PTRACE How-to

2015/10/8

Wooyeon Lee

ptrace

- Linux built-in process tracing mechanism.
- Provided as a system call.

- With ptrace,
  - a parent process (tracer) can trace its child processes (tracee).
  - can read/write to register and memory of tracee process.

ptrace

- #include <sys/ptrace.h>
- long ptrace(enum __ptrace_request request,
  pid_t pid, void *addr, void *data);

- => ptrace(REQ_CODE, PID, ADDR, DATA)
  - REQ_CODE: a code representing one of following available actions
    - start/end tracing, read/write data, etc.
  - PID: a process id of tracee
  - ADDR: a memory address where you read/write data from/to
  - DATA: a data that you read/write

- Enter ‘man ptrace’ in your command-line

Start tracing

- PTRACE_TRACEME
  - ptrace(PTRACE_TRACEME, 0, NULL, NULL)
  - Only ptrace method invoked by child

- PTRACE_ATTACH / PTRACE_DETACH
  - ptrace(PTRACE_ATTACH, pid, NULL, NULL)
  - ptrace(PTRACE_DETACH, pid, NULL, NULL)
  - It makes a target process as its child

Example

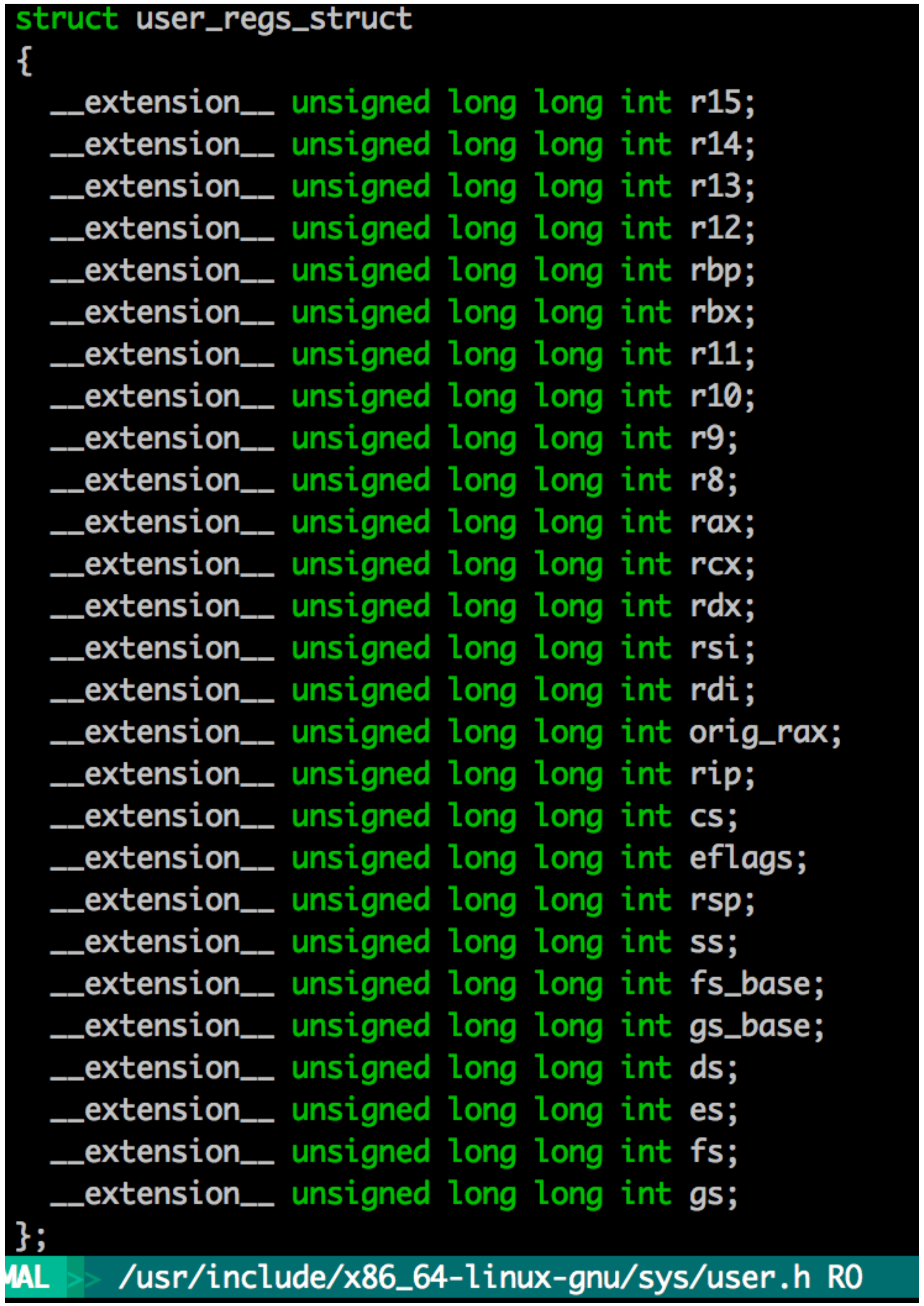
```
pid_t child;
child = fork();

if (child == 0) { /* child process */
            ptrace(PTRACE_TRACEME, 0, NULL, NULL);
            execvp(“/bin/ls”);
// INACCESSIBLE
}
else { /* parent process */
            waitpid(child, …)
            // WE GOT CHILD!
}
```

Register Access

- PTRACE_GETREGS
  - ptrace(PTRACE_GETREGS, pid, NULL, &regs)
- PTRACE_SETREGS
  - ptrace(PTRACE_SETREGS, pid, NULL, &regs)

```
// data structure representing registers
struct user_regs_struct regs;
```



```
struct user_regs_struct
{
  __extension__ unsigned long long int r15;
  __extension__ unsigned long long int r14;
  __extension__ unsigned long long int r13;
  __extension__ unsigned long long int r12;
  __extension__ unsigned long long int rbp;
  __extension__ unsigned long long int rbx;
  __extension__ unsigned long long int r11;
  __extension__ unsigned long long int r10;
  __extension__ unsigned long long int r9;
  __extension__ unsigned long long int r8;
  __extension__ unsigned long long int rax;
  __extension__ unsigned long long int rcx;
  __extension__ unsigned long long int rdx;
  __extension__ unsigned long long int rsi;
  __extension__ unsigned long long int rdi;
  __extension__ unsigned long long int orig_rax;
  __extension__ unsigned long long int rip;
  __extension__ unsigned long long int cs;
  __extension__ unsigned long long int eflags;
  __extension__ unsigned long long int rsp;
  __extension__ unsigned long long int ss;
  __extension__ unsigned long long int fs_base;
  __extension__ unsigned long long int gs_base;
  __extension__ unsigned long long int ds;
  __extension__ unsigned long long int es;
  __extension__ unsigned long long int fs;
  __extension__ unsigned long long int gs;
};
MAL  >> /usr/include/x86_64-linux-gnu/sys/user.h RO
```

Example

```
#include <sys/user.h>

struct user_regs_struct regs;

ptrace(PTRACE_GETREGS, child, NULL, &regs);

//registers
unsigned long long int rax = regs.rax;
unsigned long long int rdi = regs.rdi;
…
```

System call convention

- System call number & return value

arch/ABI	instruction	syscall #	retval
x86_64	syscall	rax	rax

- System call arguments.

arch/ABI	arg1	arg2	arg3	arg4	arg5	arg6
X86_64	rdi	rsi	rdx	r10	r8	r9

- Enter “man syscall” in command-line

Memory Access

- PTRACE_PEEKDATA/TEXT
  - data = ptrace(PTRACE_PEEKDATA, pid, addr, NULL)

- PTRACE_POKEDATA/TEXT
  - ptrace(PTRACE_POKEDATA, pid, addr, &buf)

- Transfer(copy) word data.
  - A word contains 64bits for a 64bit architecture.
  - 32bits for 32bit arch

Header Files

- /usr/include/x86_64-linux-gnu/
    - sys/reg.h
    - sys/user.h
    - sys/syscall.h -> bits/syscall.h
    - sys/ptrace.h
    - asm/ptrace-abi.h

PRACTICE

- Tracing an incremental counter
- Manipulating a behavior of program

Optional: USER area Access

- PTRACE_PEEKUSER
- PTRACE_POKEUSER
- USER area holds the registers and other information about the process (<sys/user.h>)

Thanks

sec-tas@cmslab.snu.ac.kr

References:

http://man7.org/linux/man-pages/man2/ptrace.2.html

http://www.ee.ryerson.ca/~courses/coe518/LinuxJournal/elj2002-103-ptrace1.pdf

http://delivery.acm.org/10.1145/610000/603626/6210.html?ip=147.46.246.161&id=603626&acc=ACTIVE%20SERVICE&key=0EC22F8658578FE1%2ED83A6478590749B7%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&CFID=550559368&CFTOKEN=13078748&__acm__=1444205909_e1b0402c13a974f603db2271da839e99