Implementation of a debugger under Linux (on x86 machines, of course :)

Stephane A. Sezer

December 8, 2011

Plan

1. Debugging a program on an Intel CPU
   - Interruptions
   - Generating an interrupt
   - Debug registers

Plan

Interruptions

- Redirection of the control flow to the kernel
- The kernel handle the interrupts and gives back the control to userland
- Used for debug traps

Plan

Breakpoints

- Stop execution of the program and generate an interrupt
- `int3` / `0xCC`
- `int $0x03` / `0xCD03`
- Used to implement “regular” breakpoints in GDB

Plan

Debug registers

- `DR0 - DR7`
- Used to implement GDB's `watch` and `hbreak` commands
- 3v1l ;)

Plan

2. ptrace(2)
   - Working with memory
   - Working with registers
   - Playing with the process' control flow
   - Sending signals to the process

ptrace(2)

```
long ptrace(enum __ptrace_request request,
            pid_t pid, void *addr, void *data);
```

Plan

Working with memory

- `PTRACE_PEEKDATA`, `PTRACE_POKEDATA`
- `PTRACE_PEEKTEXT`, `PTRACE_POKETEXT`
- read/write to/from the process' memory
- `ptrace(PTRACE_PEEKDATA, pid, 0x1337, NULL);`

Plan

2. ptrace(2)
   - Working with memory
   - Working with registers
   - Playing with the process' control flow
   - Sending signals to the process

Working with registers

- `struct user;`
- `struct user_regs_struct;`
- `PTRACE_PEEKREGS`, `PTRACE_POKEREGS`
- read/write to/from the process' register base

struct user_regs_struct

```
struct user_regs_struct
{
  long int ebx;
  long int ecx;
  long int edx;
  long int esi;
  [...]
  long int eip;
  long int xcs;
  long int eflags;
  long int esp;
  long int xss;
};
```

Plan

2. ptrace(2)
   - Working with memory
   - Working with registers
   - Playing with the process' control flow
   - Sending signals to the process

Playing with the process' control flow

- `PTRACE_CONT`, to continue execution
- `PTRACE_SYSCALL`, to continue execution until next syscall

Plan

2. ptrace(2)
   - Working with memory
   - Working with registers
   - Playing with the process' control flow
   - Sending signals to the process

Sending signals to processes

- `PTRACE_CONT` can take an additionnal argument
- Used to implement GDB's `signal` command

Questions ?

WUT?