**PTrace for Android**

**Introduction**

Ptrace system call provides a framework with which a process (tracer) can peek through the systems calls any other tracee process would use and modify the attributes of trace process. By attaching to another process using the ptrace call, a tool has extensive control over the operation of its target. This includes manipulation of its file descriptors, memory, and registers. It can single-step through the target's code, can observe and intercept system calls and their results, and can manipulate the target's signal handlers and both receive and send signals on its behalf. The ability to write into the target's memory allows not only its data store to be changed, but also the application's own code segment, allowing the controller to install breakpoints and patch the running code of the target.

**Design**

We designed a tracer and debugger application for android with both command line interface and GUI. The application can either spawn any other android application and attach to it giving the tracer application the ability to trace and step debug the tracee application.

**Features**

Tracer: The core feature of this tool its ability to trace the systems calls of the tracee process along with each system call’s arguments, return values. We also provided additional filters for tracing systems calls. With these features, user can provide list of system calls he/she wants to trace and tracer will just give the details of those system calls provided by the user.

Step execution (Debugger): This application also provides interface for step execution of process, a classic function of debugger. By enabling this option, the tracee process will be blocked at every system call invocation and wait for user input to proceed ahead. The user also can provide list of system calls after enabling this option. Then the tracee process will be blocked only when these system calls are invoked. This feature fine grained to control to user who wants to observer the system call execution of an application.

Blocker: Last important feature our application provides is the ability to block the processes from using certain system calls. User can provide list of system calls that should not be used on the system, for instance any network related system calls. Any application that tries accessing these blocked system calls will be terminated.

**Usage**

Command Line:

ptrace\_tool [-p <pid> [ -i <blocklist\_file> | -h <break\_point\_syscalls>]] [<cmd>]

pid: process id of tracee app.

Blocklist\_file: list of system calls to be blocked

Break\_point\_system calls: The list of systems calls at which the tracee application needs to halt for stepwise execution. This will take option ‘all’ to specify every system call should go through stepwise execution.

Cmd: command that executes a tracee process.

**References**

ptrace(2) - Linux man page - [http://linux.die.net/man/2/ptrace](http://linux.die.net/man/2/ptrace" \t "_blank)  
strace(1) - Linux man page - [http://linux.die.net/man/1/strace](http://linux.die.net/man/1/strace" \t "_blank)  
strace source for android - [https://github.com/android/platform\_external\_strace](https://github.com/android/platform_external_strace" \t "_blank)  
Strace 4.8 @XDA - [http://forum.xda-developers.com/showthread.php?t=2516002](http://forum.xda-developers.com/showthread.php?t=2516002" \t "_blank)  
MiniStrace - [https://github.com/nelhage/ministrace](https://github.com/nelhage/ministrace" \t "_blank)