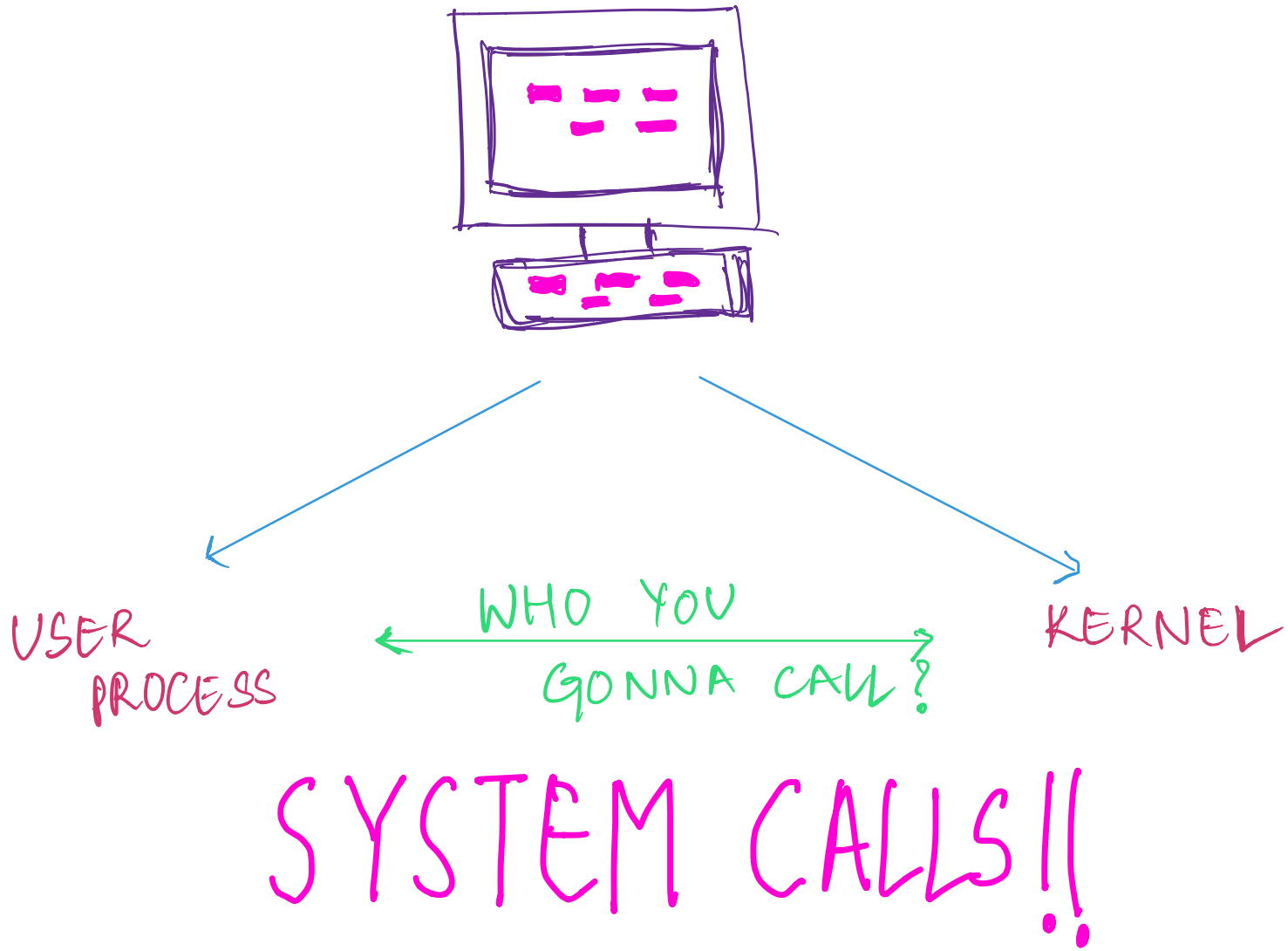
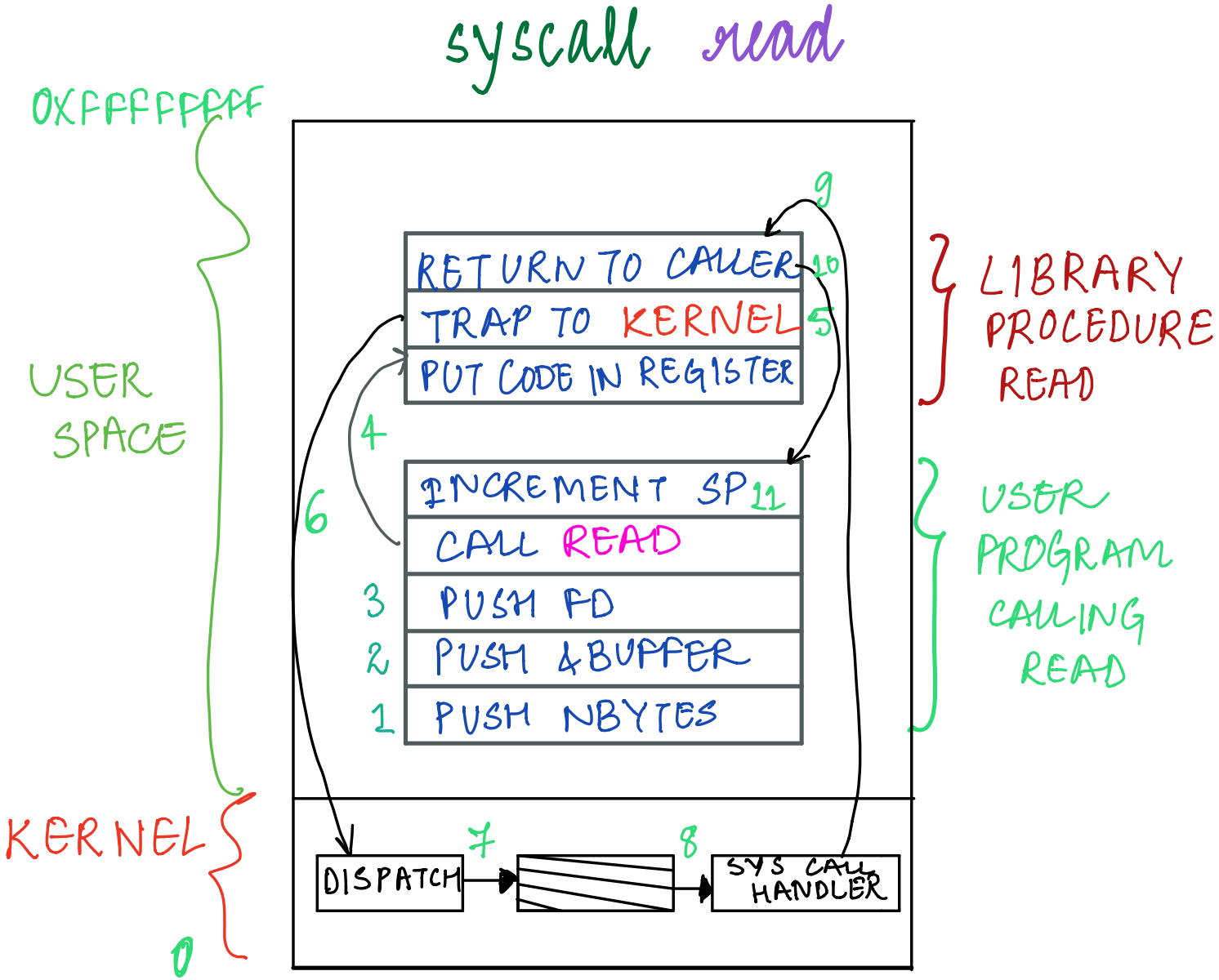
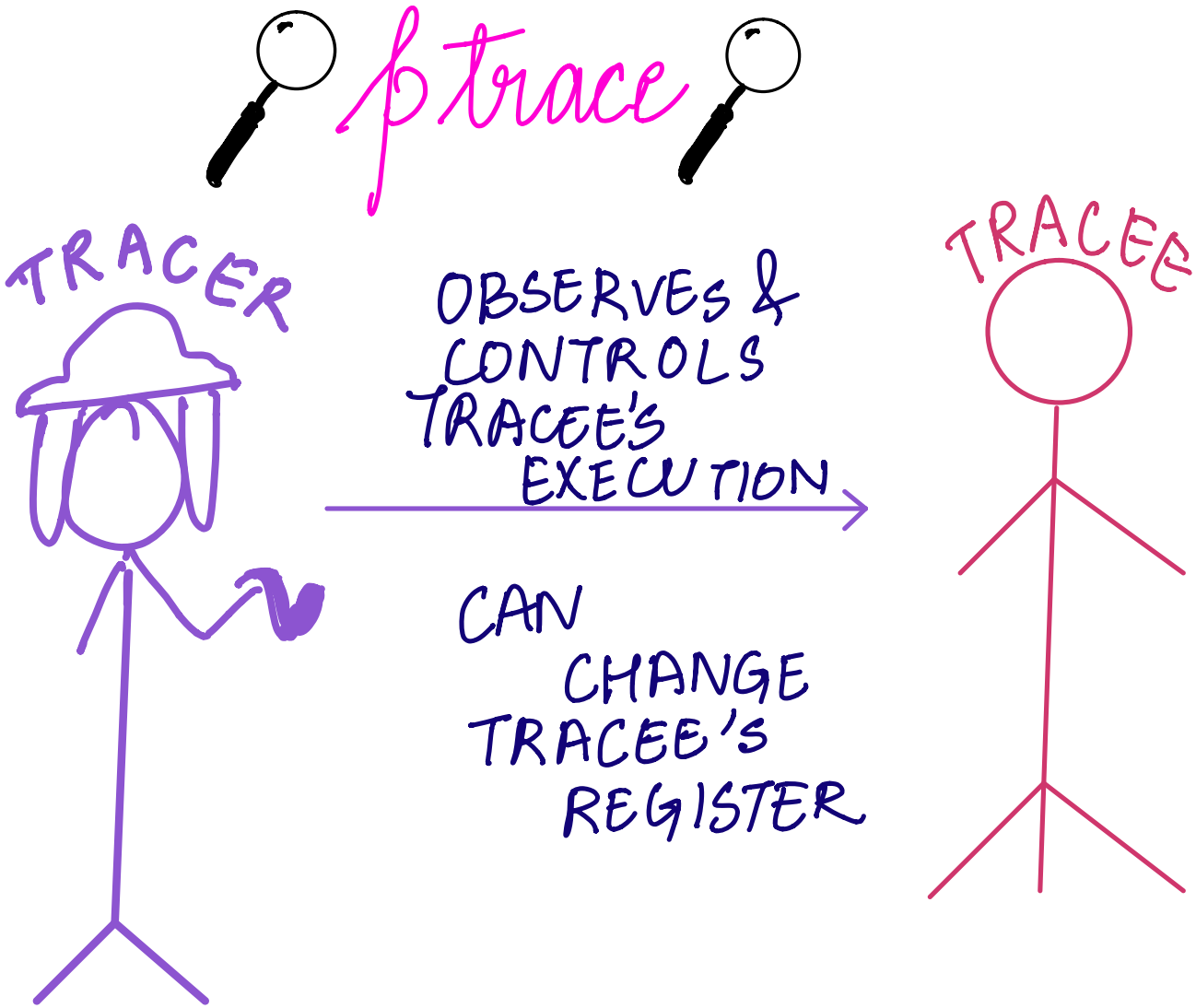
PTRACE

• SHERLOCK HOLMES OF SYSCALLS •

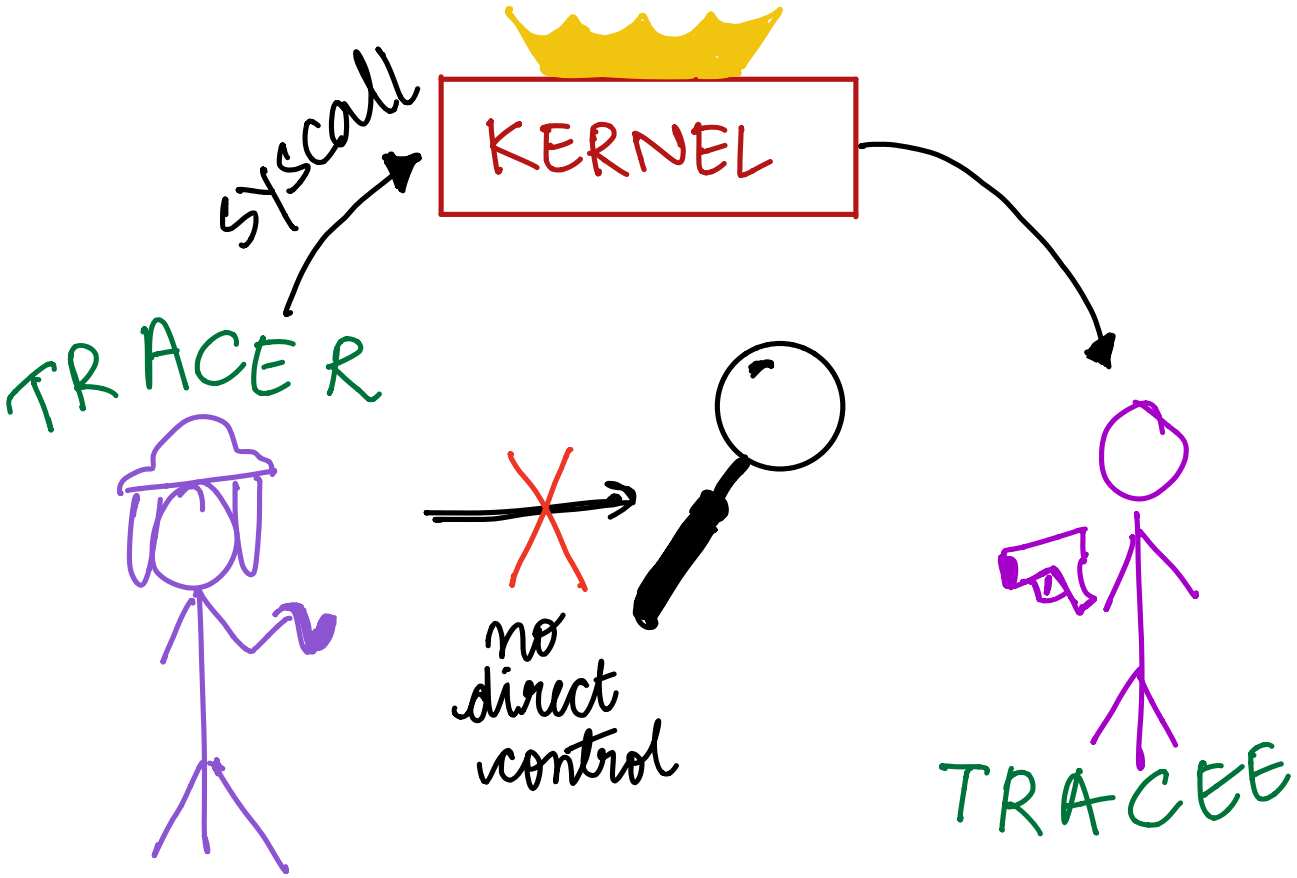
@gaurwrgi


USER PROCESS
WHO YOU GONNA CALL?
KERNEL
SYSTEM CALLS!!


syscall read
0XFFFFFFFF
USER SPACE
KERNEL
0
RETURN TO CALLER
TRAP TO KERNEL
PUT CODE IN REGISTER
INCREMENT SP
CALL READ
PUSH FD
PUSH &BUFFER
PUSH NBYTES
DISPATCH
SYS CALL HANDLER
1
2
3
4
5
6
7
8
9
10
11
LIBRARY PROCEDURE READ
USER PROGRAM CALLING READ


ptrace
TRACER
OBSERVES & CONTROLS TRACEE'S EXECUTION
CAN CHANGE TRACEE'S REGISTER
TRACEE

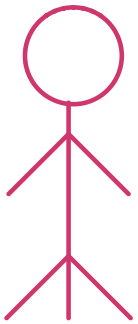
WHY IS PTRACE A SYSCALL?



HOW DO I TRACE A PROCESS?

ptrace(PTRACE_foo, pid, ...)
| > thread id

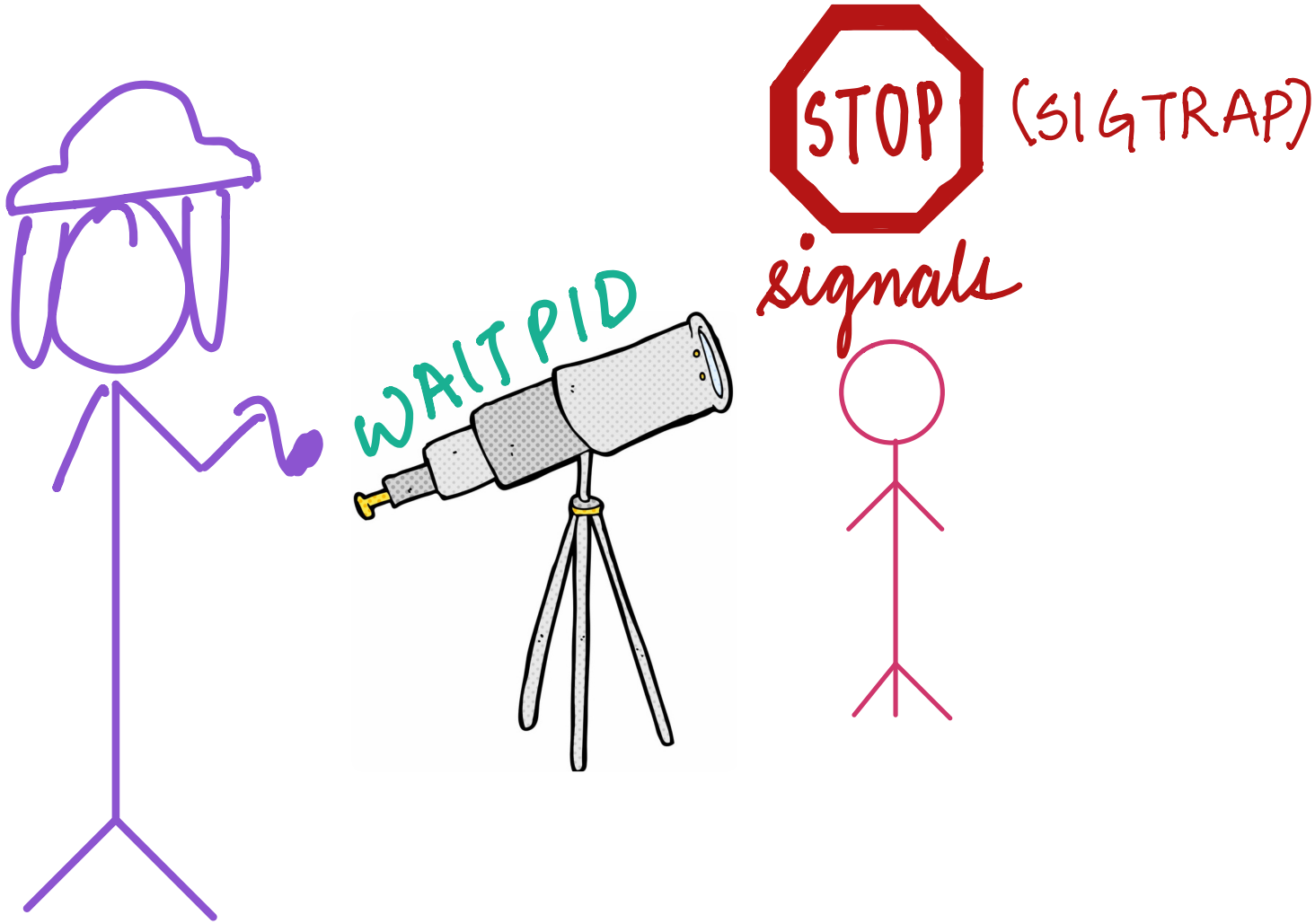
\+ PTRACE_ATTACH



or

1. PTRACE_ME!!
2. execve

PLEASE DON'T STOP THE TRACEE!



What Can Ptrace Do?

Ptrace_peekdata

Ptrace_pokedata

Ptrace_getregs

Ptrace_o_traceclone

Ptrace_o_tracesysgood

PTRACE IN THE WILD

strace

- TRACE SYSTEM CALLS & SIGNALS
- USEFUL DIAGNOSTIC, INSTRUCTIONAL & DEBUGGING TOOL.

ltrace

- DISPLAY THE CALLS A USERSPACE APPLICATION MAKES TO SHARED LIBRARIES.
- DYNAMICALLY LOADED SHARED LIBRARIES NOT STATICALLY LINKED

PYFLAME

- PYTHON PROFILER
- PERIODICALLY PTRACE_ATTACH, PEEK PYTHON STACK TRACE & DETACH
- PTRACE READS DATA FROM PYTHON PROCESS' VIRTUAL MEMORY

GDB

- gdb <EXECUTABLE>

  PTRACE_ATTACH TO PROCESS

- STOP AT LINE X

  0XCC (INT3) SOFTWARE INTERRUPT

- SIGTRAP AT BREAKPOINT

PTRACE_GETREGS TO EXAMINE STACK

PROBLEMS WITH PTRACE

- $$ SECURITY $$

- SINGLE USER ABLE TO EXAMINE MEMORY AND RUNNING STATE

- SOLUTION-
  ONLY ALLOW PTRACE DIRECTLY FROM PARENT TO CHILD

PROBLEMS WITH PTRACE

- 2 CONTEXT SWITCHES
- NOT A POSIX SYSTEM CALL
  BEHAVIOR VARIES FROM OS TO OS.
- HUGE MEMORY OVERHEAD
- TRACER MUST BECOME TRACEE'S PARENT.

Alternatives to Ptrace

- process_vm_{readv/writev}
- Transfer data between address spaces
- Data does not have to go through kernel space