, main () at a.c:8
8               pid = fork();
(gdb) n
[New process 12241]
[Switching to process 12241]
9               if (pid < 0)
(gdb)
13              else if (pid > 0)
(gdb)
17              printf("hello world\n");
(gdb)
hello world
18              return 0;
```

17 “hello world”

Linux gdb

nanxiao

```
#include <stdio.h>
#include <stdlib.h>

int main(void) {
    pid_t pid;

    pid = fork();
    if (pid < 0)
    {
        exit(1);
    }
    else if (pid > 0)
    {
        printf("Parent\n");
        exit(0);
    }
    printf("Child\n");
    return 0;
}
```

gdb gdb

```
(gdb) start
Temporary breakpoint 1 at 0x40055c: file a.c, line 7.
Starting program: /data2/home/nanxiao/a

Temporary breakpoint 1, main () at a.c:7
7           pid = fork();
(gdb) n
8           if (pid < 0)
(gdb) Child

12          else if (pid > 0)
(gdb)
14              printf("Parent\n");
(gdb)
Parent
15              exit(0);
```

8 “Child”

“ `set detach-on-fork off` ”

`detach-on-fork` `on` gdb

```
(gdb) set detach-on-fork off
(gdb) start
Temporary breakpoint 1 at 0x40055c: file a.c, line 7.
Starting program: /data2/home/nanxiao/a

Temporary breakpoint 1, main () at a.c:7
7           pid = fork();
(gdb) n
[New process 1050]
8           if (pid < 0)
(gdb)
12          else if (pid > 0)
(gdb) i inferior
  Num  Description       Executable
  2    process 1050      /data2/home/nanxiao/a
* 1    process 1046      /data2/home/nanxiao/a
(gdb) n
14              printf("Parent\n");
(gdb) n
Parent
15              exit(0);
(gdb)
[Inferior 1 (process 1046) exited normally]
(gdb)
The program is not being run.
(gdb) i inferiors
  Num  Description       Executable
  2    process 1050      /data2/home/nanxiao/a
* 1    <null>            /data2/home/nanxiao/a
(gdb) inferior 2
[Switching to inferior 2 [process 1050] (/data2/home/nanxiao/a)]
[Switching to thread 2 (process 1050)]
#0  0x00007ffff7af6cad in fork () from /lib64/libc.so.6
(gdb) bt
#0  0x00007ffff7af6cad in fork () from /lib64/libc.so.6
#1  0x0000000000400561 in main () at a.c:7
(gdb) n
Single stepping until exit from function fork,
which has no line number information.
main () at a.c:8
8           if (pid < 0)
(gdb)
12          else if (pid > 0)
(gdb)
17          printf("Child\n");
(gdb)
Child
18          return 0;
(gdb)
```

“ `set detach-on-fork off` ”
“ `i inferiors` ” `i` `info`
gdb “*”
“ `inferior infno` ”

Linux gdb

“ `set schedule-multiple on` ” `schedule-multiple` `off`

```
(gdb) set detach-on-fork off
(gdb) set schedule-multiple on
(gdb) start
Temporary breakpoint 1 at 0x40059c: file a.c, line 7.
Starting program: /data2/home/nanxiao/a

Temporary breakpoint 1, main () at a.c:7
7           pid = fork();
(gdb) n
[New process 26597]
Child
```

“Child”
gdb

nanxiao

Solaris CPU X86_64

```
(gdb) i threads
[New Thread 2 (LWP 2)]
[New Thread 3 (LWP 3)]
  Id    Target Id          Frame
  6     Thread 3 (LWP 3)  0xfeec870d in _thr_setup () from /usr/lib/
  5     Thread 2 (LWP 2)  0xfefc9661 in elf_find_sym () from /usr/li
  4     LWP    3          0xfeec870d in _thr_setup () from /usr/lib/
  3     LWP    2          0xfefc9661 in elf_find_sym () from /usr/li
* 2     Thread 1 (LWP 1)  main () at a.c:18
  1     LWP    1          main () at a.c:18
```

"i threads [Id...]"

```
  (gdb) i threads 1 2
  Id    Target Id          Frame
  2     Thread 0x7ffff782c700 (LWP 12248) 0x00007ffff78d9bcd in nanc
* 1     Thread 0x7ffff7fe9700 (LWP 12244) main () at a.c:18
```

gdb .

nanxiao

Solaris maintenance

gdb “i threads” i info
:

```
(gdb) i threads
106 process 2689429      0xff04af84 in __lwp_park () from /lib/libc
105 process 2623893      0xff04af84 in __lwp_park () from /lib/libc
104 process 2558357      0xff04af84 in __lwp_park () from /lib/libc
103 process 2492821      0xff04af84 in __lwp_park () from /lib/libc
```

Solaris gdb Solaris “maint info sol-threads” maint maintenance :

```
(gdb) maint info sol-threads
user   thread #1, lwp 1, (active)
user   thread #2, lwp 2, (active)    startfunc: monitor_thread
user   thread #3, lwp 3, (asleep)    startfunc: mem_db_thread
- Sleep func: 0x000aa32c
```

info maintenance
active asleep startfunc

gdb

nanxiao

```
#include <stdio.h>
#include <pthread.h>

void *thread_func(void *p_arg)
{
       sleep(10);
}

int main(void)
{
        pthread_t t1, t2;

        pthread_create(&t1, NULL, thread_func, "Thread 1");
        pthread_create(&t2, NULL, thread_func, "Thread 2");

        sleep(1000);
        return;
}
```

gdb :

```
(gdb) r
Starting program: /data/nan/a
[Thread debugging using libthread_db enabled]
[New Thread 1 (LWP 1)]
[New LWP    2        ]
[New LWP    3        ]
[LWP    2         exited]
[New Thread 2        ]
[LWP    3         exited]
[New Thread 3        ]
```

“ `set print thread-events off` ”

```
(gdb) set print thread-events off
(gdb) r
Starting program: /data/nan/a
[Thread debugging using libthread_db enabled]
```

gdb .

nanxiao

```
#include <stdio.h>
#include <pthread.h>
int a = 0;
int b = 0;
void *thread1_func(void *p_arg)
{
        while (1)
        {
                a++;
                sleep(1);
        }
}

void *thread2_func(void *p_arg)
{
        while (1)
        {
                b++;
                sleep(1);
        }
}

int main(void)
{
        pthread_t t1, t2;

        pthread_create(&t1, NULL, thread1_func, "Thread 1");
        pthread_create(&t2, NULL, thread2_func, "Thread 2");

        sleep(1000);
        return;
}
```

gdb

“ `step` ” “ `next` ”

:

```
(gdb) b a.c:9
Breakpoint 1 at 0x400580: file a.c, line 9.
(gdb) r
Starting program: /data2/home/nanxiao/a
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
[New Thread 0x7ffff782c700 (LWP 17368)]
[Switching to Thread 0x7ffff782c700 (LWP 17368)]

Breakpoint 1, thread1_func (p_arg=0x400718) at a.c:9
9                       a++;
(gdb) p b
$1 = 0
(gdb) s
10                      sleep(1);
(gdb) s
[New Thread 0x7ffff6e2b700 (LWP 17369)]
11              }
(gdb)

Breakpoint 1, thread1_func (p_arg=0x400718) at a.c:9
9                       a++;
(gdb)
10                      sleep(1);
(gdb) p b
$2 = 3
```

`thread1_func` `a` `thread2_func` `b`
`thread1_func` `a++` `b`
`0` `thread1_func` `b` `3`
`thread1_func` `thread2_func`

“ `set scheduler-locking on` ”

```
(gdb) b a.c:9
Breakpoint 1 at 0x400580: file a.c, line 9.
(gdb) r
Starting program: /data2/home/nanxiao/a
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
[New Thread 0x7ffff782c700 (LWP 19783)]
[Switching to Thread 0x7ffff782c700 (LWP 19783)]

Breakpoint 1, thread1_func (p_arg=0x400718) at a.c:9
9                       a++;
(gdb) set scheduler-locking on
(gdb) p b
$1 = 0
(gdb) s
10                      sleep(1);
(gdb)
11              }
(gdb)

Breakpoint 1, thread1_func (p_arg=0x400718) at a.c:9
9                       a++;
(gdb)
10                      sleep(1);
(gdb)
11              }
(gdb) p b
$2 = 0
```

`thread1_func` `b` `0`
`thread1_func` `thread2_func`

“ `set scheduler-locking` ” `off` `on`
`off` `step` " `step` "
" `next` "

gdb .

nanxiao

“$_thread”

```
#include <stdio.h>
#include <pthread.h>

int a = 0;

void *thread1_func(void *p_arg)
{
        while (1)
        {
                a++;
                sleep(10);
        }
}

void *thread2_func(void *p_arg)
{
        while (1)
        {
                a++;
                sleep(10);
        }
}

int main(void)
{
        pthread_t t1, t2;

        pthread_create(&t1, NULL, thread1_func, "Thread 1");
        pthread_create(&t2, NULL, thread2_func, "Thread 2");

        sleep(1000);
        return;
}
```

gdb 7.2 `$_thread` “`convenience variable`”

:

```
(gdb) wa a
Hardware watchpoint 2: a
(gdb) command 2
Type commands for breakpoint(s) 2, one per line.
End with a line saying just "end".
>printf "thread id=%d\n", $_thread
>end
```

“wa a” `wa` `watch` `a`
`commands`

```
(gdb) c
Continuing.
[New Thread 0x7ffff782c700 (LWP 20928)]
[Switching to Thread 0x7ffff782c700 (LWP 20928)]
Hardware watchpoint 2: a

Old value = 0
New value = 1
thread1_func (p_arg=0x400718) at a.c:11
11                       sleep(10);
thread id=2
(gdb) c
Continuing.
[New Thread 0x7ffff6e2b700 (LWP 20929)]
[Switching to Thread 0x7ffff6e2b700 (LWP 20929)]
Hardware watchpoint 2: a

Old value = 1
New value = 2
thread2_func (p_arg=0x400721) at a.c:20
20                       sleep(10);
thread id=3
```

“ `thread id=2` ” “ `thread id=3` ”
gdb .

nanxiao

gdb

```
a.c:
#include <stdio.h>
int func(int a, int b)
{
        int c = a * b;
        printf("c is %d\n", c);
}

int main(void)
{
        func(1, 2);
        return 0;
}


b.c:
#include <stdio.h>

int func1(int a)
{
        return 2 * a;
}

int func2(int a)
{
        int c = 0;
        c = 2 * func1(a);
        return c;
}

int func3(int a)
{
        int c = 0;
        c = 2 * func2(a);
        return c;
}

int main(void)
{
        printf("%d\n", func3(10));
        return 0;
}
```

gdb a

```
root@bash:~$ gdb a
GNU gdb (Ubuntu 7.7-0ubuntu3) 7.7
Copyright (C) 2014 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show c
and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from a...done.
(gdb) start
Temporary breakpoint 1 at 0x400568: file a.c, line 10.
Starting program: /home/nanxiao/a
```

“ `add-inferior [ -copies n ] [ -exec executable ]` ”
`b` `n` 1

```
(gdb) add-inferior -copies 2 -exec b
Added inferior 2
Reading symbols from b...done.
Added inferior 3
Reading symbols from b...done.
(gdb) i inferiors
  Num  Description       Executable
  3    <null>            /home/nanxiao/b
  2    <null>            /home/nanxiao/b
* 1    process 1586      /home/nanxiao/a
(gdb) inferior 2
[Switching to inferior 2 [<null>] (/home/nanxiao/b)]
(gdb) start
Temporary breakpoint 2 at 0x400568: main. (3 locations)
Starting program: /home/nanxiao/b

Temporary breakpoint 2, main () at b.c:24
24              printf("%d\n", func3(10));
(gdb) i inferiors
  Num  Description       Executable
  3    <null>            /home/nanxiao/b
* 2    process 1590      /home/nanxiao/b
  1    process 1586      /home/nanxiao/a
```

`b`

“ `clone-inferior [ -copies n ] [ infno ]` ” `inferior` `n` 1 `infno` `inferior`

```
(gdb) i inferiors
  Num  Description       Executable
  3    <null>            /home/nanxiao/b
* 2    process 1590      /home/nanxiao/b
  1    process 1586      /home/nanxiao/a
(gdb) clone-inferior -copies 1
Added inferior 4.
(gdb) i inferiors
  Num  Description       Executable
  4    <null>            /home/nanxiao/b
  3    <null>            /home/nanxiao/b
* 2    process 1590      /home/nanxiao/b
  1    process 1586      /home/nanxiao/a
```

`b`

gdb .

nanxiao

```
a.c:
#include <stdio.h>
int func(int a, int b)
{
        int c = a * b;
        printf("c is %d\n", c);
}

int main(void)
{
        func(1, 2);
        return 0;
}


b.c:
#include <stdio.h>

int func1(int a)
{
        return 2 * a;
}

int func2(int a)
{
        int c = 0;
        c = 2 * func1(a);
        return c;
}

int func3(int a)
{
        int c = 0;
        c = 2 * func2(a);
        return c;
}

int main(void)
{
        printf("%d\n", func3(10));
        return 0;
}
```

gdb “ `maint info program-spaces` ”

```
[root@localhost nan]# gdb a
GNU gdb (GDB) 7.8.1
......
Reading symbols from a...done.
(gdb) start
Temporary breakpoint 1 at 0x4004f9: file a.c, line 10.
Starting program: /home/nan/a

Temporary breakpoint 1, main () at a.c:10
10              func(1, 2);
(gdb) add-inferior -exec b
Added inferior 2
Reading symbols from b...done.
(gdb) i inferiors b
Args must be numbers or '$' variables.
(gdb) i inferiors
  Num  Description       Executable
  2    <null>            /home/nan/b
* 1    process 15753     /home/nan/a
(gdb) inferior 2
[Switching to inferior 2 [<null>] (/home/nan/b)]
(gdb) start
Temporary breakpoint 2 at 0x4004f9: main. (2 locations)
Starting program: /home/nan/b

Temporary breakpoint 2, main () at b.c:24
24              printf("%d\n", func3(10));
(gdb) i inferiors
  Num  Description       Executable
* 2    process 15902     /home/nan/b
  1    process 15753     /home/nan/a
(gdb) clone-inferior -copies 2
Added inferior 3.
Added inferior 4.
(gdb) i inferiors
  Num  Description       Executable
  4    <null>            /home/nan/b
  3    <null>            /home/nan/b
* 2    process 15902     /home/nan/b
  1    process 15753     /home/nan/a
(gdb) maint info program-spaces
  Id   Executable
  4    /home/nan/b
        Bound inferiors: ID 4 (process 0)
  3    /home/nan/b
        Bound inferiors: ID 3 (process 0)
* 2    /home/nan/b
        Bound inferiors: ID 2 (process 15902)
  1    /home/nan/a
        Bound inferiors: ID 1 (process 15753)
```

“ `maint info program-spaces` ” 4 `program-spaces` 1 4 `program-spaces` `inferior`

gdb .

nanxiao

“$_exitcode”

```
int main(void)
{
    return 0;
}
```

gdb `$_exitcode`
“ `convenience variable` ” “ `exit code` ”
:

```
[root@localhost nan]# gdb -q a
Reading symbols from a...done.
(gdb) start
Temporary breakpoint 1 at 0x400478: file a.c, line 3.
Starting program: /home/nan/a

Temporary breakpoint 1, main () at a.c:3
3               return 0;
(gdb) n
4       }
(gdb)
0x00000034e421ed1d in __libc_start_main () from /lib64/libc.so.6
(gdb)
Single stepping until exit from function __libc_start_main,
which has no line number information.
[Inferior 1 (process 1185) exited normally]
(gdb) p $_exitcode
$1 = 0
```

`$_exitcode` `0`
`1`

```
int main(void)
{
    return 0;
}
```

```
[root@localhost nan]# gdb -q a
Reading symbols from a...done.
(gdb) start
Temporary breakpoint 1 at 0x400478: file a.c, line 3.
Starting program: /home/nan/a

Temporary breakpoint 1, main () at a.c:3
3                return 1;
(gdb)
(gdb) n
4       }
(gdb)
0x00000034e421ed1d in __libc_start_main () from /lib64/libc.so.6
(gdb)
Single stepping until exit from function __libc_start_main,
which has no line number information.
[Inferior 1 (process 2603) exited with code 01]
(gdb) p $_exitcode
$1 = 1
```

`$_exitcode` `1`

gdb .

nanxiao

core dump

core dump

gdb core dump
“generate-core-file” core dump

```
(gdb) help generate-core-file
Save a core file with the current state of the debugged process.
Argument is optional filename.  Default filename is 'core.<process_

(gdb) start
Temporary breakpoint 1 at 0x8050c12: file a.c, line 9.
Starting program: /data1/nan/a
[Thread debugging using libthread_db enabled]
[New Thread 1 (LWP 1)]
[Switching to Thread 1 (LWP 1)]

Temporary breakpoint 1, main () at a.c:9
9           change_var();
(gdb) generate-core-file
Saved corefile core.12955
```

“gcore”

```
(gdb) help gcore
Save a core file with the current state of the debugged process.
Argument is optional filename.  Default filename is 'core.<process_
(gdb) gcore
Saved corefile core.13256
```

gdb

nanxiao

core dump

```
#include <stdio.h>

int main(void) {
        int *p = NULL;
        printf("hello world\n");
        *p = 0;
        return 0;
}
```

crash core dump gdb
core dump “gdb path/to/the/executable
path/to/the/coredump” gdb crash

```
bash-3.2# gdb -q /data/nan/a /var/core/core.a.22268.1402638140
Reading symbols from /data/nan/a...done.
[New LWP 1]
[Thread debugging using libthread_db enabled]
[New Thread 1 (LWP 1)]
Core was generated by `./a'.
Program terminated with signal 11, Segmentation fault.
#0  0x0000000000400cdb in main () at a.c:6
6               *p = 0;
```

gdb core dump
“file” “core” core-file “file”
“core” core dump

```
bash-3.2# gdb -q
(gdb) file /data/nan/a
Reading symbols from /data/nan/a...done.
(gdb) core /var/core/core.a.22268.1402638140
[New LWP 1]
[Thread debugging using libthread_db enabled]
[New Thread 1 (LWP 1)]
Core was generated by `./a'.
Program terminated with signal 11, Segmentation fault.
#0  0x0000000000400cdb in main () at a.c:6
6               *p = 0;
```

gdb crash

gdb

nanxiao

```
#include <stdio.h>
int global_var;

void change_var(){
    global_var=100;
}

int main(void){
    change_var();
    return 0;
}
```

Intel x86 gdb AT&T

```
(gdb) disassemble main
Dump of assembler code for function main:
   0x08050c0f <+0>:     push   %ebp
   0x08050c10 <+1>:     mov    %esp,%ebp
   0x08050c12 <+3>:     call   0x8050c00 <change_var>
   0x08050c17 <+8>:     mov    $0x0,%eax
   0x08050c1c <+13>:    pop    %ebp
   0x08050c1d <+14>:    ret
End of assembler dump.
```

“set disassembly-flavor” intel

```
(gdb) set disassembly-flavor intel
(gdb) disassemble main
Dump of assembler code for function main:
   0x08050c0f <+0>:     push   ebp
   0x08050c10 <+1>:     mov    ebp,esp
   0x08050c12 <+3>:     call   0x8050c00 <change_var>
   0x08050c17 <+8>:     mov    eax,0x0
   0x08050c1c <+13>:    pop    ebp
   0x08050c1d <+14>:    ret
End of assembler dump.
```

“set disassembly-flavor” Intel x86
“intel” “att”

gdb

nanxiao

```
#include <stdio.h>
int global_var;

void change_var(){
    global_var=100;
}

int main(void){
    change_var();
    return 0;
}
```

“b func” b break

```
(gdb) b main
Breakpoint 1 at 0x8050c12: file a.c, line 9.
(gdb) r
Starting program: /data1/nan/a
[Thread debugging using libthread_db enabled]
[New Thread 1 (LWP 1)]
[Switching to Thread 1 (LWP 1)]

Breakpoint 1, main () at a.c:9
9           change_var();
(gdb) disassemble
Dump of assembler code for function main:
   0x08050c0f <+0>:     push   %ebp
   0x08050c10 <+1>:     mov    %esp,%ebp
=> 0x08050c12 <+3>:     call   0x8050c00 <change_var>
   0x08050c17 <+8>:     mov    $0x0,%eax
   0x08050c1c <+13>:    pop    %ebp
   0x08050c1d <+14>:    ret
End of assembler dump.
```

“b *func”

```
(gdb) b *main
Breakpoint 1 at 0x8050c0f: file a.c, line 8.
(gdb) r
Starting program: /data1/nan/a
[Thread debugging using libthread_db enabled]
[New Thread 1 (LWP 1)]
[Switching to Thread 1 (LWP 1)]

Breakpoint 1, main () at a.c:8
8       int main(void){
(gdb) disassemble
Dump of assembler code for function main:
=> 0x08050c0f <+0>:     push   %ebp
   0x08050c10 <+1>:     mov    %esp,%ebp
   0x08050c12 <+3>:     call   0x8050c00 <change_var>
   0x08050c17 <+8>:     mov    $0x0,%eax
   0x08050c1c <+13>:    pop    %ebp
   0x08050c1d <+14>:    ret
End of assembler dump.
```

nanxiao

```
(gdb) set disassemble-next-line on
(gdb) start
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Temporary breakpoint 3 at 0x400543: file 1.c, line 14.
Starting program: /home/teawater/tmp/a.out

Temporary breakpoint 3, main (argc=1, argv=0x7fffffffdf38, envp=0x7
14      printf("1\n");
=> 0x0000000000400543 <main+19>:    bf f0 05 40 00  mov    $0x4005f
   0x0000000000400548 <main+24>:    e8 c3 fe ff ff  callq  0x40041
(gdb) si
0x0000000000400548  14      printf("1\n");
0x0000000000400543 <main+19>:    bf f0 05 40 00  mov    $0x4005f0,%
=> 0x0000000000400548 <main+24>:    e8 c3 fe ff ff  callq  0x40041
(gdb)
0x0000000000400410 in puts@plt ()
=> 0x0000000000400410 <puts@plt+0>: ff 25 02 0c 20 00   jmpq   *0x2

(gdb) set disassemble-next-line auto
(gdb) start
Temporary breakpoint 1 at 0x400543: file 1.c, line 14.
Starting program: /home/teawater/tmp/a.out

Temporary breakpoint 1, main (argc=1, argv=0x7fffffffdf38, envp=0x7
14      printf("1\n");
(gdb) si
0x0000000000400548  14      printf("1\n");
(gdb)
0x0000000000400410 in puts@plt ()
=> 0x0000000000400410 <puts@plt+0>: ff 25 02 0c 20 00   jmpq   *0x2
(gdb)
0x0000000000400416 in puts@plt ()
=> 0x0000000000400416 <puts@plt+6>: 68 00 00 00 00  pushq  $0x0
```

```
(gdb) set disassemble-next-line on
```

```
(gdb) set disassemble-next-line auto
```

```
(gdb) set disassemble-next-line off
```

teawater

```
#include <stdio.h>

typedef struct
{
        int a;
        int b;
        int c;
        int d;
}ex_st;

int main(void) {
        ex_st st = {1, 2, 3, 4};
        printf("%d,%d,%d,%d\n", st.a, st.b, st.c, st.d);
        return 0;
}
```

“disas /m fun” disas disassemble

```
(gdb) disas /m main
Dump of assembler code for function main:
11      int main(void) {
   0x00000000004004c4 <+0>:     push   %rbp
   0x00000000004004c5 <+1>:     mov    %rsp,%rbp
   0x00000000004004c8 <+4>:     push   %rbx
   0x00000000004004c9 <+5>:     sub    $0x18,%rsp

12              ex_st st = {1, 2, 3, 4};
   0x00000000004004cd <+9>:     movl   $0x1,-0x20(%rbp)
   0x00000000004004d4 <+16>:    movl   $0x2,-0x1c(%rbp)
   0x00000000004004db <+23>:    movl   $0x3,-0x18(%rbp)
   0x00000000004004e2 <+30>:    movl   $0x4,-0x14(%rbp)

13              printf("%d,%d,%d,%d\n", st.a, st.b, st.c, st.d);
   0x00000000004004e9 <+37>:    mov    -0x14(%rbp),%esi
   0x00000000004004ec <+40>:    mov    -0x18(%rbp),%ecx
   0x00000000004004ef <+43>:    mov    -0x1c(%rbp),%edx
   0x00000000004004f2 <+46>:    mov    -0x20(%rbp),%ebx
   0x00000000004004f5 <+49>:    mov    $0x400618,%eax
   0x00000000004004fa <+54>:    mov    %esi,%r8d
   0x00000000004004fd <+57>:    mov    %ebx,%esi
   0x00000000004004ff <+59>:    mov    %rax,%rdi
   0x0000000000400502 <+62>:    mov    $0x0,%eax
   0x0000000000400507 <+67>:    callq  0x4003b8 <printf@plt>

14              return 0;
   0x000000000040050c <+72>:    mov    $0x0,%eax

15      }
   0x0000000000400511 <+77>:    add    $0x18,%rsp
   0x0000000000400515 <+81>:    pop    %rbx
   0x0000000000400516 <+82>:    leaveq
   0x0000000000400517 <+83>:    retq

End of assembler dump.
```

C

```
(gdb) i line 13
Line 13 of "foo.c" starts at address 0x4004e9 <main+37> and ends at
```

“ `disassemble [Start],[End]` ”

```
(gdb) disassemble 0x4004e9, 0x40050c
Dump of assembler code from 0x4004e9 to 0x40050c:
   0x00000000004004e9 <main+37>:        mov    -0x14(%rbp),%esi
   0x00000000004004ec <main+40>:        mov    -0x18(%rbp),%ecx
   0x00000000004004ef <main+43>:        mov    -0x1c(%rbp),%edx
   0x00000000004004f2 <main+46>:        mov    -0x20(%rbp),%ebx
   0x00000000004004f5 <main+49>:        mov    $0x400618,%eax
   0x00000000004004fa <main+54>:        mov    %esi,%r8d
   0x00000000004004fd <main+57>:        mov    %ebx,%esi
   0x00000000004004ff <main+59>:        mov    %rax,%rdi
   0x0000000000400502 <main+62>:        mov    $0x0,%eax
   0x0000000000400507 <main+67>:        callq  0x4003b8 <printf@pl
End of assembler dump.
```

gdb

nanxiao

xmj

```
#include <stdio.h>
int global_var;

void change_var(){
    global_var=100;
}

int main(void){
    change_var();
    return 0;
}
```

gdb “`display /i $pc`”

```
(gdb) start
Temporary breakpoint 1 at 0x400488: file a.c, line 9.
Starting program: /data2/home/nanxiao/a

Temporary breakpoint 1, main () at a.c:9
9           change_var();
(gdb) display /i $pc
1: x/i $pc
=> 0x400488 <main+4>:   mov    $0x0,%eax
(gdb) si
0x000000000040048d      9           change_var();
1: x/i $pc
=> 0x40048d <main+9>:   callq  0x400474 <change_var>
(gdb)
change_var () at a.c:4
4       void change_var(){
1: x/i $pc
=> 0x400474 <change_var>:       push   %rbp
```

```
(gdb) display /3i $pc
2: x/3i $pc
=> 0x400474 <change_var>:       push   %rbp
   0x400475 <change_var+1>:     mov    %rsp,%rbp
   0x400478 <change_var+4>:     movl   $0x64,0x2003de(%rip)
```

`3`

`undisplay`

gdb

nanxiao

gdb “i registers” i info
:

```
(gdb) i registers
rax            0x7ffff7dd9f60   140737351884640
rbx            0x0      0
rcx            0x0      0
rdx            0x7fffffffe608   140737488348680
rsi            0x7fffffffe5f8   140737488348664
rdi            0x1      1
rbp            0x7fffffffe510   0x7fffffffe510
rsp            0x7fffffffe4c0   0x7fffffffe4c0
r8             0x7ffff7dd8300   140737351877376
r9             0x7ffff7deb9e0   140737351956960
r10            0x7fffffffe360   140737488348000
r11            0x7ffff7a68be0   140737348275168
r12            0x4003e0 4195296
r13            0x7fffffffe5f0   140737488348656
r14            0x0      0
r15            0x0      0
rip            0x4004cd 0x4004cd <main+9>
eflags         0x206    [ PF IF ]
cs             0x33     51
ss             0x2b     43
ds             0x0      0
es             0x0      0
fs             0x0      0
gs             0x0      0
```

“i all-registers”

```
(gdb) i all-registers
    rax            0x7ffff7dd9f60   140737351884640
    rbx            0x0      0
    rcx            0x0      0
    rdx            0x7fffffffe608   140737488348680
    rsi            0x7fffffffe5f8   140737488348664
    rdi            0x1      1
    rbp            0x7fffffffe510   0x7fffffffe510
    rsp            0x7fffffffe4c0   0x7fffffffe4c0
    r8             0x7ffff7dd8300   140737351877376
    r9             0x7ffff7deb9e0   140737351956960
    r10            0x7fffffffe360   140737488348000
    r11            0x7ffff7a68be0   140737348275168
    r12            0x4003e0 4195296
    r13            0x7fffffffe5f0   140737488348656
    r14            0x0      0
    r15            0x0      0
    rip            0x4004cd 0x4004cd <main+9>
    eflags         0x206    [ PF IF ]
    cs             0x33     51
    ss             0x2b     43
    ds             0x0      0
    es             0x0      0
    fs             0x0      0
    gs             0x0      0
    st0            0        (raw 0x00000000000000000000)
    st1            0        (raw 0x00000000000000000000)
    st2            0        (raw 0x00000000000000000000)
    st3            0        (raw 0x00000000000000000000)
    st4            0        (raw 0x00000000000000000000)
    st5            0        (raw 0x00000000000000000000)
    st6            0        (raw 0x00000000000000000000)
    st7            0        (raw 0x00000000000000000000)
    ......
```

“i registers regname” “p $regname”

```
(gdb) i registers eax
eax            0xf7dd9f60       -136470688
(gdb) p $eax
$1 = -136470688
```

gdb .

nanxiao

```
#include <stdio.h>

int main(void)
{
        printf("Hello, world\n");
        return 0;
}
```

“disassemble /r”   16

```
(gdb) disassemble /r main
Dump of assembler code for function main:
   0x0000000000400530 <+0>:     55      push   %rbp
   0x0000000000400531 <+1>:     48 89 e5        mov    %rsp,%rbp
   0x0000000000400534 <+4>:     bf e0 05 40 00  mov    $0x4005e0,%e
   0x0000000000400539 <+9>:     e8 d2 fe ff ff  callq  0x400410 <pu
   0x000000000040053e <+14>:    b8 00 00 00 00  mov    $0x0,%eax
   0x0000000000400543 <+19>:    5d      pop    %rbp
   0x0000000000400544 <+20>:    c3      retq
End of assembler dump.
(gdb) disassemble /r 0x0000000000400534,+4
Dump of assembler code from 0x400534 to 0x400538:
   0x0000000000400534 <main+4>: bf e0 05 40 00  mov    $0x4005e0,%e
End of assembler dump.
```

gdb

nanxiao

```
#include <stdio.h>

int main(void)
{
    char p1[] = "Sam";
    char *p2 = "Bob";

    printf("p1 is %s, p2 is %s\n", p1, p2);
    return 0;
}
```

gdb “ `set` ”

```
(gdb) start
Temporary breakpoint 1 at 0x8050af0: file a.c, line 5.
Starting program: /data1/nan/a
[Thread debugging using libthread_db enabled]
[New Thread 1 (LWP 1)]
[Switching to Thread 1 (LWP 1)]

Temporary breakpoint 1, main () at a.c:5
5               char p1[] = "Sam";
(gdb) n
6               char *p2 = "Bob";
(gdb)
8               printf("p1 is %s, p2 is %s\n", p1, p2);
(gdb) set main::p1="Jil"
(gdb) set main::p2="Bill"
(gdb) n
p1 is Jil, p2 is Bill
9               return 0;
```

`p1` `p2`

```
Starting program: /data1/nan/a
[Thread debugging using libthread_db enabled]
[New Thread 1 (LWP 1)]
[Switching to Thread 1 (LWP 1)]

Temporary breakpoint 2, main () at a.c:5
5               char p1[] = "Sam";
(gdb) n
6               char *p2 = "Bob";
(gdb) p p1
$1 = "Sam"
(gdb) p &p1
$2 = (char (*)[4]) 0x80477a4
(gdb) set {char [4]} 0x80477a4 = "Ace"
(gdb) n
8               printf("p1 is %s, p2 is %s\n", p1, p2);
(gdb)
p1 is Ace, p2 is Bob
9               return 0;
```

stackoverflow.

nanxiao

```
#include <stdio.h>

int func(void)
{
    int i = 2;

    return i;
}

int main(void)
{
    int a = 0;

    a = func();
    printf("%d\n", a);
    return 0;
}
```

gdb “ `set var variable=expr` ”

```
Breakpoint 2, func () at a.c:5
5                   int i = 2;
(gdb) n
7                   return i;
(gdb) set var i = 8
(gdb) p i
$4 = 8
```

`func` `set` `i` `8`

“ `set {type}address=expr` ” `address` `type`

```
Breakpoint 2, func () at a.c:5
5                   int i = 2;
(gdb) n
7                   return i;
(gdb) p &i
$5 = (int *) 0x8047a54
(gdb) set {int}0x8047a54 = 8
(gdb) p i
$6 = 8
```

`i` `8`

```
Breakpoint 2, func () at a.c:5
5                   int i = 2;
(gdb)
(gdb) n
7                   return i;
(gdb)
8               }
(gdb) set var $eax = 8
(gdb) n
main () at a.c:15
15                  printf("%d\n", a);
(gdb)
8
16                  return 0;
```

eax eax
`8` `8`

gdb

nanxiao

PC

```
#include <stdio.h>
int main(void)
{
        int a =0;

        a++;
        a++;
        printf("%d\n", a);
        return 0;
}
```

PC

“ `a=2` ” PC

```
4               int a =0;
(gdb) disassemble main
Dump of assembler code for function main:
0x08050921 <main+0>:    push   %ebp
0x08050922 <main+1>:    mov    %esp,%ebp
0x08050924 <main+3>:    sub    $0x8,%esp
0x08050927 <main+6>:    and    $0xfffffff0,%esp
0x0805092a <main+9>:    mov    $0x0,%eax
0x0805092f <main+14>:   add    $0xf,%eax
0x08050932 <main+17>:   add    $0xf,%eax
0x08050935 <main+20>:   shr    $0x4,%eax
0x08050938 <main+23>:   shl    $0x4,%eax
0x0805093b <main+26>:   sub    %eax,%esp
0x0805093d <main+28>:   movl   $0x0,-0x4(%ebp)
0x08050944 <main+35>:   lea    -0x4(%ebp),%eax
0x08050947 <main+38>:   incl   (%eax)
0x08050949 <main+40>:   lea    -0x4(%ebp),%eax
0x0805094c <main+43>:   incl   (%eax)
0x0805094e <main+45>:   sub    $0x8,%esp
0x08050951 <main+48>:   pushl  -0x4(%ebp)
0x08050954 <main+51>:   push   $0x80509b4
0x08050959 <main+56>:   call   0x80507cc <printf@plt>
0x0805095e <main+61>:   add    $0x10,%esp
0x08050961 <main+64>:   mov    $0x0,%eax
0x08050966 <main+69>:   leave
0x08050967 <main+70>:   ret
End of assembler dump.
(gdb) info line 6
Line 6 of "a.c" starts at address 0x8050944 <main+35> and ends at
(gdb) info line 7
Line 7 of "a.c" starts at address 0x8050949 <main+40> and ends at
```

“ `info line 6` ” “ `info line 7` ” “ `a++;` ”
`0x8050944` `0x8050949`

```
(gdb) n
6               a++;
(gdb) p $pc
$3 = (void (*)()) 0x8050944 <main+35>
(gdb) set var $pc=0x08050949
```

“ `a++;` ” `pc` `pc`
`0x8050944` “ `info line 6` ” `pc`
`0x8050949` “ `info line 7` ” “ `a++;` ”

```
(gdb) n
8               printf("a=%d\n", a);
(gdb)
a=1
9               return 0;
```

“ `a=1` ” “ `a++;` ”

nanxiao

```
#include <stdio.h>

void fun (int x)
{
  if (x < 0)
    puts ("error");
}

int main (void)
{
  int i = 1;

  fun (i--);
  fun (i--);
  fun (i--);

  return 0;
}
```

```
(gdb) n
13        fun (i--);
(gdb)
14        fun (i--);
(gdb)
15        fun (i--);
(gdb)
error
17        return 0;
```

15 fun 15
`run`

`jump` 15

```
(gdb) b 15
Breakpoint 2 at 0x40056a: file jump.c, line 15.
(gdb) j 15
Continuing at 0x40056a.

Breakpoint 2, main () at jump.c:15
15        fun (i--);
(gdb) s
fun (x=-2) at jump.c:5
5       if (x < 0)
(gdb) n
6         puts ("error");
```

1. `jump` pc
   i
2.

gdb

xmj

```
#include <stdio.h>
#include <stdlib.h>

void drawing (int n)
{
  if (n != 0)
    puts ("Try again?\nAll you need is a dollar, and a dream.");
  else
    puts ("You win $3000!");
}

int main (void)
{
  int n;

  srand (time (0));
  n = rand () % 10;
  printf ("Your number is %d\n", n);
  drawing (n);

  return 0;
}
```

```
(gdb) b drawing
Breakpoint 1 at 0x40064d: file win.c, line 6.
(gdb) command 1
Type commands for breakpoint(s) 1, one per line.
End with a line saying just "end".
>silent
>set variable n = 0
>continue
>end
(gdb) r
Starting program: /home/xmj/tmp/a.out
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_d
Your number is 6
You win $3000!
[Inferior 1 (process 4134) exited normally]
```

n 0

bug gdb

gdb

xmj

```
#include <stdio.h>
#include <stdlib.h>

void drawing (int n)
{
  if (n != 0)
    puts ("Try again?\nAll you need is a dollar, and a dream.");
  else
    puts ("You win $3000!");
}

int main (void)
{
  int n;

  srand (time (0));
  n = rand () % 10;
  printf ("Your number is %d\n", n);
  drawing (n);

  return 0;
}
```

gdb

gdb

```
$ gcc -write ./a.out
(gdb) show write
Writing into executable and core files is on.
```

gdb

```
(gdb) set write on
(gdb) file ./a.out
```

```
(gdb) disassemble /mr drawing
Dump of assembler code for function drawing:
5    {
   0x0000000000400642 <+0>:    55    push   %rbp
   0x0000000000400643 <+1>:    48 89 e5    mov    %rsp,%rbp
   0x0000000000400646 <+4>:    48 83 ec 10    sub    $0x10,%rsp
   0x000000000040064a <+8>:    89 7d fc    mov    %edi,-0x4(%rbp)

6      if (n != 0)
   0x000000000040064d <+11>:    83 7d fc 00    cmpl   $0x0,-0x4(%rb
   0x0000000000400651 <+15>:    74 0c    je     0x40065f <drawing+2

7        puts ("Try again?\nAll you need is a dollar, and a dream."
   0x0000000000400653 <+17>:    bf e0 07 40 00    mov    $0x4007e0,
   0x0000000000400658 <+22>:    e8 b3 fe ff ff    callq  0x400510 <
   0x000000000040065d <+27>:    eb 0a    jmp    0x400669 <drawing+3

8      else
9        puts ("You win $3000!");
   0x000000000040065f <+29>:    bf 12 08 40 00    mov    $0x400812,
   0x0000000000400664 <+34>:    e8 a7 fe ff ff    callq  0x400510 <

10    }
   0x0000000000400669 <+39>:    c9    leaveq
   0x000000000040066a <+40>:    c3    retq

End of assembler dump.
```

```
(gdb) set variable *(short*)0x400651=0x0ceb
(gdb) disassemble /mr drawing
Dump of assembler code for function drawing:
5    {
   0x0000000000400642 <+0>:     55      push   %rbp
   0x0000000000400643 <+1>:     48 89 e5        mov    %rsp,%rbp
   0x0000000000400646 <+4>:     48 83 ec 10     sub    $0x10,%rsp
   0x000000000040064a <+8>:     89 7d fc        mov    %edi,-0x4(%rbp)

6       if (n != 0)
   0x000000000040064d <+11>:    83 7d fc 00     cmpl   $0x0,-0x4(%rb
   0x0000000000400651 <+15>:    eb 0c   jmp    0x40065f <drawing+2

7         puts ("Try again?\nAll you need is a dollar, and a dream.'
   0x0000000000400653 <+17>:    bf e0 07 40 00  mov    $0x4007e0,
   0x0000000000400658 <+22>:    e8 b3 fe ff ff  callq  0x400510 <
   0x000000000040065d <+27>:    eb 0a   jmp    0x400669 <drawing+3

8       else
9         puts ("You win $3000!");
   0x000000000040065f <+29>:    bf 12 08 40 00  mov    $0x400812,
   0x0000000000400664 <+34>:    e8 a7 fe ff ff  callq  0x400510 <

10    }
   0x0000000000400669 <+39>:    c9      leaveq
   0x000000000040066a <+40>:    c3      retq

End of assembler dump.
```

"je" "jmp"

```
$ ./a.out
Your number is 2
You win $3000!
```

gdb

xmj

```
#include <stdio.h>
#include <signal.h>

void handler(int sig);

void handler(int sig)
{
        signal(sig, handler);
        printf("Receive signal: %d\n", sig);
}

int main(void) {
        signal(SIGINT, handler);
        signal(SIGALRM, handler);

        while (1)
        {
                sleep(1);
        }
        return 0;
}
```

gdb “ `i signals` ” “ `i handle` ” `i` `info` gdb :

```
(gdb) i signals
Signal        Stop      Print   Pass to program Description

SIGHUP        Yes       Yes     Yes             Hangup
SIGINT        Yes       Yes     No              Interrupt
SIGQUIT       Yes       Yes     Yes             Quit
......
SIGALRM       No        No      Yes             Alarm clock
......
```

`Signal` `Stop` gdb

`Print` gdb

`Pass to program` gdb `Description`

`SIGINT` gdb `SIGALRM` gdb

gdb `SIGINT` `SIGALRM`

```
Program received signal SIGINT, Interrupt.
0xfeeeae55 in ___nanosleep () from /lib/libc.so.1
(gdb) c
Continuing.
Receive signal: 14
```

`SIGINT` `SIGINT` `SIGALRM` `SIGALRM`

gdb .

nanxiao

```
#include <stdio.h>
#include <signal.h>

void handler(int sig);

void handler(int sig)
{
        signal(sig, handler);
        printf("Receive signal: %d\n", sig);
}

int main(void) {
        signal(SIGHUP, handler);

        while (1)
        {
                sleep(1);
        }
        return 0;
}
```

gdb “ `handle signal stop/nostop` ” :

```
(gdb) i signals
Signal        Stop      Print   Pass to program Description

SIGHUP        Yes       Yes     Yes             Hangup
......

(gdb) r
Starting program: /data1/nan/test
[Thread debugging using libthread_db enabled]
[New Thread 1 (LWP 1)]

Program received signal SIGHUP, Hangup.
[Switching to Thread 1 (LWP 1)]
0xfeeeae55 in ___nanosleep () from /lib/libc.so.1
(gdb) c
Continuing.
Receive signal: 1
```

`SIGHUP` gdb
`continue`

“ `handle SIGHUP nostop` ” `SIGHUP` gdb

```
(gdb) handle SIGHUP nostop
Signal        Stop      Print   Pass to program Description
SIGHUP        No        Yes     Yes             Hangup
(gdb) c
Continuing.

Program received signal SIGHUP, Hangup.
Receive signal: 1
```

`SIGHUP`

“ `handle SIGHUP stop` ”
`stop` `print` `print`

gdb .

nanxiao

```
#include <stdio.h>
#include <signal.h>

void handler(int sig);

void handler(int sig)
{
        signal(sig, handler);
        printf("Receive signal: %d\n", sig);
}

int main(void) {
        signal(SIGHUP, handler);

        while (1)
        {
                sleep(1);
        }
        return 0;
}
```

gdb “ `handle signal print/noprint` ” :

```
(gdb) i signals
Signal        Stop      Print   Pass to program Description

SIGHUP        Yes       Yes     Yes             Hangup
......

(gdb) r
Starting program: /data1/nan/test
[Thread debugging using libthread_db enabled]
[New Thread 1 (LWP 1)]

Program received signal SIGHUP, Hangup.
[Switching to Thread 1 (LWP 1)]
0xfeeeae55 in ___nanosleep () from /lib/libc.so.1
(gdb) c
Continuing.
Receive signal: 1
```

`SIGHUP` gdb
`continue`

“ `handle SIGHUP noprint` ” `SIGHUP` gdb

```
(gdb) handle SIGHUP noprint
Signal        Stop      Print   Pass to program Description
SIGHUP        No        No      Yes             Hangup
(gdb) r
Starting program: /data1/nan/test
[Thread debugging using libthread_db enabled]
Receive signal: 1
```

`noprint` `nostop`
`SIGHUP`

“ `handle SIGHUP print` ”

gdb .

nanxiao

```
#include <stdio.h>
#include <signal.h>

void handler(int sig);

void handler(int sig)
{
        signal(sig, handler);
        printf("Receive signal: %d\n", sig);
}

int main(void) {
        signal(SIGHUP, handler);

        while (1)
        {
                sleep(1);
        }
        return 0;
}
```

gdb
“ `handle signal pass(noignore)/nopass(ignore)` ”
. `pass` `noignore`
`nopass` `ignore` :

```
(gdb) i signals
Signal        Stop      Print   Pass to program Description

SIGHUP        Yes       Yes     Yes             Hangup
......

(gdb) r
Starting program: /data1/nan/test
[Thread debugging using libthread_db enabled]
[New Thread 1 (LWP 1)]

Program received signal SIGHUP, Hangup.
[Switching to Thread 1 (LWP 1)]
0xfeeeae55 in ___nanosleep () from /lib/libc.so.1
(gdb) c
Continuing.
Receive signal: 1
```

`SIGHUP` gdb

“ `handle SIGHUP nopass` ” `SIGHUP` gdb

```
(gdb) handle SIGHUP nopass
Signal        Stop      Print   Pass to program Description
SIGHUP        Yes       Yes     No              Hangup
(gdb) c
Continuing.

Program received signal SIGHUP, Hangup.
0xfeeeae55 in ___nanosleep () from /lib/libc.so.1
(gdb) c
Continuing.
```

`SIGHUP` “Receive signal: 1” gdb

“ `handle SIGHUP pass` ”

gdb .

nanxiao

```
#include <stdio.h>
#include <signal.h>

void handler(int sig);

void handler(int sig)
{
        signal(sig, handler);
        printf("Receive signal: %d\n", sig);
}

int main(void) {
        signal(SIGHUP, handler);

        while (1)
        {
                sleep(1);
        }
        return 0;
}
```

gdb
“ `signal signal_name` ”
:

```
(gdb) r
`/data1/nan/test' has changed; re-reading symbols.
Starting program: /data1/nan/test
[Thread debugging using libthread_db enabled]
^C[New Thread 1 (LWP 1)]

Program received signal SIGINT, Interrupt.
[Switching to Thread 1 (LWP 1)]
0xfeeeae55 in ___nanosleep () from /lib/libc.so.1
(gdb) signal SIGHUP
Continuing with signal SIGHUP.
Receive signal: 1
```

`signal SIGHUP` gdb

“ `signal 0` ”

```
Program received signal SIGHUP, Hangup.
0xfeeeae55 in ___nanosleep () from /lib/libc.so.1
(gdb) signal 0
Continuing with no signal.
```

`SIGHUP` gdb
“ `signal 0` ” `SIGHUP`

`signal` shell `kill`
shell `kill` gdb
`signal`

gdb .

nanxiao

“$_siginfo”

```
#include <stdio.h>
#include <signal.h>

void handler(int sig);

void handler(int sig)
{
        signal(sig, handler);
        printf("Receive signal: %d\n", sig);
}

int main(void) {
        signal(SIGHUP, handler);

        while (1)
        {
                sleep(1);
        }
        return 0;
}
```

Linux gdb gdb
`$_siginfo`
`kernel` :

```
Program received signal SIGHUP, Hangup.
0x00000034e42accc0 in __nanosleep_nocancel () from /lib64/libc.so.
Missing separate debuginfos, use: debuginfo-install glibc-2.12-1.1
(gdb) ptype $_siginfo
type = struct {
    int si_signo;
    int si_errno;
    int si_code;
    union {
        int _pad[28];
        struct {...} _kill;
        struct {...} _timer;
        struct {...} _rt;
        struct {...} _sigchld;
        struct {...} _sigfault;
        struct {...} _sigpoll;
    } _sifields;
}
(gdb) ptype $_siginfo._sifields._sigfault
type = struct {
    void *si_addr;
}
(gdb) p $_siginfo._sifields._sigfault.si_addr
$1 = (void *) 0x850e
```

`$_siginfo`

gdb .

nanxiao

```
#include <hiredis/hiredis.h>

int main(void)
{
        char a[1026] = {0};
        redisContext *c = NULL;
        void *reply = NULL;

        memset(a, 'a', (sizeof(a) - 1));
        c = redisConnect("127.0.0.1", 6379);
        if (NULL != c)
        {
              reply = redisCommand(c, "set 1 %s", a);
              freeReplyObject(reply);

              reply = redisCommand(c, "get 1");
              freeReplyObject(reply);

              redisFree(c);
        }
        return 0;
}
```

"`info sharedlibrary regex`"
`regex` `regex`
`regex` :

```
(gdb) start
Temporary breakpoint 1 at 0x109f0: file a.c, line 5.
Starting program: /export/home/nan/a
[Thread debugging using libthread_db enabled]
[New Thread 1 (LWP 1)]
[Switching to Thread 1 (LWP 1)]

Temporary breakpoint 1, main () at a.c:5
5                       char a[1026] = {0};
(gdb) info sharedlibrary
From        To          Syms Read   Shared Object Library
0xff3b44a0  0xff3e3490  Yes (*)     /usr/lib/ld.so.1
0xff3325f0  0xff33d4b4  Yes         /usr/local/lib/libhiredis.so.0
0xff3137f0  0xff31a9f4  Yes (*)     /lib/libsocket.so.1
0xff215fd4  0xff28545c  Yes (*)     /lib/libnsl.so.1
0xff0a3a20  0xff14fedc  Yes (*)     /lib/libc.so.1
0xff320400  0xff3234c8  Yes (*)     /platform/SUNW,UltraAX-i2/lib/
(*): Shared library is missing debugging information.
```

“`*`”

```
(gdb) i sharedlibrary hiredi*
From        To          Syms Read   Shared Object Library
0xff3325f0  0xff33d4b4  Yes         /usr/local/lib/libhiredis.so.0
```

gdb .

nanxiao

gdb init

gdb HOME
“.gdbinit”

“.gdbinit”

```
#    STL
python
import sys
sys.path.insert(0, "/home/xmj/project/gcc-trunk/libstdc++-v3/pytho
from libstdcxx.v6.printers import register_libstdcxx_printers
register_libstdcxx_printers (None)
end

#
set history filename ~/.gdb_history
set history save on

#
set confirm off

#
set print object on

#
set print array-indexes on

#
set print pretty on
```

xmj

```
#include <stdio.h>

typedef struct
{
        int a;
        int b;
        int c;
        int d;
}ex_st;

int main(void) {
        ex_st st = {1, 2, 3, 4};
        printf("%d,%d,%d,%d\n", st.a, st.b, st.c, st.d);
        return 0;
}
```

gdb gdb “.gdbinit” gdb “.gdbinit” gdb python

gdb “ `set script-extension` ” 3

a `off` gdb

b `soft` gdb python

c `strict` gdb python

```
(gdb) start
Temporary breakpoint 1 at 0x4004cd: file a.c, line 12.
Starting program: /data2/home/nanxiao/a

Temporary breakpoint 1, main () at a.c:12
12              ex_st st = {1, 2, 3, 4};
(gdb) q
A debugging session is active.

        Inferior 1 [process 24249] will be killed.

Quit anyway? (y or n) y
```

gdb

gdb.py gdb python
gdb

```
set confirm off
```

```
(gdb) start
Temporary breakpoint 1 at 0x4004cd: file a.c, line 12.
Starting program: /data2/home/nanxiao/a

Temporary breakpoint 1, main () at a.c:12
12              ex_st st = {1, 2, 3, 4};
(gdb) show script-extension
Script filename extension recognition is "soft".
(gdb) source gdb.py
  File "gdb.py", line 1
    set confirm off
              ^
SyntaxError: invalid syntax
```

“`script-extension`” `soft`
“`source gdb.py`”, pyhton gdb.py
gdb

```
(gdb) start
Temporary breakpoint 1 at 0x4004cd: file a.c, line 12.
Starting program: /data2/home/nanxiao/a

Temporary breakpoint 1, main () at a.c:12
12              ex_st st = {1, 2, 3, 4};
(gdb) set script-extension off
(gdb) source gdb.py
(gdb) q
[root@linux:~]$
```

“`script-extension`” `off` gdb

gdb

nanxiao

gdb

```
(gdb) set history save on
```

.gdb_history

```
(gdb) set history filename fname
```

$HOME/.gdbinit

```
set history filename ~/.gdb_history
set history save on
```

gdb

gdb

xmj

```
#include <stdio.h>
#include <time.h>

int main(void) {
        time_t now = time(NULL);
        struct tm local = {0};
        struct tm gmt = {0};

        localtime_r(&now, &local);
        gmtime_r(&now, &gmt);

        return 0;
}
```

gdb
`directory`

```
(gdb) start
Temporary breakpoint 1 at 0x400560: file a.c, line 5.
Starting program: /home/nan/a

Temporary breakpoint 1, main () at a.c:5
5       a.c: No such file or directory.
(gdb) directory ../ki/
Source directories searched: /home/nan/../ki:$cdir:$cwd
(gdb) n
6               struct tm local = {0};
(gdb)
7               struct tm gmt = {0};
(gdb)
9               localtime_r(&now, &local);
(gdb)
10              gmtime_r(&now, &gmt);
(gdb) q
```

`directory` `dir`) gdb

gdb code gdb
gdb `-d`

```
gdb -q a.out -d /search/code/some
```

gdb .

nanxiao

```
#include <stdio.h>
#include <time.h>

int main(void) {
        time_t now = time(NULL);
        struct tm local = {0};
        struct tm gmt = {0};

        localtime_r(&now, &local);
        gmtime_r(&now, &gmt);

        return 0;
}
```

`set substitute-path from to` `to`
`from`

```
(gdb) start
Temporary breakpoint 1 at 0x400560: file a.c, line 5.
Starting program: /home/nan/a

Temporary breakpoint 1, main () at a.c:5
5       a.c: No such file or directory.
(gdb) set substitute-path /home/nan /home/ki
(gdb) n
6                       struct tm local = {0};
(gdb)
7                       struct tm gmt = {0};
(gdb)
9                       localtime_r(&now, &local);
(gdb)
10                      gmtime_r(&now, &gmt);
(gdb)
12                      return 0;
```

`/home/ki` gdb
`set substitute-path /home/nan /home/ki`
gdb
gdb .

nanxiao

```
#include <stdio.h>

void fun1(void)
{
        int i = 0;

        i++;
        i = i * 2;
        printf("%d\n", i);
}

void fun2(void)
{
        int j = 0;

        fun1();
        j++;
        j = j * 2;
        printf("%d\n", j);
}

int main(void)
{
        fun2();
        return 0;
}
```

gdb “`-tui`” `gdb -tui program` gdb “`Crtl+X+A`”

```
   ┌──a.c──────────────────────────────────────────────────────────────
   │17              j++;
   │18              j = j * 2;
   │19              printf("%d\n", j);
   │20      }
   │21
   │22      int main(void)
   │23      {
B+>│24              fun2();
   │25              return 0;
   │26      }
   │27
   │28
   │29
   │30
   │31
   │32
   └───────────────────────────────────────────────────────────────────
native process 22141 In: main
Type "apropos word" to search for commands related to "word"...
Reading symbols from a...done.
(gdb) start
Temporary breakpoint 1 at 0x40052b: file a.c, line 24.
Starting program: /home/nan/a

Temporary breakpoint 1, main () at a.c:24
(gdb)
```

PC

“ `Crtl+X+A` ”

gdb .

nanxiao

```
#include <stdio.h>

void fun1(void)
{
        int i = 0;

        i++;
        i = i * 2;
        printf("%d\n", i);
}

void fun2(void)
{
        int j = 0;

        fun1();
        j++;
        j = j * 2;
        printf("%d\n", j);
}

int main(void)
{
        fun2();
        return 0;
}
```

gdb “`layout asm`”

```
          ┌──────────────────────────────────────────────────────────
  >│0x40052b <main+4>                callq  0x4004f3 <fun2>
   │0x400530 <main+9>                mov    $0x0,%eax
   │0x400535 <main+14>               leaveq
   │0x400536 <main+15>               retq
   │0x400537                         nop
   │0x400538                         nop
   │0x400539                         nop
   │0x40053a                         nop
   │0x40053b                         nop
   │0x40053c                         nop
   │0x40053d                         nop
   │0x40053e                         nop
   │0x40053f                         nop
   │0x400540 <__libc_csu_fini>       repz retq
   │0x400542                         data16 data16 data16 data16 nop
   │0x400550 <__libc_csu_init>       mov    %rbp,-0x28(%rsp)
   └──────────────────────────────────────────────────────────────
native process 44658 In: main

(gdb) start
Temporary breakpoint 1 at 0x40052b: file a.c, line 24.
Starting program: /home/nan/a

Temporary breakpoint 1, main () at a.c:24
(gdb)
```

“ `layout split` ”

```
   ┌──a.c─────────────────────────────────────────────────────────────
  >│24              fun2();
   │25              return 0;
   │26      }
   │27
   │28
   │29
   │30
   └──────────────────────────────────────────────────────────────────
  >│0x40052b <main+4>       callq  0x4004f3 <fun2>
   │0x400530 <main+9>       mov    $0x0,%eax
   │0x400535 <main+14>      leaveq
   │0x400536 <main+15>      retq
   │0x400537                nop
   │0x400538                nop
   │0x400539                nop
   │0x40053a                nop
   └──────────────────────────────────────────────────────────────────
native process 44658 In: main

(gdb) start
Temporary breakpoint 1 at 0x40052b: file a.c, line 24.
Starting program: /home/nan/a

Temporary breakpoint 1, main () at a.c:24
(gdb)
```

gdb .

nanxiao

```
#include <stdio.h>

void fun1(void)
{
        int i = 0;

        i++;
        i = i * 2;
        printf("%d\n", i);
}

void fun2(void)
{
        int j = 0;

        fun1();
        j++;
        j = j * 2;
        printf("%d\n", j);
}

int main(void)
{
        fun2();
        return 0;
}
```

gdb “ `layout regs` ”

```
┌──Register group: general─────────────────────────────────────────────
│rax            0x34e4590f60     227169341280     rbx            0x
│rcx            0x0      0                        rdx            0x
│rsi            0x7fffffffe4a8   140737488348328  rdi            0x
│rbp            0x7fffffffe3c0   0x7fffffffe3c0   rsp            0x
│r8             0x34e458f300     227169334016     r9             0x
│r10            0x7fffffffe210   140737488347664  r11            0x
│r12            0x4003e0 4195296                  r13            0x
└──────────────────────────────────────────────────────────────────────
   │17              j++;
   │18              j = j * 2;
   │19              printf("%d\n", j);
   │20      }
   │21
   │22      int main(void)
   │23      {
  >│24              fun2();
   └──────────────────────────────────────────────────────────────────
native process 12552 In: main
Reading symbols from a...done.
(gdb) start
Temporary breakpoint 1 at 0x40052b: file a.c, line 24.
Starting program: /home/nan/a

Temporary breakpoint 1, main () at a.c:24
(gdb)
```

“ `tui reg float` ”

```
┌──Register group: float─────────────────────────────────────────────────
│st0            0        (raw 0x00000000000000000000)
│st1            0        (raw 0x00000000000000000000)
│st2            0        (raw 0x00000000000000000000)
│st3            0        (raw 0x00000000000000000000)
│st4            0        (raw 0x00000000000000000000)
│st5            0        (raw 0x00000000000000000000)
│st6            0        (raw 0x00000000000000000000)
└────────────────────────────────────────────────────────────────────────
   │16              fun1();
   │17              j++;
   │18              j = j * 2;
   │19              printf("%d\n", j);
   │20      }
   │21
   │22      int main(void)
   │23      {
   └─────────────────────────────────────────────────────────────────────
native process 12552 In: main
Temporary breakpoint 1 at 0x40052b: file a.c, line 24.
Starting program: /home/nan/a

Temporary breakpoint 1, main () at a.c:24
(gdb) tui reg float
```

“ `tui reg system` ”

```
┌──Register group: system──────────────────────────────────────────────
│orig_rax       0xffffffffffffffff       -1
│
│
│
│
│
│
└──────────────────────────────────────────────────────────────────────
   │16              fun1();
   │17              j++;
   │18              j = j * 2;
   │19              printf("%d\n", j);
   │20      }
   │21
   │22      int main(void)
   │23      {
   └──────────────────────────────────────────────────────────────────
native process 12552 In: main

Temporary breakpoint 1, main () at a.c:24
(gdb) tui reg system
(gdb)
```

“ `tui reg general` ”

```
┌──Register group: general─────────────────────────────────────────────
│rax            0x34e4590f60     227169341280     rbx            0x
│rcx            0x0      0                        rdx            0x
│rsi            0x7fffffffe4a8   140737488348328  rdi            0x
│rbp            0x7fffffffe3c0   0x7fffffffe3c0   rsp            0x
│r8             0x34e458f300     227169334016     r9             0x
│r10            0x7fffffffe210   140737488347664  r11            0x
│r12            0x4003e0 4195296                  r13            0x
└──────────────────────────────────────────────────────────────────────
   │16              fun1();
   │17              j++;
   │18              j = j * 2;
   │19              printf("%d\n", j);
   │20      }
   │21
   │22      int main(void)
   │23      {
   └──────────────────────────────────────────────────────────────────
native process 12552 In: main
(gdb) tui reg general
(gdb)
```

gdb .

nanxiao

```
#include <stdio.h>

void fun1(void)
{
        int i = 0;

        i++;
        i = i * 2;
        printf("%d\n", i);
}

void fun2(void)
{
        int j = 0;

        fun1();
        j++;
        j = j * 2;
        printf("%d\n", j);
}

int main(void)
{
        fun2();
        return 0;
}
```

gdb
“`winheight <win_name> [+ | -]count`” `winheight`
`win` `win_name` `src` `cmd` `asm` `regs`
`src`

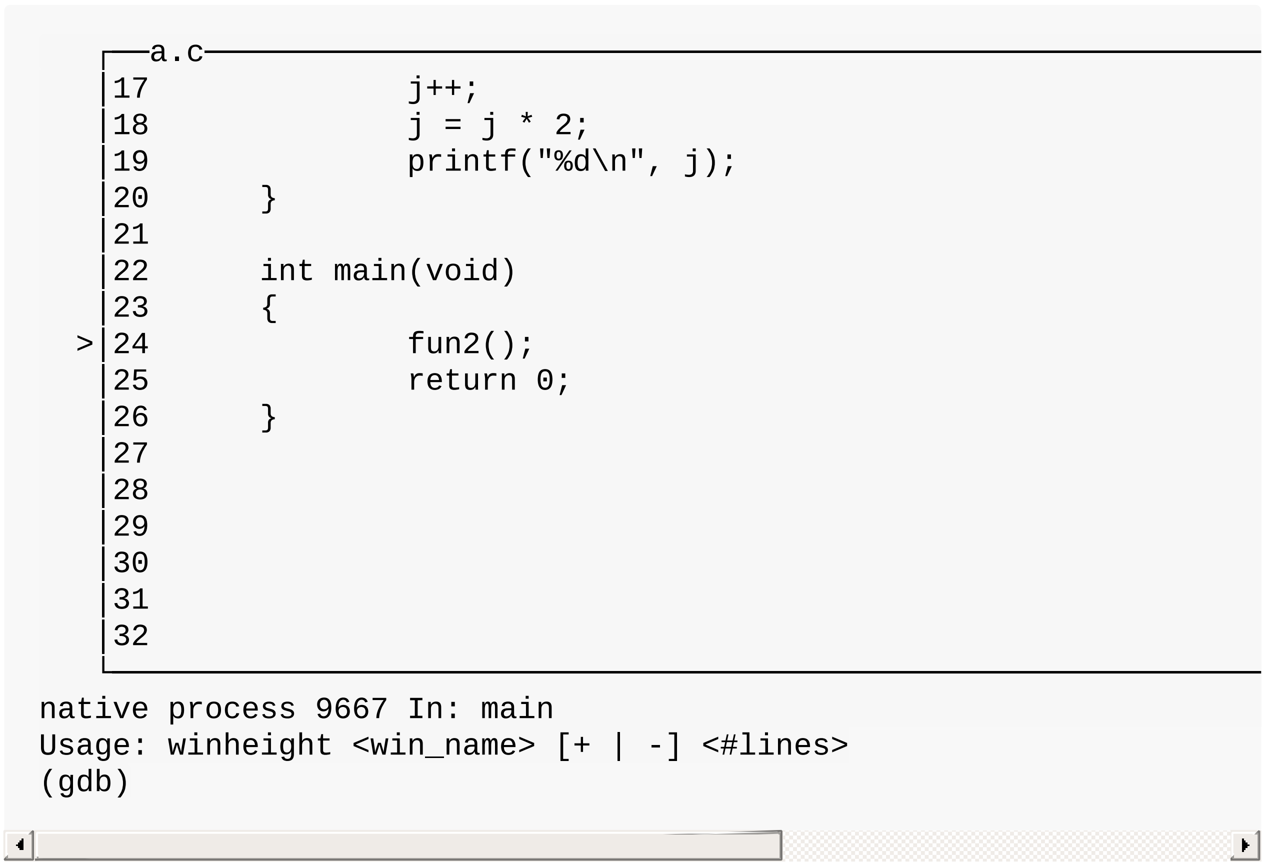
```
   ┌──a.c──────────────────────────────────────────────────────────
   │17              j++;
   │18              j = j * 2;
   │19              printf("%d\n", j);
   │20      }
   │21      int main(void)
   │23      {
   │24              fun2();
B+>│25
   │              return 0;
   │26      }
   │27
   │
   │
   │
   │
   │
   └───────────────────────────────────────────────────────────────
native process 9667 In: main
Usage: winheight <win_name> [+ | -] <#lines>
(gdb) start
Temporary breakpoint 1 at 0x40052b: file a.c, line 24.
Starting program: /home/nan/a

Temporary breakpoint 1, main () at a.c:24
```

“ `winheight src -5` ”

```
   ┌──a.c──────────────────────────────────────────────────────────
   │17              j++;
   │18              j = j * 2;
   │19              printf("%d\n", j);
   │20      }
   │21
   │22      int main(void)
   │23      {
  >│24              fun2();
   │25              return 0;
   │26      }
   │27
   └───────────────────────────────────────────────────────────────
native process 9667 In: main
Usage: winheight <win_name> [+ | -] <#lines>
(gdb)
```

“ `winheight src +5` ”



```
   ┌──a.c─────────────────────────────────────────────────
   │17              j++;
   │18              j = j * 2;
   │19              printf("%d\n", j);
   │20      }
   │21
   │22      int main(void)
   │23      {
  >│24              fun2();
   │25              return 0;
   │26      }
   │27
   │28
   │29
   │30
   │31
   │32
   └──────────────────────────────────────────────────────
native process 9667 In: main
Usage: winheight <win_name> [+ | -] <#lines>
(gdb)
```

gdb .

nanxiao

gdb

gdb “-” “--”

```
$ gdb -help
$ gdb --help

$ gdb -args ./a.out a b c
$ gdb --args ./a.out a b c
```

xmj

```
#include <stdio.h>

#define NAME "Joe"

int main()
{
  printf ("Hello %s\n", NAME);
  return 0;
}
```

`gcc -g`

```
(gdb) p NAME
No symbol "NAME" in current context.
```

gdb `gcc -g3`

```
(gdb) p NAME
$1 = "Joe"
```

gdb

xmj

gdb

gdb
tab

```
b -> break
c -> continue
d -> delete
f -> frame
i -> info
j -> jump
l -> list
n -> next
p -> print
r -> run
s -> step
u -> until
```

```
aw -> awatch
bt -> backtrace
dir -> directory
disas -> disassemble
fin -> finish
ig -> ignore
ni -> nexti
rw -> rwatch
si -> stepi
tb -> tbreak
wa -> watch
win -> winheight
```

xmj

nanxiao

gdb shell make

gdb shell

```
(gdb) shell ls
```

```
(gdb) !ls
```

"!"

(build) make

```
(gdb) make CFLAGS="-g -O0"
```

gdb

xmj

gdb cd pwd

gdb

gdb gdb

```
(gdb) pwd
Working directory /home/xmj.
(gdb) cd tmp
Working directory /home/xmj/tmp.
```

gdb

xmj

```
$ gdb -q `which gdb`
Reading symbols from /home/xmj/install/binutils-gdb-git/bin/gdb...
(gdb) r -q
Starting program: /home/xmj/install/binutils-gdb-git/bin/gdb -q
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_d
(gdb)
```

gdb gdb gdb

```
$ gdb -q `which gdb`
Reading symbols from /home/xmj/install/binutils-gdb-git/bin/gdb...
(gdb) set prompt (main gdb)
(main gdb) r -q
Starting program: /home/xmj/install/binutils-gdb-git/bin/gdb -q
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_d
(gdb)
```

`set prompt (main gdb)`

gdb

xmj

gdb

```
$ gdb -args ./a.out a b c
```

gdb

```
(gdb) set args a b c
(gdb) show args
Argument list to give program being debugged when it is started is
```

```
(gdb) r a b
Starting program: /home/xmj/tmp/a.out a b
(gdb) show args
Argument list to give program being debugged when it is started is
(gdb) r
Starting program: /home/xmj/tmp/a.out a b
```

`run`

```
(gdb) set args
```

gdb

xmj

```
(gdb) u 309
Warning: couldn't activate thread debugging using libthread_db: Ca
Warning: couldn't activate thread debugging using libthread_db: Ca
warning: Unable to find libthread_db matching inferior's thread li
```

gdb `set env varname=value`

LD_PRELOAD

```
set env LD_PRELOAD=/lib/x86_64-linux-gnu/libpthread.so.0
```

~/.gdbinit

gdb

xmj

`help` gdb

1 `help`

```
(gdb) help
List of classes of commands:

aliases -- Aliases of other commands
breakpoints -- Making program stop at certain points
data -- Examining data
files -- Specifying and examining files
internals -- Maintenance commands
obscure -- Obscure features
running -- Running the program
stack -- Examining the stack
status -- Status inquiries
support -- Support facilities
tracepoints -- Tracing of program execution without stopping the pr
user-defined -- User-defined commands

Type "help" followed by a class name for a list of commands in tha
Type "help all" for the list of all commands.
Type "help" followed by command name for full documentation.
Type "apropos word" to search for commands related to "word".
Command name abbreviations are allowed if unambiguous.
```

2 `help class`

```
(gdb) help data
Examining data.

List of commands:

append -- Append target code/data to a local file
append binary -- Append target code/data to a raw binary file
append binary memory -- Append contents of memory to a raw binary f
append binary value -- Append the value of an expression to a raw b
append memory -- Append contents of memory to a raw binary file
append value -- Append the value of an expression to a raw binary f
call -- Call a function in the program
disassemble -- Disassemble a specified section of memory
display -- Print value of expression EXP each time the program stop
dump -- Dump target code/data to a local file
dump binary -- Write target code/data to a raw binary file
dump binary memory -- Write contents of memory to a raw binary file
dump binary value -- Write the value of an expression to a raw bina
......
```

3 `help command`

```
(gdb) help mem
Define attributes for memory region or reset memory region handling
Usage: mem auto
   mem <lo addr> <hi addr> [<mode> <width> <cache>],
where <mode>  may be rw (read/write), ro (read-only) or wo (write-o
  <width> may be 8, 16, 32, or 64, and
  <cache> may be cache or nocache
```

4 `apropos regexp` `regexp`

```
(gdb) apropos set
awatch -- Set a watchpoint for an expression
b -- Set breakpoint at specified line or function
br -- Set breakpoint at specified line or function
bre -- Set breakpoint at specified line or function
brea -- Set breakpoint at specified line or function
......
```

gdb

nanxiao

gdb

```
#include <stdio.h>
#include <wchar.h>

int main(void)
{
        char str1[] = "abcd";
        wchar_t str2[] = L"abcd";

        return 0;
}
```

gdb “`set logging on`” gdb
“`gdb.txt`”
“`set logging file file`”

```
(gdb) set logging file log.txt
(gdb) set logging on
Copying output to log.txt.
(gdb) start
Temporary breakpoint 1 at 0x8050abe: file a.c, line 6.
Starting program: /data1/nan/a
[Thread debugging using libthread_db enabled]
[New Thread 1 (LWP 1)]
[Switching to Thread 1 (LWP 1)]

Temporary breakpoint 1, main () at a.c:6
6               char str1[] = "abcd";
(gdb) n
7               wchar_t str2[] = L"abcd";
(gdb) x/s str1
0x804779f:      "abcd"
(gdb) n
9               return 0;
(gdb) x/ws str2
0x8047788:      U"abcd"
(gdb) q
A debugging session is active.

        Inferior 1 [process 9931    ] will be killed.

Quit anyway? (y or n) y
```

log.txt

```
bash-3.2# cat log.txt
Temporary breakpoint 1 at 0x8050abe: file a.c, line 6.
Starting program: /data1/nan/a
[Thread debugging using libthread_db enabled]
[New Thread 1 (LWP 1)]
[Switching to Thread 1 (LWP 1)]

Temporary breakpoint 1, main () at a.c:6
6               char str1[] = "abcd";
7               wchar_t str2[] = L"abcd";
0x804779f:      "abcd"
9               return 0;
0x8047788:      U"abcd"
A debugging session is active.

        Inferior 1 [process 9931    ] will be killed.

Quit anyway? (y or n)
```

log.txt gdb

“ `set logging overwrite on` ”
“ `set logging redirect on` ” gdb
gdb .

nanxiao